Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – IF184802

# PENERAPAN TEKNOLOGI REALITAS VIRTUAL IMERSIF DAN INTERAKTIF UNTUK VISUALISASI DATA PENELITI


**FARIZ ARDIN ADHIYASA**
**NRP 05111640000158**

**Dosen Pembimbing I**
**Dr. Eng. Nanik Suciati, S.Kom, M.Kom.**

**Dosen Pembimbing II**
**Hadziq Fabroyir, S.Kom., Ph.D**

**DEPARTEMEN TEKNIK INFORMATIKA**
**Fakultas Teknologi Elektronika dan Informatika Cerdas**
**Institut Teknologi Sepuluh Nopember**
**Surabaya 2020**


**TUGAS AKHIR – IF184802**

# PENERAPAN TEKNOLOGI REALITAS VIRTUAL IMERSIF DAN INTERAKTIF UNTUK VISUALISASI DATA PENELITI

**FARIZ ARDIN ADHIYASA**
**NRP 05111640000158**

**Dosen Pembimbing I**
**Dr. Eng. Nanik Suciati, S.Kom, M.Kom.**

**Dosen Pembimbing II**
**Hadziq Fabroyir, S.Kom., Ph.D**

**DEPARTEMEN TEKNIK INFORMATIKA**
**Fakultas Teknologi Elektronika dan Informatika Cerdas**
**Institut Teknologi Sepuluh Nopember**
**Surabaya 2020**

*[Halaman ini sengaja dikosongkan]*


**UNDERGRADUATE THESIS – IF184802**

# IMPLEMENTATION OF IMMERSIVE VIRTUAL REALITY TECHNOLOGY AND INTERACTIVE FOR RESEARCHERS DATA VISUALIZATION

**FARIZ ARDIN ADHIYASA**
**NRP 05111640000158**

**Advisor I**
**Dr. Eng. Nanik Suciati, S.Kom, M.Kom.**

**Advisor II**
**Hadziq Fabroyir, S.Kom., Ph.D**

**DEPARTEMENT OF INFORMATICS**
**Faculty of Intelligent Electrical and Informatics Technology**
**Institut Teknologi Sepuluh Nopember**
**Surabaya 2020**

*[Halaman ini sengaja dikosongkan]*

## LEMBAR PENGESAHAN

## PENERAPAN TEKNOLOGI REALITAS VIRTUAL IMERSIF DAN INTERAKTIF UNTUK VISUALISASI DATA PENELITI

# TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Interaksi, Grafika, dan Seni
Program Studi S-1 Departemen Teknik Informatika
Fakultas Teknologi Elektronika dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember

Oleh:
**FARIZ ARDIN ADHIYAKSA**
NRP. 05111640000158

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Dr.Eng. Nanik Suciati, S.Kom., M.Kom. .....
NIP: 197104281994122001 (pembimbing 1)

Hadziq Fabroyir, S.Kom., Ph.D. .....
NIP: 198602272019031006 (pembimbing 2)

**SURABAYA**
**JUNI, 2020**

*[Halaman ini sengaja dikosongkan]*

**PENERAPAN TEKNOLOGI REALITAS VIRTUAL IMERSIF DAN INTERAKTIF UNTUK VISUALISASI DATA PENELITI**


| | |
|---|---|
| **Nama Mahasiswa** | **: Fariz Ardin Adhiyaksa** |
| **NRP** | **: 05111640000158** |
| **Departemen** | **: Teknik Informatika FTEIC-ITS** |
| **Dosen Pembimbing I** | **: Dr. Eng. Nanik Suciati, S.Kom., M.Kom.** |
| **Dosen Pembimbing II** | **: Hadziq Fabroyir, S.Kom., Ph.D.** |


## ABSTRAK


*Teknologi imersif adalah teknologi yang memungkinkan seseorang merasa berada di dalam dunia virtual dan mampu berinteraksi secara langsung dengan dunia tersebut seakan-akan tidak ada pemisah antara dunia virtual dengan dunia nyata. Salah satu teknologi imersif adalah perangkat Realitas Virtual perangkat realitas virtual mampu menciptakan lingkungan 3-dimensi yang imersif senhingga pengguna merasa berada didalam dunia virtual*

*Penggunaan realitas virtual umumnya digunakan pada game. Namun penggunaan perangkat Realitas Virtual juga dapat digunakan untuk visualisasi data. Penggunaan teknologi imersif seperti perangkat Realitas Virtual tergolong baru. Pengguna mampu mengeksplorasi data secara imersif menggunakan perangkat Realitas Virtual. Ide dalam Tugas Akhir ini adalah membangun aplikasi untuk visualisasi data peneliti menggunakan perangkat Realitas Virtual. Aplikasi ini dibangun menggunakan arsitektur client-server serta menggunakan kerangka kerja Unity3D dan Flask micro web framework.*

*Hasil dari pengujian dari Tugas Akhir ini telah terpenuhi dan dapat disimpulkan bahwa Tugas Akhir berjudul Penerapan Teknologi Realitas Virtual Imersif dan Interaktif untuk Visualisasi Data Peneliti ini dapat berjalan menggunakan arsitektur client-server dengan baik secara fungsional. Selain mendapatkan hasil*

*yang baik dari sisi fungsional, hasil pengujian dari pengguna juga mendapatkan hasil baik dengan penilaian 1-6 yaitu mendapatkan 5.27 untuk kejelasan informasi, 5.09 untuk kemudahan akses dan 5.18 dalam pengaturan tata letak pada aplikasi Realitas Virtual untuk visualisasi data peneliti.*

***Kata kunci:  Realitas Virtual, Visualisasi Data.***

**IMPLEMENTATION OF IMMERSIVE VIRTUAL REALITY TECHNOLOGY AND INTERACTIVE FOR RESEARCERS DATA VISUALIZATION**


| | |
|---|---|
| **Student Name** | **: Fariz Ardin Adhiyaksa** |
| **NRP** | **: 05111640000158** |
| **Major** | **: Teknik Informatika FTEIC-ITS** |
| **Advisor I** | **: Dr. Eng. Nanik Suciati, S.Kom., M.Kom.** |
| **Advisor II** | **: Hadziq Fabroyir, S.Kom., Ph.D.** |


## ABSTRACT


*Immersive technology is technology that allows a person to feel inside the virtual world and be able to interact directly with the world as if there was no separation between the virtual world and the real world. One of the immersive technologies is the Virtual Reality device, a virtual reality device capable of creating an immersive 3-dimensional environment so users feel they are in a virtual world.*

*Virtual Reality is commonly used in games. But Virtual Reality devices can also be used for data visualization. The use of immersive technology such as Virtual Reality devices is relatively new. Users are able to explore data immersively using Virtual Reality devices. The idea in this thesis is to build applications for data visualization of researchers using the Virtual Reality device. This application is built using a client-server architecture and uses the Unity3D framework and Flask micro web framework.*

*The results of the testing of this Thesis have been fulfilled and it can be concluded that the Thesis entitled The Application of Immersive and Interactive Virtual Reality Technology for Researchers Data Visualization can run using the client-server architecture functionally well. In addition to getting good results from the functional side, the test results from users also get good results with ratings 1-6 namely getting 5.2 for clarity of*

*information, 5.1 for ease of access and 5.2 in layout settings in the Virtual Reality application for visualizing researcher's data.*

***Keyword: Virtual Reality, Data Visualization.***

*[Halaman ini sengaja dikosongkan]*

# KATA PENGANTAR

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ

Puji syukur bagi Allah SWT, yang telah melimpahkan rahmat serta hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul **Penerapan Teknologi Realitas Virtual Imersif dan Interaktif untuk Visualisasi Data Peneliti.** Pengerjaan Tugas Akhir ini merupakan suatu langkah yang sangat baik bagi penulis. Dengan pengerjaan Tugas Akhir ini, penulis dapat belajar lebih banyak, menggali lebih dalam serta meningkatkan apa yang telah didapatkan penulis selama menempuh perkuliahan di Teknik Informatika ITS. Dengan Tugas Akhir ini penulis juga dapat menghasilkan suatu implementasi dari apa yang telah penulis pelajari. Selesainya Tugas Akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak. Sehingga pada kesempatan ini penulis mengucapkan syukur dan terima kasih kepada:

1. Allah Tuhan Yang Maha Esa atas
2. Kedua orang tua penulis, Bapak Amarudin dan Ibu Sulasmi yang selalu mencurahkan doa, memberi dukungan, perhatian, serta kasih sayang kepada penulis.
3. Bapak Dr.Eng. Nanik Suciati, S.Kom, M.Kom. Selaku pembimbing Tugas Akhir pertama yang telah memberikan arahan dalam mengerjakan Tugas Akhir ini.
4. Bapak Hadziq Fabroyir, S.Kom., Ph.D. Selaku pembimbing Tugas Akhir kedua yang dengan sabar membimbing penulis dalam mengerjakan Tugas Akhir ini.
5. TC’16 yang telah menjadi keluarga besar penulis selama masa perkuliahan
6. Teman-teman UKM IFLS yang telah menemani penulis selama berada di perantauan.
7. Teman-teman seangkatan yang menjalani Tugas Akhir yang bersama-sama berjuang dan saling mendukung hingga lulus bersama.

8. Serta semua pihak yang tidak dapat disebutkan satu per satu, yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan. Sehingga dengan kerendahan hati, penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depannya.

Surabaya, Juni 2020

Fariz Ardin Adhiyaksa

# DAFTAR ISI

LEMBAR PENGESAHAN ..... v
ABSTRAK ..... vii
ABSTRACT ..... ix
KATA PENGANTAR ..... xii
DAFTAR ISI ..... xiv
DAFTAR GAMBAR ..... xxi
DAFTAR TABEL ..... xxvi
DAFTAR KODE SUMBER ..... xxx
BAB I PENDAHULUAN ..... 1
1.1 Latar Belakang ..... 1
1.2 Rumusan Masalah ..... 2
1.3 Batasan Masalah ..... 2
1.4 Tujuan ..... 2
1.5 Manfaat ..... 3
1.6 Metodologi ..... 3
BAB II TINJAUAN PUSTAKA ..... 8
2.1 Realitas Virtual ..... 8
2.2 Unity3D ..... 8
2.3 Pemodelan 3-dimensi ..... 9
2.4 Bahasa Pemrograman C# ..... 9
2.5 Bahasa Pemrograman Python ..... 10
2.6 Google VR ..... 10
2.7 VR Box ..... 10
2.8 phpMyAdmin ..... 11

2.9 XAMPP ..... 11
2.10 Flask Micro Web Framework ..... 11
BAB III ANALISIS DAN PERANCANGAN ..... 13
3.1 Analisis Sistem ..... 13
3.1.1 Spesifikasi Kebutuhan Fungsional ..... 13
3.1.2 Spesifikasi Kebutuhan Non-Fungsional ..... 14
3.1.3 Spesifikasi *Use Case* ..... 14
3.1.3.1 Melihat Beranda ..... 15
3.1.3.2 Melihat Grafik Rata Rata H-Index berdasarkan Afiliasi ..... 16
3.1.3.3 Melihat Grafik Batang Total Dokumen Peneliti berdasarkan Afiliasi ..... 18
3.1.3.4 Melihat Grafik Total Dokumen Scopus berdasarkan Afiliasi ..... 20
3.1.3.5 Melihat Grafik Total Sitasi berdasarkan Afiliasi .21
3.1.3.6 Melihat Daftar Peneliti ..... 23
3.1.3.7 Mencari Peneliti ..... 24
3.1.3.8 Melakukan Filter Peneliti ..... 27
3.1.3.9 Filter berdasarkan Keahlian ..... 28
3.1.3.10 Filter berdasarkan Subjek ..... 30
3.1.3.11 Filter berdasarkan Afiliasi ..... 32
3.1.3.12 Mengurutkan Daftar Peneliti ..... 34
3.1.3.13 Urutkan berdasarkan Nama ..... 36
3.1.3.14 Urutkan berdasarkan H-Index ..... 38
3.1.3.15 Urutkan berdasarkan Total Dokumen ..... 39
3.1.3.16 Urutkan berdasarkan Total Dokumen Scopus ..... 41

3.1.3.17 Urutkan berdasarkan Total Sitasi ..... 43
3.1.3.18 Melihat Detail Peneliti..... 44
3.1.3.19 Melihat Daftar Afiliasi ..... 46
3.1.3.20 Melihat Detail Afiliasi..... 47
3.1.3.21 Melihat Peneliti berdasarkan Detail Afiliasi ..... 49
3.1.3.22 Melihat Daftar Subjek Penelitian ..... 51
3.1.3.23 Melihat Detail Subjek Penelitian..... 52
3.1.3.24 Melihat Peneliti berdasarkan Detail Subjek Penelitian..... 54
3.1.3.25 Melihat Diagram *Scatter* ..... 56
3.2 Perancangan Sistem.....58
3.2.1 Deskripsi Umum Sistem.....58
3.2.2 Arsitektur Sistem.....58
3.3 Perancangan Basis Data .....60
3.4 Perancangan Antarmuka Pengguna.....64
3.4.1 Rancangan Antarmuka Panel Konfigurasi .....64
3.4.2 Rancangan Antarmuka Beranda .....64
3.4.3 Rancangan Antarmuka Daftar Peneliti .....65
3.4.4 Rancangan Antarmuka Detail Peneliti .....66
3.4.5 Rancangan Antarmuka Daftar Afiliasi .....66
3.4.6 Rancangan Antarmuka Detail Afiliasi.....67
3.4.7 Rancangan Antarmuka Daftar Subjek Penelitian .....67
3.4.8 Rancangan Antarmuka Detail Subjek Penelitian.....68
3.4.9 Rancangan Antarmuka Diagram *Scatter* .....68
3.5 Perancangan Objek dalam Aplikasi Realitas Virtual .....69
3.5.1 Perancangan Koridor.....70

3.5.2 Perancangan Panel Kontrol ..... 70
BAB IV IMPLEMENTASI SISTEM..... 72
4.1 Lingkungan Operasi Sistem ..... 72
4.2 Implementasi Basis Data..... 73
4.2.1 Implementasi Tabel Afiliasi ..... 73
4.2.2 Implementasi Tabel Departemen..... 74
4.2.3 Implementasi Tabel Detail Subjek Area..... 75
4.2.4 Implementasi Tabel Fakultas..... 77
4.2.5 Implementasi Tabel Keahlian..... 77
4.2.6 Implementasi Tabel Peneliti ..... 78
4.2.7 Implementasi Tabel Subjek Area ..... 80
4.2.8 Implementasi Tabel Subjek Topik Tier 1 ..... 81
4.2.9 Implementasi Tabel Subjek Topik Tier 2 ..... 82
4.3 Implementasi Aplikasi Realitas Virtual ..... 83
4.3.1 Implementasi Aplikasi *Client* ..... 83
4.3.1.1 Implementasi Antarmuka Panel Konfigurasi .....83
4.3.1.2 Implementasi Antarmuka Beranda .....85
4.3.1.1 Implementasi Antarmuka Daftar Peneliti .....109
4.3.1.2 Implementasi Antarmuka Detail Peneliti .....120
4.3.1.3 Implementasi Antarmuka Daftar Subjek Penelitian .....124
4.3.1.4 Implementasi Antarmuka Detail Subjek Penelitian .....128
4.3.1.5 Implementasi Antarmuka Daftar Afiliasi .....131
4.3.1.6 Implementasi Antarmuka Detail Afiliasi .....134
4.3.1.7 Implementasi Antarmuka Diagram *Scatter* .....137

4.3.2 Implementasi Aplikasi Server .....142
4.3.2.1 API *End Point* Beranda ..... 143
4.3.2.2 API *End Point* Daftar Peneliti ..... 145
4.3.2.3 API *End Point* Detail Peneliti..... 149
4.3.2.4 API *End Point* Afiliasi ..... 152
4.3.2.5 API *End Point* Detail Afiliasi..... 153
4.3.2.6 API *End Point* Subjek Penelitian ..... 155
4.3.2.7 API *End Point* Detail Subjek Penelitian..... 157
4.3.2.8 API *End Point* Tekstur Diagram *Scatter* ..... 159
4.3.2.9 API *End Point* Tekstur Diagram *Pie Chart*..... 161
4.3.2.10 API *End Point* Perintah Suara ..... 162
BAB V PENGUJIAN DAN EVALUASI .....164
5.1 Pengujian Fungsional .....164
5.1.1 Lingkungan Operasi Sistem Pengujian Fungsional.164
5.1.2 Skenario Pengujian Fungsionalitas.....165
5.1.2.1 Uji Coba pada Panel Navigasi..... 166
5.1.2.2 Uji Coba pada Beranda..... 167
5.1.2.3 Uji Coba pada Daftar Peneliti..... 169
5.1.2.4 Uji Coba pada Daftar Subjek Penelitian..... 172
5.1.2.5 Uji Coba pada Detail Subjek Penelitian ..... 172
5.1.2.6 Uji Coba pada Daftar Afiliasi..... 173
5.1.2.7 Uji Coba pada Detail Afiliasi ..... 174
5.1.2.8 Uji Coba pada Diagram *Scatter*..... 175
5.2 Pengujian Pengguna .....176
5.2.1 Daftar Responden .....176

5.2.2 Skenario Pengujian Pengguna ..... 177
5.2.3 Hasil Pengujian Pengguna ..... 179
5.3 Evaluasi Pengujian ..... 184
5.3.1 Evaluasi Hasil Pengujian Fungsional ..... 184
5.3.2 Evaluasi Hasil Pengujian Pengguna ..... 185
BAB VI PENUTUP ..... 188
6.1 Kesimpulan ..... 188
6.2 Kritik dan Saran ..... 189
LAMPIRAN ..... 192
BIODATA PENULIS ..... 207

*[Halaman ini sengaja dikosongkan]*

# DAFTAR GAMBAR

Gambar 3.1 Diagram *Use Case* Aplikasi ..... 15
Gambar 3.2 Diagram Aktivitas Melihat Beranda ..... 16
Gambar 3.3 Diagram Aktivitas Melihat Grafik Rata Rata H Index Berdasarkan Afiliasi ..... 18
Gambar 3.4 Diagram Aktivitas Melihat Grafik Total Dokumen Berdasarkan Afiliasi ..... 20
Gambar 3.5 Diagram Aktivitas Melihat Grafik Total Dokumen Scopus Berdasarkan Afiliasi ..... 21
Gambar 3.6 Diagram Aktivitas Melihat Grafik Total Sitasi Berdasarkan Afiliasi ..... 23
Gambar 3.7 Diagram Aktivitas Melihat Daftar Peneliti ..... 24
Gambar 3.8 Mencari Peneliti ..... 26
Gambar 3.9 Melakukan Filter Peneliti ..... 28
Gambar 3.10 Filter Berdasarkan Keahlian ..... 30
Gambar 3.11 Filter Berdasarkan Subjek ..... 32
Gambar 3.12 Filter Berdasarkan Afiliasi ..... 34
Gambar 3.13 Mengurutkan Daftar Peneliti ..... 36
Gambar 3.14 Urutkan Berdasarkan Nama ..... 38
Gambar 3.15 Urutkan Berdasarkan H Index ..... 39
Gambar 3.16 Urutkan Berdasarkan Total Dokumen ..... 41
Gambar 3.17 Urutkan Berdasarkan Total Dokumen Scopus ..... 43
Gambar 3.18 Urutkan Berdasarkan Total Sitasi ..... 44
Gambar 3.19 Melihat Detail Peneliti ..... 46
Gambar 3.20 Diagram Aktivitas Melihat Daftar Afiliasi ..... 47
Gambar 3.21 Melihat Detail Afiliasi ..... 49
Gambar 3.22 Melihat Peneliti berdasarkan Detail Afiliasi ..... 51
Gambar 3.23 Diagram Aktivitas Melihat Daftar Subjek Penelitian ..... 52
Gambar 3.24 Melihat Detail Subjek Penelitian ..... 54
Gambar 3.25 Melihat Detail Afiliasi ..... 56
Gambar 3.26 Diagram Aktivitas Melihat Daftar Subjek Penelitian ..... 57

Gambar 3.27 Arsitektur Aplikasi Realitas Virtual untuk Visualisasi Data peneliti.....59
Gambar 3.28 Alur Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti.....59
Gambar 3.29 Model Basis Data Realitas Virtual untuk Visualisasi Data Peneliti .....63
Gambar 3.30 Rancangan Antarmuka Panel Konfigurasi.....64
Gambar 3.31 Rancangan Antarmuka Beranda .....65
Gambar 3.32 Rancangan Antarmuka Daftar Peneliti .....65
Gambar 3.33 Rancangan Antarmuka Detail Peneliti.....66
Gambar 3.34 Rancangan Antarmuka Daftar Afiliasi .....66
Gambar 3.35 Rancangan Antarmuka Detail Afiliasi.....67
Gambar 3.36 Rancangan Antarmuka Daftar Subjek Penelitian ..67
Gambar 3.37 Rancangan Antarmuka Daftar Subjek Penelitian ..68
Gambar 3.38 Rancangan Antarmuka Diagram Scatter.....69
Gambar 3.39 Rancangan Antarmuka Diagram Scatter.....69
Gambar 3.40 Rancangan Koridor.....70
Gambar 3.41 Rancangan Panel Kontrol .....71
Gambar 4.1 Implementasi Tabel Afiliasi .....74
Gambar 4.2 Contoh Data Afiliasi.....74
Gambar 4.3 Implementasi Tabel Departemen.....75
Gambar 4.4 Contoh Data Departemen .....75
Gambar 4.5 Implementasi Tabel Detail Subjek Area.....76
Gambar 4.6 Contoh Data Detail Subjek Area .....76
Gambar 4.7 Implementasi Tabel Fakultas.....77
Gambar 4.8 Contoh Data Fakultas .....77
Gambar 4.9 Implementasi Tabel Keahlian.....78
Gambar 4.10 Contoh Data Keahlian.....78
Gambar 4.11 Implementasi Tabel Peneliti .....79
Gambar 4.12 Contoh Data Detail Subjek Area .....80
Gambar 4.13 Implementasi Tabel Subjek Area.....80
Gambar 4.14 Contoh Data Subjek Area .....81
Gambar 4.15 Implementasi Tabel Subjek Topik Tier 1 .....81
Gambar 4.16 Contoh Data Subjek Subjek Topik Tier 1.....82
Gambar 4.17 Implementasi Tabel Subjek Topik Tier 1 .....82

Gambar 4.18 Contoh Data Subjek Topik Tier 2 ..... 83
Gambar 4.19 Panel untuk *Icon* Afiliasi ..... 84
Gambar 4.20 Cube untuk Beranda ..... 86
Gambar 4.21 Panel untuk Diagram Batang pada Beranda ..... 87
Gambar 4.22 Panel untuk Keterangan Diagram Batang ..... 87
Gambar 4.23 Text untuk Beranda ..... 88
Gambar 4.24 Panel untuk Icon Afiliasi ..... 88
Gambar 4.25 Tampilan Beranda ..... 109
Gambar 4.26 Tampilan Fitur Panel Daftar Peneliti ..... 109
Gambar 4.27 Tampilan Fitur Panel Peneliti ..... 110
Gambar 4.28 Tampilan Panel Daftar Peneliti ..... 120
Gambar 4.29 Tampilan Panel Detail Peneliti ..... 121
Gambar 4.30 Prefab Tombol DaftarAfiliasi ..... 125
Gambar 4.31 Tampilan Panel Daftar Subjek Penelitan ..... 126
Gambar 4.32 Tampilan Detail Subjek Penelitian ..... 129
Gambar 4.33 Prefab Tombol DaftarAfiliasi ..... 131
Gambar 4.34 Tampilan Panel Daftar Afiliasi ..... 132
Gambar 4.35 Tampilan Panel Detail Afiliasi ..... 135
Gambar 4.36 Tampilan Panel Diagram Scatter ..... 138
Gambar 4.37 Tampilan Diagram *Scatter* dalam Bentuk 3-dimensi ..... 139
Gambar C. 1 Pengujian Beranda ..... 199
Gambar C. 2 Pengujian Diagram Batang Rata-rata H-Index ..... 199
Gambar C. 3 Pengujian Diagram Batang Total Dokumen ..... 200
Gambar C. 4 Pengujian Diagram Batang Total Dokumen dalam Scopus ..... 200
Gambar C. 5 Pengujian Diagram Batang Total Sitasi ..... 200
Gambar C. 6 Pengujian Daftar Peneliti ..... 201
Gambar C. 7 Pengujian Filter Keahlian Daftar Peneliti ..... 201
Gambar C. 8 Pengujian Filter Afiliasi Daftar Peneliti ..... 201
Gambar C. 9 Pengujian Filter Subjek Penelitian Daftar Peneliti ..... 202
Gambar C. 10 Pengujian Urutkan Nama Daftar Peneliti ..... 202
Gambar C. 11 Pengujian Urutkan H-Index Daftar Peneliti ..... 202
Gambar C. 12 Pengujian Urutkan Dokumen Daftar Peneliti ..... 203

Gambar C. 13 Pengujian Urutkan Dokumen Scopus Daftar Peneliti.....203
Gambar C. 14 Pengujian Urutkan Total Sitasi Daftar Peneliti.203
Gambar C. 15 Pengujian Pencarian Daftar Peneliti.....204
Gambar C. 16 Pengujian Detail Peneliti.....204
Gambar C. 17 Pengujian Daftar Subjek Penelitian .....204
Gambar C. 18 Pengujian Detail Subjek Penelitian.....205
Gambar C. 19 Pengujian Daftar Afiliasi .....205
Gambar C. 20 Pengujian Detail Afiliasi.....205
Gambar C. 21 Pengujian Diagram *Scatter* .....206
Gambar C. 22 Pengujian Diagram *Scatter* 3-dimensi .....206

*[Halaman ini sengaja dikosongkan]*

# DAFTAR TABEL

Tabel 3.1 Kebutuhan Fungsional.....13
Tabel 3.2 Kebutuhan Non-Fungsional.....14
Tabel 3.3 UC-01: Melihat Beranda .....15
Tabel 3.4 UC-02: Melihat Grafik Rata-rata H Index Berdasarkan Afiliasi .....17
Tabel 3.5 UC-03: Melihat Grafik Total Dokumen berdasarkan Afiliasi .....18
Tabel 3.6 UC-04: Melihat Total Dokumen Scopus Berdasarkan Afiliasi .....20
Tabel 3.7 UC-05: Melihat Grafik Total Sitasi Berdasarkan Afiliasi .....22
Tabel 3.8 UC-06: Melihat Daftar Peneliti .....23
Tabel 3.9 UC-07 : Mencari Peneliti.....25
Tabel 3.10 UC-08: Melakukan Filter Peneliti .....27
Tabel 3.11 UC-09: Filter Berdasarkan Keahlian.....29
Tabel 3.12 UC-10: Filter Berdasarkan Subjek .....30
Tabel 3.13 UC-11: Filter Berdasarkan Afiliasi .....33
Tabel 3.14 UC-12: Mengurutkan Daftar Peneliti .....35
Tabel 3.15 UC-13: Urutkan Berdasarkan Nama.....37
Tabel 3.16 UC-14: Urutkan Berdasarkan H Index .....38
Tabel 3.17 UC-15: Urutkan Berdasarkan Total Dokumen.....40
Tabel 3.18 UC-16: Urutkan Berdasarkan Total Dokumen.....42
Tabel 3.19 UC-17: Urutkan Berdasarkan Total Sitasi.....43
Tabel 3.20 UC-18: Urutkan Berdasarkan Total Dokumen.....45
Tabel 3.21 UC-19: Melihat Daftar Afiliasi .....46
Tabel 3.22 UC-20: Melihat Detail Afiliasi.....48
Tabel 3.23 UC-21: Melihat Peneliti berdasarkan Afiliasi .....50
Tabel 3.24 UC-22: Melihat Daftar Subjek Penelitian .....51
Tabel 3.25 UC-23: Melihat Detail Subjek Penelitian.....53
Tabel 3.26 UC-24: Melihat Peneliti berdasarkan Subjek Penelitian .....55
Tabel 3.27 UC-25: Melihat Diagram *Scatter* .....56

Tabel 3.28 Daftar Tabel Basis Data Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti.....60
Tabel 4.1 Lingkungan Operasi Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti.....72
Tabel 5.1 Lingkungan Operasi Pengujian Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti.....164
Tabel 5.2 Skenario Uji Coba Fungsionalitas.....165
Tabel 5.3 Hasil Uji Coba pada Panel Navigasi .....166
Tabel 5.4 Hasil Uji Coba pada Beranda .....167
Tabel 5.5 Hasil Uji Coba pada Daftar Peneliti .....169
Tabel 5.6 Hasil Uji Coba pada Daftar Subjek Penelitian .....172
Tabel 5.7 Hasil Uji Coba pada Detail Subjek Penelitian.....173
Tabel 5.8 Hasil Uji Coba pada Daftar Afiliasi .....174
Tabel 5.9 Hasil Uji Coba pada Detail Afiliasi.....174
Tabel 5.10 Hasil Uji Coba pada Diagram *Scatter* .....175
Tabel 5.11 Daftar Responden.....176
Tabel 5.12 Penugasan Pengguna .....177
Tabel 5.13 Pertanyaan Mengemukakan Pendapat.....178
Tabel 5.14 Pertanyaan Iya dan Tidak.....178
Tabel 5.15 Pertanyaan Berskala .....178
Tabel 5.16 Parameter Nilai Pertanyaan Berskala .....179
Tabel 5.17 Waktu Respon Penguji Terhadap Setiap Penugasan .....179
Tabel 5.18 Pendapat Penguji terhadap Antarmuka Aplikasi Realitas Virtual.....180
Tabel 5.19 Penilaian Penguji Terhadap Aplikasi Realitas Virtual .....181
Tabel 5.20 Pendapat Penguji terhadap Inovasi Visualisasi Data menggunakan Aplikasi Realitas Virtual.....181
Tabel 5.21 Penemuan Kesulitan oleh Penguji terhadap Aplikasi Realitas Virtual.....181
Tabel 5.22 Penjelasan Penguji terkait Kesulitan terhadap Aplikasi Realitas Virtual.....182
Tabel 5.23 Saran Penguji terhadap Antarmuka Aplikasi Realitas Virtual.....182

Tabel 5.24 Rangkuman Hasil Pengujian Fungsional.....184
Tabel 5.25 Rangkuman Waktu Respon Penguji terhadap Aplikasi Realitas Virtual.....185
Tabel 5.26 Penilaian Penguji terhadap Aplikasi Realitas Virtual .....187
Tabel 5.27 Persentase Jumlah Penguji yang Mengalami Kesulitan .....187
Tabel 5.28 Persentase Jumlah Penguji yang Setuju Invoasi Visualisasi Data menggunakan Realitas Virtual.....187
Tabel A. 1 Contoh Data Peneliti Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti.....193
Tabel B. 2 Contoh Data Departemen Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti.....194
Tabel C. 3 Contoh Data Afiliasi Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti.....195
Tabel D. 4 Contoh Data Fakultas Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti.....196
Tabel E. 5 Contoh Data Subjek Area Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti.....197
Tabel F. 6 Contoh Data Detail Subjek Area Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti .....197
Tabel G. 7 Contoh Data Subjek Topik Tier 1 Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti .....198
Tabel H. 8 Contoh Data Subjek Topik Tier 2 Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti .....198
Tabel I. 9 Contoh Data Keahlian Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti.....198

*[Halaman ini sengaja dikosongkan]*

# DAFTAR KODE SUMBER

Kode Sumber 4.1 *Class Config* .....85
Kode Sumber 4.2 Pemanggilan *Class Config* .....85
Kode Sumber 4.3 Fungsi Dashboard.....90
Kode Sumber 4.4 Fungsi Dashboard Panel Event.....93
Kode Sumber 4.5 Fungsi Rerata H Index.....96
Kode Sumber 4.7 Fungsi Total Dokumen .....100
Kode Sumber 4.8 Fungsi Total Dokumen Scopus .....103
Kode Sumber 4.9 Fungsi Total Sitasi.....107
Kode Sumber 4.10 Fungsi Keterangan.....108
Kode Sumber 4.11 Fungsi *Get Data* Peneliti .....111
Kode Sumber 4.12 Fungsi *Get Data* Peneliti .....112
Kode Sumber 4.13 Fungsi *Get Urutkan* Data Peneliti .....113
Kode Sumber 4.14 Fungsi Data Peneliti .....117
Kode Sumber 4.15 Fungsi Data Filter.....118
Kode Sumber 4.16 Fungsi *Record Audio* dan *Stop Record Audio* .....120
Kode Sumber 4.17 Fungsi *Get Detail* Peneliti dan Detail Peneliti .....124
Kode Sumber 4.18 Fungsi *Get Data* Subjek Penelitian dan Data Subjek Penelitian.....128
Kode Sumber 4.19 Fungsi *Get* Detail Subjek Penelitian dan Detail Subjek.....131
Kode Sumber 4.20 Fungsi *Get Data* Afiliasi dan Data Afiliasi 134
Kode Sumber 4.21 Fungsi *Get* Detail Afiliasi dan Detail Afiliasi .....137
Kode Sumber 4.22 Fungsi *Get Scatter Plot* dan *Three D Scatter* .....142
Kode Sumber 4.22 Potongan Fungsi *Index* Aplikasi Server .....144
Kode Sumber 4.23 Contoh Penyusunan Data .....145
Kode Sumber 4.24 Fungsi Peneliti Aplikasi Server .....149
Kode Sumber 4.25 Fungsi Detail Peneliti Aplikasi Server .....151
Kode Sumber 4.26 Fungsi Afiliasi Aplikasi Server .....153
Kode Sumber 4.27 Fungsi Detail Afiliasi Aplikasi Server.....155

Kode Sumber 4.28 Fungsi Subjek Aplikasi Server ..... 157
Kode Sumber 4.29 Fungsi Detail Subjek Aplikasi Server ..... 159
Kode Sumber 4.30 Fungsi *Scatter* Aplikasi Server ..... 160
Kode Sumber 4.31 Fungsi *Pie Chart Texture* Aplikasi Server . 162
Kode Sumber 4.32 Fungsi *Voice Recognizer* Aplikasi Server .. 163

*[Halaman ini sengaja dikosongkan]*

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang

Teknologi imersif adalah teknologi yang memungkinkan seseorang merasa berada di dalam dunia virtual dan mampu berinteraksi secara langsung dengan dunia tersebut seakan-akan tidak ada pemisah antara dunia virtual dengan dunia nyata [1]. Teknologi imersif bisa digunakan untuk berbagai macam tujuan seperti simulator, game, hingga visualisasi data. Teknologi imersif dalam visualisasi data merupakan hal baru sebagai media eksplorasi data, dimana data yang telah diolah dapat diambil dan direpresentasikan ke dalam bentuk objek virtual yang yang divisualisasikan di dalam ruang 3-dimensi. Selain itu, pengguna juga dapat melakukan eksplorasi data sesuai dengan apa yang dibutuhkan [2].

Untuk mendukung hal tersebut, beberapa perangkat diperlukan untuk memvisualisasikan lingkungan virtual 3-dimensi. Salah satu perangkat tersebut adalah Realitas Virtual. Dengan perangkat Realitas Virtual, pengguna dapat merasakan pengalaman yang imersif dengan objek-objek 3-dimensi dalam dunia virtual.

Dalam Tugas Akhir ini penulis menggunakan data peneliti sebagai data yang divisualisasikan. Pengguna dapat melakukan eksplorasi data peneliti dengan kesan holografik yang menampilkan identitas peneliti, bidang keahlian, statistik penelitian serta pengguna juga dapat melihat grafik data klaster peneliti.

Tujuan dari Tugas Akhir ini adalah sebagai proyeksi masa depan bahwa teknologi imersif akan masif digunakan sebagai media presentasi atau sebagai media eksplorasi data layaknya hologram mengingat perkembangan teknologi untuk berinteraksi dengan ruang virtual semakin pesat serta perangkat keras yang didesain semakin minimalis.

## 1.2 Rumusan Masalah

Rumusan masalah dalam topik Tugas Akhir ini dapat dijabarkan sebagai berikut :

1. Bagaimana pengambilan dan penampilan data ke dalam ruang 3-dimensi?
2. Bagaimana pengaturan desain antarmuka aplikasi Realitas Virtual untuk visualisasi data peneliti?
3. Bagaimana cara pengguna berinteraksi dengan visualisasi data peneliti di aplikasi realitas virtual?

## 1.3 Batasan Masalah

Batasan masalah dalam topik Tugas Akhir ini dapat dijabarkan sebagai berikut :

1. Aplikasi ini dibangun berbasis Realitas Virtual Android.
2. Aplikasi ini mengacu pada data peneliti sebagai data yang divisualisasikan.
3. Aplikasi ini hanya membaca data visualisasi, tetapi tidak menuliskan hasil interaksi sebagai data baru.

## 1.4 Tujuan

Tujuan dari pembuatan Tugas Akhir ini adalah membangun aplikasi untuk visualisasi data peneliti secara imersif dan interaktif menggunakan Realitas Virtual. Aplikasi ini memungkinkan pengguna untuk melakukan eksplorasi terkait data peneliti. Selain itu, tujuan dari Tugas Akhir ini adalah sebagai proyeksi bahwa masa depan dimana teknologi imersif seperti Realitas Virtual, *Augmented Reality* ataupun *Mixed Reality* akan masif digunakan. Bahkan saat ini para peneliti telah mengembangkan teknologi holografi dimana teknologi ini memiliki banyak versi hologram dan masih perlu dilakukan pengembangan lebih lanjut.

## 1.5 Manfaat

1. Memberikan informasi terkait peneliti sesuai dengan bidangnya.
2. Memberikan pengalaman eksplorasi data secara holografis.

## 1.6 Metodologi

Tahapan-tahapan yang dilakukan dalam pengerjaan Tugas Akhir adalah sebagai berikut :

### A. Penyusunan Proposal Tugas Akhir

Langkah awal dalam penyusunan Tugas Akhir adalah menyusun proposal Tugas Akhir. Proposal Tugas Akhir terdiri dari latar belakang, rumusan masalah, batasan masalah, tujuan pembuatan Tugas Akhir, serta metodologi yang berisi tentang proses dan tahapan penyusunan Tugas Akhir. Selain itu, terdapat juga tinjauan pustaka yang digunakan untuk referensi pendukung dalam pengerjaan Tugas Akhir, ringkasan Tugas Akhir yang menjelaskan secara singkat tentang cara kerja aplikasi Realitas Virtual ini, dan juga timeline pengerjaan Tugas Akhir.

### B. Studi Literatur

Dalam studi literatur, berisikan tentang referensi yang dapat dipelajari untuk melakukan implementasi, diantaranya adalah sebagai berikut:

a. Realitas Virtual sebagai perangkat visualisasi.
b. Unity3D untuk merancang aplikasi Realitas Virtual.
c. Bahasa Pemrograman C# untuk mengimplementasikan logika dalam aplikasi Realitas Virtual.
d. Bahasa Pemrograman PHP sebagai bahasa yang menangani permintaan data.
e. Pemodelan 3-dimensi untuk merancang lingkungan 3-dimensi.
f. *phpMyAdmin* sebagai basis data.

g. *Leap Motion Controller* sebagai pengontrol untuk berinteraksi.
h. *OpenVR* SDK sebagai modul pembantu dalam pengembangan aplikasi Realitas Virtual.

Lumen *micro-framework* sebagai API *end point* untuk menangani *request* dari aplikasi Realitas Virtual.

### C. Analisis dan Desain Sistem

Dalam tahap analisis dan desain sistem ini dilakukan analisis terhadap kebutuhan fungsional dan non-fungsional dimana nantinya akan menghasilkan use-case yang menjelaskan tentang apa saja yang dapat dilakukan oleh user dan proses apa saja yang ada di dalam aplikasi Realitas Virtual ini. Selain itu, dalam tahap ini juga akan dilakukan desain antarmuka serta desain interaksi.

### D. Implementasi Sistem

Implementasi perangkat lunak merupakan tahap untuk membangun aplikasi dengan rancangan yang telah dibuat sebelumnya. Sehingga dalam tahap implementasi ini menghasilkan aplikasi yang sesuai dengan apa yang dirancang sebelumnya.

Dalam menyelesaikan Tugas Akhir ini, aplikasi akan diimplementasikan menggunakan IDE Visual Studio dan *Unity3D game engine* menggunakan bahasa pemrograman C# dengan arsitektur *client* dan server untuk melayani permintaan pengguna dalam visualisasi data peneliti.

### E. Pengujian dan Evaluasi

Tahap pengujian merupakan tahap untuk menguji apakah aplikasi sudah sesuai dengan apa yang diharapkan. Dalam tahap ini dilakukan pengujian fungsionalitas perangkat lunak apakah aplikasi Realitas Virtual ini terdapat error atau bug.

Selain itu, untuk pengujian terhadap aplikasi Realitas Virtual ini juga menggunakan metode *Time Completion.* Metode *Time Completion* adalah metode pengujian yang dilakukan dengan melibatkan pengguna dalam pengujian aplikasi. Pengguna akan

diberi penugasan untuk menyelesaikan tugas tersebut dan menghitung waktu yang dibutuhkan oleh pengguna dalam menyelesaikan tugas yang telah diberikan.

### F. Penyusunan Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam Tugas Akhir ini. Pada tahap ini juga disertakan hasil dari implementasi metode dan algoritma yang telah dibuat. Sistematika penulisan buku Tugas Akhir ini secara garis besar antara lain:

1. Pendahuluan.
    a. Latar Belakang.
    b. Rumusan Masalah.
    c. Batasan Tugas Akhir.
    d. Tujuan.
    e. Metodologi.
    f. Sistematika Penulisan.
2. Tinjauan Pustaka.
3. Analisis dan Desain.
4. Implementasi.
5. Pengujian dan Evaluasi.
6. Kesimpulan dan Saran.
7. Daftar Pustaka.

## 1.7 Sistematika Penulisan

Buku tugas akhir ini terdiri dari beberapa bab yang dijelaskan sebagai berikut:

**BAB I PENDAHULUAN**

Bab ini berisi tentang latar belakang masalah, rumusan dan batasan permasalahan, tujuan dan manfaat pembuatan tugas akhir, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

**BAB II TINJAUAN PUSTAKA**

Bab ini membahas dasar pembuatan dan beberapa teori penunjang yang berhubungan dengan pokok pembahasan yang mendasari pembuatan tugas akhir ini.

**BAB III ANALISIS DAN PERANCANGAN**

Bab ini membahas analisis dari sistem yang dibuat meliputi analisis permasalahan, deskripsi umum perangkat lunak, spesifikasi kebutuhan, dan identifikasi pengguna. Kemudian membahas rancangan dari sistem yang dibuat meliputi rancangan skenario kasus penggunaan, data, dan antarmuka.

**BAB IV IMPLEMENTASI SISTEM**

Bab ini membahas implementasi dari rancangan sistem yang dilakukan pada tahap perancangan. Penjelasan implementasi meliputi implementasi pembuatan objek, implementasi pembuatan aplikasi, dan implementasi pembuatan simulasi.

**BAB V PENGUJIAN DAN EVALUASI**

Bab ini membahas pengujian dari aplikasi yang dibuat dengan melihat keluaran yang dihasilkan oleh aplikasi dan evaluasi untuk mengetahui kemampuan aplikasi.

**BAB VI PENUTUP**

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan serta saran untuk pengembangan aplikasi selanjutnya.

*[Halaman ini sengaja dikosongkan]*

# BAB II
# TINJAUAN PUSTAKA

Dalam pengerjaan Tugas Akhir ini, tinjauan pustaka yang akan digunakan adalah sebagai berikut :

## 2.1 Realitas Virtual

Realitas Virtual merupakan teknologi yang memungkinkan pengguna merasa di dalam dunia virtual dengan lingkungan yang divisualisasikan oleh komputer. Kelebihan dari Realitas Virtual adalah pengguna mendapatkan sensasi dunia nyata didalam dunia virtual Secara umum realitas virtual memberikan pengalaman secara visual kepada pengguna. Tetapi beberapa perangkat juga mendukung pengalaman secara kinestetik di dalam Realitas Virtual [3].

Realitas Virtual sangat berguna untuk membantu mensimulasikan hal-hal yang sulit dihadirkan secara nyata atau yang memiliki resiko tinggi. Salah satu contohnya adalah pemanfaatan Realitas Virtual dalam bidang penerbangan. Dalam hal ini, Pilot dapat berlatih untuk menerbangkan pesawat yang sudah didesain semirip mungkin dengan yang asli secara virtual sebelum menggunakan pesawat sebenarnya.

## 2.2 Unity3D

Unity3D merupakan *game engine* lintas *platform* yang digunakan untuk merancang game yang dapat menghasilkan game untuk berbagai macam *platform* seperti Mac, Windows, Wii, iPhone, iPad, Android, dll. Unity3D didesain untuk pengembangan game dengan grafis tingkat tinggi seperti OpenGL dan DirectX.

Unity3D bukan merupakan software modeling, sehingga diperlukan software lain untuk melakukan modeling seperti Blender, Sketchup, dan masih banyak lagi. Unity3D saat ini dibekali beberapa fitur, diantaranya adalah particle system untuk

membuat partikel, animation untuk membuat animasi, dan masih banyak lagi.

Unity3D didukung dengan bahasa pemrograman C# untuk menjalankan logika yang telah diprogram oleh programmer sehingga mekanisme simulasi atau game dapat berjalan sesuai apa yang telah diprogram [4].

## 2.3 Pemodelan 3-dimensi

Pemodelan 3-dimensi adalah proses pembuatan objek secara matematis dalam 3-dimensi yang menggunakan perangkat lunak khusus untuk pemodelan 3-dimensi. Proses ini digunakan untuk menciptakan sebuah objek virtual yang menggambarkan objek sebenarnya secara yang dinamakan model 3-dimensi.

Model 3-dimensi dibuat dengan konsep geometri dengan menggunakan titik-titik dimana titik-titik tersebut terhubung satu sama lain sehingga menciptakan garis, bidang datar dalam, dan permukaan yang melengkung yang menghasilkan bentuk 3-dimensi menyerupai objek yang dijadikan model [5].

## 2.4 Bahasa Pemrograman C#

Bahasa pemrograman C# adalah bahasa pemrograman hasil pengembangan Microsoft dengan merekrut Anders Helsberg yang digunakan untuk tujuan umum. Artinya, bahasa pemrograman ini dapat digunakan dalam berbagai variasi fungsi misalnya untuk pemrograman server pada website, membangun aplikasi multiplatform, pemrograman game dan sebagainya.

Bahasa Pemrograman C# dibangun berbasis C++ yang telah dipengaruhi oleh aspek maupun fitur dari bahasa pemrograman Java, Delphi, Visual Basic, dan lain-lain dengan beberapa penyederhanaan. Bahasa pemrograman C# merupakan bahasa pemrograman berorientasi objek dimana C# mengusung konsep objek seperti inheritance, class, polymorphism dan encapsulation. Bahasa pemrograman C# sangat bergantung dengan framework yang disebut .NET. Framework inilah yang

akan digunakan untuk melakukan kompilasi dan menjalankan kode C# [6].

## 2.5 Bahasa Pemrograman Python

Bahasa pemrograman Python adalah bahasa pemrograman interpretatif yang mudah untuk dipahami secara sintaks karena bahasa pemrograman ini menekankan pada keterbacaan kode serta telah menyediakan beragam *library*.

Bahasa pemrograman Python dirancang oleh Guido van Rossum pada tahun 1991 dan saat ini dikembangkan oleh Python Software Foundation. Bahasa ini telah kompatibel dibeberapa sistem operasi dan bahkan telah tersedia diberbagai distro Linux secara default [7].

## 2.6 Google VR

Google VR adalah SDK (Software Development Kit) dan API (Application Programming Interface) adalah modul-modul pembantu yang disediakan dan dikembangkan oleh Google untuk mendungkung Realitas Virtual pada perangkat bergerak.

Meskipun Google VR adalah SDK yang dikembangkan oleh Google dan dikenal dengan Android-nya, Google VR dapat digunakan pada perangkat bergerak lain seperti iOS.

## 2.7 VR Box

VR Box adalah alat yang digunakan untuk mengubah perangkat bergerak menjadi perangkat Realitas Virtual. VR Box kompatibel dengan seluruh perangkat bergerak sehingga sangat fleksibel digunakan pada perangkat bergerak apapun.

VR Box sebenarnya adalah replika dari Google Cardboard namun memiliki ketahanan yang lebih baik dari Google Cardboard karena terbuat dari bahan plastik.

Dengan menggunakan VR Box, pengguna mampu merasakan sensasi menggunakan Realitas Virtual dengan biaya yang murah dibandingkan dengan perangkat Realitas Virtual yang lain.

## 2.8 phpMyAdmin

phpMyAdmin adalah perangkat lunak yang ditulis menggunakan bahasa pemrograman PHP yang digunakan untuk menangani administrasi MySQL melalui World Wide Web [8]. phpMyAdmin dikembangkan mulai tahun 1998 oleh Tobias Ratschiller seorang yang merupakan konsultan IT.

Dengan adanya phpMyAdmin, seseorang dapat membuat basis data, membuat tabel, mengisi data, dan mengelola basis data dengan mudah tanpa harus menghafal perintah untuk melakukan pengelolaan basis data.

## 2.9 XAMPP

XAMPP adalah sebuah perangkat lunak untuk *localhost* yang merupakan kompilasi beberapa program yaitu Apache, MySQL, HTTP Server dan penerjemah bahasa yang ditulis menggunakan bahasa PHP dan Perl.

Nama XAMPP sendiri diambil dari singkatan X yang artinya *cross platform*, Apache, MySQL atau MariaDB, PHP dan Perl. XAMPP merupakan web server yang mudah digunakan dan tersedia dalam GNU *General Public Lisence* dan bebas [9].

## 2.10 Flask Micro Web Framework

Flask adalah *micro web framework* yang ditulis menggunakan bahasa permrograman Python dan kompatibel dengan *library* bahasa Python. Dikarenakan Flask adalah micro web framework, Flask memiliki ukuran yang sangat kecil dan ringan. Flask *micro web framework* memiliki mengandalkan *Werkzeug* dan juga Jinja2 sebagai dependensinya [10].

Flask biasanya digunakan untuk membuat aplikasi dalam skala yang lebih kecil dan fleksibel dalam skalabilitas aplikasi ketika ingin membuat aplikasi yang lebih besar. Flask dapat digunakan untuk membuat API end point dan memiliki fitur ORM atau *Object Relational Mapper* untuk memudahkan dalam

pengambilan data pada basis data dengan menggunakan *library* SQL Alchemy.

# BAB III
# ANALISIS DAN PERANCANGAN

Bab ini berisi tentang analisis dan perancangan pada aplikasi Realitas Virtual untuk visualisasi data peneliti. Pada bab ini akan dijelaskan mulai dari analisis sistem, hingga desain antarmuka dan interaksi yang akan diterapkan pada aplikasi Realitas Virtual.

## 3.1 Analisis Sistem

Pada tahap analisis sistem akan menjelaskan tentang deskripsi umum sistem, lingkungan operasi, deskripsi kebutuhan sistem dan deskripsi kebutuhan fungsional pada aplikasi Realitas Virtual untuk visualisasi data peneliti.

### 3.1.1 Spesifikasi Kebutuhan Fungsional

Aplikasi Realitas Virtual untuk visualisasi data peneliti memiliki spesifikasi kebutuhan fungsional yang dapat dilihat pada Tabel 3.1.

*Tabel 3.1 Kebutuhan Fungsional*

| Kode | Deskripsi |
|---|---|
| F-01 | Sistem mampu menampilkan grafik statistik pada beranda. |
| F-02 | Sistem mampu menampilkan daftar peneliti. |
| F-03 | Sistem memungkinkan pengguna untuk mencari peneliti. |
| F-04 | Sistem memungkinkan pengguna untuk mengurutkan data peneliti, |
| F-05 | Sistem mampu menampilkan detail peneliti. |
| F-06 | Sistem mampu menampilkan daftar subjek. |
| F-07 | Sistem mampu menampilkan detail subjek. |
| F-08 | Sistem mampu menampilkan daftar afiliasi. |

| Kode | Deskripsi |
|---|---|
| F-09 | Sistem mampu menampilkan detail afiliasi. |
| F-10 | Sistem mampu menampilkan diagram *scatter*. |
| F-11 | Sistem server mampu mengirimkan data untuk aplikasi *client*. |

### 3.1.2 Spesifikasi Kebutuhan Non-Fungsional

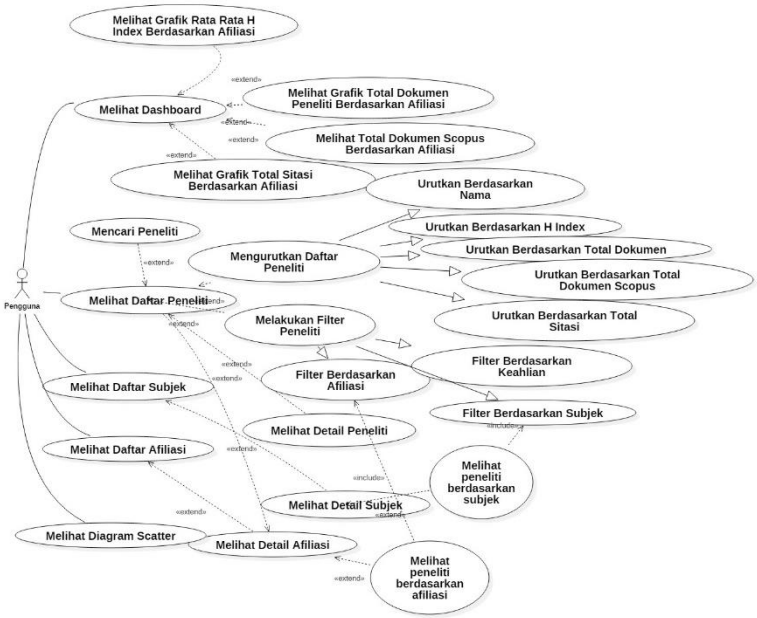
Aplikasi Realitas Virtual untuk visualisasi data peneliti memiliki spesifikasi kebutuhan non-fungsional yang dapat dilihat pada Tabel 3.2.

*Tabel 3.2 Kebutuhan Non-Fungsional*

| Kode | Deskripsi |
|---|---|
| NF-01 | Sistem berjalan menggunakan basis aplikasi Realitas Virtual. |
| NF-02 | Sistem berjalan menggunakan arsitektur *client* dan server. |
| NF-03 | Sistem server berjalan menggunakan bahasa Python. |
| NF-04 | Sistem *client* berjalan menggunakan bahasa C#. |
| NF-04 | Sistem server menggunakan DBMS MySQL. |

### 3.1.3 Spesifikasi *Use Case*

Spesifikasi *use case* berisi tentang skenario bagaimana cara pengguna berinteraksi dengan sistem dari setiap *use case* pada aplikasi Realitas Virtual untuk visualisasi data peneliti. Untuk memudahkan mendeskripsikan apa saja yang dapat dilakukan pengguna terhadap sistem, maka diperlukan diagram *use case.* Diagram ini digunakan untuk memodelkan interaksi antara aktor dengan sistem.Diagram *use case* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Gambar 3.2.




*Gambar 3.1 Diagram Use Case Aplikasi*

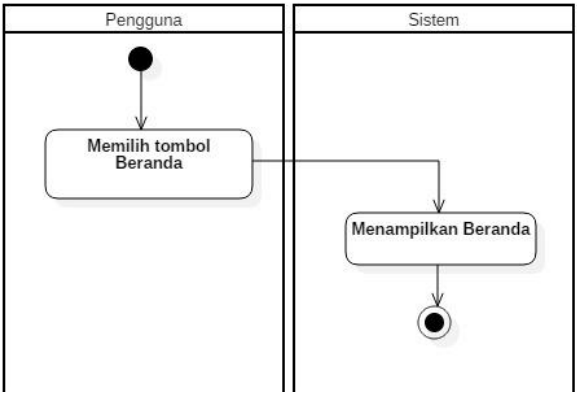
### 3.1.3.1 Melihat Beranda

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki beranda yang merupakan halaman depan. Beranda pada aplikasi Realitas Virtual untuk visualisasi data peneliti ini dapat melihat beberapa grafik batang. Skenario untuk *use case Melihat Beranda* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.3 serta diagram aktivitas untuk use case *Melihat Beranda* dapat dilihat pada Gambar 3.2.

*Tabel 3.3 UC-01: Melihat Beranda*

| Nama Use Case | Melihat Beranda |
|---|---|
| **Kode Use Case** | UC-01 |
| **Aktor** | Pengguna |

| | | | |
|---|---|---|---|
| **Deskripsi** | | Pengguna melihat beranda pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | - | |
| **Trigger** | | Pengguna memilih “Beranda” | |
| **Kondisi Awal** | | Pengguna belum memasuki Beranda | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | Pengguna memilih tombol Beranda | | |
| | | 2 | Sistem menampilkan Beranda |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Pengguna di teruskan ke Beranda | |
| **Eksepsi : -** | | | |




*Gambar 3.2 Diagram Aktivitas Melihat Beranda*

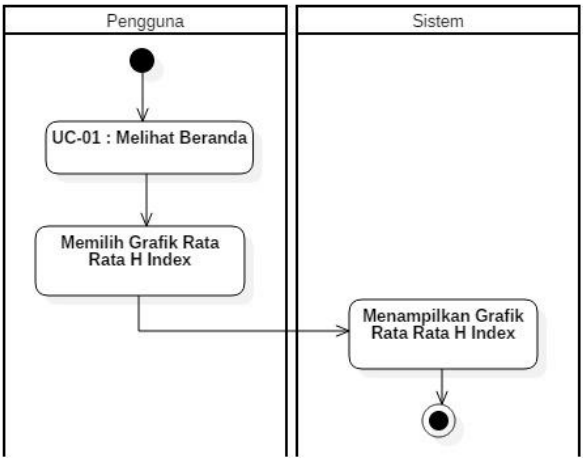
### 3.1.3.2 Melihat Grafik Rata Rata H-Index berdasarkan Afiliasi

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki grafik rata-rata h-index yang terdapat pada beranda.

Grafik rata-rata h-index ini merepresentasikan rata-rata h-index peneliti dan dikelompokkan berdasarkan afiliasi. *Use case Melihat Grafik Rata-rata H-Index* merupakan langkah lanjutan dari *use case* sebelumnya yaitu *Melihat Beranda*. Skenario untuk *use case Melihat Grafik Rata-rata H-Index* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.4 dan diagram aktivitas untuk use case *Melihat Grafik Rata-rata H-Index* dapat dilihat pada Gambar 3.3.

*Tabel 3.4 UC-02: Melihat Grafik Rata-rata H Index Berdasarkan Afiliasi*

| **Nama Use Case** | | **Melihat Grafik Rata Rata H Index Berdasarkan Afiliasi** | |
|---|---|---|---|
| **Kode Use Case** | | UC-02 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna melihat *Melihat Grafik Rata Rata H Index Berdasarkan Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-01 | |
| **Trigger** | | Pengguna memilih "Grafik Rata Rata H Index" | |
| **Kondisi Awal** | | Pengguna belum memasuki Grafik Rata Rata H Index | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-01: Melihat Beranda | | |
| 2 | Pengguna memilih Grafik Rata Rata H Index | | |
| | | 2 | Sistem menampilkan Grafik Rata Rata H Index |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Pengguna di teruskan ke Grafik Rata Rata H Index | |
| **Eksepsi : -** | | | |




*Gambar 3.3 Diagram Aktivitas Melihat Grafik Rata Rata H Index Berdasarkan Afiliasi*

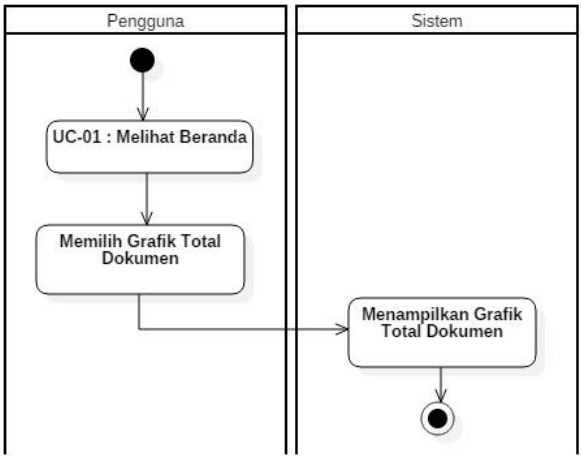
#### 3.1.3.3 Melihat Grafik Batang Total Dokumen Peneliti berdasarkan Afiliasi

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki grafik total dokumen pada beranda. Grafik total dokumen ini merepresentasikan total dokumen penelitian yang dikelompokkan berdasarkan afiliasi. *Use case Melihat Grafik Total Dokumen berdasarkan Afiliasi* merupakan langkah lanjutan dari *use case* sebelumnya yaitu *Melihat Beranda*. Skenario untuk *use case Melihat Grafik Total Dokumen berdasarkan Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.5 dan diagram aktivitas untuk use case *Melihat Grafik Total Dokumen berdasarkan Afiliasi* dilihat pada Gambar 3.4.

*Tabel 3.5 UC-03: Melihat Grafik Total Dokumen berdasarkan Afiliasi*

| Nama Use Case | Melihat Grafik Total Dokumen Peneliti Berdasarkan Afiliasi |
|---|---|
| **Kode Use Case** | UC-03 |
| **Aktor** | Pengguna |

| | | | |
|---|---|---|---|
| **Deskripsi** | | Pengguna melihat *Melihat Grafik Total Dokumen Peneliti Berdasarkan Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-01 | |
| **Trigger** | | Pengguna memilih "Grafik Total Dokumen Peneliti" | |
| **Kondisi Awal** | | Pengguna belum memasuki Grafik Total Dokumen Peneliti | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-01: Melihat Beranda | | |
| 2 | Pengguna memilih Grafik Total Dokumen Peneliti | | |
| | | 2 | Sistem menampilkan Grafik Total Dokumen Peneliti |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Pengguna di teruskan ke Grafik Total Dokumen Peneliti | |
| **Eksepsi : -** | | | |



*Gambar 3.4 Diagram Aktivitas Melihat Grafik Total Dokumen Berdasarkan Afiliasi*

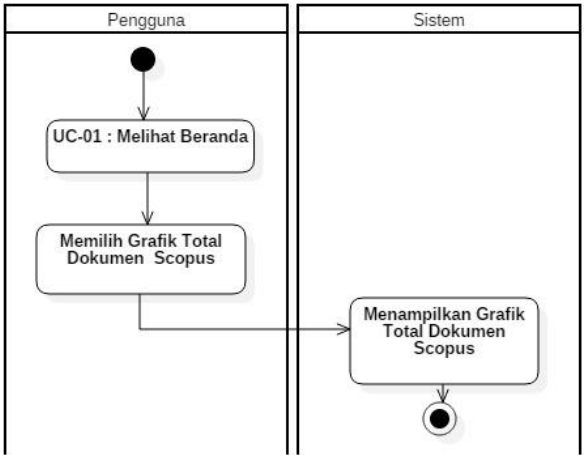
### 3.1.3.4 Melihat Grafik Total Dokumen Scopus berdasarkan Afiliasi

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki grafik total dokumen yang tedapat pada beranda. Grafik total dokumen ini merepresentasikan total dokumen penelitian yang dipublikasikan dalam scopus dan dikelompokkan berdasarkan afiliasi. *Use case Melihat Grafik Total Dokumen Scopus berdasarkan Afiliasi* merupakan langkah lanjutan dari *use case* sebelumnya yaitu *Melihat Beranda*. Skenario untuk *use case Melihat Grafik Total Dokumen Scopus berdasarkan Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.6 dan diagram aktivitas untuk use case *Melihat Grafik Total Dokumen berdasarkan Afiliasi* dapat dilihat pada Gambar 3.5.

*Tabel 3.6 UC-04: Melihat Total Dokumen Scopus Berdasarkan Afiliasi*

| **Nama Use Case** | | **Melihat Total Dokumen Scopus Berdasarkan Afiliasi** | |
|---|---|---|---|
| **Kode Use Case** | | UC-04 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna melihat *Melihat Total Dokumen Scopus Berdasarkan Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-01 | |
| **Trigger** | | Pengguna memilih "Grafik Total Dokumen Scopus" | |
| **Kondisi Awal** | | Pengguna belum memasuki Grafik Total Dokumen Scopus | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-01: Melihat Beranda | | |

| | | | |
|---|---|---|---|
| 2 | Pengguna memilih Grafik Total Dokumen Scopus | | |
| | | 2 | Sistem menampilkan Grafik Total Dokumen Scopus |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Pengguna di teruskan ke Grafik Total Dokumen Scopus | |
| **Eksepsi : -** | | | |




*Gambar 3.5 Diagram Aktivitas Melihat Grafik Total Dokumen Scopus Berdasarkan Afiliasi*

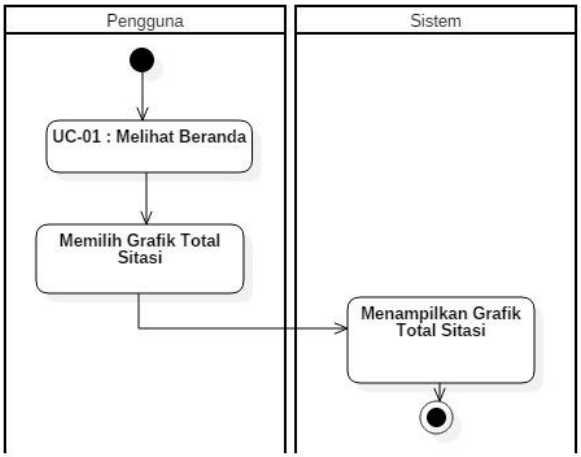
### 3.1.3.5 Melihat Grafik Total Sitasi berdasarkan Afiliasi

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki grafik total dokumen berdasarkan afiliasi yang terdapat pada beranda. Grafik total sitasi ini merepresentasikan total sitasi terhadap dokumen penelitian yang dipublikasikan dan dikelompokkan berdasarkan afiliasi. *Use case Melihat Grafik Total Dokumen Scopus berdasarkan Afiliasi* merupakan langkah lanjutan dari *use case* sebelumnya yaitu *Melihat Beranda*. Skenario

untuk *use case Melihat Grafik Total Dokumen Scopus berdasarkan Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.7 dan diagram aktivitas untuk use case *Melihat Grafik Total Dokumen berdasarkan Afiliasi* dapat dilihat pada Gambar 3.6.

*Tabel 3.7 UC-05: Melihat Grafik Total Sitasi Berdasarkan Afiliasi*

| **Nama Use Case** | | **Melihat Grafik Total Sitasi Berdasarkan Afiliasi** | |
|---|---|---|---|
| **Kode Use Case** | | UC-05 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna melihat *Melihat Total Sitasi Berdasarkan Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-01 | |
| **Trigger** | | Pengguna memilih "Grafik Total Sitasi" | |
| **Kondisi Awal** | | Pengguna belum memasuki Grafik Total Sitasi | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-01: Melihat Beranda | | |
| 2 | Pengguna memilih Grafik Total Sitasi | | |
| | | 2 | Sistem menampilkan Grafik Total Sitasi |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Pengguna di teruskan ke Grafik Total Sitasi | |
| **Eksepsi : -** | | | |




*Gambar 3.6 Diagram Aktivitas Melihat Grafik Total Sitasi Berdasarkan Afiliasi*

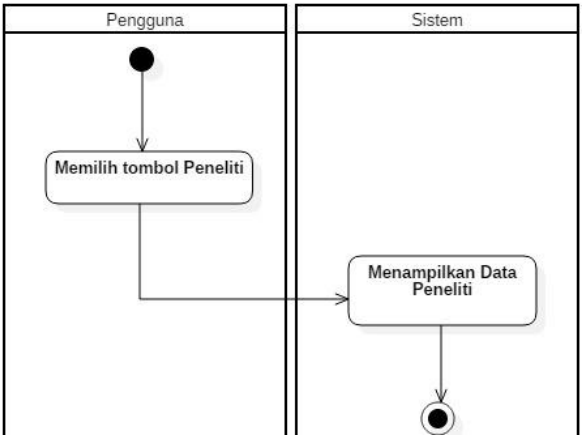
### 3.1.3.6 Melihat Daftar Peneliti

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki daftar peneliti. Daftar ini berisi tentang peneliti-peneliti yang terdapat pada aplikasi Realitias Virtual ini. Skenario untuk *use case Melihat Daftar Peneliti* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.8 dan diagram aktivitas untuk use case *Melihat Daftar Peneliti* dapat dilihat pada Gambar 3.7.

*Tabel 3.8 UC-06: Melihat Daftar Peneliti*

| Nama Use Case | Melihat Daftar Peneliti |
|---|---|
| **Kode Use Case** | UC-06 |
| **Aktor** | Pengguna |
| **Deskripsi** | Pengguna melihat *Melihat Daftar Peneliti* pada aplikasi Realitas Virtual untuk visualisasi data peneliti |
| **Relasi** | - |

| | | | |
|---|---|---|---|
| **Trigger** | | Pengguna memilih tombol "Peneliti" | |
| **Kondisi Awal** | | Pengguna belum memasuki Daftar Peneliti | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | Pengguna memilih tombol Peneliti | | |
| | | 2 | Sistem menampilkan Grafik Total Sitasi |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Pengguna di teruskan ke Daftar Peneliti | |
| **Eksepsi : -** | | | |




*Gambar 3.7 Diagram Aktivitas Melihat Daftar Peneliti*

### 3.1.3.7 Mencari Peneliti

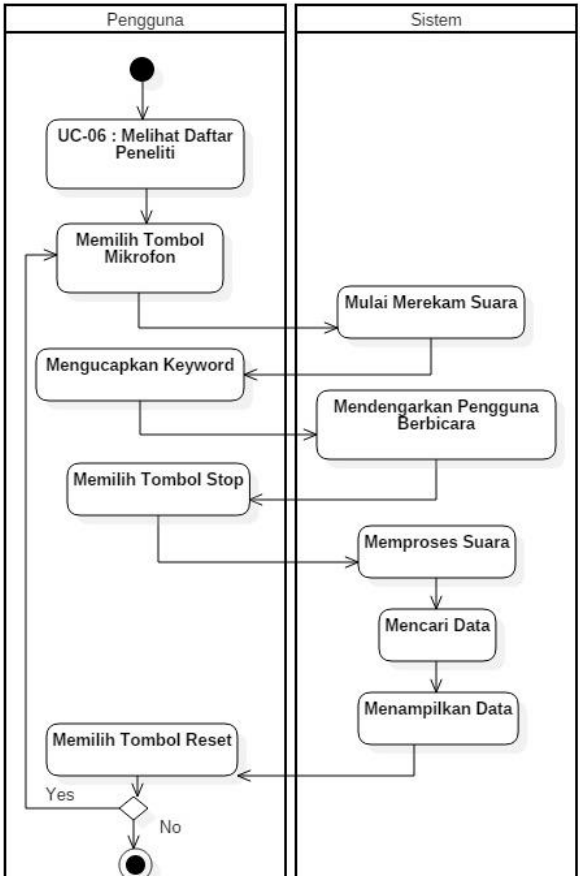
Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur pencarian peneliti. fitur ini digunakan untuk mencari peneliti berdasarkan nama. *Use case Mencari Peneliti* merupakan langkah lanjutan dari *use case* sebelumnya yaitu *Melihat Daftar Peneliti*. Skenario untuk *use case Melihat Daftar Peneliti* pada

aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.9 dan diagram aktivitas untuk use case *Mencari Peneliti* dapat dilihat pada Gambar 3.8.

*Tabel 3.9 UC-07 : Mencari Peneliti*

| **Nama Use Case** | | **Mencari Peneliti** | |
|---|---|---|---|
| **Kode Use Case** | | UC-07 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Mencari Peneliti* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih tombol "Mikrofon" | |
| **Kondisi Awal** | | Pengguna belum memilih tombol Mikrofon | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna memilih tombol Mikrofon | | |
| | | 3 | Sistem mulai merekam suara |
| 4 | Pengguna mengucapkan *keyword* | | |
| | | 5 | Sistem mendengarkan pengguna berbicara |
| 7 | Pengguna memilih tombol stop | | |
| | | 8 | Sistem memproses suara |
| | | 9 | Sistem mencari data peneliti |
| | | 10 | Sistem menampilkan data peneliti |
| **Alur alternatif : -** | | | |
| 11 | Pengguna memilih tombol Reset | | |

| | |
|---|---|
| **Kondisi Akhir** | Sistem menampilkan data peneliti berdasarkan kata kunci |
| **Eksepsi :** Sistem tidak dapat mendeteksi suara | |




*Gambar 3.8 Mencari Peneliti*

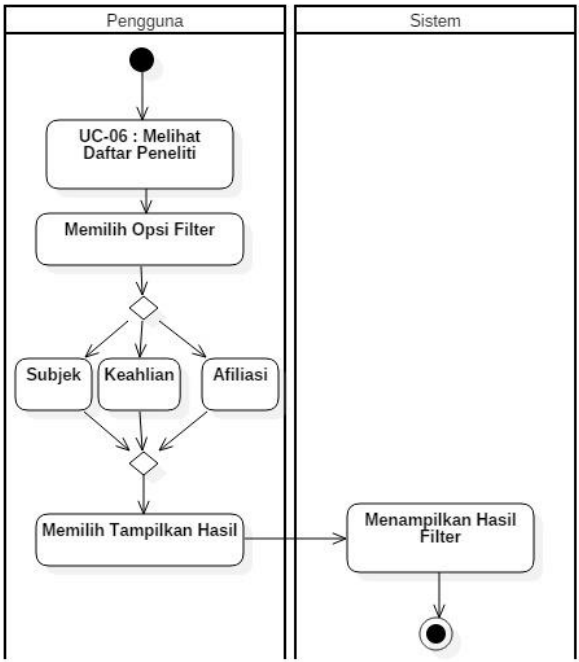
### 3.1.3.8 Melakukan Filter Peneliti

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur filter peneliti. fitur ini digunakan memfilter peneliti berdasarkan filter keahlian, subjek penelitian dan afiliasi. *Use case Melakukan Filter Peneliti* merupakan langkah lanjutan dari *use case* sebelumnya yaitu *Melihat Daftar Peneliti*. Skenario untuk *use case Melakukan Filter Peneliti* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.10 dan diagram aktivitas untuk use case *Melakukan Filter Peneliti* dapat dilihat pada Gambar 3.9.

*Tabel 3.10 UC-08: Melakukan Filter Peneliti*

| **Nama Use Case** | | **Melakukan Filter Peneliti** | |
|---|---|---|---|
| **Kode Use Case** | | UC-08 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Melakukan Filter Peneliti* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih tombol “Tampilkan Hasil” | |
| **Kondisi Awal** | | Pengguna belum memilih opsi filter | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna opsi filter | | |
| 3 | Pengguna memilih tombol Tampilkan Hasil | | |
| | | 4 | Sistem menampilkan hasil filter |
| **Alur alternatif : -** | | | |
| 2.1 | Pengguna memilih opsi filter Subjek | | |

| | | | |
|---|---|---|---|
| 2.2 | Pengguna memilih opsi filter Afiliasi | | |
| 2.3 | Pengguna memilih opsi filter Keahlian | | |
| **Kondisi Akhir** | | Sistem menampilkan hasil filter | |
| **Eksepsi : -** | | | |




*Gambar 3.9 Melakukan Filter Peneliti*

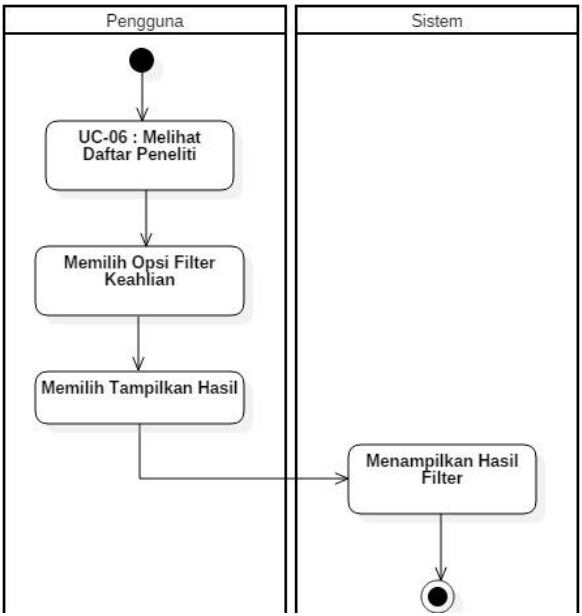
### 3.1.3.9 Filter berdasarkan Keahlian

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur filter peneliti berdasarkan keahlian. *Use case Filter Berdasarkan Keahlian* merupakan generalisasi dari *use case* sebelumnya yaitu *Melakukan Filter Peneliti*. Skenario untuk *use case Filter berdasarkan Keahlian* pada aplikasi Realitas Virtual

untuk visualisasi data peneliti dapat dilihat pada Tabel 3.11 dan diagram aktivitas untuk use case *Filter berdasarkan Keahlian* dapat dilihat pada Gambar 3.10.

*Tabel 3.11 UC-09: Filter Berdasarkan Keahlian*

| **Nama Use Case** | | **Filter Berdasarkan Keahlian** | |
|---|---|---|---|
| **Kode Use Case** | | UC-09 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna melakukan *Filter Berdasarkan Keahlian* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih tombol “Tampilkan Hasil” | |
| **Kondisi Awal** | | Pengguna belum memilih opsi filter | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna opsi filter Keahlian | | |
| 3 | Pengguna memilih tombol Tampilkan Hasil | | |
| | | 4 | Sistem menampilkan hasil filter |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan hasil filter | |
| **Eksepsi : -** | | | |




*Gambar 3.10 Filter Berdasarkan Keahlian*

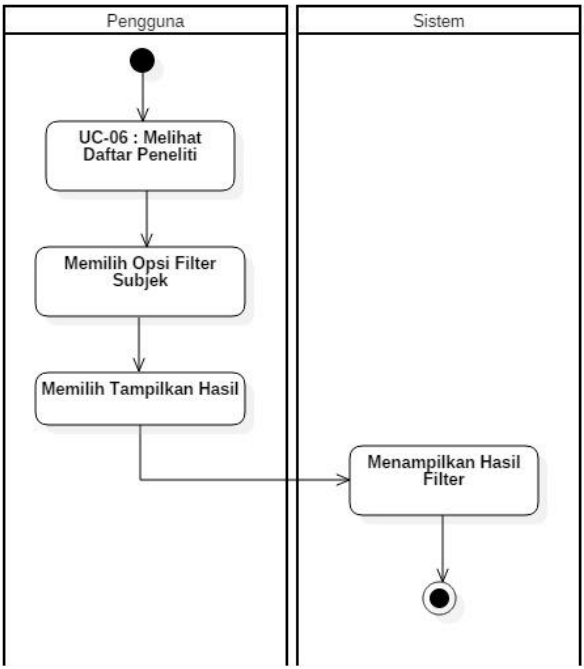
### 3.1.3.10 Filter berdasarkan Subjek

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur filter peneliti berdasarkan subjek. *Use case Filter Berdasarkan Subjek* merupakan generalisasi dari *use case* sebelumnya yaitu *Melakukan Filter Peneliti*. Skenario untuk *use case Filter berdasarkan Subjek* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.12 dan diagram aktivitas untuk use case *Filter berdasarkan Subjek* dapat dilihat pada Gambar 3.11.

*Tabel 3.12 UC-10: Filter Berdasarkan Subjek*

| Nama Use Case | Filter Berdasarkan Keahlian |
|---|---|
| **Kode Use Case** | UC-10 |
| **Aktor** | Pengguna |

| | | | |
|---|---|---|---|
| **Deskripsi** | | Pengguna melakukan *Filter Berdasarkan Subjek* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih tombol "Tampilkan Hasil" | |
| **Kondisi Awal** | | Pengguna belum memilih opsi filter | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna opsi filter Subjek | | |
| 3 | Pengguna memilih tombol Tampilkan Hasil | | |
| | | 4 | Sistem menampilkan hasil filter |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan hasil filter | |
| **Eksepsi : -** | | | |



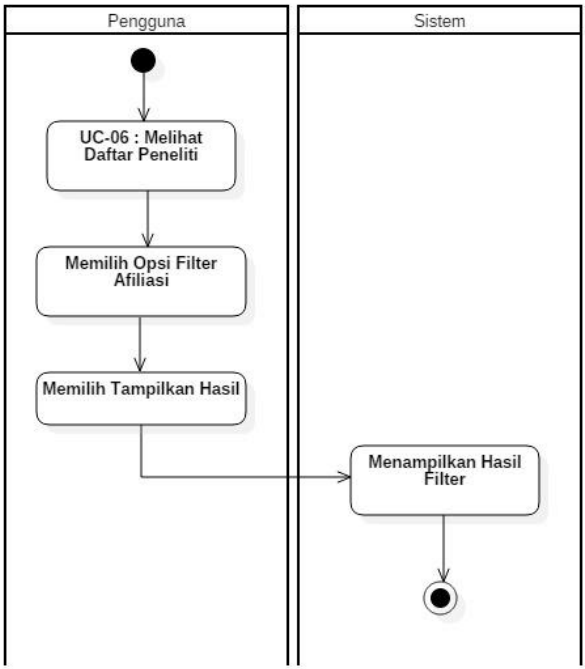

*Gambar 3.11 Filter Berdasarkan Subjek*

### 3.1.3.11 Filter berdasarkan Afiliasi

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur filter peneliti berdasarkan afiliasi. *Use case Filter Berdasarkan Subjek* merupakan generalisasi dari *use case* sebelumnya yaitu *Melakukan Filter Peneliti*. Skenario untuk *use case Filter berdasarkan Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.13 dan diagram aktivitas untuk use case *Filter berdasarkan Afiliasi* dapat dilihat pada Gambar 3.12.

*Tabel 3.13 UC-11: Filter Berdasarkan Afiliasi*

| **Nama Use Case** | | **Filter Berdasarkan Keahlian** | |
|---|---|---|---|
| **Kode Use Case** | | UC-11 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna melakukan *Filter Berdasarkan Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih tombol "Tampilkan Hasil" | |
| **Kondisi Awal** | | Pengguna belum memilih opsi filter | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna opsi filter Subjek | | |
| 3 | Pengguna memilih tombol Tampilkan Hasil | | |
| | | 4 | Sistem menampilkan hasil filter |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan hasil filter | |
| **Eksepsi : -** | | | |



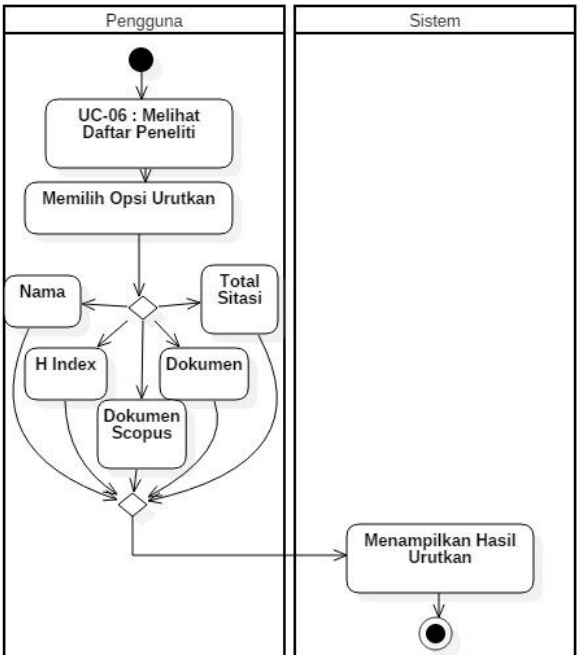

*Gambar 3.12 Filter Berdasarkan Afiliasi*

### 3.1.3.12 Mengurutkan Daftar Peneliti

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur mengurutkan daftar peneliti. Fitur urutkan daftar peneliti memiliki opsi urutkan berdasarkan nama, h-index, total dokumen, total dokumen scopus dan total sitasi. Skenario untuk *use case Mengurutkan Daftar Peneliti* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.14 dan diagram aktivitas untuk use case *Mengurutkan Daftar Peneliti* dapat dilihat pada Gambar 3.13.

*Tabel 3.14 UC-12: Mengurutkan Daftar Peneliti*

| **Nama Use Case** | | **Mengurutkan Daftar Peneliti** | |
|---|---|---|---|
| **Kode Use Case** | | UC-12 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Mengurutkan Daftar Peneliti* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih opsi “Urutkan” | |
| **Kondisi Awal** | | Pengguna belum memilih opsi urutkan | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna memilih opsi urutkan | | |
| | | 3 | Sistem menampilkan hasil urutkan |
| **Alur alternatif : -** | | | |
| 2.1 | Pengguna memilih opsi urutkan Nama | | |
| 2.2 | Pengguna memilih opsi urutkan H Index | | |
| 2.3 | Pengguna memilih opsi urutkan Dokumen | | |
| 2.4 | Pengguna memilih opsi urutkan Dokumen Scopus | | |
| 2.5 | Pengguna memilih opsi urutkan Total Sitasi | | |
| **Kondisi Akhir** | | Sistem menampilkan hasil urutkan | |
| **Eksepsi : -** | | | |



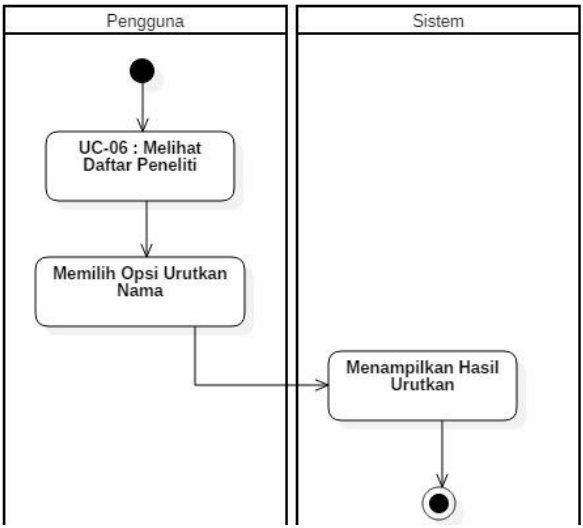

*Gambar 3.13 Mengurutkan Daftar Peneliti*

### 3.1.3.13 Urutkan berdasarkan Nama

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur mengurutkan daftar peneliti berdasarkan nama dari peneliti. *Use case Urutkan berdasarkan Nama* merupakan generalisasi dari *use case Mengurutkan Daftar Peneliti*. Skenario untuk *use case Urutkan berdasarkan Nama* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.15 dan diagram aktivitas untuk use case *Urutkan berdasarkan Nama* dapat dilihat pada Gambar 3.14.

*Tabel 3.15 UC-13: Urutkan Berdasarkan Nama*

| **Nama Use Case** | | **Urutkan Berdasarkan Nama** | |
|---|---|---|---|
| **Kode Use Case** | | UC-13 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Urutkan Berdasarkan Nama* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih opsi “Urutkan” | |
| **Kondisi Awal** | | Pengguna belum memilih opsi urutkan | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna memilih opsi urutkan Nama | | |
| | | 3 | Sistem menampilkan hasil urutkan |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan hasil urutkan | |
| **Eksepsi : -** | | | |




*Gambar 3.14 Urutkan Berdasarkan Nama*

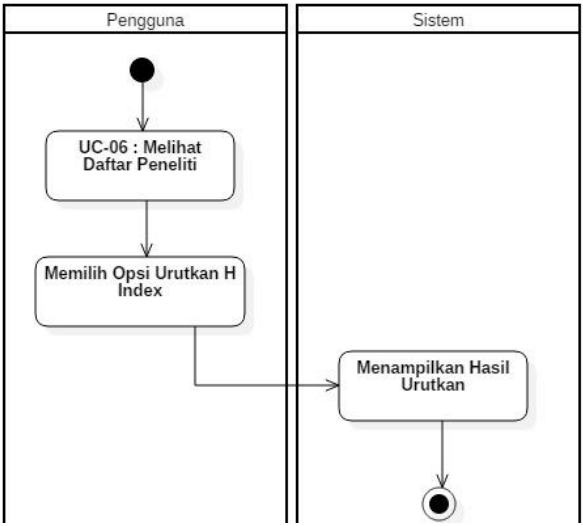
### 3.1.3.14 Urutkan berdasarkan H-Index

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur mengurutkan daftar peneliti berdasarkan h-index peneliti. *Use case Urutkan berdasarkan H-Index* merupakan generalisasi dari *use case Mengurutkan Daftar Peneliti*. Skenario untuk *use case Urutkan berdasarkan H-Index* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.16 dan diagram aktivitas untuk use case *Urutkan berdasarkan H-Index* dapat dilihat pada Gambar 3.15.

*Tabel 3.16 UC-14: Urutkan Berdasarkan H Index*

| Nama Use Case | Urutkan Berdasarkan H Index |
|---|---|
| **Kode Use Case** | UC-14 |
| **Aktor** | Pengguna |
| **Deskripsi** | Pengguna *Urutkan Berdasarkan H Index* pada aplikasi Realitas Virtual untuk visualisasi data peneliti |

| | | | |
|---|---|---|---|
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih opsi “Urutkan” | |
| **Kondisi Awal** | | Pengguna belum memilih opsi urutkan | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna memilih opsi urutkan H Index | | |
| | | 3 | Sistem menampilkan hasil urutkan |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan hasil urutkan | |
| **Eksepsi : -** | | | |




*Gambar 3.15 Urutkan Berdasarkan H Index*

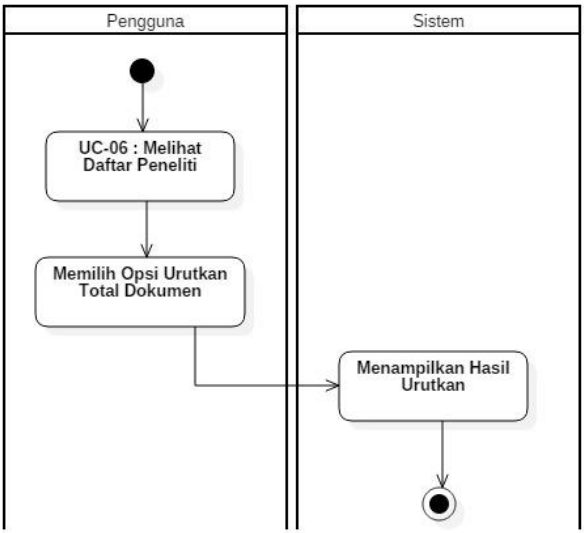
### 3.1.3.15 Urutkan berdasarkan Total Dokumen

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur mengurutkan daftar peneliti berdasarkan total

dokumen peneliti. *Use case Urutkan berdasarkan Total Dokumen* merupakan generalisasi dari *use case Mengurutkan Daftar Peneliti*. Skenario untuk *use case Urutkan berdasarkan Total Dokumen* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.17 dan diagram aktivitas untuk use case *Urutkan berdasarkan Total Dokumen* dapat dilihat pada Gambar 3.16.

*Tabel 3.17 UC-15: Urutkan Berdasarkan Total Dokumen*

| **Nama Use Case** | | **Urutkan Berdasarkan Total Dokumen** | |
|---|---|---|---|
| **Kode Use Case** | | UC-15 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Urutkan Berdasarkan Total Dokumen* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih opsi “Urutkan” | |
| **Kondisi Awal** | | Pengguna belum memilih opsi urutkan | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna memilih opsi urutkan Total Dokumen | | |
| | | 3 | Sistem menampilkan hasil urutkan |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan hasil urutkan | |
| **Eksepsi : -** | | | |



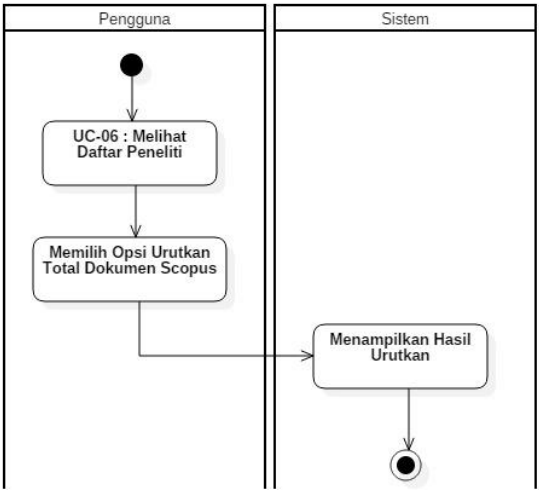

*Gambar 3.16 Urutkan Berdasarkan Total Dokumen*

### 3.1.3.16 Urutkan berdasarkan Total Dokumen Scopus

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur mengurutkan daftar peneliti berdasarkan total dokumen scopus peneliti. *Use case Urutkan berdasarkan Total Dokumen Scopus* merupakan generalisasi dari *use case Mengurutkan Daftar Peneliti*. Skenario untuk *use case Urutkan berdasarkan Total Dokumen Scopus* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.18 dan diagram aktivitas untuk use case *Urutkan berdasarkan Total Dokumen Scopus* dapat dilihat pada Gambar 3.17.

*Tabel 3.18 UC-16: Urutkan Berdasarkan Total Dokumen*

| **Nama Use Case** | | **Urutkan Berdasarkan Total Dokumen Scopus** | |
|---|---|---|---|
| **Kode Use Case** | | UC-16 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Urutkan Berdasarkan Total Dokumen Scopus* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih opsi “Urutkan” | |
| **Kondisi Awal** | | Pengguna belum memilih opsi urutkan | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna memilih opsi urutkan Total Dokumen Scopus | | |
| | | 3 | Sistem menampilkan hasil urutkan |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan hasil urutkan | |
| **Eksepsi : -** | | | |




*Gambar 3.17 Urutkan Berdasarkan Total Dokumen Scopus*

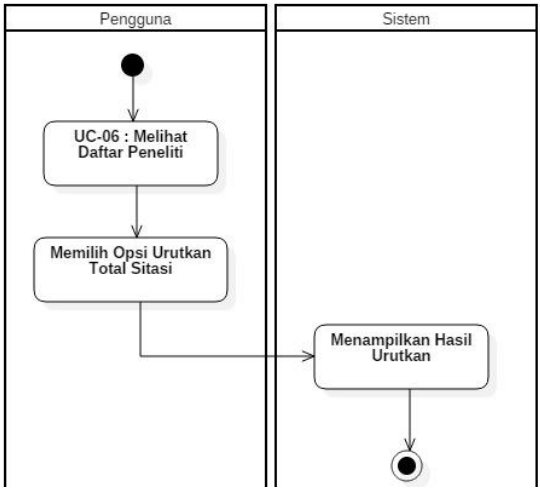
### 3.1.3.17 Urutkan berdasarkan Total Sitasi

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki fitur mengurutkan daftar peneliti berdasarkan total sitasi peneliti. *Use case Urutkan berdasarkan Total Sitasi* merupakan generalisasi dari *use case Mengurutkan Daftar Peneliti*. Skenario untuk *use case Urutkan berdasarkan Total Sitasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.19 dan diagram aktivitas untuk use case *Urutkan berdasarkan Total Sitasi* dapat dilihat pada Gambar 3.18.

*Tabel 3.19 UC-17: Urutkan Berdasarkan Total Sitasi*

| Nama Use Case | Urutkan Berdasarkan Total Sitasi |
|---|---|
| **Kode Use Case** | UC-17 |
| **Aktor** | Pengguna |
| **Deskripsi** | Pengguna *Urutkan Berdasarkan Total Sitasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti |
| **Relasi** | Extend dari UC-06 |

| | | | |
|---|---|---|---|
| **Trigger** | | Pengguna memilih opsi “Urutkan” | |
| **Kondisi Awal** | | Pengguna belum memilih opsi urutkan | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna memilih opsi urutkan Total Sitasi | | |
| | | 3 | Sistem menampilkan hasil urutkan |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan hasil urutkan | |
| **Eksepsi : -** | | | |




*Gambar 3.18 Urutkan Berdasarkan Total Sitasi*

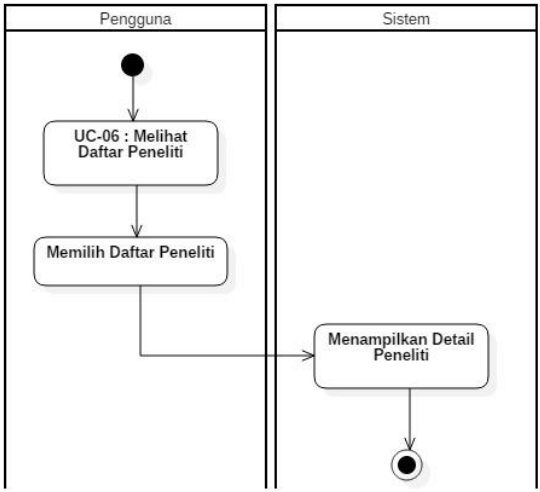
### 3.1.3.18 Melihat Detail Peneliti

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki detail peneliti yang berguna untuk melihat informasi lengkap dari peneliti. *Use case Melihat Detail Peneliti* merupakan

langkah lanjutan dari *use case Melihat Daftar Peneliti*. Skenario untuk *use case Melihat Detail Peneliti* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.20 dan diagram aktivitas untuk use case *Melihat Detail Peneliti* dapat dilihat pada Gambar 3.19.

*Tabel 3.20 UC-18: Urutkan Berdasarkan Total Dokumen*

| **Nama Use Case** | | **Urutkan Berdasarkan Total Sitasi** | |
|---|---|---|---|
| **Kode Use Case** | | UC-18 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Melihat Detail Peneliti* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-06 | |
| **Trigger** | | Pengguna memilih opsi “Urutkan” | |
| **Kondisi Awal** | | Pengguna belum memilih opsi urutkan | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-06 : Melihat Daftar Peneliti | | |
| 2 | Pengguna memilih daftar peneliti | | |
| | | 3 | Sistem menampilkan detail peneliti |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan detail peneliti | |
| **Eksepsi : -** | | | |




*Gambar 3.19 Melihat Detail Peneliti*

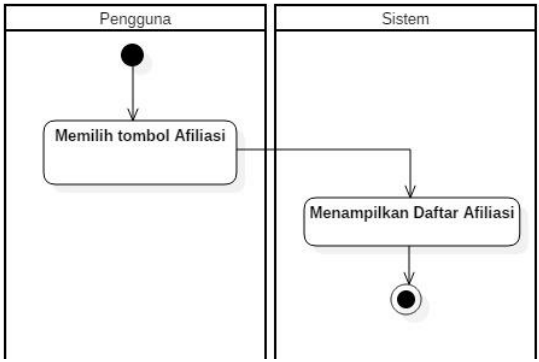
### 3.1.3.19 Melihat Daftar Afiliasi

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki daftar afiliasi. Daftar ini berisi tentang afiliasi-afiliasi yang terdapat pada aplikasi Realitias Virtual ini. Skenario untuk *use case Melihat Daftar Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.21 dan diagram aktivitas untuk use case *Melihat Daftar Afiliasi* dapat dilihat pada Gambar 3.20.

*Tabel 3.21 UC-19: Melihat Daftar Afiliasi*

| Nama Use Case | Melihat Daftar Afiliasi |
|---|---|
| **Kode Use Case** | UC-19 |
| **Aktor** | Pengguna |
| **Deskripsi** | Pengguna melihat *Melihat Daftar Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti |
| **Relasi** | - |
| **Trigger** | Pengguna memilih tombol “Afiliasi” |

| | | | |
|---|---|---|---|
| **Kondisi Awal** | | Pengguna belum memasuki Daftar Afiliasi | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | Pengguna memilih tombol Afiliasi | | |
| | | 2 | Sistem menampilkan Daftar Afiliasi |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Pengguna di teruskan ke Daftar Afiliasi | |
| **Eksepsi : -** | | | |



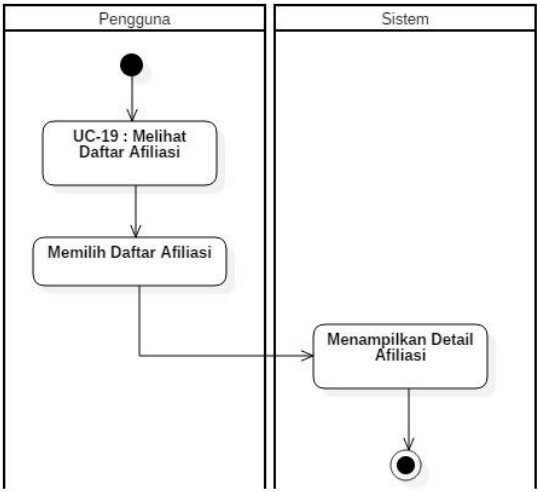

*Gambar 3.20 Diagram Aktivitas Melihat Daftar Afiliasi*

### 3.1.3.20 Melihat Detail Afiliasi

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki detail afiliasi yang berguna untuk melihat informasi lengkap dari afiliasi. *Use case Melihat Detail Afiliasi* merupakan langkah lanjutan dari *use case Melihat Daftar Afiliasi*. Skenario untuk *use case Melihat Detail Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.22 dan diagram aktivitas untuk use case *Melihat Detail Afiliasi* dapat dilihat pada Gambar 3.21.

*Tabel 3.22 UC-20: Melihat Detail Afiliasi*

| **Nama Use Case** | | **Melihat Detail Afiliasi** | |
|---|---|---|---|
| **Kode Use Case** | | UC-20 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Melihat Detail Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-19 | |
| **Trigger** | | Pengguna memilih tombol pada Daftar Afiliasi | |
| **Kondisi Awal** | | Pengguna belum memilih opsi urutkan | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-19 : Melihat Daftar Afiliasi | | |
| 2 | Pengguna memilih daftar afiliasi | | |
| | | 3 | Sistem menampilkan detail afiliasi |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan detail afiliasi | |
| **Eksepsi : -** | | | |



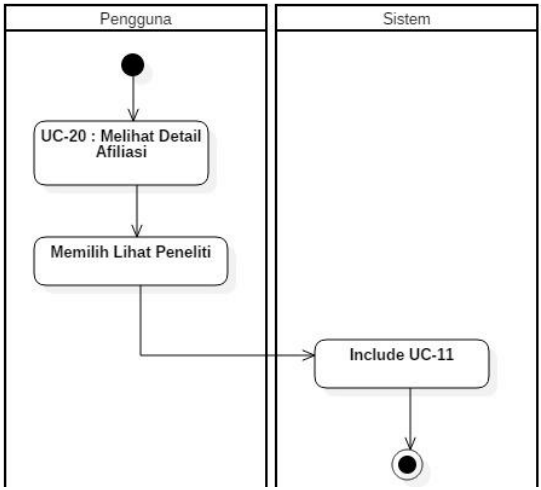

*Gambar 3.21 Melihat Detail Afiliasi*

### 3.1.3.21 Melihat Peneliti berdasarkan Detail Afiliasi

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki dapat melihat peneliti berdasarkan afiliasi setelah berada pada detail afiliasi. Daftar ini berisi tentang peneliti-peneliti berdasarkan detail afiliasi terkait yang terdapat pada aplikasi Realitias Virtual ini. Skenario untuk *use case Melihat Peneliti berdasarkan Detail Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.23 dan diagram aktivitas untuk use case *Melihat Peneliti berdasarkan Detail Afiliasi* dapat dilihat pada Gambar 3.22.

*Tabel 3.23 UC-21: Melihat Peneliti berdasarkan Afiliasi*

| **Nama Use Case** | | **Melihat Peneliti berdasarkan Detail Afiliasi** | |
|---|---|---|---|
| **Kode Use Case** | | UC-21 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Melihat Peneliti berdasarkan Detail Afiliasi* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-20<br>Include dari UC-11 | |
| **Trigger** | | Pengguna memilih tombol Lihat Peneliti pada Detail Afiliasi | |
| **Kondisi Awal** | | Pengguna berada pada detail afiliasi | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-20 : Melihat Detail Afiliasi | | |
| 2 | Pengguna memilih Lihat Peneliti | | |
| **Include UC-11 : Filter berdasarkan Afiliasi** | | | |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan peneliti berdasarkan afiliasi | |
| **Eksepsi : -** | | | |




*Gambar 3.22 Melihat Peneliti berdasarkan Detail Afiliasi*

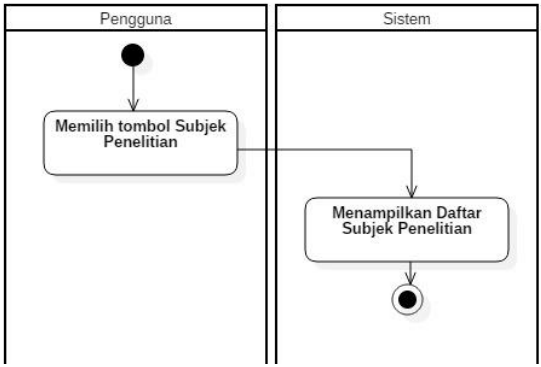
### 3.1.3.22 Melihat Daftar Subjek Penelitian

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki daftar subjek penelitian. Daftar ini berisi tentang subjek-subjek yang diteliti pada aplikasi Realitias Virtual ini. Skenario untuk *use case Melihat Daftar Subjek Penelitian* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.24 dan diagram aktivitas untuk use case *Melihat Daftar Subjek Penelitian* dapat dilihat pada Gambar 3.23.

*Tabel 3.24 UC-22: Melihat Daftar Subjek Penelitian*

| Nama Use Case | Melihat Daftar Subjek Penelitian |
|---|---|
| **Kode Use Case** | UC-22 |
| **Aktor** | Pengguna |
| **Deskripsi** | Pengguna melihat *Melihat Daftar Subjek Penelitian* pada aplikasi Realitas Virtual untuk visualisasi data peneliti |

| | | | |
|---|---|---|---|
| **Relasi** | | - | |
| **Trigger** | | Pengguna memilih tombol “Subjek Penelitian” | |
| **Kondisi Awal** | | Pengguna belum memasuki Daftar Subjek Penelitian | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | Pengguna memilih tombol Subjek Penelitian | | |
| | | 2 | Sistem menampilkan Daftar Subjek Penelitian |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Pengguna di teruskan ke Daftar Subjek Penelitian | |
| **Eksepsi : -** | | | |




*Gambar 3.23 Diagram Aktivitas Melihat Daftar Subjek Penelitian*

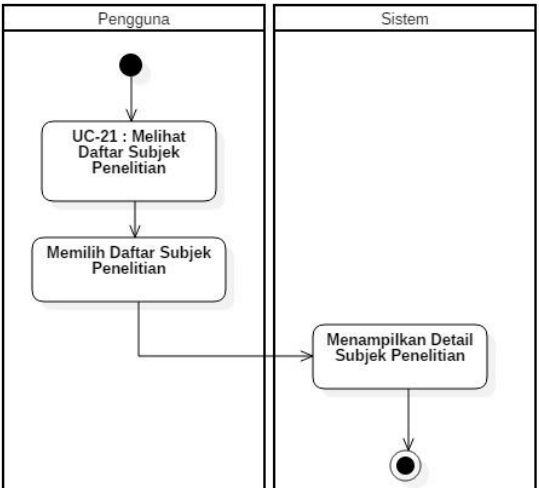
### 3.1.3.23 Melihat Detail Subjek Penelitian

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki detail subjek penelitian yang berguna untuk melihat informasi lengkap dari subjek penelitian. *Use case Melihat Detail Subjek Penelitian* merupakan langkah lanjutan dari *use case*

*Melihat Daftar Subjek Penelitian*. Skenario untuk *use case Melihat Detail Subjek Penelitian* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.25 dan diagram aktivitas untuk use case *Melihat Detail Subjek Penelitian* dapat dilihat pada Gambar 3.24.

*Tabel 3.25 UC-23: Melihat Detail Subjek Penelitian*

| **Nama Use Case** | | **Melihat Detail Subjek Penelitian** | |
|---|---|---|---|
| **Kode Use Case** | | UC-23 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Melihat Detail Subjek Penelitian* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-21 | |
| **Trigger** | | Pengguna memilih tombol pada Daftar Subjek Penelitian | |
| **Kondisi Awal** | | Pengguna belum Melihat Subjek Penelitian | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-21 : Melihat Daftar Subjek Penelitian | | |
| 2 | Pengguna memilih tombol Subjek Penelitian | | |
| | | 3 | Sistem menampilkan Detail Subjek Penelitian |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan Detail Subjek Penelitian | |
| **Eksepsi : -** | | | |



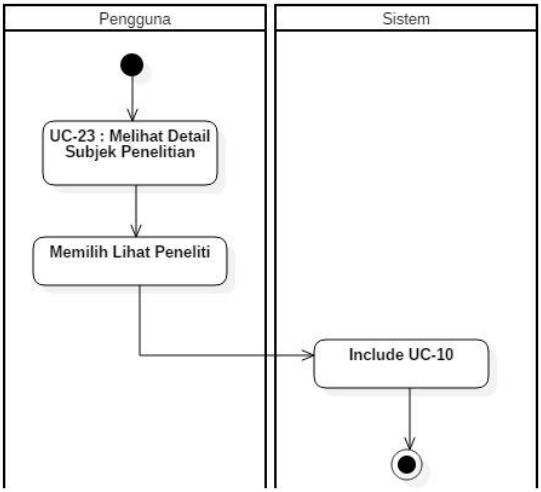

*Gambar 3.24 Melihat Detail Subjek Penelitian*

### 3.1.3.24 Melihat Peneliti berdasarkan Detail Subjek Penelitian

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki dapat melihat peneliti berdasarkan subjek penelitian setelah berada pada detail afiliasi. Daftar ini berisi tentang peneliti-peneliti berdasarkan detail subjek penelitian terkait yang terdapat pada aplikasi Realitias Virtual ini. Skenario untuk *use case Melihat Peneliti berdasarkan Detail Subjek Penelitian* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.26 dan diagram aktivitas untuk use case *Melihat Peneliti berdasarkan Detail Subjek Penelitian* dapat dilihat pada Gambar 3.25.

*Tabel 3.26 UC-24: Melihat Peneliti berdasarkan Subjek Penelitian*

| **Nama Use Case** | | **Melihat Peneliti berdasarkan Detail Subjek Penelitian** | |
|---|---|---|---|
| **Kode Use Case** | | UC-24 | |
| **Aktor** | | Pengguna | |
| **Deskripsi** | | Pengguna *Melihat Peneliti berdasarkan Detail Subjek Penelitian* pada aplikasi Realitas Virtual untuk visualisasi data peneliti | |
| **Relasi** | | Extend dari UC-23<br>Include dari UC-10 | |
| **Trigger** | | Pengguna memilih tombol Lihat Peneliti pada Detail Subjek Penelitian | |
| **Kondisi Awal** | | Pengguna berada pada detail subjek penelitian | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | UC-23 : Melihat Detail Subjek Penelitian | | |
| 2 | Pengguna memilih Lihat Peneliti | | |
| **Include UC-10 : Filter berdasarkan Subjek Penelitian** | | | |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Sistem menampilkan peneliti berdasarkan subjek penelitian | |
| **Eksepsi : -** | | | |




*Gambar 3.25 Melihat Detail Afiliasi*

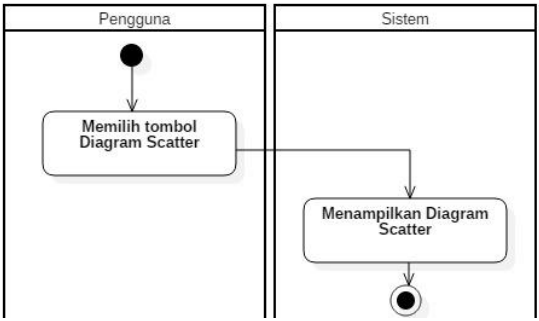
### 3.1.3.25 Melihat Diagram *Scatter*

Aplikasi Realitas Virtual untuk visualisasi data peneliti ini memiliki diagram *scatter*. Diagram *scatter* ini memperlihatkan pemetaan peneliti. Skenario untuk *use case Melihat Diagram Scatter* pada aplikasi Realitas Virtual untuk visualisasi data peneliti dapat dilihat pada Tabel 3.27 dan diagram aktivitas untuk use case *Melihat Diagram Scatter* dapat dilihat pada Gambar 3.26.

*Tabel 3.27 UC-25: Melihat Diagram Scatter*

| Nama Use Case | Melihat Diagram Scatter |
|---|---|
| **Kode Use Case** | UC-25 |
| **Aktor** | Pengguna |
| **Deskripsi** | Pengguna melihat *Melihat Diagram Scatter* pada aplikasi Realitas Virtual untuk visualisasi data peneliti |
| **Relasi** | - |

| | | | |
|---|---|---|---|
| **Trigger** | | Pengguna memilih tombol "Diagram *Scatter*" | |
| **Kondisi Awal** | | Pengguna belum memasuki tampilan Diagram *Scatter* | |
| **Alur Normal:** | | **Sistem:** | |
| 1 | Pengguna memilih tombol Diagram *Scatter* | | |
| | | 2 | Sistem menampilkan Diagram *Scatter* |
| **Alur alternatif : -** | | | |
| **Kondisi Akhir** | | Pengguna di teruskan ke Diagram *Scatter* | |
| **Eksepsi : -** | | | |




*Gambar 3.26 Diagram Aktivitas Melihat Daftar Subjek Penelitian*

## 3.2 Perancangan Sistem

Tahap perancangan ini akan dibagi menjadi tiga bagian yaitu perancangan basis data, perancangan antar muka dan perancangan objek dalam aplikasi Realitas Virtual untuk visualisasi data peneliti.

### 3.2.1 Deskripsi Umum Sistem

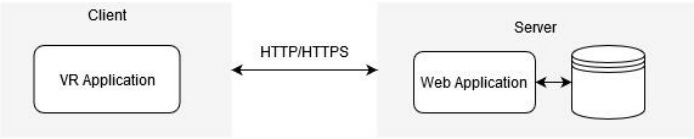
Penerapan Realitas Virtual untuk visualisasi data peneliti adalah sistem dimana pengguna dapat melakukan explorasi data tentang peneliti seperti total sitasi, total dokumen penelitian, total dokumen penelitian yang dipublikasikan di scopus, dan juga h index peneliti.

Dalam aplikasi Realitas Virtual ini, pengguna dapat melakukan pencarian sesuai dengan permintaan pengguna. Pengguna dapat berinteraksi dengan lingkungan 3-dimensi seperti menyentuh, mengambil, dan memindahkan. Selain explorasi, sistem ini memungkinkan untuk melakukan visualisasi data peneliti sesuai dengan apa yang diminta oleh pengguna.

Sistem pada aplikasi Realitas Virtual ini menggunakan arsitektur *client-server* dalam pengambilan datanya. *Client* akan mengirimkan permintaan kepada server. Setelah permintaan diterima server, server akan memproses permintaan lalu mengirimkan respon dalam bentuk JSON kepada *client.*

### 3.2.2 Arsitektur Sistem

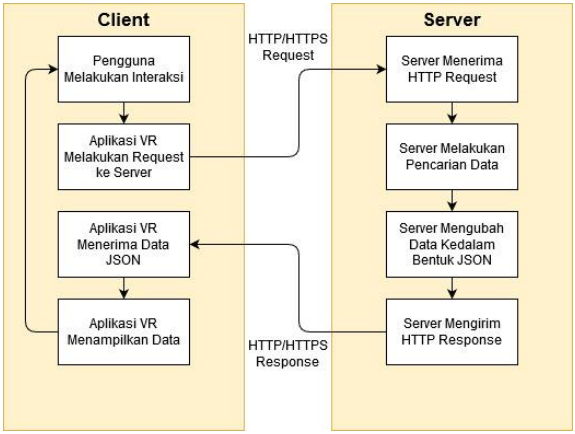
Aplikasi Realitas Virtual untuk Visualisasi data peneliti ini menerapkan arsitektur *client-server* dimana arsitektur ini melibatkan aplikasi *client* berbasis Android yang dibangun menggunakan Unity3D sebagai aplikasi yang menampilkan data dari *server* seperti yang ditunjukkan pada Gambar 3.27.




*Gambar 3.27 Arsitektur Aplikasi Realitas Virtual untuk Visualisasi Data peneliti*

Server untuk aplikasi Realitas Virtual ini dibangun menggunakan *Flask micro framework* sebagai penyimpanan dan pemrosesan data yang berkomunikasi dengan aplikasi *cilent* menggunakan port HTTP atau HTTPS menggunakan API *end point*.

Aplikasi *client* akan mengirimkan permintaan melalui HTTP *Request* menggunakan API *end point* dari aplikasi web lalu akan diproses oleh backend untuk mengambil data pada basis data dan dikembalikan kepada aplikasi *client* dalam bentuk objek JSON menggunakan HTTP *Response*. Alur aplikasi Realitas Virtual untuk visualisasi data peneliti ditunjukkan pada Gambar 3.1.




*Gambar 3.28 Alur Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

## 3.3 Perancangan Basis Data

Aplikasi Realitas Virtual untuk visualisasi data peneliti membutuhkan basis data guna menyimpan data peneliti pada server. Basis data ini terdiri dari beberapa tabel yaitu Afiliasi, Peneliti, Subjek Area, Departemen, Detail Subjek Area, Fakultas, Keahlian, Subjek Topik Tier 1 dan Subjek Topik Tier 2.

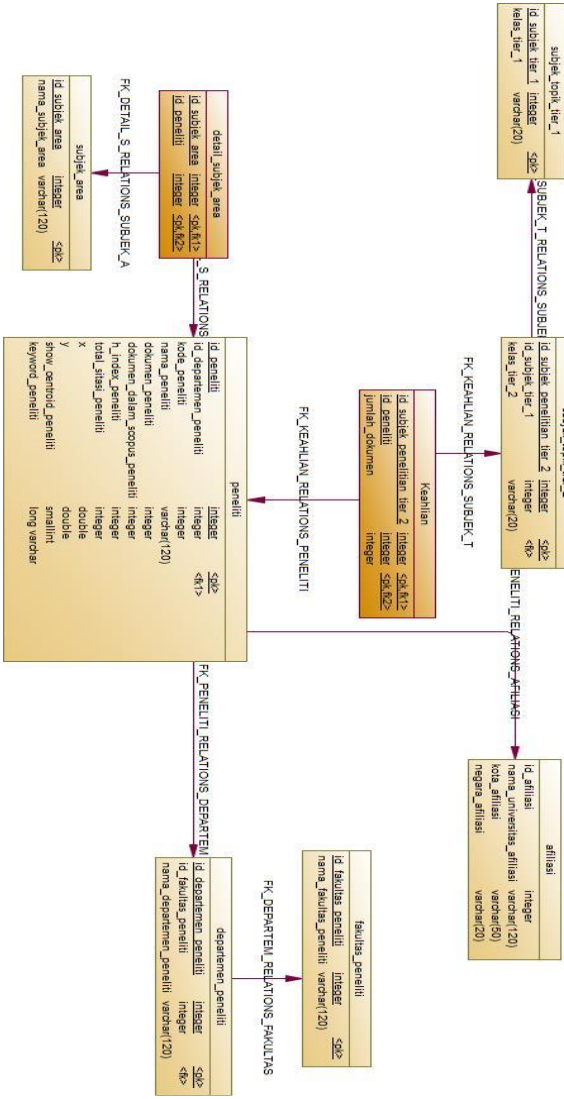
Untuk lebih detail mengenai tabel basis data untuk aplikasi Realitas Virtual untuk visualisasi data peneliti, daftar tabel dapat dilihat pada Tabel 3.28. Sedangkan gambar model basis data untuk aplikasi Realitas Virtual ini dapat dilihat pada Gambar 3.29.

*Tabel 3.28 Daftar Tabel Basis Data Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

| Nama Tabel | Atribut | Tipe Data | Null | Primary Key (PK)/ Foreign Key (FK) |
|---|---|---|---|---|
| Afiliasi | id_afiliasi | Int(11) | Tidak | PK |
| | universitas | Varchar(22) | Ya | - |
| | kota | Varchar(22) | Ya | - |
| | negara | Varchar(22) | Ya | - |
| | logo | text | Ya | - |
| departemen | id_departemen | Int(11) | Tidak | PK |
| | nama_ departemen | Varchar(120) | Tidak | - |
| | id_fakultas_ departemen | Int(11) | Tidak | FK (fakultas) |
| detail_ subjek_area | id_detail_ subjek_area | Int(11) | Tidak | PK |
| | id_subjek_area | Int(11) | Tidak | FK |

| | | | | |
|---|---|---|---|---|
| | | | | (subjek_ area) |
| | id_peneliti | Int(11) | Tidak | FK (peneliti) |
| fakultas | id_fakultas | Int(11) | Tidak | PK |
| | nama_fakultas | Varchar(120) | Tidak | - |
| keahlian | id_keahlian | Int(11) | Tidak | PK |
| | id_subjek_ topik_tier_2 | Int(11) | Tidak | FK (subjek_topik _tier_1) |
| | id_peneliti | Int(11) | Tidak | FK (peneliti) |
| | jumlah_ dokumen | Int(11) | Tidak | |
| peneliti | id_peneliti | Int(11) | Tidak | PK |
| | kode_peneliti | BigInt(20) | Tidak | |
| | nama_peneliti | Varchar(120) | Tidak | |
| | dokumen_ peneliti | Int(11) | Tidak | |
| | dokumen_dalam_ scopus_penelit i | Int(11) | Tidak | |
| | h_index_ peneliti | Int(11) | Tidak | |
| | total_sitasi_ peneliti | Int(11) | Tidak | |
| | x | Double | Tidak | |
| | y | Double | Tidak | |
| | keyword_ peneliti | | Tidak | |
| | show_centroid _ peneliti | TinyInt(1) | Tidak | |

| | | | | |
|---|---|---|---|---|
| | id_departemen | Int(11) | Tidak | FK (departemen) |
| | id_afiliasi | Int(11) | Tidak | FK (afiliasi) |
| subjek_area | id_subjek_area | Int(11) | Tidak | PK |
| | nama_subjek_area | Varchar(120) | Tidak | |
| subjek_topik _tier_1 | id_subjek_topik _tier_1 | Int(11) | Tidak | PK |
| | kelas_tier_1 | Varchar(20) | Tidak | |
| subjek_ topik_tier_2 | id_subjek_ topik_tier_2 | Int(11) | Tidak | PK |
| | kelas_tier_2 | Varchar(20) | Tidak | |
| | id_subjek_ topik_tier_1 | Int(11) | Tidak | FK (subjek_topik _tier_1) |



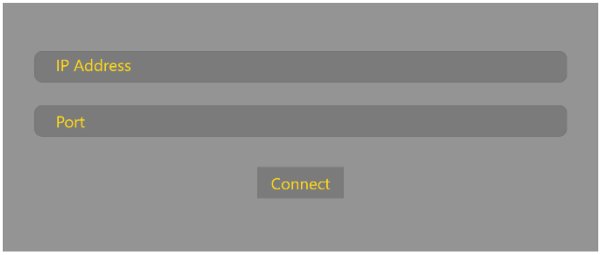

*Gambar 3.29 Model Basis Data Realitas Virtual untuk Visualisasi Data Peneliti*

## 3.4 Perancangan Antarmuka Pengguna

Pada bagian ini akan membahas tentang bagaimana rancangan antarmuka pengguna yang nantinya akan diterapkan pada aplikasi Realitas Virtual untuk visualisasi data peneliti. Antarmuka aplikasi ini terdapat beberapa bagian yaitu beranda, daftar peneliti, detail peneliti, daftar subjek, detail subjek, daftar afiliasi, detail afiliasi, dan diagram *scatter*.

### 3.4.1 Rancangan Antarmuka Panel Konfigurasi

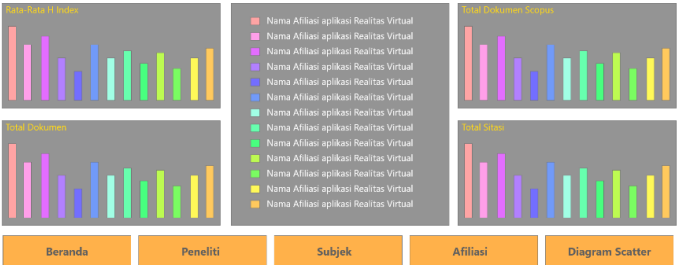
Panel konfigurasi adalah panel yang digunakan untuk memudahkan dalam pengaturan URL atau alamat IP yang digunakan untuk komunikasi dari *client* ke *server*. Tampilan beranda dapat dilihat pada Gambar 3.30.




*Gambar 3.30 Rancangan Antarmuka Panel Konfigurasi*

### 3.4.2 Rancangan Antarmuka Beranda

Beranda adalah halaman awal dari aplikasi Realitas Virtual untuk visualisasi data peneliti dimana terdapat empat diagram batang yang menunjukkan rata-rata h-index, total dokumen, total dokumen dalam aplikasi realitas virtual. Tampilan beranda dapat dilihat pada Gambar 3.31.




*Gambar 3.31 Rancangan Antarmuka Beranda*

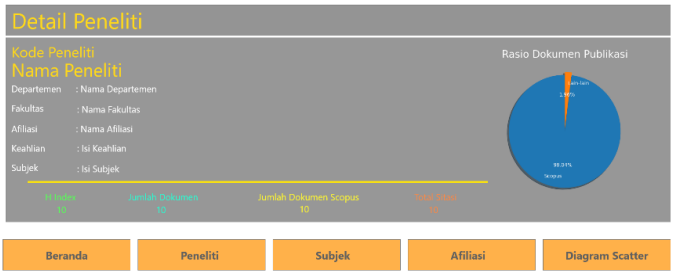
### 3.4.3 Rancangan Antarmuka Daftar Peneliti

Daftar peneliti adalah daftar yang menyajikan informasi secara ringkas dari peneliti pada aplikasi Realitas Virtual ini. Daftar peneliti memiliki fitur urutkan yang dapat mengurutkan daftar peneliti berdasarkan nama, h-index, total dokumen, total dokumen scopus dan total sitasi. Selain itu daftar peneliti memiliki fitur filter yang dapat melakukan filter peneliti berdasarkan afiliasi, subjek dan keahlian dari peneliti. Daftar peneliti juga memiliki fitur pencarian menggunakan perintah suara dimana saat pengguna ingin mencari peneliti tertentu. Tampilan daftar peneliti dapat dilihat pada Gambar 3.32.


*Gambar 3.32 Rancangan Antarmuka Daftar Peneliti*

### 3.4.4 Rancangan Antarmuka Detail Peneliti

Detail peneliti adalah halaman yang menunjukkan secara lengkap informasi dari peneliti pada aplikasi Realitas Virtual. Beberapa informasi yang ditampilkan antara lain adalah nama, departemen, fakultas, afiliasi, keahlian, subjek penelitian, h index, jumlah dokumen, jumlah dokumen scopus, total sitasi serta diagram yang menunjukkan rasio dokumen dan dokumen scopus pada peneliti. Tampilan detail peneliti dapat dilihat pada Gambar 3.33.




*Gambar 3.33 Rancangan Antarmuka Detail Peneliti*

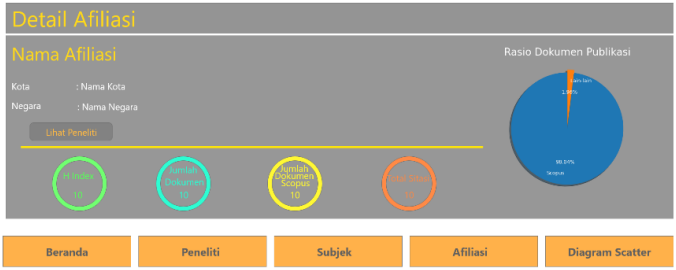
### 3.4.5 Rancangan Antarmuka Daftar Afiliasi

Daftar afiliasi adalah daftar yang menyajikan nama-nama dari afiliasi peneliti pada aplikasi Realitas Virtual. Daftar afiliasi ini memiliki tampilan yang dapat dilihat pada Gambar 3.34.


*Gambar 3.34 Rancangan Antarmuka Daftar Afiliasi*

### 3.4.6 Rancangan Antarmuka Detail Afiliasi

Detail peneliti adalah halaman yang menunjukkan secara lengkap informasi dari peneliti pada aplikasi Realitas Virtual. Beberapa informasi yang ditampilkan antara lain adalah nama, kota, negara, h index, jumlah dokumen, jumlah dokumen scopus, total sitasi serta diagram yang menunjukkan rasio dokumen dan dokumen scopus pada afiliasi. Tampilan detail afiliasi dapat dilihat pada Gambar 3.35.




*Gambar 3.35 Rancangan Antarmuka Detail Afiliasi*

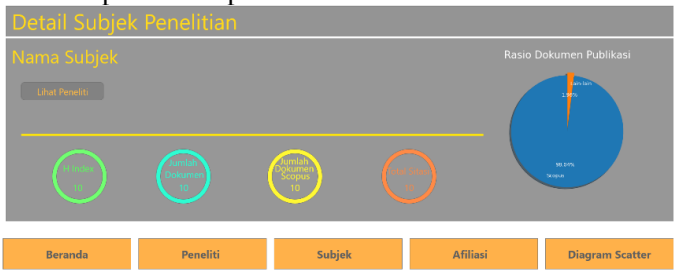
### 3.4.7 Rancangan Antarmuka Daftar Subjek Penelitian

Daftar subjek penelitian adalah daftar yang menyajikan subjek-subjek yang diteliti oleh peneliti pada aplikasi Realitas Virtual. Daftar afiliasi ini memiliki tampilan yang dapat dilihat pada Gambar 3.36.


*Gambar 3.36 Rancangan Antarmuka Daftar Subjek Penelitian*

### 3.4.8 Rancangan Antarmuka Detail Subjek Penelitian

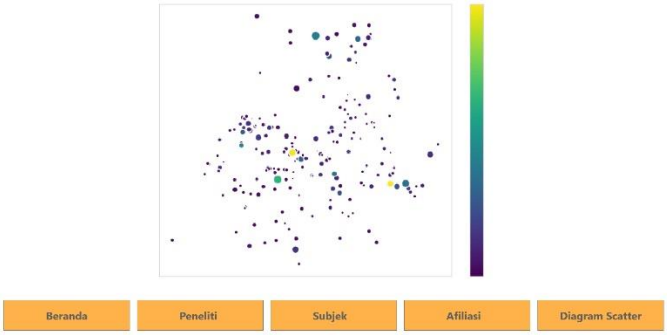
Detail subjek penelitian adalah halaman yang menunjukkan secara lengkap informasi dari subjek pada aplikasi Realitas Virtual. Beberapa informasi yang ditampilkan antara lain adalah nama, h index, jumlah dokumen, jumlah dokumen scopus, total sitasi serta diagram yang menunjukkan rasio dokumen dan dokumen scopus pada subjek penelitian. Tampilan detail subjek penelitian dapat dilihat pada Gambar 3.37.



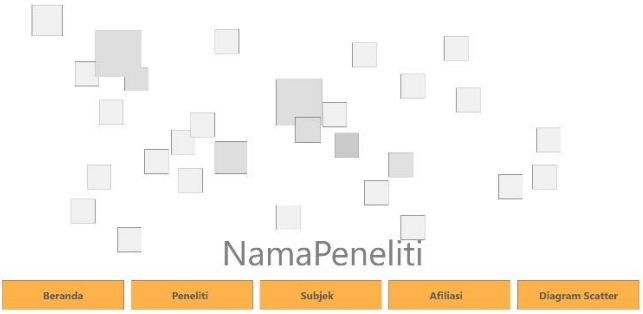

*Gambar 3.37 Rancangan Antarmuka Daftar Subjek Penelitian*

### 3.4.9 Rancangan Antarmuka Diagram *Scatter*

Diagram *scatter* adalah diagram yang memvisualisasikan pemetaan atau pengklasteran dari peneliti. Dalam diagram *scatter* ini memiliki dua buah variabel yaitu h index peneliti dan jumlah dokumen yang dipublikasikan. Semakin tinggi jumlah dokumen yang dipublikasikan akan menghasilkan warna yang cerah. Sedangkan semakin tinggi H Index peneliti akan menghasilkan ukuran yang lebih besar. Tampilan diagram *scatter* dalam bentuk 2-dimensi dapat dilihat pada Gambar 3.38 serta tampilan diagram *scatter* dalam bentuk 3-dimensi dapat dilihat pada Gambar 3.39 serta




*Gambar 3.38 Rancangan Antarmuka Diagram Scatter*




*Gambar 3.39 Rancangan Antarmuka Diagram Scatter*

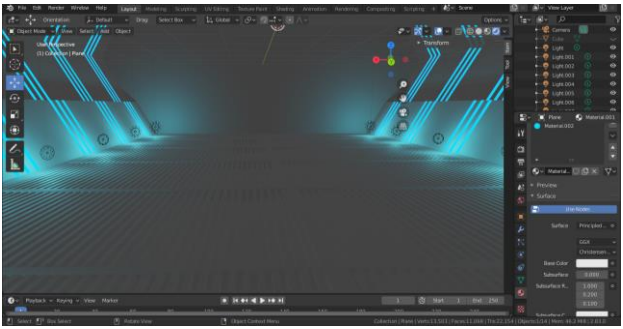
## 3.5 Perancangan Objek dalam Aplikasi Realitas Virtual

Pada bagian ini akan membahas tentang bagaimana rancangan objek dalam aplikasi Realitas Virtual untuk visualisasi data peneliti. Dikarenakan aplikasi Realitas Virtual untuk visualisasi data peneliti ini memanfaatkan ruang virtual 3-dimensi, maka aplikasi ini dilengkapi dengan model 3-dimensi sebagai pelengkapnya. Model 3-dimensi ini dirancang dengan menggunakan perangkat lunak Blender. Perancangan ini ditujukan agar menciptakan lingkungan dengan kesan futuristik dan realistik.

Dalam perancangan ini dibagi menjadi dua bagian yaitu perancangan koridor dan panel kontrol.

### 3.5.1 Perancangan Koridor

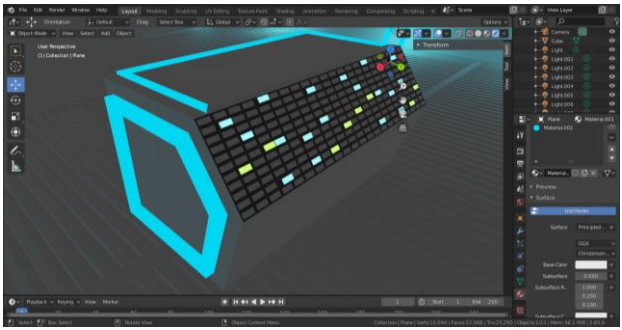
Pada bagian ini akan membahas tentang bagaimana rancangan objek koridor. Koridor ini berkonsep fiksi sains sehingga pengguna akan merasakan seperti di masa depan. Koridor ini dibentuk dari objek *plane* dan objek *cube* kemudian dimodifikasi sedemikian rupa menggunakan *modifier array* dan *mirror* sehingga menyerupai koridor. Selain itu koridor ini memiliki atap dan lantai berongga serta dinding yang bersifat emisif sehingga menyebabkan dinding seperti menyala dan menambah kesan futuristik pada koridor. Gambar rancangan koridor dapat dilihat pada Gambar 3.38.



*Gambar 3.40 Rancangan Koridor*

### 3.5.2 Perancangan Panel Kontrol

Pada bagian ini akan membahas tentang bagaimana rancangan objek panel kontrol. Panel kontrol ini juga berkonsep fiksi sains dengan dipenuhi oleh tombol yang menyala. Panel kontrol ini dibentuk dari objek *cube* kemudian dimodifikasi sedemikian rupa sehingga menyerupai panel kontrol. Gambar rancangan panel kontrol dapat dilihat pada Gambar 3.39.



*Gambar 3.41 Rancangan Panel Kontrol*

# BAB IV
# IMPLEMENTASI SISTEM

Bab ini membahas tentang bagaimana implementasi sistem yang telah dirancang. Bahasa pemrograman yang digunakan dalam implementasi sistem ini meliputi Bahasa pemrograman C# untuk aplikasi *client*. Sedangkan untuk aplikasi server menggunakan Bahasa pemrograman Python serta basis data menggunakan MySQL pada PhpMyAdmin.

## 4.1 Lingkungan Operasi Sistem

Linkungan operasi yang digunakan untuk mengembangkan aplikasi Realitas Virtual ini ditunjukkan pada Tabel 4.1.

*Tabel 4.1 Lingkungan Operasi Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

| Perangkat | Spesifikasi |
|---|---|
| Perangkat Keras Server | Sistem Operasi Server: Windows 10<br>Prosesor Server: Intel(R) Core(TM) i7 6700HQ CPU<br>GPU Server: NVIDIA Geforce GTX 950, 4GB VRAM<br>Memori Server : 8 GB RAM<br>Penyimpanan Server : 1TB HDD & 240 GB SSD |
| Perangkat Keras Klien | Sistem Operasi Klien : Android 9.0 (Pie)<br>Chipset Klien : Qualcomm SDM660 Snapdragon 660 (14 nm)<br>CPU Klien : Octa-core (4x2.2 GHz Kryo 260 Gold & 4x1.8 GHz Kryo 260 Silver)<br>GPU Klien : Adreno 512<br>Memori Klien : 4GB RAM<br>Penyimpanan Klien : 64GB Internal |

| Perangkat | Spesifikasi |
|---|---|
| Perangkat Lunak | DBMS : MySQL<br>Bahasa : C# , Python<br>Kerangka Kerja : Unity3D 10.3, Flask *micro framework*.<br>Perangkat Lunak Pendukung : StarUML, Adobe XD, Power Designer, Microsoft Excel, Microsoft Word, Microsoft Visual Studio Code, Microsoft Visual Studio Community, Blender. |

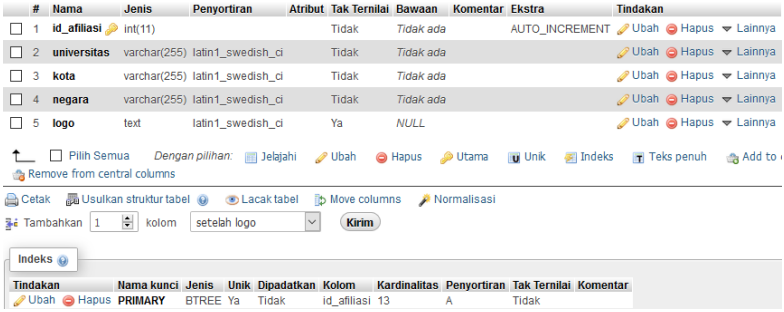
## 4.2 Implementasi Basis Data

Pada bagian ini membahas tentang bagaimana implementasi basis data sebagai penyimpanan data peneliti. Basis data yang telah dirancang pada tahap perancangan kemudian diimplementasikan kedalam PhpMyAdmin. Data peneliti untuk aplikasi ini dinormalisasi kedalam tabel yang telah dirancang sebelumnya.

Langkah awal dalam implementasi basis data aplikasi Realitas Virtual untuk visualisasi data peneliti adalah membuat tabel pada basis data yang telah dirancang sebelumnya,

### 4.2.1 Implementasi Tabel Afiliasi

Implementasi tabel afiliasi dilakukan dengan membuat tabel dengan nama afiliasi dengan atribut id_afiliasi sebagai *primary key* dengan tipe data int(11), universitas dengan tipe data varchar(22), kota dengan tipe data varchar(22), negara dengan tipe data varchar(22) dan logo menggunakan tipe data *text*. Implementasi tabel afiliasi dapat dilihat pada Gambar 4.1.



| # | Nama | Jenis | Penyortiran | Atribut | Tak Ternilai | Bawaan | Komentar | Ekstra | Tindakan |
|---|---|---|---|---|---|---|---|---|---|
| 1 | id_afiliasi | int(11) | | | Tidak | *Tidak ada* | | AUTO_INCREMENT | Ubah Hapus Lainnya |
| 2 | universitas | varchar(255) | latin1_swedish_ci | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 3 | kota | varchar(255) | latin1_swedish_ci | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 4 | negara | varchar(255) | latin1_swedish_ci | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 5 | logo | text | latin1_swedish_ci | | Ya | *NULL* | | | Ubah Hapus Lainnya |

Pilih Semua *Dengan pilihan:* Jelajahi Ubah Hapus Utama Unik Indeks Teks penuh Add to
Remove from central columns

Cetak Usulkan struktur tabel Lacak tabel Move columns Normalisasi

Tambahkan 1 kolom setelah logo Kirim

Indeks

| Tindakan | Nama kunci | Jenis | Unik | Dipadatkan | Kolom | Kardinalitas | Penyortiran | Tak Ternilai | Komentar |
|---|---|---|---|---|---|---|---|---|---|
| Ubah Hapus | PRIMARY | BTREE | Ya | Tidak | id_afiliasi | 13 | A | Tidak | |

*Gambar 4.1 Implementasi Tabel Afiliasi*

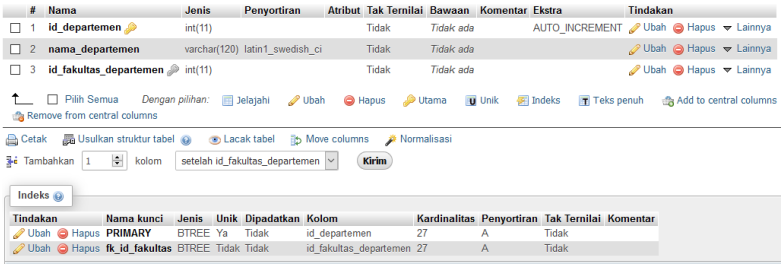
Setelah tabel dibuat, langkah selanjutnya yaitu memasukkan data afiliasi kedalam tabel. Sehingga tabel memiliki data seperti yang ditunjukkan pada Gambar 4.2.

| | id_afiliasi | universitas | kota | negara | logo |
|---|---|---|---|---|---|
| Ubah Salin Hapus | 1 | Institut Teknologi Sepuluh Nopember | Surabaya | Indonesia | ./logo_universitas/its.png |
| Ubah Salin Hapus | 2 | Telkom University | Bandung West Java | Indonesia | ./logo_universitas/telu.png |
| Ubah Salin Hapus | 3 | Politeknik Elektronika Negeri Surabaya | Surabaya | Indonesia | ./logo_universitas/pens.png |
| Ubah Salin Hapus | 4 | Universitas Airlangga | Surabaya | Indonesia | ./logo_universitas/unair.png |
| Ubah Salin Hapus | 5 | Universitas Islam Darul 'Ulum | | Indonesia | ./logo_universitas/unisda.jpg |
| Ubah Salin Hapus | 6 | Institut Teknologi Sepuluh November (ITS) | | Indonesia | ./logo_universitas/its.png |
| Ubah Salin Hapus | 7 | Institut Tehnologi Sepuluh Nopember | Surabaya | Indonesia | ./logo_universitas/its.png |
| Ubah Salin Hapus | 8 | Gadjah Mada University | Yogyakarta | Indonesia | ./logo_universitas/ugm.png |
| Ubah Salin Hapus | 9 | Universitas Nahdlatul Ulama | | Indonesia | ./logo_universitas/unu.jpg |
| Ubah Salin Hapus | 10 | Faculty of Mathematics Computation and Data Scienc... | | Indonesia | |
| Ubah Salin Hapus | 11 | Faculty of Technology Information | Surabaya | Indonesia | |
| Ubah Salin Hapus | 12 | Kementerian Agama Republik Indonesia | Banda Aceh | Indonesia | ./logo_universitas/kemenag.png |
| Ubah Salin Hapus | 13 | Universitas Gadjah Mada | Yogyakarta | Indonesia | ./logo_universitas/ugm.png |

*Gambar 4.2 Contoh Data Afiliasi*

### 4.2.2 Implementasi Tabel Departemen

Implementasi tabel afiliasi dilakukan dengan membuat tabel dengan nama departemen dengan atribut id_departemen sebagai *primary key* dengan tipe data int(11), nama_departemen dengan tipe data varchar(120), dan id_fakultas_departemen sebagai *foreign key* yang mereferensi ke tabel fakultas. Implementasi tabel departemen dapat dilihat pada Gambar 4.3.



| # | Nama | Jenis | Penyortiran | Atribut | Tak Ternilai | Bawaan | Komentar | Ekstra | Tindakan |
|---|---|---|---|---|---|---|---|---|---|
| 1 | id_departemen | int(11) | | | Tidak | *Tidak ada* | | AUTO_INCREMENT | Ubah Hapus Lainnya |
| 2 | nama_departemen | varchar(120) | latin1_swedish_ci | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 3 | id_fakultas_departemen | int(11) | | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |

Pilih Semua *Dengan pilihan:* Jelajahi Ubah Hapus Utama Unik Indeks Teks penuh Add to central columns
Remove from central columns

Cetak Usulkan struktur tabel Lacak tabel Move columns Normalisasi
Tambahkan 1 kolom setelah id_fakultas_departemen Kirim

Indeks

| Tindakan | Nama kunci | Jenis | Unik | Dipadatkan | Kolom | Kardinalitas | Penyortiran | Tak Ternilai | Komentar |
|---|---|---|---|---|---|---|---|---|---|
| Ubah Hapus | PRIMARY | BTREE | Ya | Tidak | id_departemen | 27 | A | Tidak | |
| Ubah Hapus | fk_id_fakultas | BTREE | Tidak | Tidak | id_fakultas_departemen | 27 | A | Tidak | |

*Gambar 4.3 Implementasi Tabel Departemen*

Setelah tabel dibuat, langkah selanjutnya yaitu memasukkan data departemen kedalam tabel. Sehingga tabel memiliki data seperti yang ditunjukkan pada Gambar 4.4.

| | id_departemen | nama_departemen | id_fakultas_departemen |
|---|---|---|---|
| Ubah Salin Hapus | 1 | S1 MATEMATIKA | 1 |
| Ubah Salin Hapus | 2 | S1 TEKNIK MESIN | 2 |
| Ubah Salin Hapus | 3 | S1 TEKNIK BIOMEDIK | 3 |
| Ubah Salin Hapus | 4 | S1 SISTEM INFORMASI | 4 |
| Ubah Salin Hapus | 5 | S1 TEKNIK MATERIAL | 2 |
| Ubah Salin Hapus | 6 | S1 TEKNIK KOMPUTER | 3 |
| Ubah Salin Hapus | 7 | S1 TEKNIK SISTEM PERKAPALAN | 5 |
| Ubah Salin Hapus | 8 | S1 TEKNIK FISIKA | 2 |
| Ubah Salin Hapus | 9 | S1 TEKNIK KIMIA | 2 |
| Ubah Salin Hapus | 10 | S1 STATISTIKA | 1 |
| Ubah Salin Hapus | 11 | S1 TEKNIK INFORMATIKA | 4 |
| Ubah Salin Hapus | 12 | S1 TEKNIK ELEKTRO | 3 |
| Ubah Salin Hapus | 13 | S1 TEKNIK LINGKUNGAN | 6 |
| Ubah Salin Hapus | 14 | S1 FISIKA | 7 |

*Gambar 4.4 Contoh Data Departemen*

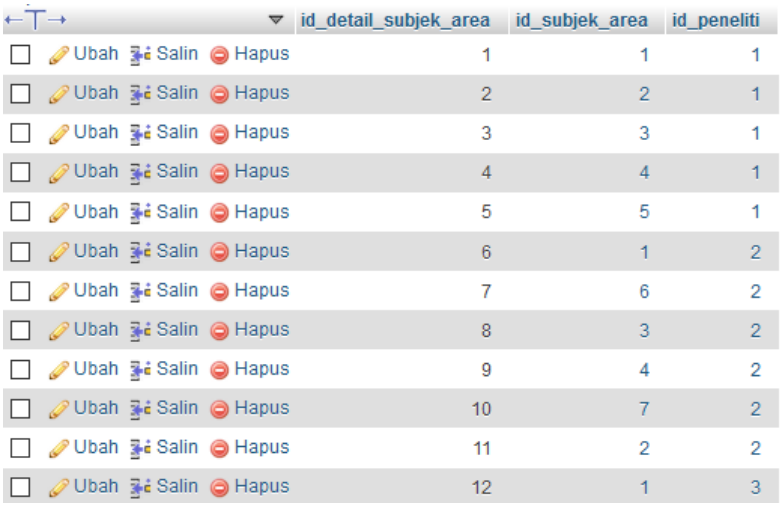
### 4.2.3 Implementasi Tabel Detail Subjek Area

Implementasi tabel detail subjek area dilakukan dengan membuat tabel dengan nama detail_subjek_area dengan atribut id_detail_subjek_area sebagai *primary key* dengan tipe data int(11), id_subjek_area sebagai *foreign key* yang mereferensi ke tabel subjek area dengan tipe data int(11), id_peneliti dengan tipe data

int(11) sebagai *foreign key* yang mereferensi ke tabel peneliti. Implementasi tabel detail_subjek_area dapat dilihat pada Gambar 4.5.


*Gambar 4.5 Implementasi Tabel Detail Subjek Area*

Setelah tabel dibuat, langkah selanjutnya yaitu memasukkan data afiliasi kedalam tabel. Sehingga tabel memiliki data seperti yang ditunjukkan pada Gambar 4.6.




*Gambar 4.6 Contoh Data Detail Subjek Area*

### 4.2.4 Implementasi Tabel Fakultas

Implementasi tabel fakultas dilakukan dengan membuat tabel dengan nama fakultas dengan atribut id_fakultas sebagai *primary key* dengan tipe data int(11) dan nama_fakultas dengan tipe data varchar(120). Implementasi tabel fakultas dapat dilihat pada Gambar 4.7.


*Gambar 4.7 Implementasi Tabel Fakultas*

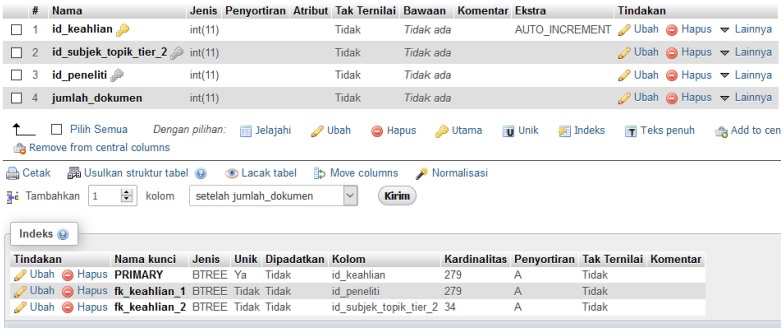
Setelah tabel dibuat, langkah selanjutnya yaitu memasukkan data fakultas kedalam tabel. Sehingga tabel memiliki data seperti yang ditunjukkan pada Gambar 4.8.


*Gambar 4.8 Contoh Data Fakultas*

### 4.2.5 Implementasi Tabel Keahlian

Implementasi tabel keahlian dilakukan dengan membuat tabel dengan nama keahlian dengan atribut id_keahlian sebagai *primary key* dengan tipe data int(11), id_subjek_topik_tier_2

sebagai *foreign key* yang mereferensi ke tabel subjek_topik_tier_2 dengan tipe data int(11), id_peneliti sebagai *foreign key* yang mereferensi ke tabel peneliti dengan tipe data int(11) dan jumlah_dokumen dengan tipe data int(11). Implementasi tabel keahlian dapat dilihat pada Gambar 4.9.




*Gambar 4.9 Implementasi Tabel Keahlian*

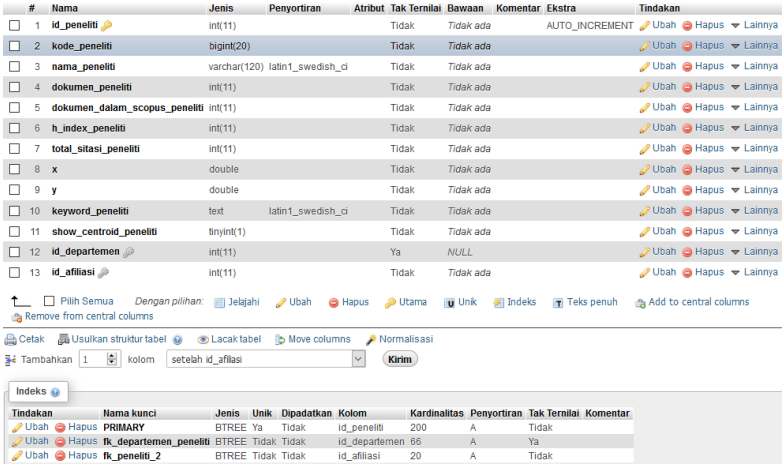
Setelah tabel dibuat, langkah selanjutnya yaitu memasukkan data keahlian kedalam tabel. Sehingga tabel memiliki data seperti yang ditunjukkan pada Gambar 4.10.


*Gambar 4.10 Contoh Data Keahlian*

### 4.2.6 Implementasi Tabel Peneliti

Implementasi tabel keahlian dilakukan dengan membuat tabel dengan nama keahlian dengan atribut id_peneliti sebagai *primary key* dengan tipe data int(11), kode_peneliti dengan tipe

data bigint(11), nama_peneliti dengan tipe data varchar(120), dokumen_peneliti dengan tipe data int(11), h_index_peneliti dengan tipe data int(11), total_sitasi_peneliti dengan tipe data int(11), *x* dengan tipe data double, *y* dengan tipe data double, keyword_peneliti dengan tipe data text, show_centroid_peneliti menggunakan tipe data tinyint(1), id_departemen *foreign key* yang mereferensi ke tabel departemen dengan tipe data int(11) dan id_afiliasi sebagai *foreign key* yang mereferensi ke tabel afiliasi dengan tipe data int(11). Implementasi tabel peneliti dapat dilihat pada Gambar 4.11.



| # | Nama | Jenis | Penyortiran | Atribut | Tak Ternilai | Bawaan | Komentar | Ekstra | Tindakan |
|---|---|---|---|---|---|---|---|---|---|
| 1 | id_peneliti | int(11) | | | Tidak | *Tidak ada* | | AUTO_INCREMENT | Ubah Hapus Lainnya |
| 2 | kode_peneliti | bigint(20) | | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 3 | nama_peneliti | varchar(120) | latin1_swedish_ci | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 4 | dokumen_peneliti | int(11) | | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 5 | dokumen_dalam_scopus_peneliti | int(11) | | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 6 | h_index_peneliti | int(11) | | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 7 | total_sitasi_peneliti | int(11) | | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 8 | x | double | | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 9 | y | double | | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 10 | keyword_peneliti | text | latin1_swedish_ci | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 11 | show_centroid_peneliti | tinyint(1) | | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |
| 12 | id_departemen | int(11) | | | Ya | *NULL* | | | Ubah Hapus Lainnya |
| 13 | id_afiliasi | int(11) | | | Tidak | *Tidak ada* | | | Ubah Hapus Lainnya |

Pilih Semua *Dengan pilihan:* Jelajahi Ubah Hapus Utama Unik Indeks Teks penuh Add to central columns Remove from central columns

Cetak Usulkan struktur tabel Lacak tabel Move columns Normalisasi

Tambahkan 1 kolom setelah id_afiliasi Kirim

Indeks

| Tindakan | Nama kunci | Jenis | Unik | Dipadatkan | Kolom | Kardinalitas | Penyortiran | Tak Ternilai | Komentar |
|---|---|---|---|---|---|---|---|---|---|
| Ubah Hapus | PRIMARY | BTREE | Ya | Tidak | id_peneliti | 200 | A | Tidak | |
| Ubah Hapus | fk_departemen_peneliti | BTREE | Tidak | Tidak | id_departemen | 66 | A | Ya | |
| Ubah Hapus | fk_peneliti_2 | BTREE | Tidak | Tidak | id_afiliasi | 20 | A | Tidak | |

*Gambar 4.11 Implementasi Tabel Peneliti*

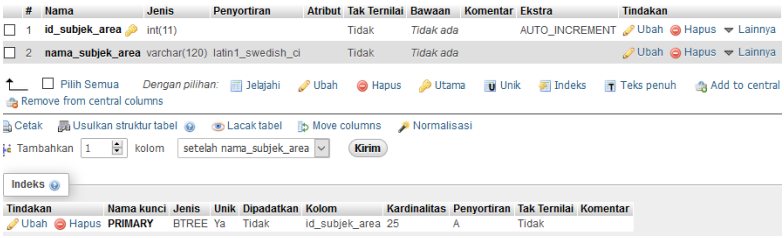
Setelah tabel dibuat, langkah selanjutnya yaitu memasukkan data peneliti kedalam tabel. Sehingga tabel memiliki data seperti yang ditunjukkan pada Gambar 4.12.

| id_peneliti | kode_peneliti | nama_peneliti | dokumen_peneliti | dokumen_dalam_scopus_peneliti | h_index_peneliti | total_sitasi_peneliti | x | y | keyword_peneliti |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 10641836800 | Subchan, Subchan | 42 | 22 | 5 | 155 | 11.08873663 | -11.00289261 | estimation,control,d |
| 2 | 12761914600 | Pramujati, Bambang | 86 | 44 | 7 | 139 | 11.74061567 | -5.705990654 | controller,part,surfac |
| 3 | 12774830800 | Pramono, Agus Sigit | 30 | 15 | 3 | 36 | 14.90394307 | -4.018126063 | study,simulation,var |
| 4 | 13205413800 | Arifin, Achmad | 40 | 21 | 5 | 82 | 4.607845503 | -5.635650208 | processing,charact |
| 5 | 15026300000 | Er, Mahendrawathi | 28 | 16 | 5 | 71 | 1.502382915 | -21.50821012 | movement,analysis, |
| 6 | 15072626100 | Ardhyananta, Hosta | 40 | 21 | 6 | 194 | 16.17482775 | 7.877630114 | performance,enhan |
| 7 | 16026156200 | Wulandari, Diah Puspito | 20 | 12 | 4 | 34 | 2.67657784 | -5.218235449 | gamelan,music,ons |
| 8 | 16041627800 | Alasiry, Ali Husein | 22 | 12 | 2 | 16 | 13.18972361 | -19.51363217 | implementation,tech |
| 9 | 16303005900 | Ariana, I. Made | 16 | 8 | 1 | 3 | 19.56816899 | -9.744566654 | water,plate,collector, |
| 10 | 16550792300 | Sardjono, Tri Arief | 50 | 33 | 5 | 91 | -0.077819598 | -4.043530293 | curvature,determina |
| 11 | 16636376400 | Guntur, Harus Laksana | 34 | 17 | 3 | 36 | 17.12046106 | -4.837950262 | control,vibration,isol |
| 12 | 16644746700 | Biyanto, Totok Ruki | 60 | 54 | 5 | 68 | 18.7200082 | -9.103724628 | design,plant,control |
| 13 | 22986633400 | Susanto, Tony Dwi | 26 | 13 | 4 | 80 | -9.381336106 | -23.97286356 | level,model,governn |

*Gambar 4.12 Contoh Data Detail Subjek Area*

### 4.2.7 Implementasi Tabel Subjek Area

Implementasi tabel subjek area dilakukan dengan membuat tabel dengan nama subjek_area dengan atribut id_subjek_area sebagai *primary key* dengan tipe data int(11) dan nama_subjek_area dengan tipe data varchar(120). Implementasi tabel subjek area dapat dilihat pada Gambar 4.13.




*Gambar 4.13 Implementasi Tabel Subjek Area*

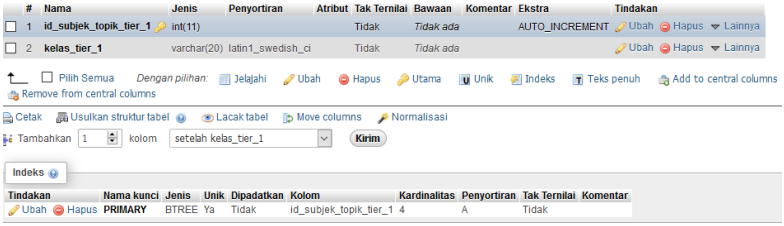
Setelah tabel dibuat, langkah selanjutnya yaitu memasukkan data subjek area kedalam tabel. Sehingga tabel memiliki data seperti yang ditunjukkan pada Gambar 4.14.

| | id_subjek_area | nama_subjek_area |
|---|---|---|
| Ubah Salin Hapus | 1 | Engineering |
| Ubah Salin Hapus | 2 | Mathematics |
| Ubah Salin Hapus | 3 | Physics and Astronomy |
| Ubah Salin Hapus | 4 | Computer Science |
| Ubah Salin Hapus | 5 | Decision Sciences |
| Ubah Salin Hapus | 6 | Materials Science |
| Ubah Salin Hapus | 7 | Chemical Engineering |
| Ubah Salin Hapus | 8 | Social Sciences |
| Ubah Salin Hapus | 9 | Energy |
| Ubah Salin Hapus | 10 | Business, Management and Accounting |
| Ubah Salin Hapus | 11 | Chemistry |

*Gambar 4.14 Contoh Data Subjek Area*

### 4.2.8 Implementasi Tabel Subjek Topik Tier 1

Implementasi tabel subjek topik tier 1 dilakukan dengan membuat tabel dengan nama subjek_topik_tier_1 dengan atribut id_subjek_topik_tier_1 sebagai *primary key* dengan tipe data int(11) dan kelas_tier_1 dengan tipe data varchar(20). Implementasi tabel subjek topik tier 1 dapat dilihat pada Gambar 4.15.



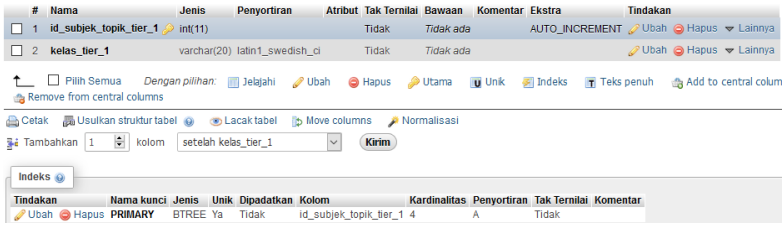

*Gambar 4.15 Implementasi Tabel Subjek Topik Tier 1*

Setelah tabel dibuat, langkah selanjutnya yaitu memasukkan data subjek topik tier 1 kedalam tabel. Sehingga tabel memiliki data seperti yang ditunjukkan pada Gambar.


*Gambar 4.16 Contoh Data Subjek Subjek Topik Tier 1*

### 4.2.9 Implementasi Tabel Subjek Topik Tier 2

Implementasi tabel subjek topik tier 2 dilakukan dengan membuat tabel dengan nama subjek_topik_tier_2 dengan atribut id_subjek_topik_tier_2 sebagai *primary key* dengan tipe data int(11), kelas_tier_2 dengan tipe data varchar(20), dan id_subjek_topik_tier_1 sebagai *foreign key* yang mereferensi ke tabel subjek_topik_tier_1 dengan tipe data int(11). Implementasi tabel subjek topik tier 2 dapat dilihat pada Gambar 4.17.




*Gambar 4.17 Implementasi Tabel Subjek Topik Tier 1*

Setelah tabel dibuat, langkah selanjutnya yaitu memasukkan data subjek topik tier 2 kedalam tabel. Sehingga tabel memiliki data seperti yang ditunjukkan pada Gambar 4.18.

| | id_subjek_topik_tier_2 | kelas_tier_2 | id_subjek_topik_tier_1 |
|---|---|---|---|
| Ubah Salin Hapus | 1 | MEDI | 1 |
| Ubah Salin Hapus | 2 | NURS | 1 |
| Ubah Salin Hapus | 3 | DENT | 1 |
| Ubah Salin Hapus | 4 | VETE | 1 |
| Ubah Salin Hapus | 5 | HEAL | 1 |
| Ubah Salin Hapus | 6 | AGRI | 2 |
| Ubah Salin Hapus | 7 | NEUR | 2 |
| Ubah Salin Hapus | 8 | IMMU | 2 |
| Ubah Salin Hapus | 9 | BIOC | 2 |
| Ubah Salin Hapus | 10 | PHAR | 2 |

*Gambar 4.18 Contoh Data Subjek Topik Tier 2*

## 4.3 Implementasi Aplikasi Realitas Virtual

Dalam tahap implementasi aplikasi Realitas Virtual untuk visualisasi data peneliti ini dibagi menjadi dua bagian yaitu implementasi aplikasi *client* dan implementasi aplikasi web server. Implementasi aplikasi *client* menggunakan bahasa pemrograman C# dengan menggunakan Unity3D. Sedangkan implementasi aplikasi web server menggunakan bahasa pemrograman Python menggunakan kerangka kerja Flask.

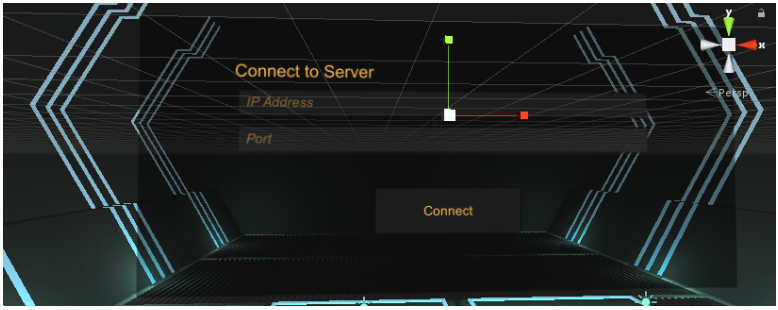
### 4.3.1 Implementasi Aplikasi *Client*

Bagian ini akan menjelaskan mengenai implementasi aplikasi Realitas Virtual untuk visualisasi data peneliti pada sisi *client*. Dalam aplikasi *client* ini memiliki beberapa tampilan yaitu tampilan beranda, peneliti detail peneliti, subjek penelitian, detail subjek penelitian, afiliasi, detail afiliasi dan diagram *scatter*.

#### 4.3.1.1 Implementasi Antarmuka Panel Konfigurasi

Panel konfigurasi adalah panel yang digunakan untuk konfigurasi koneksi ke server dengan mengkonfigurasikan alamat

*IP* dan juga *Port* agar memudahkan koneksi ke server. Tampilan panel konfigurasi dapat dilihat pada Gambar 4.19.




*Gambar 4.19 Panel untuk Icon Afiliasi*

Panel konfigurasi mengimplementasikan ***Class Config*** adalah class yang digunakan untuk konfigurasi koneksi ke server dengan mengkonfigurasikan alamat *IP* dan juga *Port*. *Class* ini memiliki fungsi ***SetUrl()*** yang terletak pada Kode Sumber 4.1 baris 11. Fungsi ***SetUrl()*** digunakan untuk mengatur URL. Kemudian terdapat fungsi ***SetPort()*** pada Kode Sumber 4.1 baris 15 untuk mengatur *port* dan ***GetWebAPI()*** pada Kode Sumber 4.1 baris 19 untuk mendapatkan konfigurasi untuk koneksi ke server. Pemanggilan ***Class Config*** ditunjukkan pada Kode Sumber 4.2.

```
1.  using System.Collections;
2.  using System.Collections.Generic;
3.  using UnityEngine;
4.
5.  public class Config
6.  {
7.    private string URL = "127.0.0.1";
8.    private string port = "5000";
9.    private string webAPI;
10.
11.   public void SetUrl(string URL)
12.   {
13.     this.URL = URL;
```

```
14.   }
15.   public string GetUrl()
16.   {
17.     return this.URL;
18.   }
19.   public void SetPort(string port)
20.   {
21.     this.port = port;
22.   }
23.   public string GetPort()
24.   {
25.     return this.port;
26.   }
27.   public string GetWebAPI()
28.   {
29.     string prefix = "https://";
30.     this.webAPI = prefix + URL + ":" + port;
31.     return this.webAPI;
32.   }
33. }
```

*Kode Sumber 4.1 Class Config*

```
1.  public void SetURL()
2.  {
3.    Config config = new Config();
4.    config.SetUrl(ip.text);
5.    config.SetPort(port.text);
6.    ApplyURL(config);
7.    configPanel.SetActive(false);
8.  }
```

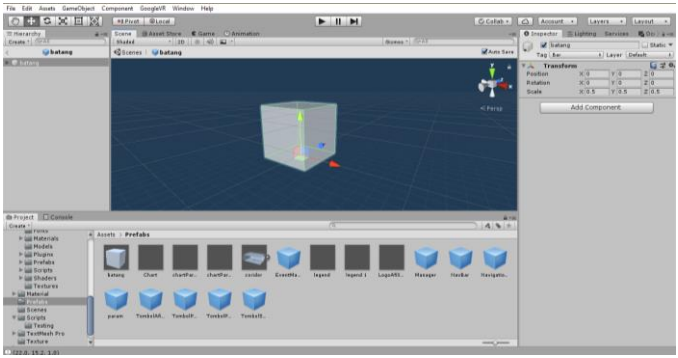
*Kode Sumber 4.2 Pemanggilan Class Config*

#### 4.3.1.2 Implementasi Antarmuka Beranda

Didalam beranda memiliki beberapa objek dasar yaitu *cube*, *panel*, *plane* dan *text* yang dijadikan prefab. Objek-objek ini akan diinstansiasi secara dinamis melalui script sesuai dengan data yang diminta untuk beranda. Data yang akan ditampilkan

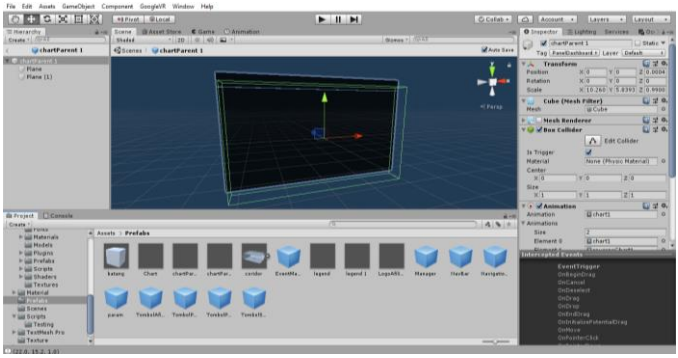
diberanda antara lain adalah rata-rata h-index, total dokumen, total dokumen scopus, dan total sitasi berdasarkan afiliasi. Data ini dikirimkan oleh server kemudian ditampilkan kedalam beranda.

Objek *cube* pada beranda digunakan untuk diagram batang dan juga warna untuk keterangan. Objek *cube* untuk diagram batang akan ditransformasi baik ukuran dan tinggi serta akan diubah warnanya sesuai dengan data dari server. Kemudian objek cube untuk keterangan akan ditransformasi berdasarkan posisi serta diubah warnanya sesuai data dari server. Objek *cube* dapat dilihat pada Gambar 4.20.

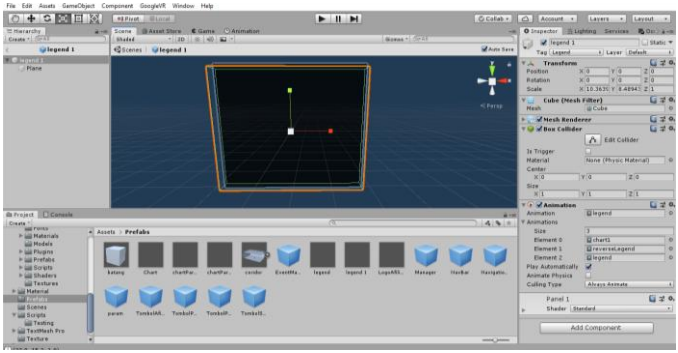


*Gambar 4.20 Cube untuk Beranda*

Objek *panel* pada beranda digunakan untuk tempat diagram batang dan juga tempat untuk keterangan. Objek *cube* akan ditransformasikan menjadi *child* dari panel tersebut keterangan. Objek panel diagram batang dapat dilihat pada Gambar 4.21. Sedangkan objek *panel* keterangan dapat dilihat pada Gambar 4.22.

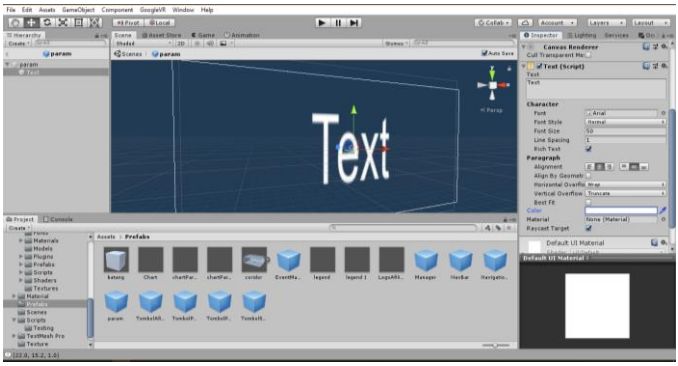


*Gambar 4.21 Panel untuk Diagram Batang pada Beranda*



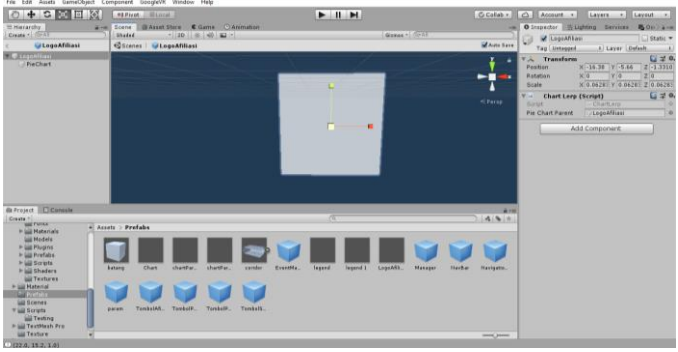
*Gambar 4.22 Panel untuk Keterangan Diagram Batang*

Objek *text* pada beranda digunakan untuk parameter diagram batang serta nama afiliasi. *Text* ini akan diinstansiasi dan diubah sesuai data yang akan ditampilkan. Objek *text* dapat dilihat pada Gambar 4.23.




*Gambar 4.23 Text untuk Beranda*

Objek *plane* pada beranda digunakan untuk icon afiliasi. *Plane* ini akan diinstansiasi dan diubah sesuai data yang akan ditampilkan sehingga memiliki *texture* berupa *icon* afiliasi. Objek plane dapat dilihat pada Gambar 4.24.



*Gambar 4.24 Panel untuk Icon Afiliasi*

Antarmuka beranda memiliki beberapa fungsi yaitu fungsi ***Dashboard()*** pada Kode Sumber 4.3, fungsi ***DashboardPanelEvent()*** pada Kode Sumber 4.4, fungsi ***ReRataHIndex()*** pada Kode Sumber 4.5, fungsi ***TotalDokumen()*** pada Kode Sumber 4.6, fungsi ***TotalDokumenScopus()*** pada Kode

Sumber 4.7, fungsi ***TotalSitasi()*** pada Kode Sumber 4.8 dan fungsi ***Keterangan()*** pada Kode Sumber 4.9.

***Dashboard()*** adalah fungsi yang digunakan untuk meminta data beranda ke server menggunakan URL https://<alamat_ip>:<port>/ dan server akan mengirimkan kembali data berupa objek JSON.

Setelah data didapatkan, fungsi ***Dashboard()*** akan memanggil fungsi ***ReratаHIndex()***, ***TotalDokumen()***, ***TotalDokumenScopus()***, ***TotalSitasi()*** dan ***Keterangan()*** untuk menampilkan data. Fungsi ***Dashboard()*** dapat dilihat pada Kode Sumber 4.3.

```
1.  public void Dashboard()
2.    {
3.      tombolBeranda.GetComponent<Button>().interactab
    le = false;
4.      tombolPeneliti.GetComponent<Button>().interacta
    ble = false;
5.      tombolAfiliasi.GetComponent<Button>().interacta
    ble = false;
6.      tombolSubjekPenelitian.GetComponent<Button>().i
    nteractable = false;
7.      tombolDiagramScatter.GetComponent<Button>().int
    eractable = false;
8.      connectionMessage.text = "Connecting...";
9.      connectionMessagePanel.SetActive(true);
10.     RequestHandler requestHandler = new RequestHand
    ler();
11.     requestHandler.URL = URL;
12.     StartCoroutine(requestHandler.RequestData((resu
    lt) => {
13.       ReratаHIndex(result);
14.       TotalDokumen(result);
15.       TotalDokumenScopus(result);
16.       TotalSitasi(result);
17.       Keterangan(result);
18.       navBarPanel.SetActive(true);
19.       connectionMessagePanel.SetActive(false);
```

```
      tombolBeranda.GetComponent<Button>().interactable = true;
      tombolPeneliti.GetComponent<Button>().interactable = true;
      tombolAfiliasi.GetComponent<Button>().interactable = true;
      tombolSubjekPenelitian.GetComponent<Button>().interactable = true;
      tombolDiagramScatter.GetComponent<Button>().interactable = true;
    }, (error) => {
      if (error != "")
      {
        retryMessage.text = error;
        retry.SetActive(true);
        connectionMessagePanel.SetActive(false);
      }
    }));
  }
```

*Kode Sumber 4.3 Fungsi Dashboard*

***DashboardPanelEvent()*** adalah fungsi yang digunakan untuk berinteraksi dengan panel diagram batang. Fungsi ***DashboardPanelEvent()*** menggunakan *raycast* untuk berinteraksi dengan panel diagram batang. Ketika pointer menyentuh panel diagram batang dan mengklik kiri *mouse*, panel yang akan melakukan *sliding* dan menyembunyikan panel diagram batang lain. ***DashboardPanelEvent()*** dapat dilihat pada Kode Sumber 4.4.

```
public void DashboardPanelEvent()
  {
    if (Input.GetMouseButtonDown(0) && dashboardSelectable == true)
    {
      RaycastHit hit;
      Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);
      if (Physics.Raycast(pointer.transform.position, pointer.transform.forward, out hit, 100.0f))
```

```
8.        {
9.          if (hit.transform != null)
10.         {
11.           if (hit.transform.gameObject.tag == "Pane
    lDashboard")
12.           {
13.             GameObject[] panelDashboard = GameObjec
    t.FindGameObjectsWithTag("PanelDashboard");
14.             GameObject[] panelLegend = GameObject.F
    indGameObjectsWithTag("Legend");
15.             kembali.SetActive(true);
16.             dashboardSelectable = false;
17.             tombolBeranda.SetActive(false);
18.             tombolPeneliti.SetActive(false);
19.             tombolSubjekPenelitian.SetActive(false)
    ;
20.             tombolAfiliasi.SetActive(false);
21.             tombolDiagramScatter.SetActive(false);

22.             int i = 0;
23.
24.             foreach (GameObject panel in panelDashb
    oard)
25.             {
26.               if (panel != hit.transform.gameObject
     && panel.activeSelf == true && panel.tag == "Panel
    Dashboard")
27.               {
28.                 inactivePanel[i] = panel;
29.                 panel.SetActive(false);
30.                 i += 1;
31.               }
32.               else
33.               {
34.                 if (panel.name == "chartParent 1(Cl
    one)")
35.                 {
36.
37.                   activePanel.Add(panel);
38.
39.                   foreach (GameObject legend in pan
    elLegend)
```

```
40.                     {
41.                       activePanel.Add(legend);
42.                     }
43.
44.                     foreach (GameObject active in act
    ivePanel)
45.                     {
46.                       if(active.tag == "PanelDashboar
    d")
47.                       {
48.                         defaultPosition = active.tran
    sform.position;
49.                         defaultRotation = active.tran
    sform.rotation;
50.                         defaultScaleChart = active.tr
    ansform.localScale;
51.                         Quaternion endRotation = Quat
    ernion.Euler(0, 0, 0);
52.                         Debug.Log("Default" + endMark
    er);
53.                         Debug.Log("Default" + endRota
    tion);
54.                         Debug.Log("Default" + endScal
    eLegend);
55.                         slideIn = DashboardLerp(activ
    e, endRotation, endMarker, endScaleChart);
56.                         StartCoroutine(slideIn);
57.                       }
58.                       else
59.                       {
60.                         defaultScaleLegend = active.t
    ransform.localScale;
61.                         defaultPositionLegend = activ
    e.transform.position;
62.                         defaultRotationLegend = Quate
    rnion.Euler(0, 0, 0);
63.                         Quaternion endRotation = Quat
    ernion.Euler(0, 0, 0);
64.                         Debug.Log("Default Legend" +
    endMarkerLegend);
65.                         Debug.Log("Default Legend" +
    endScaleLegend);
```

```
                        Debug.Log("Default Legend" +
endRotation);
                        slideIn = DashboardLerp(activ
e, endRotation, endMarkerLegend, endScaleLegend);
                        StartCoroutine(slideIn);
                    }
                }

                }
            }
          }
          i = 0;
        }
      }
    }
  }
```

*Kode Sumber 4.4 Fungsi Dashboard Panel Event*

***ReratаHIndex()*** adalah fungsi yang digunakan untuk menginstansiasi objek-objek dasar untuk diagram batang rata-rata h-index. Objek-objek tersebut akan ditransformasi kedalam posisi yang telah ditentukan. Fungsi ***ReratаHIndex()*** dapat dilihat pada Kode Sumber 4.5.

```
public void ReratаHIndex(RawData rawData)
{
    float posX = -0.45f;
    float posY = -0.45f;
    float posZ = 0f;
    GameObject parent = (GameObject)Instantiate(par
entBar);
    GameObject judul = (GameObject)Instantiate(para
mPrefab);
    parent.transform.GetChild(0).GetComponent<Rende
rer>().material.renderQueue = 2999;
    parent.transform.GetChild(1).GetComponent<Rende
rer>().material.renderQueue = 2999;
    judul.GetComponentInChildren<Text>().text = "Ra
ta-Rata H Index";
```

```
    judul.GetComponentInChildren<Text>().color = warnaJudul;
    judul.GetComponentInChildren<Text>().alignment = TextAnchor.MiddleLeft;
    judul.GetComponentInChildren<Text>().fontSize = fontSizeJudul;
    judul.transform.SetParent(parent.transform);
    judul.transform.localPosition = judulPos;
    judul.transform.GetChild(0).GetComponent<RectTransform>().SetSizeWithCurrentAnchors(RectTransform.Axis.Horizontal, 1000f);
    judul.transform.GetChild(0).GetComponent<RectTransform>().localScale = new Vector3(judul.transform.GetChild(0).GetComponent<RectTransform>().localScale.x + 0.5f, judul.transform.GetChild(0).GetComponent<RectTransform>().localScale.y, judul.transform.GetChild(0).GetComponent<RectTransform>().localScale.z);
    int i = 0;
    foreach(var raw in (dynamic)(rawData.data[0].data_dashboard[0].rerata_h_index))
    {
        GameObject chartBar = (GameObject)Instantiate(barPrefab);
        chartBar.GetComponentInChildren<Renderer>().material.color = colorList[(raw.id_afiliasi-1)%13];
        chartBar.transform.SetParent(parent.transform);
        chartBar.transform.localPosition = new Vector3(posX, posY, posZ);
        chartBar.transform.localScale = new Vector3(chartBar.transform.localScale.x, raw.rata_rata_h_index, chartBar.transform.localScale.z);
            GameObject logo = (GameObject)Instantiate(logoAfiliasi);
        logo.transform.SetParent(parent.transform);

        logo.transform.localPosition = new Vector3(posX, posY, -0.5f);
        Renderer rendererLogo = logo.transform.GetChild(0).GetComponent<Renderer>();
```

```
30.         rendererLogo.material.mainTexture = Texture
    Encoder(raw.logo_afiliasi);
31.         posX += 0.075f;
32.         i += 1;
33.     }
34.
35.     float higherChild = 0f;
36.     foreach(Transform child in parent.transform)
37.     {
38.         if (higherChild < child.transform.localScal
    e.y)
39.         {
40.             higherChild = child.transform.localScal
    e.y;
41.         }
42.         }
43.     List<Vector3> trans = new List<Vector3>();
44.     foreach (Transform child in parent.transform)
45.     {
46.         if(child.tag == "Bar")
47.         {
48.             child.transform.localScale = new Vector
    3(child.transform.localScale.x, (child.transform.lo
    calScale.y * 0.75f / higherChild) + 0.02f, child.tr
    ansform.localScale.z);
49.             var x = new Vector3(child.transform.loc
    alPosition.x, child.transform.localScale.y - 0.4f,
    child.transform.localPosition.z);
50.             trans.Add(x);
51.         }
52.     }
53.
54.     int transIndex = 0;
55.     foreach (var raw in (dynamic)(rawData.data[0].d
    ata_dashboard[0].rerata_h_index))
56.     {
57.         GameObject param = (GameObject)Instantiate(
    paramPrefab);
58.         param.GetComponentInChildren<Text>().text =
     raw.rata_rata_h_index.ToString();
59.         param.GetComponentInChildren<Text>().color
    = Color.white;
```

```
60.         param.transform.SetParent(parent.transform)
    ;
61.         param.transform.localPosition = trans[trans
    Index];
62.         param.transform.localScale = new Vector3(pa
    ram.transform.localScale.x * 1.5f, param.transform.
    localScale.y * 1.5f, param.transform.localScale.z *
     1.5f);
63.         transIndex += 1;
64.     }
65.     parent.transform.localScale = new Vector3(paren
    t.transform.localScale.x * 3 / 4, parent.transform.
    localScale.y * 3 / 4, parent.transform.localScale.z
     * 3 / 4);
66.     parent.transform.localPosition = new Vector3(-
    18f, 5.5f, 0);
67.     parent.transform.localRotation = Quaternion.Eul
    er(0, rotationYHIndex, 0);
68.
69.     EventTrigger panelOnTrigger = parent.GetCompone
    nt<EventTrigger>();
70.     EventTrigger.Entry newTriggerClick = new EventT
    rigger.Entry();
71.     newTriggerClick.eventID = EventTriggerType.Poin
    terClick;
72.     newTriggerClick.callback.AddListener((data) =>
    { DashboardPanelEvent(); });
73.     panelOnTrigger.triggers.Add(newTriggerClick);
74. }
```

*Kode Sumber 4.5 Fungsi Rerata H Index*

***TotalDokumen()*** adalah fungsi yang digunakan untuk menginstansiasi objek-objek dasar untuk diagram batang total dokumen. Objek-objek tersebut akan ditransformasi kedalam posisi yang telah ditentukan. Fungsi ***TotalDokumen()*** dapat dilihat pada Kode Sumber 4.6.

```
1.  public void TotalDokumen(RawData rawData)
2.      {
3.
```

```
4.          GameObject parent = (GameObject)Instantiate
    (parentBar);
5.          GameObject judul = (GameObject)Instantiate(
    paramPrefab);
6.          parent.transform.GetChild(0).GetComponent<R
    enderer>().material.renderQueue = 2999;
7.          parent.transform.GetChild(1).GetComponent<R
    enderer>().material.renderQueue = 2999;
8.          judul.GetComponentInChildren<Text>().text =
     "Total Dokumen";
9.          judul.GetComponentInChildren<Text>().color
    = warnaJudul;
10.         judul.GetComponentInChildren<Text>().alignm
    ent = TextAnchor.MiddleLeft;
11.         judul.GetComponentInChildren<Text>().fontSi
    ze = fontSizeJudul;
12.         judul.transform.SetParent(parent.transform)
    ;
13.         judul.transform.localPosition = judulPos;
14.         judul.transform.GetChild(0).GetComponent<Re
    ctTransform>().SetSizeWithCurrentAnchors(RectTransf
    orm.Axis.Horizontal, 1000f);
15.         judul.transform.GetChild(0).GetComponent<Re
    ctTransform>().localScale = new Vector3(judul.trans
    form.GetChild(0).GetComponent<RectTransform>().loca
    lScale.x + 0.5f, judul.transform.GetChild(0).GetCom
    ponent<RectTransform>().localScale.y, judul.transfo
    rm.GetChild(0).GetComponent<RectTransform>().localS
    cale.z);
16.         int i = 0;
17.         foreach (var raw in (dynamic)(rawData.data[
    0].data_dashboard[0].total_dokumen))
18.         {
19.             GameObject chartBar = (GameObject)Insta
    ntiate(barPrefab);
20.             chartBar.GetComponentInChildren<Rendere
    r>().material.color = colorList[(raw.id_afiliasi -
    1) % 13];
21.             chartBar.transform.SetParent(parent.tra
    nsform);
22.             chartBar.transform.localPosition = new
    Vector3(posX, posY, posZ);
```

```
23.             chartBar.transform.localScale = new Vec
    tor3(chartBar.transform.localScale.x, raw.total_dok
    umen, chartBar.transform.localScale.z);
24.             GameObject logo = (GameObject)Instantia
    te(logoAfiliasi);
25.             logo.transform.SetParent(parent.transfo
    rm);
26.             logo.transform.localPosition = new Vect
    or3(posX, posY, -0.5f);
27.
28.             Renderer rendererLogo = logo.transform.
    GetChild(0).GetComponent<Renderer>();
29.             rendererLogo.material.mainTexture = Tex
    tureEncoder(raw.logo_afiliasi);
30.             posX += 0.075f;
31.             i += 1;
32.         }
33.
34.         float higherChild = 0f;
35.         foreach (Transform child in parent.transfor
    m)
36.         {
37.             if (child.tag == "Bar")
38.             {
39.                 if (higherChild < child.transform.l
    ocalScale.y)
40.                 {
41.                     higherChild = child.transform.l
    ocalScale.y;
42.                 }
43.             }
44.         }
45.         List<Vector3> trans = new List<Vector3>();

46.         foreach (Transform child in parent.transfor
    m)
47.         {
48.             if (child.tag == "Bar")
49.             {
50.                 child.transform.localScale = new Ve
    ctor3(child.transform.localScale.x, (child.transfor
```

```
    m.localScale.y * 0.75f / higherChild) + 0.02f, chil
    d.transform.localScale.z);
51.                 var x = new Vector3(child.transform
    .localPosition.x, child.transform.localScale.y - 0.
    4f, child.transform.localPosition.z);
52.                 trans.Add(x);
53.             }
54.         }
55.
56.         int transIndex = 0;
57.         foreach (var raw in (dynamic)(rawData.data[
    0].data_dashboard[0].total_dokumen))
58.         {
59.             GameObject param = (GameObject)Instanti
    ate(paramPrefab);
60.             param.GetComponentInChildren<Text>().te
    xt = raw.total_dokumen.ToString();
61.             param.GetComponentInChildren<Text>().co
    lor = Color.white;
62.             param.transform.SetParent(parent.transf
    orm);
63.             param.transform.localPosition = trans[t
    ransIndex];
64.             param.transform.localScale = new Vector
    3(param.transform.localScale.x * 1.5f, param.transf
    orm.localScale.y * 1.5f, param.transform.localScale
    .z * 1.5f);
65.             transIndex += 1;
66.         }
67.         parent.transform.localScale = new Vector3(p
    arent.transform.localScale.x * 3 / 4, parent.transf
    orm.localScale.y * 3 / 4, parent.transform.localSca
    le.z * 3 / 4);
68.         parent.transform.localPosition = new Vector
    3(-18f, -3f, 0);
69.         parent.transform.localRotation = Quaternion
    .Euler(0, rotationYTotalDokumen, 0);
70.
71.         EventTrigger panelOnTrigger = parent.GetCom
    ponent<EventTrigger>();
72.         EventTrigger.Entry newTriggerClick = new Ev
    entTrigger.Entry();
```

```
73.         newTriggerClick.eventID = EventTriggerType.
    PointerClick;
74.         newTriggerClick.callback.AddListener((data)
     => { DashboardPanelEvent(); });
75.         panelOnTrigger.triggers.Add(newTriggerClick
    );
76.     }
```

*Kode Sumber 4.6 Fungsi Total Dokumen*

***TotalDokumenScopus()*** adalah fungsi yang digunakan untuk menginstansiasi objek-objek dasar untuk diagram batang total dokumen scopus. Objek-objek tersebut akan ditransformasi kedalam posisi yang telah ditentukan. Fungsi ***TotalDokumenScopus()*** dapat dilihat pada Kode Sumber 4.7.

```
1.  public void TotalDokumenScopus(RawData rawData)
2.      {
3.
4.          GameObject parent = (GameObject)Instantiate
    (parentBar);
5.          GameObject judul = (GameObject)Instantiate(
    paramPrefab);
6.          parent.transform.GetChild(0).GetComponent<R
    enderer>().material.renderQueue = 2999;
7.          parent.transform.GetChild(1).GetComponent<R
    enderer>().material.renderQueue = 2999;
8.          judul.GetComponentInChildren<Text>().text =
     "Total Dokumen dalam Scopus";
9.          judul.GetComponentInChildren<Text>().color
    = warnaJudul;
10.         judul.GetComponentInChildren<Text>().alignm
    ent = TextAnchor.MiddleLeft;
11.         judul.GetComponentInChildren<Text>().fontSi
    ze = fontSizeJudul;
12.         judul.transform.SetParent(parent.transform)
    ;
13.         judul.transform.localPosition = judulPos;
14.         judul.transform.GetChild(0).GetComponent<Re
    ctTransform>().SetSizeWithCurrentAnchors(RectTransf
    orm.Axis.Horizontal, 1000f);
```

```
15.         judul.transform.GetChild(0).GetComponent<Re
    ctTransform>().localScale = new Vector3(judul.trans
    form.GetChild(0).GetComponent<RectTransform>().loca
    lScale.x + 0.5f, judul.transform.GetChild(0).GetCom
    ponent<RectTransform>().localScale.y, judul.transfo
    rm.GetChild(0).GetComponent<RectTransform>().localS
    cale.z);
16.         int i = 0;
17.         foreach (var raw in (dynamic)(rawData.data[
    0].data_dashboard[0].total_dokumen_scopus))
18.         {
19.             GameObject chartBar = (GameObject)Insta
    ntiate(barPrefab);
20.             chartBar.GetComponentInChildren<Rendere
    r>().material.color = colorList[(raw.id_afiliasi -
    1) % 13];
21.             chartBar.transform.SetParent(parent.tra
    nsform);
22.             chartBar.transform.localPosition = new
    Vector3(posX, posY, posZ);
23.             chartBar.transform.localScale = new Vec
    tor3(chartBar.transform.localScale.x, raw.total_dok
    umen_scopus, chartBar.transform.localScale.z);
24.             GameObject logo = (GameObject)Instantia
    te(logoAfiliasi);
25.             logo.transform.SetParent(parent.transfo
    rm);
26.             logo.transform.localPosition = new Vect
    or3(posX, posY, -0.5f);
27.
28.             Renderer rendererLogo = logo.transform.
    GetChild(0).GetComponent<Renderer>();
29.             rendererLogo.material.mainTexture = Tex
    tureEncoder(raw.logo_afiliasi);
30.             posX += 0.075f;
31.             i += 1;
32.         }
33.
34.         float higherChild = 0f;
35.         foreach (Transform child in parent.transfor
    m)
36.         {
```

```
37.             if (higherChild < child.transform.local
    Scale.y)
38.             {
39.                 higherChild = child.transform.local
    Scale.y;
40.             }
41.         }
42.         List<Vector3> trans = new List<Vector3>();
43.         foreach (Transform child in parent.transfor
    m)
44.         {
45.             if (child.tag == "Bar")
46.             {
47.                 child.transform.localScale = new Ve
    ctor3(child.transform.localScale.x, (child.transfor
    m.localScale.y * 0.75f / higherChild) + 0.02f, chil
    d.transform.localScale.z);
48.                 var x = new Vector3(child.transform
    .localPosition.x, child.transform.localScale.y - 0.
    4f, child.transform.localPosition.z);
49.                 trans.Add(x);
50.             }
51.         }
52.
53.         int transIndex = 0;
54.         foreach (var raw in (dynamic)(rawData.data[
    0].data_dashboard[0].total_dokumen_scopus))
55.         {
56.             GameObject param = (GameObject)Instanti
    ate(paramPrefab);
57.             param.GetComponentInChildren<Text>().te
    xt = raw.total_dokumen_scopus.ToString();
58.             param.GetComponentInChildren<Text>().co
    lor = Color.white;
59.             param.transform.SetParent(parent.transf
    orm);
60.             param.transform.localPosition = trans[t
    ransIndex];
61.             param.transform.localScale = new Vector
    3(param.transform.localScale.x * 1.5f, param.transf
```

```
orm.localScale.y * 1.5f, param.transform.localScale
.z * 1.5f);
            transIndex += 1;
        }
        parent.transform.localScale = new Vector3(p
arent.transform.localScale.x * 3 / 4, parent.transf
orm.localScale.y * 3 / 4, parent.transform.localSca
le.z * 3 / 4);
        parent.transform.localPosition = new Vector
3(18f, 5.5f, 0);
        parent.transform.localRotation = Quaternion
.Euler(0, rotationYTotalDokumenScopus, 0);

        EventTrigger panelOnTrigger = parent.GetCom
ponent<EventTrigger>();
        EventTrigger.Entry newTriggerClick = new Ev
entTrigger.Entry();
        newTriggerClick.eventID = EventTriggerType.
PointerClick;
        newTriggerClick.callback.AddListener((data)
 => { DashboardPanelEvent(); });
        panelOnTrigger.triggers.Add(newTriggerClick
);
    }
```

*Kode Sumber 4.7 Fungsi Total Dokumen Scopus*

***TotalSitasi()*** adalah fungsi yang digunakan untuk menginstansiasi objek-objek dasar untuk diagram batang total sitasi. Objek-objek tersebut akan ditransformasi kedalam posisi yang telah ditentukan. Fungsi ***TotalSitasi()*** dapat dilihat pada Kode Sumber 4.8.

```
public void TotalSitasi(RawData rawData)
    {

        GameObject parent = (GameObject)Instantiate
(parentBar);
        GameObject judul = (GameObject)Instantiate(
paramPrefab);
```

```
6.          parent.transform.GetChild(0).GetComponent<R
    enderer>().material.renderQueue = 2999;
7.          parent.transform.GetChild(1).GetComponent<R
    enderer>().material.renderQueue = 2999;
8.          judul.GetComponentInChildren<Text>().text =
     "Total Sitasi";
9.          judul.GetComponentInChildren<Text>().color
    = warnaJudul;
10.         judul.GetComponentInChildren<Text>().alignm
    ent = TextAnchor.MiddleLeft;
11.         judul.GetComponentInChildren<Text>().fontSi
    ze = fontSizeJudul;
12.         judul.transform.SetParent(parent.transform)
    ;
13.         judul.transform.localPosition = judulPos;
14.         judul.transform.GetChild(0).GetComponent<Re
    ctTransform>().SetSizeWithCurrentAnchors(RectTransf
    orm.Axis.Horizontal, 1000f);
15.         judul.transform.GetChild(0).GetComponent<Re
    ctTransform>().localScale = new Vector3(judul.trans
    form.GetChild(0).GetComponent<RectTransform>().loca
    lScale.x +0.5f, judul.transform.GetChild(0).GetComp
    onent<RectTransform>().localScale.y, judul.transfor
    m.GetChild(0).GetComponent<RectTransform>().localSc
    ale.z);
16.         int i = 0;
17.         foreach (var raw in (dynamic)(rawData.data[
    0].data_dashboard[0].total_sitasi))
18.         {
19.             GameObject chartBar = (GameObject)Insta
    ntiate(barPrefab);
20.             chartBar.GetComponentInChildren<Rendere
    r>().material.color = colorList[(raw.id_afiliasi -
    1) % 13];
21.             chartBar.transform.SetParent(parent.tra
    nsform);
22.             chartBar.transform.localPosition = new
    Vector3(posX, posY, posZ);
23.             chartBar.transform.localScale = new Vec
    tor3(chartBar.transform.localScale.x, raw.total_sit
    asi_peneliti, chartBar.transform.localScale.z);
```

```
24.             GameObject logo = (GameObject)Instantia
    te(logoAfiliasi);
25.             logo.transform.SetParent(parent.transfo
    rm);
26.             logo.transform.localPosition = new Vect
    or3(posX, posY, -0.5f);
27.
28.             Renderer rendererLogo = logo.transform.
    GetChild(0).GetComponent<Renderer>();
29.             rendererLogo.material.mainTexture = Tex
    tureEncoder(raw.logo_afiliasi);
30.             posX += 0.075f;
31.             i += 1;
32.         }
33.
34.         float higherChild = 0f;
35.         foreach (Transform child in parent.transfor
    m)
36.         {
37.             if (higherChild < child.transform.local
    Scale.y)
38.             {
39.                 higherChild = child.transform.local
    Scale.y;
40.             }
41.         }
42.         List<Vector3> trans = new List<Vector3>();

43.         foreach (Transform child in parent.transfor
    m)
44.         {
45.             if (child.tag == "Bar")
46.             {
47.                 child.transform.localScale = new Ve
    ctor3(child.transform.localScale.x, (child.transfor
    m.localScale.y * 0.75f / higherChild) + 0.02f, chil
    d.transform.localScale.z);
48.                 var x = new Vector3(child.transform
    .localPosition.x, child.transform.localScale.y - 0.
    4f, child.transform.localPosition.z);
49.                 trans.Add(x);
50.             }
```

```
51.         }
52.
53.         int transIndex = 0;
54.         foreach (var raw in (dynamic)(rawData.data[
    0].data_dashboard[0].total_sitasi))
55.         {
56.             GameObject param = (GameObject)Instanti
    ate(paramPrefab);
57.             param.GetComponentInChildren<Text>().te
    xt = raw.total_sitasi_peneliti.ToString();
58.             param.GetComponentInChildren<Text>().co
    lor = Color.white;
59.             param.transform.SetParent(parent.transf
    orm);
60.             param.transform.localPosition = trans[t
    ransIndex];
61.             param.transform.localScale = new Vector
    3(param.transform.localScale.x * 1.5f, param.transf
    orm.localScale.y * 1.5f, param.transform.localScale
    .z * 1.5f);
62.             transIndex += 1;
63.         }
64.         parent.transform.localScale = new Vector3(p
    arent.transform.localScale.x * 3 / 4, parent.transf
    orm.localScale.y * 3 / 4, parent.transform.localSca
    le.z * 3 / 4);
65.         parent.transform.localPosition = new Vector
    3(18f, -3f, 0);
66.         parent.transform.localRotation = Quaternion
    .Euler(0, rotationYTotalSitasi, 0);
67.
68.         EventTrigger panelOnTrigger = parent.GetCom
    ponent<EventTrigger>();
69.         EventTrigger.Entry newTriggerClick = new Ev
    entTrigger.Entry();
70.         newTriggerClick.eventID = EventTriggerType.
    PointerClick;
71.         newTriggerClick.callback.AddListener((data)
     => { DashboardPanelEvent(); });
72.         panelOnTrigger.triggers.Add(newTriggerClick
    );
73.     }
```

*Kode Sumber 4.8 Fungsi Total Sitasi*

***Keterangan()*** adalah fungsi yang digunakan untuk menginstansiasi objek-objek dasar untuk keterangan diagram batang. Objek-objek tersebut akan ditransformasi kedalam posisi yang telah ditentukan. Fungsi ***Keterangan()*** dapat dilihat pada Kode Sumber 4.9.

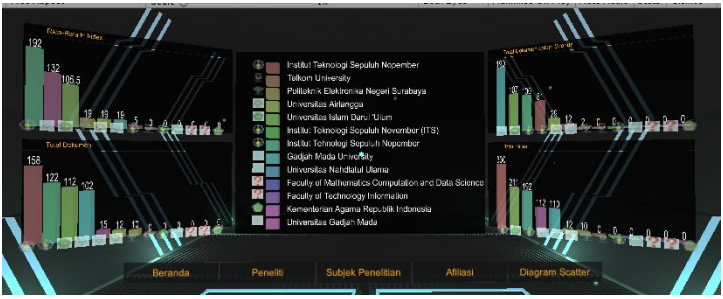
```
1.  public void Keterangan(RawData rawData)
2.      {
3.          GameObject keterangan = (GameObject)Instant
    iate(legend);
4.          keterangan.transform.GetChild(0).GetCompone
    nt<Renderer>().material.renderQueue = 2999;
5.          keterangan.transform.localPosition = new Ve
    ctor3(0, 1.25f, 2);
6.          foreach (var raw in (dynamic)(rawData.data[
    0].data_dashboard[0].afiliasi))
7.          {
8.              GameObject chartBar = (GameObject)Insta
    ntiate(barPrefab);
9.              GameObject afiliasi = (GameObject)Insta
    ntiate(paramPrefab);
10.             chartBar.GetComponentInChildren<Rendere
    r>().material.color = colorList[(raw.id_afiliasi -
    1) % 13];
11.             chartBar.transform.SetParent(keterangan
    .transform);
12.             afiliasi.GetComponentInChildren<RectTra
    nsform>().SetSizeWithCurrentAnchors(RectTransform.A
    xis.Horizontal, 2000);
13.             afiliasi.transform.GetChild(0).GetCompo
    nent<RectTransform>().SetSizeWithCurrentAnchors(Rec
    tTransform.Axis.Horizontal, 2000f);
14.             afiliasi.transform.GetChild(0).GetCompo
    nent<RectTransform>().localScale = new Vector3(afil
    iasi.transform.GetChild(0).GetComponent<RectTransfo
    rm>().localScale.x +0.5f, afiliasi.transform.GetChi
    ld(0).GetComponent<RectTransform>().localScale.y, a
    filiasi.transform.GetChild(0).GetComponent<RectTran
    sform>().localScale.z);
```

```
15.             afiliasi.GetComponentInChildren<Text>()
    .color = Color.white;
16.             afiliasi.GetComponentInChildren<Text>()
    .text = raw.nama_afiliasi.ToString();
17.             afiliasi.GetComponentInChildren<Text>()
    .alignment = TextAnchor.MiddleLeft;
18.             afiliasi.transform.SetParent(keterangan
    .transform);
19.             chartBar.transform.localPosition = new
    Vector3(legenPosX+0.05f, legenPosY, legenPosZ);
20.             afiliasi.transform.localPosition = new
    Vector3(legenPosX + 0.7f+ 0.05f, univPosY, legenPos
    Z);
21.             GameObject logo = (GameObject)Instantia
    te(logoAfiliasi);
22.             logo.transform.SetParent(keterangan.tra
    nsform);
23.             logo.transform.localPosition = new Vect
    or3(legenPosX-0.01f, legenPosY+0.025f, 0);
24.             logo.transform.localScale = new Vector3
    (logo.transform.localScale.x * 0.75f, logo.transfor
    m.localScale.y * 0.75f, logo.transform.localScale.z
     * 0.75f);
25.
26.             Renderer rendererLogo = logo.transform.
    GetChild(0).GetComponent<Renderer>();
27.             rendererLogo.material.mainTexture = Tex
    tureEncoder(raw.logo_afiliasi);
28.             legenPosY -= 0.075f;
29.             univPosY -= 0.076f;
30.         }
31.
32.         keterangan.transform.localScale = new Vecto
    r3(22.01331f, 15.23f, 1);
33. }
34.
```

*Kode Sumber 4.9 Fungsi Keterangan*

Objek-objek dasar untuk beranda akan diinstansiasi kedalam ruang 3-dimensi. Ketika objek-objek tersebut telah

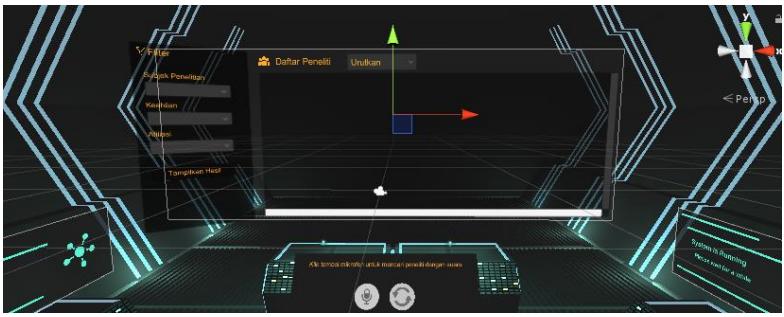
diinstansiasi dengan data untuk beranda, tampilan beranda akan tampak seperti pada Gambar 4.25.




*Gambar 4.25 Tampilan Beranda*

### 4.3.1.1 Implementasi Antarmuka Daftar Peneliti

Antarmuka daftar peneliti memiliki tiga buah fitur utama, yaitu fitur urutkan, filter, dan cari. Fitur urutkan dan filter menggunakan objek *dropdown*. *List* fitur *dropdown* akan diinstansiasi secara dinamis sesuai dengan data pada basis data. Sedangkan fitur pencarian memiliki dua buah tombol yaitu tombol mikrofon untuk merekam suara sebagai kata kunci dan juga tombol reset untuk melakukan reset pencarian. Tampilan fitur-fitur tersebut dapat dilihat pada Gambar 4.26.




*Gambar 4.26 Tampilan Fitur Panel Daftar Peneliti*

Selain tiga buah fitur tersebut, panel daftar peneliti memiliki tombol daftar peneliti. Tombol ini memuat informasi singkat tentang peneliti. Tombol ini akan diinstansiasi sesuai data pada basis data. Jika tombol ini di klik, pengguna akan diteruskan ke panel detail peneliti. Tombol untuk daftar peneliti dapat dilihat pada Gambar 4.27.


*Gambar 4.27 Tampilan Fitur Panel Peneliti*

Antarmuka daftar peneliti memiliki beberapa fungsi yaitu fungsi ***GetDataPeneliti()*** pada Kode Sumber 4.10, fungsi ***GetFilterPeneliti()*** pada Kode Sumber 4.11, fungsi ***GetUrutkanDataPeneliti()*** pada Kode Sumber 4.12, ***DataPeneliti()*** pada Kode Sumber 4.13, fungsi ***DataFilter()*** pada Kode Sumber 4.14, ***RecordAudio()*** pada Kode Sumber 4.15 baris pertama, dan ***StopRecordAudio()*** pada Kode Sumber 4.15 baris ke-15.

***GetDataPeneliti()*** adalah fungsi yang digunakan untuk meminta data peneliti ke server menggunakan URL https://<alamat_ip>:<port>/peneliti dan server akan mengirimkan kembali data berupa objek JSON.

Setelah data didapatkan, fungsi ***GetDataPeneliti()*** akan memanggil fungsi ***DataPeneliti()*** untuk menampilkan data. Fungsi ***GetDataPeneliti()*** dapat dilihat pada Kode Sumber 4.10.

```
1.  public void GetDataPeneliti()
2.    {
3.
4.      StartCoroutine(requestPeneliti.RequestData((res
    ult) => {
5.        panelPeneliti.SetActive(true);
6.        DataPeneliti(result);
7.        DataFilter(result);
8.
9.        tombolBeranda.GetComponent<Button>().interact
    able = true;
10.       tombolPeneliti.GetComponent<Button>().interac
    table = true;
11.       tombolAfiliasi.GetComponent<Button>().interac
    table = true;
12.       tombolSubjekPenelitian.GetComponent<Button>()
    .interactable = true;
13.       tombolDiagramScatter.GetComponent<Button>().i
    nteractable = true;
14.       connectionMessagePanel.SetActive(false);
15.     }, error=> {
16.       if (error != "")
17.       {
18.         retryMessage.text = error;
19.         retry.SetActive(true);
20.         connectionMessagePanel.SetActive(false);
21.       }
22.     }
23.     ));
24.
25.   }
```

*Kode Sumber 4.10 Fungsi Get Data Peneliti*

***GetFilterPeneliti()*** adalah fungsi yang digunakan untuk meminta data peneliti ke server menggunakan URL https://<alamat_ip>:<port>/peneliti?keahlian=id_keahlian&afiliasi=id_afiliasi&id_subjek_area=id_subjek&keyword=& dengan parameter filter. Parameter filter meliputi afiliasi, keahlian dan subjek. Setelah data didapatkan, fungsi ***GetFilterPeneliti()*** akan memanggil fungsi ***DataPeneliti()*** ***DataFilter*** untuk menampilkan

data. Fungsi ***GetFilterPeneliti()*** dapat dilihat pada Kode Sumber 4.11.

```
1.  public void GetFilterPeneliti(string parameter)
2.    {
3.
4.      StartCoroutine(requestFilterPeneliti.RequestDat
    a((result) => {
5.        DataPeneliti(result);
6.        DataFilter(result);
7.        loadPeneliti = true;
8.        connectionMessagePanel.SetActive(false);
9.        tombolBeranda.GetComponent<Button>().interact
    able = true;
10.       tombolPeneliti.GetComponent<Button>().interac
    table = true;
11.       tombolAfiliasi.GetComponent<Button>().interac
    table = true;
12.       tombolSubjekPenelitian.GetComponent<Button>()
    .interactable = true;
13.       tombolDiagramScatter.GetComponent<Button>().i
    nteractable = true;
14.     }, error => {
15.       if (error != "")
16.       {
17.         retryMessage.text = error;
18.         retry.SetActive(true);
19.         connectionMessagePanel.SetActive(false);
20.       }
21.     }
22.     ));
23.   }
```

*Kode Sumber 4.11 Fungsi Get Data Peneliti*

***GetUrutkanDataPeneliti()*** adalah fungsi yang digunakan untuk meminta data peneliti ke server menggunakan URL https://<alamat_ip>:<port>/peneliti?urutkan=parameter dengan parameter urutkan. Parameter filter meliputi nama, h index, total dokumen, total dokumen scopus dan total sitasi. Setelah data

didapatkan, fungsi ***GetUrutkanDataPeneliti()*** akan memanggil fungsi ***DataPeneliti()*** untuk menampilkan data. Fungsi ***GetUrutkanDataPeneliti()*** dapat dilihat pada Kode Sumber 4.12.

```
1.  public void GetUrutkanDataPeneliti(string param)

2.    {
3.      StartCoroutine(requestUrutkanPeneliti.RequestDa
    ta((result) => {
4.        connectionMessagePanel.SetActive(false);
5.        DataPeneliti(result);
6.        loadPeneliti = true;
7.        tombolBeranda.GetComponent<Button>().interact
    able = true;
8.        tombolPeneliti.GetComponent<Button>().interac
    table = true;
9.        tombolAfiliasi.GetComponent<Button>().interac
    table = true;
10.       tombolSubjekPenelitian.GetComponent<Button>()
    .interactable = true;
11.       tombolDiagramScatter.GetComponent<Button>().i
    nteractable = true;
12.
13.     }, (error) => {
14.       if (error != "")
15.       {
16.         retryMessage.text = error;
17.         retry.SetActive(true);
18.         connectionMessagePanel.SetActive(false);

19.       }
20.     }
21.     ));
22.   }
```

*Kode Sumber 4.12 Fungsi Get Urutkan Data Peneliti*

***DataPeneliti()*** adalah fungsi yang digunakan untuk menginstansiasi objek tombol menjadi bentuk daftar. Setiap tombol akan memiliki aksi sesuai dengan ID dari peneliti dan

menambahkan infomasi ringkas dari peneliti kedalam tombol. Aksi dari setiap tombol nantinya akan digunakan untuk meminta detail peneliti dengan cara mengklik tombol pada daftar peneliti yang akan memanggil fungsi ***GetDetailPeneliti()*** yang menggunakan parameter id dari peneliti untuk mendapatkan detail peneliti. Fungsi ***DataPeneliti()*** dapat dilihat pada Kode Sumber 4.13.

```
1.  public void DataPeneliti(RawData rawData)
2.    {
3.      foreach (Transform list in listPeneliti.transfo
    rm)
4.      {
5.        Destroy(list.gameObject);
6.      }
7.
8.      float y = -80f;
9.      float height = 0f;
10.     for (int i = 0; i < (dynamic)rawData.data_len;
    i++)
11.     {
12.       height += 155f;
13.     }
14.     RectTransform lP = listPeneliti.GetComponent<Re
    ctTransform>();
15.     lP.sizeDelta = new Vector2(lP.sizeDelta.x, heig
    ht);
16.     foreach (var raw in (dynamic)(rawData.data[0].d
    ata_peneliti))
17.     {
18.       GameObject daftarPeneliti = (GameObject)Insta
    ntiate(tombolListPeneliti);
19.       string id_peneliti = raw.id_peneliti.ToString
    ();
20.       daftarPeneliti.GetComponentInChildren<Button>
    ().onClick.AddListener(() => GetDetailPeneliti(id_p
    eneliti));
21.       Debug.Log(tombolListPeneliti.GetComponentInCh
    ildren<Button>().onClick);
22.       daftarPeneliti.transform.GetChild(0).GetCompo
    nent<Text>().text = raw.nama_peneliti.ToString();
```

```
23.       daftarPeneliti.transform.GetChild(1).GetCompo
    nent<Text>().text = raw.kode_peneliti.ToString();
24.       daftarPeneliti.transform.GetChild(2).GetCompo
    nent<Text>().text = "#" + raw.h_index_rank.ToString
    ();
25.       Debug.Log(raw.nama_fakultas);
26.       daftarPeneliti.transform.GetChild(4).GetCompo
    nent<Text>().text = raw.nama_fakultas.ToString();
27.       daftarPeneliti.transform.GetChild(6).GetCompo
    nent<Text>().text = raw.nama_departemen.ToString();

28.       string nama_subjek = "";
29.       string nama_keahlian = "";
30.       int x = 0;
31.       if (raw.detail_subjek_area.Count > 0)
32.       {
33.         foreach (var subjek in raw.detail_subjek_ar
    ea)
34.         {
35.           if (x > 0)
36.           {
37.             nama_subjek += ", " + subjek.nama_subje
    k_area.ToString();
38.           }
39.           else
40.           {
41.             nama_subjek = subjek.nama_subjek_area.T
    oString();
42.           }
43.           x += 1;
44.         }
45.       }
46.       else
47.       {
48.         nama_subjek = "-";
49.       }
50.
51.
52.       x = 0;
53.       if(raw.keahlian_peneliti.Count > 0)
54.       {
```

```
55.         foreach (var keahlian in raw.keahlian_penel
    iti)
56.         {
57.           if (x > 0)
58.           {
59.             nama_keahlian += ", " + keahlian.nama_k
    eahlian_tier_2 + "(" + keahlian.nama_keahlian_tier_
    1 + ")";
60.           }
61.           else
62.           {
63.             nama_keahlian = keahlian.nama_keahlian_
    tier_2 + "(" + keahlian.nama_keahlian_tier_1 + ")";

64.           }
65.           x += 1;
66.         }
67.       }
68.       else
69.       {
70.         nama_keahlian = "-";
71.       }
72.
73.
74.       daftarPeneliti.transform.GetChild(8).GetCompo
    nent<Text>().text = nama_subjek;
75.       daftarPeneliti.transform.GetChild(10).GetComp
    onent<Text>().text = raw.h_index_peneliti.ToString(
    );
76.       daftarPeneliti.transform.GetChild(12).GetComp
    onent<Text>().text = raw.dokumen_peneliti.ToString(
    );
77.       daftarPeneliti.transform.GetChild(14).GetComp
    onent<Text>().text = raw.dokumen_peneliti_dalam_sco
    pus.ToString();
78.       daftarPeneliti.transform.GetChild(16).GetComp
    onent<Text>().text = raw.total_sitasi_peneliti.ToSt
    ring();
79.       daftarPeneliti.transform.GetChild(18).GetComp
    onent<Text>().text = raw.nama_afiliasi.ToString();
```

```
80.       daftarPeneliti.transform.GetChild(20).GetComp
    onent<Text>().text = nama_keahlian;
81.       daftarPeneliti.transform.SetParent(listPeneli
    ti.transform, false);
82.       daftarPeneliti.GetComponent<RectTransform>().
    localPosition = new Vector3(400f, y, 0);
83.       Debug.Log(daftarPeneliti.GetComponent<RectTra
    nsform>().localPosition.y);
84.       daftarPeneliti.transform.localScale = new Vec
    tor3(1, 1, 1);
85.       y -= (150f + 5f);
86.     }
87.   }
```

*Kode Sumber 4.13 Fungsi Data Peneliti*

***DataFilter()*** adalah fungsi yang digunakan untuk menambahkan data filter kedalam daftar *dropdown. Dropdown* ini digunakan untuk memilih parameter filter yang akan digunakan. Fungsi ***DataFilter()*** dapat dilihat pada Kode Sumber 4.14.

```
1.  public void DataFilter(RawData rawData)
2.      {
3.          List<string> listAfiliasi = new List<string
    >() { "semua" };
4.          List<string> listKeahlian = new List<string
    >() { "semua" };
5.          List<string> listSubjekPenelitian = new Lis
    t<string>() { "semua" };
6.
7.          Dropdown dropdownAfiliasi = filterAfiliasi.
    GetComponent<Dropdown>();
8.          Dropdown dropdownKeahlian = filterKeahlian.
    GetComponent<Dropdown>();
9.          Dropdown dropdownSubjekPenelitian = filterS
    ubjek.GetComponent<Dropdown>();
10.
11.         foreach(var i in (dynamic)(rawData.data[0].
    data_afiliasi))
12.         {
```

```
13.             listAfiliasi.Add(i.nama_afiliasi);
14.         }
15.
16.         foreach (var i in (dynamic)(rawData.data[0]
    .data_subjek_topik))
17.         {
18.             listKeahlian.Add(i.kelas_tier_2);
19.         }
20.
21.         foreach (var i in (dynamic)(rawData.data[0]
    .data_subjek_area))
22.         {
23.             listSubjekPenelitian.Add(i.nama_subjek_
    area);
24.         }
25.
26.         dropdownAfiliasi.AddOptions(listAfiliasi);

27.         dropdownKeahlian.AddOptions(listKeahlian);

28.         dropdownSubjekPenelitian.AddOptions(listSub
    jekPenelitian);
29.     }
```

*Kode Sumber 4.14 Fungsi Data Filter*

***RecordAudio()*** adalah fungsi yang digunakan untuk tombol merekam suara dengan mikrofon dan disimpan kedalam file audio. File audio ini nantinya dikirimkan ke server menggunakan URL https://<alamat_ip>:<port>/voicerecognizer. Ketika tombol merekam suara diklik, tombol akan berubah menjadi tombol stop. Ketika tombol stop diklik, tombol akan memanggil fungsi ***StopRecordAudio()***. Ketika fungsi ***StopRecordAudio()*** dipanggil, mikrofon akan berhenti merekam suara dan mengirimkan suara ke server menggunakan fungsi ***SendAudioToServer()*** untuk diproses menjadi kata kunci pencarian. Fungsi ***RecordAudio()*** dapat dilihat di Kode Sumber 4.15 baris ke-15 sedangkan fungsi ***SendAudioToServer()*** dapat dilihat di Kode Sumber 4.15 baris ke-33.

```
1.  public void RecordAudio()
2.  {
3.      startRecord.SetActive(false);
4.      stopRecord.SetActive(true);
5.      if (!Permission.HasUserAuthorizedPermission(Per
    mission.Microphone))
6.      {
7.          Permission.RequestUserPermission(Permission
    .Microphone);
8.      }
9.      else
10.     {
11.         voice = Microphone.Start(null, false, 10, 4
    4100);
12.     }
13. }
14.
15. public void StopRecordAudio()
16. {
17.     textSuara.text = "Sedang Menganalisa Suara";
18.     startRecord.SetActive(true);
19.     stopRecord.SetActive(false);
20.     Microphone.End(null);
21.
22.     string fullFileName = Path.Combine(Application.
    persistentDataPath, "voice.wav");
23.     SavWav.Save(fullFileName, voice, false);
24.     bool checkFile = File.Exists(fullFileName);
25.
26.     Debug.Log(checkFile);
27.     if (checkFile)
28.     {
29.         SendAudioToServer(fullFileName);
30.     }
31. }
32.
33. public void SendAudioToServer(string file)
34. {
35.     RequestAudioHandler getAudioResult = new Reques
    tAudioHandler();
36.     getAudioResult.URL = URL + "/voicerecognizer";
```

```
37.     getAudioResult.file = file;
38.
39.     StartCoroutine(getAudioResult.RequestAudioResul
    t((result) => {
40.         string hasilAudio = result.data[0].hasil_au
    dio;
41.         textSuara.text = "Mencari " + hasilAudio;
42.         string parameter = "/peneliti?keahlian=&afi
    liasi=&id_subjek_area=&keyword=" + hasilAudio;
43.         GetFilterPeneliti(parameter);
44.     }, (msg) =>
45.     {
46.         if (msg != "")
47.         {
48.             textSuara.text = msg;
49.         }
50.     }
51.     ));
52. }
```

*Kode Sumber 4.15 Fungsi Record Audio dan Stop Record Audio*

Ketika tampilan antarmuka daftar peneliti tersebut sudah diinstansiasi menggunakan data peneliti dari basis data, tampilan panel daftar peneliti akan tampak seperti pada Gambar 4.28.


*Gambar 4.28 Tampilan Panel Daftar Peneliti*

### 4.3.1.2 Implementasi Antarmuka Detail Peneliti

Antarmuka detail peneliti adalah tampilan yang memuat tentang informasi detail peneliti. Dalam tampilan memuat beberapa informasi yaitu kode peneliti, nama peneliti, departemen peneliti, fakultas peneliti, keahlian peneliti, subjek penelitian, h index, total dokumen, total dokumen scopus, total sitasi serta *pie chart* untuk rasio dokumen publikasi.

Antarmuka detail peneliti memiliki dua buah fungsi utama yaitu fungsi ***GetDetailPeneliti()*** di Kode Sumber 4.16 pada baris pertama dan ***DetailPeneliti()*** di Kode Sumber 4.16 pada baris ke-37. ***GetDetailPeneliti()*** memiliki fungsi untuk meminta data detail peneliti ke server menggunakan URL https://<alamat_ip>:<port>/detail-peneliti?id_peneliti=id_peneliti dengan parameter ID peneliti. Setelah permintaan dikirimkan, server akan mengirimkan kembali data berupa objek JSON. Data tersebut akan ditampilkan menggunakan fungsi ***DetailPeneliti()*.** Fungsi ***DetailPeneliti()*** akan memodifikasi tampilan dasar untuk detail peneliti sehingga memiliki antarmuka detail peneliti seperti pada Gambar 4.29.


*Gambar 4.29 Tampilan Panel Detail Peneliti*

```
1.  public void GetDetailPeneliti(string id_peneliti)
2.  {
3.
```

```
    StartCoroutine(getDetailPeneliti.RequestData((result) => {
      DetailPeneliti(result);
      int scopus = result.data[0].detail_peneliti[0].dokumen_peneliti_dalam_scopus;
      int total = result.data[0].detail_peneliti[0].dokumen_peneliti;
      RequestImageHandler getTexture = new RequestImageHandler();
      getTexture.URL = URL + "/piechart-texture?scopus=" + scopus + "&total=" + total;
      StartCoroutine(getTexture.RequestImageTexture((textureImg) => {
        Renderer pieChartRenderer = pieChart.GetComponent<Renderer>();
        pieChartRenderer.material.renderQueue = 3001;

        pieChartRenderer.material.mainTexture = textureImg;
        panelDetailPeneliti.SetActive(true);
        connectionMessagePanel.SetActive(false);
      }, (msg) =>
      {
        if(msg != "")
        {
          retryMessage.text = msg;
          retry.SetActive(true);
          connectionMessagePanel.SetActive(false);
        }
      }
      ));
    }, (error) => {
      if (error != "")
      {
        retryMessage.text = error;
        retry.SetActive(true);
        connectionMessagePanel.SetActive(false);
      }
    }
    ));
}

```

```
37. public void DetailPeneliti(RawData rawData)
38. {
39.   detailNamaPeneliti.text = rawData.data[0].detail_
    peneliti[0].nama_peneliti.ToString();
40.   detailKodePeneliti.text = rawData.data[0].detail_
    peneliti[0].kode_peneliti.ToString();
41.   detailAfiliasiPeneliti.text = rawData.data[0].det
    ail_peneliti[0].nama_afiliasi.ToString();
42.   detailDepartemenPeneliti.text = rawData.data[0].d
    etail_peneliti[0].nama_departemen.ToString();
43.   detailFakultasPeneliti.text = rawData.data[0].det
    ail_peneliti[0].nama_fakultas.ToString();
44.   string stringKeahlian = "";
45.   string stringSubjek = "";
46.   foreach (var keahlian in rawData.data[0].detail_p
    eneliti[0].keahlian_peneliti)
47.   {
48.     if(rawData.data[0].detail_peneliti[0].keahlian_
    peneliti.Count > 1)
49.     {
50.       if (stringKeahlian == "")
51.       {
52.         stringKeahlian += keahlian.nama_keahlian_ti
    er_2 + "(" + keahlian.nama_keahlian_tier_1 + ")";
53.       }
54.       else
55.       {
56.         stringKeahlian += ", " + keahlian.nama_keah
    lian_tier_2 + "(" + keahlian.nama_keahlian_tier_1 +
     ")";
57.       }
58.     }
59.     else
60.     {
61.       stringKeahlian = keahlian.nama_keahlian_tier_
    2 + "(" + keahlian.nama_keahlian_tier_1 + ")";
62.     }
63.   }
64.
65.   foreach (var subjekArea in rawData.data[0].detail
    _peneliti[0].detail_subjek_area)
66.   {
```

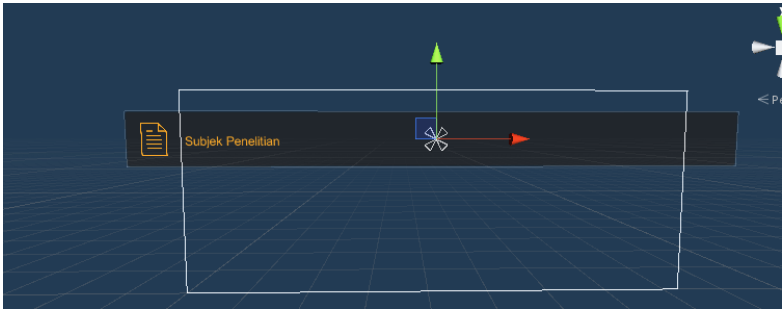
```
67.     if (rawData.data[0].detail_peneliti[0].detail_s
    ubjek_area.Count > 1)
68.     {
69.       if (stringSubjek == "")
70.       {
71.         stringSubjek += subjekArea.nama_subjek_area
    ;
72.       }
73.       else
74.       {
75.         stringSubjek += ", " + subjekArea.nama_subj
    ek_area;
76.       }
77.     }
78.     else
79.     {
80.       stringKeahlian = subjekArea.nama_subjek_area;

81.     }
82.   }
83.   detailKeahlianPeneliti.text = stringKeahlian;
84.   detailSubjekPeneliti.text = stringSubjek;
85.   totalSitasiPeneliti.text = rawData.data[0].detail
    _peneliti[0].total_sitasi_peneliti.ToString();
86.   totalDokumenPeneliti.text = rawData.data[0].detai
    l_peneliti[0].dokumen_peneliti.ToString();
87.   totalDokumenScopusPeneliti.text = rawData.data[0]
    .detail_peneliti[0].dokumen_peneliti_dalam_scopus.T
    oString();
88.   hIndexPeneliti.text = rawData.data[0].detail_pene
    liti[0].h_index_peneliti.ToString();
89. }
```

*Kode Sumber 4.16 Fungsi Get Detail Peneliti dan Detail Peneliti*

#### 4.3.1.3 Implementasi Antarmuka Daftar Subjek Penelitian

Antarmuka daftar subjek penelitian adalah tampilan yang berisi tentang daftar subjek penelitian. Tampilan ini menggunakan *prefab* tombol. Tombol ini akan diinstansiasi secara dinamis dengan menggunakan data subjek penelitian yang didapatkan dari

server. *Prefab* untuk tombol subjek penelitian dapat dilihat pada Gambar 4.30.




*Gambar 4.30 Prefab Tombol DaftarAfiliasi*

Antarmuka daftar subjek penelitian memiliki dua buah fungsi utama yaitu fungsi ***GetDataSubjekPenelitian ()*** di Kode Sumber 4.17 pada baris pertama dan ***DataSubjekPenelitian()*** di Kode Sumber 4.17 pada baris ke-48. ***GetDataSubjekPenelitian()*** memiliki fungsi untuk meminta data subjek penelitian ke server menggunakan URL https://<alamat_ip>:<port>/subjek dan server akan mengirimkan kembali data berupa objek JSON.

Setelah data tersebut didapat, informasi mengenai detail peneliti akan ditampilkan menggunakan fungsi ***DataSubjekPenelitian()*** sehingga memiliki antarmuka daftar subjek penelitian seperti pada Gambar 4.31.


*Gambar 4.31 Tampilan Panel Daftar Subjek Penelitan*

```
1.  public void GetDataSubjekPenelitian()
2.    {
3.      GameObject[] dashboard = GameObject.FindGameObj
    ectsWithTag("PanelDashboard");
4.      GameObject[] legend = GameObject.FindGameObject
    sWithTag("Legend");
5.      int i = 0;
6.      foreach (GameObject panel in dashboard)
7.      {
8.
9.        inactivePanel[i] = panel;
10.       panel.SetActive(false);
11.       i += 1;
12.     }
13.
14.     if (legend.Length != 0)
15.     {
16.       inactivePanel[i] = legend[0];
17.       legend[0].SetActive(false);
18.     }
19.
20.     if (loadSubjekPenelitian == false)
21.     {
22.       RequestHandler requestSubjekPenelitian = new
    RequestHandler();
23.       requestSubjekPenelitian.URL = URL + "/subjek"
    ;
```

```
24.       connectionMessage.text = "Loading...";
25.       connectionMessagePanel.SetActive(true);
26.       StartCoroutine(requestSubjekPenelitian.Reques
    tData((result) => {
27.         DataSubjekPenelitian(result);
28.         panelSubjekPenelitian.SetActive(true);
29.         loadSubjekPenelitian = true;
30.         connectionMessagePanel.SetActive(false);
31.
32.       }, error => {
33.         if (error != "")
34.         {
35.           retryMessage.text = error;
36.           retry.SetActive(true);
37.           connectionMessagePanel.SetActive(false);

38.         }
39.       }
40.       ));
41.     }
42.     else
43.     {
44.       panelSubjekPenelitian.SetActive(true);
45.     }
46.   }
47.
48.   public void DataSubjekPenelitian(RawData rawData)

49.   {
50.     float y = -90 / 2f;
51.     float height = 0f;
52.     for (int i = 0; i < (dynamic)rawData.data_len;
    i++)
53.     {
54.       height += 95f;
55.     }
56.     RectTransform lS = listSubjekPenelitian.GetComp
    onent<RectTransform>();
57.     lS.sizeDelta = new Vector2(lS.sizeDelta.x, heig
    ht);
58.     foreach (var dataSubjek in rawData.data[0].data
    _subjek_area)
```

```
59.     {
60.       GameObject daftarSubjek = (GameObject)Instant
    iate(tombolListSubjekPenelitian);
61.       daftarSubjek.GetComponentInChildren<Button>()
    .onClick.AddListener(() => GetDetailSubjekPenelitia
    n(dataSubjek.id_subjek_area.ToString()));
62.       daftarSubjek.transform.GetChild(0).GetCompone
    nt<Text>().text = dataSubjek.nama_subjek_area;
63.
64.       daftarSubjek.transform.SetParent(listSubjekPe
    nelitian.transform, false);
65.       daftarSubjek.GetComponent<RectTransform>().lo
    calPosition = new Vector3(510f, y, 0);
66.       daftarSubjek.transform.localScale = new Vecto
    r3(1, 1, 1);
67.       y -= (90f + 5f);
68.     }
69.   }
```

*Kode Sumber 4.17 Fungsi Get Data Subjek Penelitian dan Data Subjek Penelitian*

#### 4.3.1.4 Implementasi Antarmuka Detail Subjek Penelitian

Antarmuka detail subjek penelitian adalah tampilan yang memuat tentang informasi detail subjek penelitian. Dalam tampilan memuat beberapa informasi yaitu nama subjek penelitian, rata-rata h index, total dokumen, total dokumen scopus, total sitasi serta *pie chart* untuk rasio dokumen publikasi.

Antarmuka detail subjek penelitian memiliki dua buah fungsi utama yaitu fungsi ***GetDetailSubjekPenelitan ()*** di Kode Sumber 4.18 pada baris pertama dan ***DetailSubjek()*** di Kode Sumber 4.18 pada baris ke-39. ***GetDetailSubjekPenelitian()*** memiliki fungsi untuk meminta data detail subjek penelitian ke server menggunakan URL https://<alamat_ip>:<port>/detail-subjek?id_subjek=id_subjek dengan parameter ID subjek penelitian. Setelah permintaan dikirimkan, server akan mengirimkan kembali data berupa objek JSON. Data tersebut akan ditampilkan menggunakan fungsi ***DetailSubjek()***. Fungsi

***DetailSubjek()*** akan memodifikasi tampilan dasar untuk detail subjek penelitian sehingga memiliki antarmuka detail subjek penelitian seperti pada Gambar 4.32.


*Gambar 4.32 Tampilan Detail Subjek Penelitian*

```
1.  public void GetDetailSubjekPenelitian(string id_sub
    jek)
2.    {
3.
4.      RequestHandler getDetailSubjek = new RequestHan
    dler();
5.      getDetailSubjek.URL = URL + "/detail-
    subjek?id_subjek=" + id_subjek;
6.      connectionMessage.text = "Loading...";
7.      connectionMessagePanel.SetActive(true);
8.      StartCoroutine(getDetailSubjek.RequestData((res
    ult) => {
9.        DetailSubjek(result);
10.       int scopus = result.data[0].detail_subjek_pen
    elitian[0].total_dokumen_scopus;
11.       int total = result.data[0].detail_subjek_pene
    litian[0].total_dokumen;
12.       RequestImageHandler getTexture = new RequestI
    mageHandler();
13.       getTexture.URL = URL + "/piechart-
    texture?scopus=" + scopus + "&total=" + total;
```

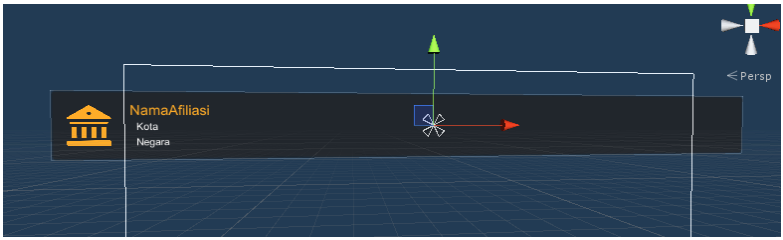
```
14.       StartCoroutine(getTexture.RequestImageTexture
    ((textureImg) => {
15.         Renderer pieChartRenderer = pieChartSubjekP
    enelitian.GetComponent<Renderer>();
16.         pieChartRenderer.material.renderQueue = 300
    1;
17.         pieChartRenderer.material.mainTexture = tex
    tureImg;
18.         panelDetailSubjekPenelitian.SetActive(true)
    ;
19.       }, (msg) =>
20.       {
21.         if (msg != "")
22.         {
23.           retryMessage.text = msg;
24.           retry.SetActive(true);
25.           connectionMessagePanel.SetActive(false);

26.         }
27.       }
28.       ));
29.     }, (error) => {
30.       if (error != "")
31.       {
32.         retryMessage.text = error;
33.         retry.SetActive(true);
34.         connectionMessagePanel.SetActive(false);
35.       }
36.     }
37.     ));
38.   }
39.   public void DetailSubjek(RawData rawData)
40.   {
41.     namaDetailSubjek.text = rawData.data[0].detail_
    subjek_penelitian[0].nama_subjek_penelitian;
42.     rerataHIndexSubjek.text = rawData.data[0].detai
    l_subjek_penelitian[0].rata_rata_h_index.ToString()
    ;
43.     totalDokumenSubjek.text = rawData.data[0].detai
    l_subjek_penelitian[0].total_dokumen.ToString();
```

```
44.     totalDokumenScopusSubjek.text = rawData.data[0]
    .detail_subjek_penelitian[0].total_dokumen_scopus.T
    oString();
45.     totalSitasiSubjek.text = rawData.data[0].detail
    _subjek_penelitian[0].total_sitasi.ToString();
46.
47.     panelDetailSubjekPenelitian.transform.GetChild(
    7).GetComponent<Button>().onClick.AddListener(() =>
     GetPenelitiFrom("", rawData.data[0].detail_subjek_
    penelitian[0].id_subjek_penelitian.ToString()));
48.   }
```

*Kode Sumber 4.18 Fungsi Get Detail Subjek Penelitian dan Detail Subjek*

### 4.3.1.5 Implementasi Antarmuka Daftar Afiliasi

Antarmuka daftar afiliasi adalah tampilan yang berisi tentang daftar afiliasi. Tampilan ini menggunakan prefab tombol. Tombol ini akan diinstansiasi secara dinamis dengan menggunakan data afiliasi yang didapatkan dari server. Prefab untuk tombol daftar peneliti dapat dilihat pada Gambar 4.33.




*Gambar 4.33 Prefab Tombol DaftarAfiliasi*

Antarmuka daftar afiliasi memiliki dua buah fungsi utama yaitu fungsi ***GetDataAfiliasi()*** di Kode Sumber 4.19 pada baris pertama dan ***DataAfiliasi()*** di Kode Sumber 4.19 pada baris ke-48. ***GetDataAfiliasi()*** memiliki fungsi untuk meminta data afiliasi ke server menggunakan URL https://<alamat_ip>:<port>/afiliasi dan server akan mengitimkan kembali data berupa objek JSON.

Setelah data tersebut didapat, informasi mengenai detail peneliti akan ditampilkan menggunakan fungsi ***DataAfiliasi()*** sehingga memiliki antarmuka daftar afiliasi seperti pada Gambar 4.34.


*Gambar 4.34 Tampilan Panel Daftar Afiliasi*

```
1.  public void GetDataAfiliasi()
2.    {
3.      GameObject[] dashboard = GameObject.FindGameObj
    ectsWithTag("PanelDashboard");
4.      GameObject[] legend = GameObject.FindGameObject
    sWithTag("Legend");
5.      int i = 0;
6.      foreach (GameObject panel in dashboard)
7.      {
8.
9.        inactivePanel[i] = panel;
10.       panel.SetActive(false);
11.       i += 1;
12.     }
13.
14.     if (legend.Length != 0)
15.     {
16.       inactivePanel[i] = legend[0];
17.       legend[0].SetActive(false);
18.     }
19.
20.     if (loadAfiliasi == false)
```

```
21.     {
22.
23.       RequestHandler requestAfiliasi = new RequestH
    andler();
24.       requestAfiliasi.URL = URL + "/afiliasi";
25.       connectionMessage.text = "Loading...";
26.       connectionMessagePanel.SetActive(true);
27.       StartCoroutine(requestAfiliasi.RequestData((r
    esult) => {
28.         connectionMessagePanel.SetActive(false);
29.         DataAfiliasi(result);
30.         loadAfiliasi = true;
31.
32.       }, error => {
33.         if (error != "")
34.         {
35.           retryMessage.text = error;
36.           retry.SetActive(true);
37.           connectionMessagePanel.SetActive(false);

38.         }
39.       }
40.       ));
41.     }
42.     else
43.     {
44.       panelAfiliasi.SetActive(true);
45.     }
46.   }
47.
48.   public void DataAfiliasi(RawData rawData)
49.   {
50.     float y = -90/2f;
51.     float height = 0f;
52.     for (int i = 0; i < (dynamic)rawData.data_len;
    i++)
53.     {
54.       height += 95f;
55.     }
56.     RectTransform lA = listAfiliasi.GetComponent<Re
    ctTransform>();
```

```
57.     lA.sizeDelta = new Vector2(lA.sizeDelta.x, heig
    ht);
58.     foreach(var dataAfiliasi in rawData.data[0].afi
    liasi)
59.     {
60.       GameObject daftarAfiliasi = (GameObject)Insta
    ntiate(tombolListAfiliasi);
61.       daftarAfiliasi.GetComponentInChildren<Button>
    ().onClick.AddListener(() => GetDetailAfiliasi(data
    Afiliasi.id_afiliasi.ToString()));
62.       daftarAfiliasi.transform.GetChild(0).GetCompo
    nent<Text>().text = dataAfiliasi.nama_afiliasi;
63.       daftarAfiliasi.transform.GetChild(1).GetCompo
    nent<Text>().text = dataAfiliasi.kota_afiliasi;
64.       daftarAfiliasi.transform.GetChild(2).GetCompo
    nent<Text>().text = dataAfiliasi.negara_afiliasi;
65.       daftarAfiliasi.transform.SetParent(listAfilia
    si.transform, false);
66.       daftarAfiliasi.GetComponent<RectTransform>().
    localPosition = new Vector3(510f, y, 0);
67.       daftarAfiliasi.transform.localScale = new Vec
    tor3(1, 1, 1);
68.       y -= (90f + 5f);
69.     }
70.   }
```

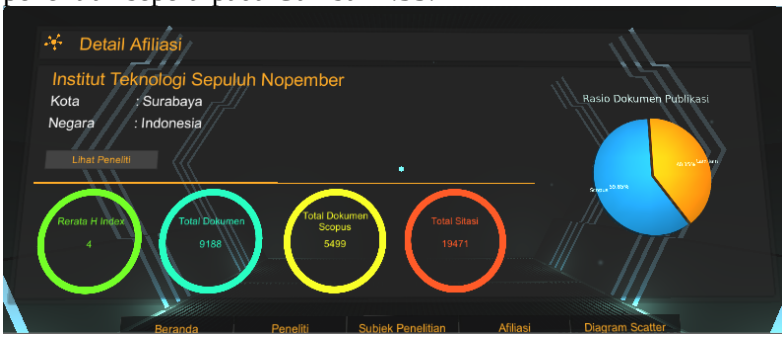
*Kode Sumber 4.19 Fungsi Get Data Afiliasi dan Data Afiliasi*

#### 4.3.1.6 Implementasi Antarmuka Detail Afiliasi

Antarmuka detail afiliasi adalah tampilan yang memuat tentang informasi detail dari afiliasi. Dalam tampilan memuat beberapa informasi yaitu nama afiliasi, kota afiliasi, negara afiliasi, rata-rata h index, total dokumen, total dokumen scopus, total sitasi serta *pie chart* untuk rasio dokumen publikasi.

Antarmuka detail afiliasi memiliki dua buah fungsi utama yaitu fungsi ***GetDetailAfiliasi()*** di Kode Sumber 4.20 pada baris pertama dan ***DetailAfiliasi()*** di Kode Sumber 4.20 pada baris ke-44. ***GetDetailAfiliasi()*** memiliki fungsi untuk meminta data detail afiliasi ke server menggunakan URL

https://<alamat_ip>:<port>/detail-afiliasi?id_afiliasi=id_afiliasi dengan parameter ID afiliasi. Setelah permintaan dikirimkan, server akan mengirimkan kembali data berupa objek JSON. Data tersebut akan ditampilkan menggunakan fungsi ***DetailAfiliasi()***. Fungsi ***DetailSubjek()*** akan memodifikasi tampilan dasar untuk detail subjek penelitian sehingga memiliki antarmuka detail subjek penelitian seperti pada Gambar 4.35.




*Gambar 4.35 Tampilan Panel Detail Afiliasi*

```
1.  public void GetDetailAfiliasi(string id_afiliasi)
2.    {
3.
4.      panelAfiliasi.SetActive(false);
5.      Debug.Log("ID Tombol :" + id_afiliasi);
6.      RequestHandler getDetailAfiliasi = new RequestH
    andler();
7.      getDetailAfiliasi.URL = URL + "/detail-
    afiliasi?id_afiliasi=" + id_afiliasi;
8.      connectionMessage.text = "Loading...";
9.      connectionMessagePanel.SetActive(true);
10.     StartCoroutine(getDetailAfiliasi.RequestData((r
    esult) => {
11.       DetailAfiliasi(result);
12.       int scopus = result.data[0].detail_afiliasi[0
    ].total_dokumen_scopus;
13.       int total = result.data[0].detail_afiliasi[0]
    .total_dokumen;
```

```
14.       RequestImageHandler getTexture = new RequestI
    mageHandler();
15.       getTexture.URL = URL + "/piechart-
    texture?scopus=" + scopus + "&total=" + total;
16.       StartCoroutine(getTexture.RequestImageTexture
    ((textureImg) => {
17.         Renderer pieChartRenderer = pieChartAfilias
    i.GetComponent<Renderer>();
18.         pieChartRenderer.material.renderQueue = 300
    1;
19.         pieChartRenderer.material.mainTexture = tex
    tureImg;
20.         panelDetailAfiliasi.SetActive(true);
21.         connectionMessagePanel.SetActive(false);
22.
23.       }, (msg) =>
24.       {
25.         if (msg != "")
26.         {
27.           retryMessage.text = msg;
28.           retry.SetActive(true);
29.           connectionMessagePanel.SetActive(false);

30.         }
31.       }
32.       ));
33.     }, (error) => {
34.       if (error != "")
35.       {
36.         retryMessage.text = error;
37.         retry.SetActive(true);
38.         connectionMessagePanel.SetActive(false);
39.       }
40.     }
41.     ));
42.   }
43.
44.   public void DetailAfiliasi(RawData rawData)
45.   {
46.     namaAfiliasi.text = rawData.data[0].detail_afil
    iasi[0].nama_afiliasi;
```

```
47.     kotaAfiliasi.text = rawData.data[0].detail_afil
    iasi[0].kota_afiliasi;
48.     negaraAfiliasi.text = rawData.data[0].detail_af
    iliasi[0].negara_afiliasi;
49.     rerataHIndexAfiliasi.text = rawData.data[0].det
    ail_afiliasi[0].rata_rata_h_index.ToString();
50.     totalDokumenAfiliasi.text = rawData.data[0].det
    ail_afiliasi[0].total_dokumen.ToString();
51.     totalDokumenScopusAfiliasi.text = rawData.data[
    0].detail_afiliasi[0].total_dokumen_scopus.ToString
    ();
52.     totalSitasiAfiliasi.text = rawData.data[0].deta
    il_afiliasi[0].total_sitasi.ToString();
53.
54.     panelDetailAfiliasi.transform.GetChild(11).GetC
    omponent<Button>().onClick.AddListener(() => GetPen
    elitiFrom(rawData.data[0].detail_afiliasi[0].id_afi
    liasi.ToString(), ""));
55.   }
```

*Kode Sumber 4.20 Fungsi Get Detail Afiliasi dan Detail Afiliasi*

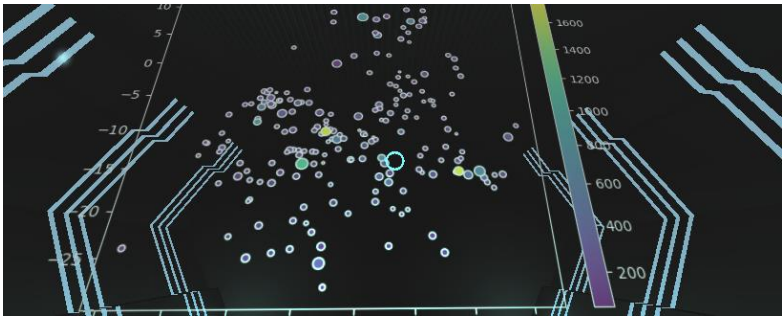
#### 4.3.1.7 Implementasi Antarmuka Diagram *Scatter*

Antarmuka diagram *scatter* adalah tampilan yang memuat tentang pemetaan peneliti berdasarkan koordinat x dan y. pemetaan ini memiliki parameter yaitu total dokumen tiap peneliti dan h index dari peneliti. H index peneliti akan menentukan ukuran *node* dari peneliti. Peneliti dengan h index tinggi akan memiliki ukuran yang lebih besar dibanding yang lain. Total dokumen peneliti juga akan mempengaruhi warna pada node peneliti. Semakin tinggi total dokumen peneliti, warna *node* peneliti akan semakin cerah.

Antarmuka diagram *scatter* memiliki fungsi ***GetScatterPlot()*** yang dapat dilihat pada Kode Sumber 4.21 pada baris pertama dan fungsi ***ThreeDScatter()*** yang dapat dilihat pada Kode Sumber 4.21 pada baris ke-51.

***GetScatterPlot()*** memiliki fungsi untuk meminta data detail afiliasi ke server menggunakan URL https://<alamat_ip>:<port>/scatter.. Setelah permintaan

dikirimkan, server akan mengirimkan kembali data berupa tekstur. Tekstur tersebut akan ditampilkan ke objek plane sehingga memiliki antarmuka diagram *scatter* seperti pada Gambar 4.36.




*Gambar 4.36 Tampilan Panel Diagram Scatter*

***ThreeDScatter()*** berfungsi untuk visualisasi diagram *scatter* kedalam bentuk 3-dimensi. Fungsi ini menginstansiasi objek *cube* dan memodifikasi transparansi warna dan juga transformasi baik posisi maupun ukuran dari objek *cube*. Transparansi warna pada *cube* akan diubah sesuai dengan total dokumen peneliti sedangkan transformasi ukuran dan posisi *cube* akan diubah berdasarkan h index dan koordinat pada sumbu x dan y. Fungsi ini akan meminta data peneliti ke server menggunakan URL https://<alamat_ip>:<port>/peneliti.. Setelah permintaan dikirimkan, server akan mengirimkan kembali data berupa objek JSON. Data tersebut akan digunakan untuk instansiasi objek *cube* kedalam ruang 3-dimensi sehingga memiliki antarmuka diagram *scatter* dalam bentuk 3-dimensi seperti pada Gambar 4.37.


*Gambar 4.37 Tampilan Diagram Scatter dalam Bentuk 3-dimensi*

```
1.  public void GetScatterPlot()
2.    {
3.      GameObject[] dashboard = GameObject.FindGameObj
    ectsWithTag("PanelDashboard");
4.      GameObject[] legend = GameObject.FindGameObject
    sWithTag("Legend");
5.      int i = 0;
6.      foreach (GameObject panel in dashboard)
7.      {
8.
9.        inactivePanel[i] = panel;
10.       panel.SetActive(false);
11.       i += 1;
12.     }
13.
14.     if (legend.Length != 0)
15.     {
16.       inactivePanel[i] = legend[0];
17.       legend[0].SetActive(false);
18.     }
19.     if(loadScatter == false)
20.     {
21.
22.       connectionMessage.text = "Loading...";
23.       connectionMessagePanel.SetActive(true);
24.       RequestImageHandler getTexture = new RequestI
    mageHandler();
25.       getTexture.URL = URL + "/scatter";
```

```
26.       StartCoroutine(getTexture.RequestImageTexture
    ((textureImg) => {
27.         loadScatter = true;
28.         connectionMessagePanel.SetActive(false);
29.         Renderer scatterRenderer = scatterPlot.GetC
    omponent<Renderer>();
30.         scatterRenderer.material.renderQueue = 3001
    ;
31.         scatterRenderer.material.mainTexture = text
    ureImg;
32.         panelScatter.SetActive(true);
33.
34.       }, (msg) =>
35.       {
36.         if (msg != "")
37.         {
38.           retryMessage.text = msg;
39.           retry.SetActive(true);
40.           connectionMessagePanel.SetActive(false);

41.         }
42.       }
43.       ));
44.     }
45.     else
46.     {
47.       panelScatter.SetActive(true);
48.     }
49.   }
50.
51.   public void ThreeDScatter()
52.   {
53.     if(loadThreeDScatter == false)
54.     {
55.
56.       requestPeneliti.URL = URL + "/peneliti?";
57.       connectionMessage.text = "Loading...";
58.       connectionMessagePanel.SetActive(true);
59.       StartCoroutine(requestPeneliti.RequestData((r
    esult) => {
60.         float highestDocument = 0;
61.         List<int> totalDokumen = new List<int>();
```

```
62.         foreach (var data in result.data[0].data_pe
    neliti)
63.         {
64.           loadThreeDScatter = true;
65.           GameObject chartBar = (GameObject)Instant
    iate(Plot3D);
66.           chartBar.transform.SetParent(ParentThreeD
    ScatterPlot.transform);
67.           chartBar.transform.localPosition = new Ve
    ctor3((float)data.x * 3, (float)data.y * 3, 0);
68.           chartBar.transform.localScale = new Vecto
    r3(chartBar.transform.localScale.x * data.h_index_p
    eneliti / 6, chartBar.transform.localScale.x * data
    .h_index_peneliti / 6, chartBar.transform.localScal
    e.z * data.h_index_peneliti / 6);
69.           chartBar.name = data.nama_peneliti;
70.           totalDokumen.Add(data.dokumen_peneliti);

71.           if (highestDocument < data.dokumen_peneli
    ti)
72.           {
73.             highestDocument = data.dokumen_peneliti
    ;
74.           }
75.           EventTrigger evT = chartBar.GetComponent<
    EventTrigger>();
76.           EventTrigger.Entry newTriggerExit = new E
    ventTrigger.Entry();
77.           newTriggerExit.eventID = EventTriggerType
    .PointerExit;
78.           newTriggerExit.callback.AddListener((d) =
    > { ClearNamePlot(); });
79.           evT.triggers.Add(newTriggerExit);
80.         }
81.         int i = 0;
82.         foreach(Transform child in ParentThreeDScat
    terPlot.transform)
83.         {
84.           Debug.Log(child.name);
85.           int alpha = totalDokumen[i] % 255;
```

```
86.           child.gameObject.GetComponent<Renderer>()
    .material.SetColor("_Color", new Color32(50, 255, 2
    00, (byte)alpha));
87.           i += 1;
88.         }
89.         connectionMessagePanel.SetActive(false);
90.       }, error => {
91.         if (error != "")
92.         {
93.           retryMessage.text = error;
94.           retry.SetActive(true);
95.           connectionMessagePanel.SetActive(false);

96.         }
97.       }
98.       ));
99.     }
100.           else
101.           {
102.             ParentThreeDScatterPlot.SetActive(true
    );
103.           }
104.         }
```

*Kode Sumber 4.21 Fungsi Get Scatter Plot dan Three D Scatter*

### 4.3.2 Implementasi Aplikasi Server

Bagian ini akan menjelaskan mengenai implementasi aplikasi Realitas Virtual untuk visualisasi data peneliti pada sisi server. Server dalam aplikasi Realitas Virtual untuk visualisasi data peneliti ini berguna untuk pemrosesan data peneliti. Aplikasi *client* melakukan permintaan data kepada server. Server akan memproses data tersebut dan dikembalikan ke aplikasi *client* dalam bentuk JSON untuk ditampilkan.

Web server untuk aplikasi Realitas Virtual Visualisasi Data Peneliti ini dibangun menggunakan Flask *micro web framework* yang ditulis menggunakan bahasa pemrograman Python dan memiliki beberapa fungsi untuk API *end point* yaitu fungsi ***index(), peneliti(), peneliti(), detailPeneliti(), afiliasi(),***

***detailAfiliasi(), subjek(), detailSubjek(), scatter(), pieChartTexture dan voicerecognizer().***

#### 4.3.2.1 API *End Point* Beranda

Fungsi ***index()*** merupakan fungsi yang digunakan untuk API *end point* beranda pada aplikasi Realitas Virtual dengan URL https://<alamat_ip>:<port>/. Fungsi ini mengambil data beranda dengan menggunakan ORM untuk memudahkan dalam melakukan *query* data. Data tersebut disusun dan dikembalikan dalam bentuk objek JSON. Data-data tersebut meliputi afiliasi, rata-rata h index, total dokumen, total dokumen scopus dan total sitasi. Potongan fungsi ***index()*** dapat dilihat pada Kode Sumber 4.22 serta contoh serialisasi data ORM dapat dilihat pada kode sumber 4.23.

```
1.  @app.route('/',methods=['GET','POST'])
2.  def index():
3.      arrayQueryAfiliasi = []
4.      session = Session()
5.      dataAfiliasi = session.query(Afiliasi).all()
6.      session.commit()
7.
8.      session = Session()
9.      arrayQueryRerataHIndex = []
10.     subqueryRerataHIndex = session.query(Peneliti.i
    d_afiliasi, func.avg(Peneliti.h_index_peneliti).lab
    el('rata_rata_h_index')).group_by(Peneliti.id_afili
    asi).subquery()
11.     dataRerataHIndex = session.query(Afiliasi, subq
    ueryRerataHIndex.c.rata_rata_h_index).outerjoin(sub
    queryRerataHIndex, Afiliasi.id_afiliasi == subquery
    RerataHIndex.c.id_afiliasi).order_by(desc("rata_rat
    a_h_index")).all()
12.     session.commit()
13.     temp_row = None
14.
15.     session = Session()
16.     arrayQueryTotalDokumen = []
```

```
17.     subqueryTotalDokumen = session.query(Peneliti.id_afiliasi, func.sum(Peneliti.dokumen_peneliti).label('total_dokumen')).group_by(Peneliti.id_afiliasi).subquery()
18.     dataTotalDokumen = session.query(Afiliasi, subqueryTotalDokumen.c.total_dokumen).outerjoin(subqueryTotalDokumen, Afiliasi.id_afiliasi == subqueryTotalDokumen.c.id_afiliasi).order_by(desc("total_dokumen")).all()
19.     session.commit()
20.     temp_row = None
21.
22.     session = Session()
23.     arrayQueryTotalDokumenScopus = []
24.     subqueryTotalDokumenScopus = session.query(Peneliti.id_afiliasi, func.sum(Peneliti.dokumen_dalam_scopus_peneliti).label('total_dokumen_scopus')).group_by(Peneliti.id_afiliasi).subquery()
25.     dataTotalDokumenScopus = session.query(Afiliasi, subqueryTotalDokumenScopus.c.total_dokumen_scopus).outerjoin(subqueryTotalDokumenScopus, Afiliasi.id_afiliasi == subqueryTotalDokumenScopus.c.id_afiliasi).order_by(desc("total_dokumen_scopus")).all()
26.     session.commit()
27.
28.     session = Session()
29.     arrayQueryTotalSitasi = []
30.     subqueryTotalSitasi = session.query(Peneliti.id_afiliasi, func.sum(Peneliti.total_sitasi_peneliti).label('total_sitasi')).group_by(Peneliti.id_afiliasi).subquery()
31.     dataTotalSitasi = session.query(Afiliasi, subqueryTotalSitasi.c.total_sitasi).outerjoin(subqueryTotalSitasi, Afiliasi.id_afiliasi == subqueryTotalSitasi.c.id_afiliasi).order_by(desc("total_sitasi")).all()
32.     session.commit()
```

*Kode Sumber 4.22 Potongan Fungsi Index Aplikasi Server*

```
temp_row = None
for row in dataAfiliasi:
    if row.logo != '':
        logo = Base64Encoder(row.logo)
        logo = logo.decode('ascii')
    else:
        logo = ""
    x = Serialisasi(id_afiliasi = row.id_afiliasi,

            nama_afiliasi = row.universitas,
            logo_afiliasi = logo)
    temp_row = x.__dict__
    arrayQueryAfiliasi.append(temp_row)
session = Session()
```

*Kode Sumber 4.23 Contoh Penyusunan Data*

#### 4.3.2.2 API *End Point* Daftar Peneliti

Fungsi ***peneliti()*** merupakan fungsi yang digunakan untuk API *end point* daftar peneliti pada aplikasi Realitas Virtual dengan URL https://<alamat_ip>:<port>/peneliti. Fungsi ini mengambil data peneliti dengan menggunakan ORM untuk memudahkan dalam melakukan *query* data. Data tersebut disusun dan dikembalikan ke aplikasi *clien*t berupa objek JSON. Potongan fungsi ***peneliti()*** dapat dilihat pada Kode Sumber 4.24.

```
@app.route('/peneliti', methods=['GET'])
def peneliti():
    if urutkan == '':
        arrayQuery = []
        session = Session()
        data = session.query(Peneliti, func.dense_r
ank().over(
            order_by=Peneliti.h_index_peneliti.desc
()
            ).label('rnk')
            ).outerjoin(Keahlian, Keahlian.id_penel
iti == Peneliti.id_peneliti
```

```
            ).outerjoin(Afiliasi, Afiliasi.id_afiliasi == Peneliti.id_afiliasi
            ).outerjoin(DetailSubjekArea, DetailSubjekArea.id_peneliti == Peneliti.id_peneliti
            ).filter(or_(Keahlian.id_subjek_topik_tier_2 == keahlian, keahlian == ''), or_(Peneliti.id_afiliasi == id_afiliasi, id_afiliasi == ''), or_(DetailSubjekArea.id_subjek_area == id_subjek_area, id_subjek_area == ''), or_(or_(Peneliti.nama_peneliti.like(search), search == ''), or_(Peneliti.keyword_peneliti.like(search), search == ''))).order_by('rnk').all()
        session.commit()

    if urutkan == "hindex":
        arrayQuery = []
        session = Session()
        data = session.query(Peneliti, func.dense_rank().over(
            order_by=Peneliti.h_index_peneliti.desc()
            ).label('rnk')
            ).outerjoin(Keahlian, Keahlian.id_peneliti == Peneliti.id_peneliti
            ).outerjoin(Afiliasi, Afiliasi.id_afiliasi == Peneliti.id_afiliasi
            ).outerjoin(DetailSubjekArea, DetailSubjekArea.id_peneliti == Peneliti.id_peneliti
            ).filter(or_(Keahlian.id_subjek_topik_tier_2 == keahlian, keahlian == ''), or_(Peneliti.id_afiliasi == id_afiliasi, id_afiliasi == ''), or_(DetailSubjekArea.id_subjek_area == id_subjek_area, id_subjek_area == ''), or_(or_(Peneliti.nama_peneliti.like(search), search == ''), or_(Peneliti.keyword_peneliti.like(search), search == ''))).order_by(Peneliti.h_index_peneliti.desc()).all()
        session.commit()
    if urutkan == "nama":
        session = Session()
        arrayQuery = []
        data = session.query(Peneliti, func.dense_rank().over(
```

```
30.             order_by=Peneliti.h_index_peneliti.desc
    ()
31.             ).label('rnk')
32.             ).outerjoin(Keahlian, Keahlian.id_penel
    iti == Peneliti.id_peneliti
33.             ).outerjoin(Afiliasi, Afiliasi.id_afili
    asi == Peneliti.id_afiliasi
34.             ).outerjoin(DetailSubjekArea, DetailSub
    jekArea.id_peneliti == Peneliti.id_peneliti
35.             ).filter(or_(Keahlian.id_subjek_topik_t
    ier_2 == keahlian, keahlian == ''), or_(Peneliti.id
    _afiliasi == id_afiliasi, id_afiliasi == ''), or_(D
    etailSubjekArea.id_subjek_area == id_subjek_area, i
    d_subjek_area == ''), or_(or_(Peneliti.nama_penelit
    i.like(search), search == ''), or_(Peneliti.keyword
    _peneliti.like(search), search == ''))).order_by(Pe
    neliti.nama_peneliti.asc()).all()
36.         session.commit()
37.
38.     if urutkan == "dokumen":
39.         arrayQuery = []
40.         session = Session()
41.         data = session.query(Peneliti, func.dense_r
    ank().over(
42.             order_by=Peneliti.h_index_peneliti.desc
    ()
43.             ).label('rnk')
44.             ).outerjoin(Keahlian, Keahlian.id_penel
    iti == Peneliti.id_peneliti
45.             ).outerjoin(Afiliasi, Afiliasi.id_afili
    asi == Peneliti.id_afiliasi
46.             ).outerjoin(DetailSubjekArea, DetailSub
    jekArea.id_peneliti == Peneliti.id_peneliti
47.             ).filter(or_(Keahlian.id_subjek_topik_t
    ier_2 == keahlian, keahlian == ''), or_(Peneliti.id
    _afiliasi == id_afiliasi, id_afiliasi == ''), or_(D
    etailSubjekArea.id_subjek_area == id_subjek_area, i
    d_subjek_area == ''), or_(or_(Peneliti.nama_penelit
    i.like(search), search == ''), or_(Peneliti.keyword
    _peneliti.like(search), search == ''))).order_by(Pe
    neliti.dokumen_peneliti.desc()).all()
48.         session.commit()
```

```
49.
50.     if urutkan == "dokumenscopus":
51.         arrayQuery = []
52.         session = Session()
53.         data = session.query(Peneliti, func.dense_r
    ank().over(
54.             order_by=Peneliti.h_index_peneliti.desc
    ()
55.             ).label('rnk')
56.             ).outerjoin(Keahlian, Keahlian.id_penel
    iti == Peneliti.id_peneliti
57.             ).outerjoin(Afiliasi, Afiliasi.id_afili
    asi == Peneliti.id_afiliasi
58.             ).outerjoin(DetailSubjekArea, DetailSub
    jekArea.id_peneliti == Peneliti.id_peneliti
59.             ).filter(or_(Keahlian.id_subjek_topik_t
    ier_2 == keahlian, keahlian == ''), or_(Peneliti.id
    _afiliasi == id_afiliasi, id_afiliasi == ''), or_(D
    etailSubjekArea.id_subjek_area == id_subjek_area, i
    d_subjek_area == ''), or_(or_(Peneliti.nama_penelit
    i.like(search), search == ''), or_(Peneliti.keyword
    _peneliti.like(search), search == ''))).order_by(Pe
    neliti.dokumen_dalam_scopus_peneliti.desc()).all()

60.         session.commit()
61.
62.     if urutkan == "sitasi":
63.         arrayQuery = []
64.         session = Session()
65.         data = session.query(Peneliti, func.dense_r
    ank().over(
66.             order_by=Peneliti.h_index_peneliti.desc
    ()
67.             ).label('rnk')
68.             ).outerjoin(Keahlian, Keahlian.id_penel
    iti == Peneliti.id_peneliti
69.             ).outerjoin(Afiliasi, Afiliasi.id_afili
    asi == Peneliti.id_afiliasi
70.             ).outerjoin(DetailSubjekArea, DetailSub
    jekArea.id_peneliti == Peneliti.id_peneliti
71.             ).filter(or_(Keahlian.id_subjek_topik_t
    ier_2 == keahlian, keahlian == ''), or_(Peneliti.id
```

```
    _afiliasi == id_afiliasi, id_afiliasi == ''), or_(D
    etailSubjekArea.id_subjek_area == id_subjek_area, i
    d_subjek_area == ''), or_(or_(Peneliti.nama_penelit
    i.like(search), search == ''), or_(Peneliti.keyword
    _peneliti.like(search), search == ''))).order_by(Pe
    neliti.total_sitasi_peneliti.desc()).all()
72.         session.commit()
```

*Kode Sumber 4.24 Fungsi Peneliti Aplikasi Server*

#### 4.3.2.3 API *End Point* Detail Peneliti

Fungsi ***detailPeneliti()*** merupakan fungsi yang digunakan untuk API *end point* detail peneliti pada aplikasi realitas virtual dengan URL https://<alamat_ip>:<port>/detail-peneliti. Fungsi ini mengambil data detail peneliti dengan menggunakan ORM untuk memudahkan dalam melakukan *query* data. Data tersebut disusun dan dikembalikan ke aplikasi *clien*t berupa objek JSON. Fungsi ***detailPeneliti()*** dapat dilihat pada Kode Sumber 4.25.

```
1.  @app.route('/detail-peneliti', methods=['GET'])
2.  def detailPeneliti():
3.    try:
4.      if request.method == 'GET':
5.        session = Session()
6.        idPeneliti = request.args.get('id_peneliti')
7.        arrayQuery = []
8.        data = session.query(Peneliti).filter(Penelit
    i.id_peneliti == idPeneliti).all()
9.        session.commit()
10.       data_len = len(data)
11.       for row in data:
12.         x = Serialisasi(id_peneliti = row.id_peneli
    ti,
13.           kode_peneliti = row.kode_peneliti,
14.           nama_peneliti = row.nama_peneliti,
15.           dokumen_peneliti = row.dokumen_peneliti,
```

```
16.           dokumen_peneliti_dalam_scopus = row.dokum
    en_dalam_scopus_peneliti,
17.           h_index_peneliti = row.h_index_peneliti,

18.           total_sitasi_peneliti = row.total_sitasi_
    peneliti,
19.           x = row.x,
20.           y = row.y,
21.           keyword_peneliti = row.keyword_peneliti,

22.           show_centroid_peneliti = row.show_centroi
    d_peneliti,
23.           id_departemen = row.id_departemen,
24.           nama_afiliasi = row.ambilAfiliasi.univers
    itas,
25.           nama_departemen = row.ambilDepartemen.nam
    a_departemen,
26.           nama_fakultas = row.ambilDepartemen.ambil
    Fakultas.nama_fakultas)
27.         temp_row = x.__dict__
28.         array_temp = []
29.         for detail_subjek_area in row.ambilDetailSu
    bjekArea:
30.           y = Serialisasi(id_detail_subjek_area = d
    etail_subjek_area.id_detail_subjek_area, id_subjek_
    area = detail_subjek_area.id_subjek_area, nama_subj
    ek_area = detail_subjek_area.ambilSubjekArea.nama_s
    ubjek_area)
31.           array_temp.append(y.__dict__)
32.         temp_row['detail_subjek_area'] = array_temp
33.
34.         array_temp = []
35.         for keahlian in row.ambilKeahlian:
36.           y = Serialisasi(id_keahlian = keahlian.id
    _keahlian,
37.           id_subjek_topik_tier_2 = keahlian.id_subj
    ek_topik_tier_2,
38.           nama_keahlian_tier_2 = keahlian.ambilSubj
    ekTopikTier2.kelas_tier_2,
39.           total_dokumen = keahlian.jumlah_dokumen,
```

```
40.           id_subjek_topik_tier_1 = keahlian.ambilSu
    bjekTopikTier2.ambilSubjekTopikTier1.id_subjek_topi
    k_tier_1,
41.           nama_keahlian_tier_1 = keahlian.ambilSubj
    ekTopikTier2.ambilSubjekTopikTier1.kelas_tier_1
42.           )
43.           array_temp.append(y.__dict__)
44.         temp_row['keahlian_peneliti'] = array_temp
45.
46.         arrayQuery.append(temp_row)
47.
48.       dataDetailPeneliti = {}
49.       dataDetailPeneliti['detail_peneliti'] = array
    Query
50.       all_data =[dataDetailPeneliti]
51.
52.       response = ResponseData("200","Data Berhasil
    Ditemukan", all_data, data_len)
53.       response = json.dumps(response.__dict__)
54.
55.     return str(response)
56.
57.   except Exception as e:
58.     if hasattr(e, 'message'):
59.       data = [e]
60.       data_len = ""
61.       response = ResponseData("500","Terjadi Kesala
    han Pada Server",data, data_len)
62.       response = json.dumps(response.__dict__)
63.       return str(response)
64.     else:
65.       data = []
66.       data_len = ""
67.       response = ResponseData("500","Terjadi Kesala
    han Pada Server",data, data_len)
68.       response = json.dumps(response.__dict__)
69.       return str(response)
```

*Kode Sumber 4.25 Fungsi Detail Peneliti Aplikasi Server*

#### 4.3.2.4 API *End Point* Afiliasi

Fungsi ***afiliasi()*** merupakan fungsi yang digunakan untuk API *end point* daftar afiliasi pada aplikasi realitas virtual dengan URL https://<alamat ip>:<port>/afiliasi. Fungsi ini mengambil data afiliasi dengan menggunakan ORM untuk memudahkan dalam melakukan *query* data. Data tersebut disusun dan dikembalikan ke aplikasi *clien*t berupa objek JSON. Fungsi ***afiliasi()*** dapat dilihat pada Kode Sumber 4.26.

```
1.  @app.route('/afiliasi', methods=['GET'])
2.  def afiliasi():
3.    try:
4.      session = Session()
5.      sub_query = session.query(Peneliti, func.sum(Pe
    neliti.dokumen_peneliti).label('total_dokumen'), fu
    nc.sum(Peneliti.dokumen_dalam_scopus_peneliti).labe
    l('total_dokumen_scopus'), func.sum(Peneliti.total_
    sitasi_peneliti).label('total_sitasi'), func.avg(Pe
    neliti.h_index_peneliti).label('rata_rata_h_index')
    , func.count(Peneliti.id_peneliti).label('jumlah_pe
    neliti')).group_by(Peneliti.id_afiliasi).subquery()

6.      data = session.query(Afiliasi, sub_query.c.tota
    l_dokumen, sub_query.c.total_dokumen_scopus, sub_qu
    ery.c.total_sitasi, sub_query.c.rata_rata_h_index,
    sub_query.c.jumlah_peneliti).outerjoin(sub_query, s
    ub_query.c.id_afiliasi == Afiliasi.id_afiliasi).gro
    up_by(Afiliasi.id_afiliasi).all()
7.      session.commit()
8.      data_len = len(data)
9.      arrayQuery = []
10.     temp_row = None
11.     for data in data:
12.       x = Serialisasi(id_afiliasi = data[0].id_afil
    iasi,
13.       nama_afiliasi = data[0].universitas,
14.       kota_afiliasi = data[0].kota,
15.       negara_afiliasi = data[0].negara,
16.       )
```

```
17.       temp_row = x.__dict__
18.       arrayQuery.append(temp_row)
19.
20.     dataAfiliasi = {}
21.     dataAfiliasi['afiliasi'] = arrayQuery
22.     all_data =[dataAfiliasi]
23.
24.     response = ResponseData("200","Data Berhasil Di
    temukan", all_data, data_len)
25.     response = json.dumps(response.__dict__)
26.     return str(response)
27.
28.   except Exception as e:
29.     if hasattr(e, 'message'):
30.       data = [e]
31.       data_len = ""
32.       response = ResponseData("500","Terjadi Kesala
    han Pada Server",data, data_len)
33.       response = json.dumps(response.__dict__)
34.       return str(response)
35.     else:
36.       data = []
37.       data_len = ""
38.       response = ResponseData("500","Terjadi Kesala
    han Pada Server",data, data_len)
39.       response = json.dumps(response.__dict__)
40.       return str(response)
```

*Kode Sumber 4.26 Fungsi Afiliasi Aplikasi Server*

#### 4.3.2.5 API *End Point* Detail Afiliasi

Fungsi ***detailAfiliasi()*** merupakan fungsi yang digunakan untuk API *end point* detail afiliasi pada aplikasi realitas virtual dengan URL https://<alamat_ip>:<port>/detail-afiliasi. Fungsi ini mengambil data detail afiliasi pada basis data. Data tersebut disusun dan dikembalikan ke aplikasi *clien*t berupa objek JSON. Fungsi ***detailAfiliasi()*** dapat dilihat pada Kode Sumber 4.27.

```
@app.route('/detail-afiliasi', methods=['GET'])
def detailAfiliasi():
  try:
    if request.method == 'GET':
      session = Session()
      id_afiliasi = request.args.get('id_afiliasi')

      sub_query = session.query(Peneliti, func.sum(Peneliti.dokumen_peneliti).label('total_dokumen'), func.sum(Peneliti.dokumen_dalam_scopus_peneliti).label('total_dokumen_scopus'), func.sum(Peneliti.total_sitasi_peneliti).label('total_sitasi'), func.avg(Peneliti.h_index_peneliti).label('rata_rata_h_index'), func.count(Peneliti.id_peneliti).label('jumlah_peneliti')).group_by(Peneliti.id_afiliasi).subquery()
      data = session.query(Afiliasi, sub_query.c.total_dokumen, sub_query.c.total_dokumen_scopus, sub_query.c.total_sitasi, sub_query.c.rata_rata_h_index, sub_query.c.jumlah_peneliti).outerjoin(sub_query, sub_query.c.id_afiliasi == Afiliasi.id_afiliasi).group_by(Afiliasi.id_afiliasi).filter(Afiliasi.id_afiliasi == id_afiliasi).all()
      session.commit()
      data_len = len(data)
      arrayQuery = []
      temp_row = None
      for data in data:
        x = Serialisasi(id_afiliasi = data[0].id_afiliasi,
        nama_afiliasi = data[0].universitas,
        kota_afiliasi = data[0].kota,
        negara_afiliasi = data[0].negara,
        total_dokumen = str(data[1]),
        total_dokumen_scopus = str(data[2]),
        rata_rata_h_index = str(data[4]),
        total_sitasi = str(data[3]),
        jumlah_peneliti_afiliasi = str(data[5])
        )
        temp_row = x.__dict__
        arrayQuery.append(temp_row)

```

```
27.       dataAfiliasi = {}
28.       dataAfiliasi['detail_afiliasi'] = arrayQuery

29.       all_data =[dataAfiliasi]
30.
31.       response = ResponseData("200","Data Berhasil
    Ditemukan", all_data, data_len)
32.       response = json.dumps(response.__dict__)
33.       return str(response)
34.
35.   except Exception as e:
36.     if hasattr(e, 'message'):
37.       data = [e]
38.       data_len = ""
39.       response = ResponseData("500","Terjadi Kesala
    han Pada Server",data, data_len)
40.       response = json.dumps(response.__dict__)
41.       return str(response)
42.     else:
43.       data = []
44.       data_len = ""
45.       response = ResponseData("500","Terjadi Kesala
    han Pada Server",data, data_len)
46.       response = json.dumps(response.__dict__)
47.       return str(response)
```

*Kode Sumber 4.27 Fungsi Detail Afiliasi Aplikasi Server*

#### 4.3.2.6 API *End Point* Subjek Penelitian

Fungsi ***subjek()*** merupakan fungsi yang digunakan untuk API *end point* daftar subjek penelitian pada aplikasi Realitas Virtual dengan URL https://<alamat_ip>:<port>/subjek. Fungsi ini mengambil data subjek penelitian dengan menggunakan ORM untuk memudahkan dalam melakukan *query* data. Data tersebut disusun dan dikembalikan ke aplikasi *clien*t berupa objek JSON. Fungsi ***subjek()*** dapat dilihat pada Kode Sumber 4.28.

```
1.  @app.route('/subjek', methods=['GET'])
2.  def subjek():
```

```
3.    try:
4.      arrayQuery = []
5.      session = Session()
6.      subquery1 = session.query(Peneliti, func.sum(Peneliti.dokumen_peneliti).label('total_dokumen'),func.sum(Peneliti.dokumen_dalam_scopus_peneliti).label('total_dokumen_scopus'), func.sum(Peneliti.total_sitasi_peneliti).label('total_sitasi'), func.avg(Peneliti.h_index_peneliti).label('rata_rata_h_index'), func.count(Peneliti.id_peneliti).label('jumlah_peneliti')).group_by(Peneliti.id_peneliti).subquery()
7.      subquery2 = session.query(DetailSubjekArea, subquery1.c.total_dokumen, subquery1.c.total_dokumen_scopus, subquery1.c.total_sitasi, subquery1.c.rata_rata_h_index, subquery1.c.jumlah_peneliti).outerjoin(subquery1, subquery1.c.id_peneliti == DetailSubjekArea.id_peneliti).group_by(DetailSubjekArea.id_detail_subjek_area).subquery()
8.      data = session.query(SubjekArea, func.sum(subquery2.c.total_dokumen).label('total_dokumen'), func.sum(subquery2.c.total_dokumen_scopus).label('total_dokumen_scopus'), func.sum(subquery2.c.total_sitasi).label('total_sitasi'), func.avg(subquery2.c.rata_rata_h_index).label('rata_rata_h_index'), func.count(subquery2.c.jumlah_peneliti)).outerjoin(subquery2, subquery2.c.id_subjek_area == SubjekArea.id_subjek_area).group_by(SubjekArea.id_subjek_area).all()
9.      session.commit()
10.     data_len = len(data)
11.     for data in data:
12.       x = Serialisasi(id_subjek_area = data[0].id_subjek_area, nama_subjek_area = data[0].nama_subjek_area)
13.       arrayQuery.append(x.__dict__)
14.
15.     dataSubjek = {}
16.     dataSubjek['data_subjek_area'] = arrayQuery
17.     all_data =[dataSubjek]
18.
19.     response = ResponseData("200","Data Berhasil Ditemukan", all_data, data_len)
20.     response = json.dumps(response.__dict__)
```

```
21.     return str(response)
22.
23.   except Exception as e:
24.     if hasattr(e, 'message'):
25.       data = [e]
26.       data_len = ""
27.       response = ResponseData("500","Terjadi Kesala
    han Pada Server",data, data_len)
28.       response = json.dumps(response.__dict__)
29.       return str(response)
30.     else:
31.       data = []
32.       data_len = ""
33.       response = ResponseData("500","Terjadi Kesala
    han Pada Server",data, data_len)
34.       response = json.dumps(response.__dict__)
35.       return str(response)
```

*Kode Sumber 4.28 Fungsi Subjek Aplikasi Server*

#### 4.3.2.7 API *End Point* Detail Subjek Penelitian

Fungsi ***detailSubjek()*** merupakan fungsi yang digunakan untuk API *end point* detail subjek pada aplikasi Realitas Virtual dengan URL https://<alamat_ip>:<port>/detail-subjek. Fungsi ini mengambil data detail subjek dengan menggunakan ORM untuk memudahkan dalam melakukan *query* data. Data tersebut disusun dan dikembalikan ke aplikasi *clien*t berupa objek JSON. Fungsi ***detailSubjek()*** dapat dilihat pada Kode Sumber 4.29.

```
1.  @app.route('/detail-subjek', methods=['GET'])
2.  def detailSubjek():
3.    try:
4.      if request.method == 'GET':
5.        id_subjek = request.args.get('id_subjek')
6.        arrayQuery = []
7.        session = Session()
8.        subquery1 = session.query(Peneliti, func.sum(
    Peneliti.dokumen_peneliti).label('total_dokumen'),f
    unc.sum(Peneliti.dokumen_dalam_scopus_peneliti).lab
```

```
el('total_dokumen_scopus'), func.sum(Peneliti.total_sitasi_peneliti).label('total_sitasi'), func.avg(Peneliti.h_index_peneliti).label('rata_rata_h_index'), func.count(Peneliti.id_peneliti).label('jumlah_peneliti')).group_by(Peneliti.id_peneliti).subquery()
9.        subquery2 = session.query(DetailSubjekArea, subquery1.c.total_dokumen, subquery1.c.total_dokumen_scopus, subquery1.c.total_sitasi, subquery1.c.rata_rata_h_index, subquery1.c.jumlah_peneliti).outerjoin(subquery1, subquery1.c.id_peneliti == DetailSubjekArea.id_peneliti).group_by(DetailSubjekArea.id_detail_subjek_area).subquery()
10.       data = session.query(SubjekArea, func.sum(subquery2.c.total_dokumen).label('total_dokumen'), func.sum(subquery2.c.total_dokumen_scopus).label('total_dokumen_scopus'), func.sum(subquery2.c.total_sitasi).label('total_sitasi'), func.avg(subquery2.c.rata_rata_h_index).label('rata_rata_h_index'), func.count(subquery2.c.jumlah_peneliti)).outerjoin(subquery2, subquery2.c.id_subjek_area == SubjekArea.id_subjek_area).group_by(SubjekArea.id_subjek_area).filter(SubjekArea.id_subjek_area == id_subjek).all()
11.       session.commit()
12.       data_len = len(data)
13.
14.       for data in data:
15.         x = Serialisasi(id_subjek_penelitian = data[0].id_subjek_area,
16.           nama_subjek_penelitian = data[0].nama_subjek_area,
17.           total_dokumen = str(data[1]),
18.           total_dokumen_scopus = str(data[2]),
19.           rata_rata_h_index = str(data[4]),
20.           total_sitasi = str(data[3]),
21.           jumlah_peneliti_subjek = str(data[5]))
22.           arrayQuery.append(x.__dict__)
23.
24.       dataSubjek = {}
25.       dataSubjek['detail_subjek_penelitian'] = arrayQuery
26.       all_data =[dataSubjek]
```

```
27.
28.       response = ResponseData("200","Data Berhasil
    Ditemukan", all_data, data_len)
29.       response = json.dumps(response.__dict__)
30.       return str(response)
31.
32.   except Exception as e:
33.     if hasattr(e, 'message'):
34.       data = [e]
35.       data_len = ""
36.       response = ResponseData("500","Terjadi Kesala
    han Pada Server",data, data_len)
37.       response = json.dumps(response.__dict__)
38.       return str(response)
39.     else:
40.       data = []
41.       data_len = ""
42.       response = ResponseData("500","Terjadi Kesala
    han Pada Server",data, data_len)
43.       response = json.dumps(response.__dict__)
44.       return str(response)
```

*Kode Sumber 4.29 Fungsi Detail Subjek Aplikasi Server*

#### 4.3.2.8 API *End Point* Tekstur Diagram *Scatter*

Fungsi ***scatter()*** merupakan fungsi yang digunakan untuk membuat tekstur diagram *scatter*. Fungsi ini berada pada API *end point* dengan URL https://<alamat_ip>:<port>/scatter. Fungsi ini mengambil data peneliti kemudian digunakan dalam membuat diagram *scatter* berdasarkan h index, total dokumen, serta berdasarkan koordinat x dan y. Data peneliti tersebut diambil dengan menggunakan ORM untuk memudahkan dalam melakukan *query* data. Gambar tersebut dikembalikan ke aplikasi *clien*t yang nantinya akan digunakan sebagai tekstur. Fungsi ***scatter()*** dapat dilihat pada Kode Sumber 4.30.

```
1.  @app.route('/scatter', methods=['GET'])
2.  def scatter():
```

```
    x = []
    y = []
    colors = []
    sizes = []
    session = Session()
    query = session.query(Peneliti).all()
    session.commit()
    for data in query:
      x.append(data.x)
      y.append(data.y)
      colors.append(data.dokumen_peneliti * 4)
      sizes.append(data.h_index_peneliti * 4)
    i = 0
    for i in range(2):
      plt.scatter(x, y, c=colors, s=sizes, alpha=1,

            cmap='viridis', edgecolors='white')
      plt.rcParams.update({
      "lines.color": "white",
      "patch.edgecolor": "white",
      "text.color": "white",
      "axes.facecolor": "white",
      "axes.edgecolor": "lightgray",
      "axes.labelcolor": "white",
      "xtick.color": "white",
      "ytick.color": "white",
      "grid.color": "white",
      "figure.facecolor": "black",
      "figure.edgecolor": "black",
      "savefig.facecolor": "black",
      "savefig.edgecolor": "black"})
      plt.colorbar();  # show color scale
      plt.savefig('scatter.png', transparent=True)

      filedir = './scatter.png'
      plt.close()
    return send_file(filedir, mimetype='image/png')
```

*Kode Sumber 4.30 Fungsi Scatter Aplikasi Server*

#### 4.3.2.9 API *End Point* Tekstur Diagram *Pie Chart*

Fungsi ***pieChartTexture()*** merupakan fungsi yang digunakan untuk membuat tekstur diagram *pie*. Fungsi ini berada pada API *end point* dengan URL https://<alamat_ip>:<port>/piechart-texture. Fungsi ini mengambil data rasio dokumen dan dokumen scopus kemudian digunakan dalam membuat diagram pie berupa gambar. Gambar tersebut dikembalikan ke aplikasi *clien*t yang nantinya akan digunakan sebagai tekstur. Fungsi ***pieChartTexture()*** dapat dilihat pada Kode Sumber 4.31.

```
1.  @app.route('/piechart-texture', methods=['GET'])
2.  def piechartTexture():
3.      try:
4.          scopus = request.args.get('scopus')
5.          total = request.args.get('total')
6.          labels = 'Scopus', 'Lain-lain',
7.          sizes = [int(scopus), int(total) - int(scop
    us)]
8.          explode = (0, 0.05)
9.          _fig1, ax1 = plt.subplots(figsize=(5, 5))
10.         _patches, texts, autotexts = ax1.pie(sizes,
     explode=explode, labels=labels, autopct='%1.2f%%',

11.                 shadow=True, startangle=90, labeldi
    stance=0.8, textprops={'color': 'white'})
12.         title_obj = plt.title('Rasio Dokumen Publik
    asi')
13.         plt.setp(title_obj, color='w')
14.         plt.setp(title_obj, fontsize= 20)
15.         plt.savefig('temp.png', transparent=True)
16.         filedir = './temp.png'
17.         plt.close()
18.
19.         return send_file(filedir, mimetype='image/p
    ng')
20.
21.     except Exception as e:
22.         if hasattr(e, 'message'):
```

```
            data = [e]
            data_len = ""
            response = ResponseData("500","Terjadi
Kesalahan Pada Server",data, data_len)
            response = json.dumps(response.__dict__
)
            return str(response)
        else:
            data = []
            data_len = ""
            response = ResponseData("500","Terjadi
Kesalahan Pada Server",data, data_len)
            response = json.dumps(response.__dict__
)
            return str(response)
```

*Kode Sumber 4.31 Fungsi Pie Chart Texture Aplikasi Server*

#### 4.3.2.10 API *End Point* Perintah Suara

Fungsi ***voicerecoginzer()*** merupakan fungsi yang digunakan dalam perintah suara untuk memproses suara kedalam bentuk teks. Fungsi ini memanfaatkan API google untuk menerjemahkan suara kedalam bentuk teks. Teks ini nantinya digunakan sebagai kata kunci pencarian peneliti. Fungsi ini berada pada API *end point* dengan URL https://<alamat_ip>:<port>/voicerecognizer. Fungsi ***voicerecognizer()*** dapat dilihat pada Kode Sumber 4.32.

```
@app.route('/voicerecognizer',methods=['GET','POST'
])
def voicerecognizer():
    try:
        if request.method == 'POST':
            audio = request.files['audio-file']
            filePath = "./audio"
            audio.save(filePath)
            r = sr.Recognizer()
            with sr.AudioFile(filePath) as source:
```

```
10.                 audio = r.listen(source)
11.                 text = r.recognize_google(audio,lan
    guage="id-ID")
12.                 text_list = {}
13.                 text_list["hasil_audio"] = text.rep
    lace(" ","")
14.                 data = [text_list]
15.                 response = ResponseData("200", "Aud
    io berhasil diproses",data, len(text_list))
16.                 response = json.dumps(response.__di
    ct__)
17.             return str(response)
18.
19.     except Exception as e:
20.         if hasattr(e, 'message'):
21.             data = [e]
22.             data_len = ""
23.             response = ResponseData("500","Terjadi
    Kesalahan Pada Server",data, data_len)
24.             response = json.dumps(response.__dict__
    )
25.             return str(response)
26.         else:
27.             data = []
28.             data_len = ""
29.             response = ResponseData("500","Tidak Da
    pat Mendengar Suara Anda",data, data_len)
30.             response = json.dumps(response.__dict__
    )
31.             return str(response)
```

*Kode Sumber 4.32 Fungsi Voice Recognizer Aplikasi Server*

# BAB V
# PENGUJIAN DAN EVALUASI

Tahap pengujian merupakan tahap untuk menguji apakah aplikasi sudah sesuai dengan apa yang diharapkan. Dalam tahap ini dilakukan pengujian fungsionalitas perangkat lunak apakah aplikasi Realitas Virtual ini terdapat error atau bug dan juga penggujian terhadap pengguna apakah pengguna mampu memahami antarmuka pada aplikasi Realitas Virtual ini.

## 5.1 Pengujian Fungsional

Tahap pengujian fungsional dilakukan dengan menguji aplikasi apakah aplikasi ini dapat berjalan secara funsional. Dalam pengujian aplikasi ini menggunakan perangkat Redmi Note 7 dengan sistem operasi Android 9.0 (Pie). Selain itu, pengujian ini dilakukan apakah secara fungsional aplikasi dapat berjalan dengan baik.

### 5.1.1 Lingkungan Operasi Sistem Pengujian Fungsional

Linkungan operasi yang digunakan untuk pengujian aplikasi Realitas Virtual ini ditunjukkan pada Tabel 5.1.

*Tabel 5.1 Lingkungan Operasi Pengujian Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

| Perangkat | Spesifikasi |
|---|---|
| Perangkat Server | Sistem Operasi Server: Windows 10<br>Prosesor Server: Intel(R) Core(TM) i7 6700HQ CPU<br>GPU Server: NVIDIA Geforce GTX 950, 4GB VRAM<br>Memori Server : 8 GB RAM<br>Penyimpanan Server : 1TB HDD & 240 GB SSD |

| Perangkat | Spesifikasi |
|---|---|
| Perangkat Klien | Sistem Operasi Klien : Android 9.0 (Pie)<br>Chipset Klien : Qualcomm SDM660 Snapdragon 660 (14 nm)<br>CPU Klien : Octa-core (4x2.2 GHz Kryo 260 Gold & 4x1.8 GHz Kryo 260 Silver)<br>GPU Klien : Adreno 512<br>Memori Klien : 4GB RAM<br>Penyimpanan Klien : 64GB Internal |

### 5.1.2 Skenario Pengujian Fungsionalitas

. Dalam tahap ini dilakukan pengujian fungsionalitas perangkat lunak apakah aplikasi Realitas Virtual ini terdapat *error* atau *bug*. Metode pengujian fungsional pada aplikasi Realitas Virtual untuk visualisasi data peneliti ini adalah mencoba masing-masing fitur apakah fitur berjalan sesuai dengan apa yang diharapkan menggunakan metode *blackbox* dengan skenario yang telah ditentukan mengacu pada subbab 3.1. Skenario uji coba fungsionalitas yang dilakukan terhadap aplikasi yang dibangun dijelaskan pada Tabel 5.2.

*Tabel 5.2 Skenario Uji Coba Fungsionalitas*

| Kode Uji Coba | Deskripsi Uji Coba |
|---|---|
| UF-001 | Uji coba pada panel navigasi |
| UF-002 | Uji coba pada beranda |
| UF-003 | Uji coba pada daftar peneliti |
| UF-004 | Uji coba pada daftar subjek penelitian |
| UF-005 | Uji coba pada detail subjek penelitian |
| UF-006 | Uji coba pada daftar afiliasi |
| UF-007 | Uji coba pada detail afiliasi |
| UF-008 | Uji coba pada diagram *scatter* |

Setiap skenario akan dijelaskan mengenai kondisi awal, masukkan, dan keluaran yang diharapkan, kondisi akhir, dan hasil uji coba. Berikut ini merupakan penjabaran hasil setiap uji coba yang dilakukan.

#### 5.1.2.1 Uji Coba pada Panel Navigasi

Pada bagian ini akan dijelaskan mengenai pengujian terhadap panel navigasi dengan tujuan apakah tombol tiap panel navigasi berfungsi dengan baik. Pengujian ini dilakukan dengan mendeskripsikan masukan, keluaran, hasil yang dicapai dan kondisi akhir.

Pada panel navigasi, tombol yang akan diuji adalah tombol Beranda, Peneliti, Subjek Penelitian, Afiliasi dan Diagram *Scatter*. Skenario uji dapat dilihat pada Tabel 5.3.

*Tabel 5.3 Hasil Uji Coba pada Panel Navigasi*

| **ID** | **UF-001** |
|---|---|
| Nama | Uji coba pada panel navigasi |
| Tujuan uji coba | Pengguna mengetahui fungsionalitas tombol yang ada pada panel navigasi |
| Kondisi awal | Pengguna berada pada panel lain |
| ***Skenario 1*** | ***Pengguna memilih tombol Beranda*** |
| Masukan | Memilih tombol *beranda* pada dunia virtual |
| Keluaran yang diharapkan | Pengguna berpindah ke halaman beranda |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada halaman beranda |
| ***Skenario 2*** | ***Pengguna memilih tombol Peneliti*** |
| Masukan | Memilih tombol *peneliti* pada dunia virtual |
| Keluaran yang diharapkan | Muncul panel daftar peneliti |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada halaman daftar peneliti |
| ***Skenario 3*** | ***Pengguna memilih tombol Subjek Penelitian*** |
| Masukan | Menekan tombol subjek penelitian pada dunia virtual |
| Keluaran yang diharapkan | Muncul panel daftar subjek penelitian |
| Hasil uji coba | Berhasil |

| | |
|---|---|
| Kondisi Akhir | Pengguna berada pada halaman daftar subjek penelitian |
| ***Skenario 4*** | ***Pengguna memilih tombol Afiliasi*** |
| Masukan | Menekan tombol afiliasi pada dunia virtual |
| Keluaran yang diharapkan | Muncul panel daftar afiliasi |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada halaman daftar afiliasi |
| ***Skenario 5*** | ***Pengguna memilih tombol Diagram Scatter*** |
| Masukan | Menekan tombol diagram *scatter* pada dunia virtual |
| Keluaran yang diharapkan | Muncul panel diagram *scatter* |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada halaman diagram *scatter* |

#### 5.1.2.2 Uji Coba pada Beranda

Pada bagian ini akan dijelaskan mengenai pengujian terhadap panel yang terdapat pada beranda dengan tujuan apakah panel pada beranda berfungsi dengan baik. Pengujian ini dilakukan dengan mendeskripsikan masukan, keluaran, hasil yang dicapai dan kondisi akhir.

Pada beranda, terdapat beberapa panel yang akan diuji yaitu panel diagram batang rata-rata h-index, diagram batang total dokumen, diagram batang total dokumen dalam scopus dan diagram batang total sitasi. Skenario uji dapat dilihat pada Tabel 5.4.

*Tabel 5.4 Hasil Uji Coba pada Beranda*

| ID | UF-002 |
|---|---|
| Nama | Uji coba pada beranda |
| Tujuan uji coba | Pengguna mengetahui fungsionalitas panel yang ada pada beranda |

| | |
|---|---|
| Kondisi awal | Pengguna berada pada beranda |
| ***Skenario 1*** | ***Pengguna memilih panel Diagram Batang Rata-Rata H-Index*** |
| Masukan | Memilih tombol *panel diagram batang rata-rata h-index* pada dunia virtual |
| Keluaran yang diharapkan | Pengguna berpindah ke panel diagram batang rata-rata h-index |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel diagram batang rata-rata h-index |
| ***Skenario 2*** | ***Pengguna memilih panel Diagram Batang Total Dokumen*** |
| Masukan | Memilih tombol *panel diagram total dokumen* pada dunia virtual |
| Keluaran yang diharapkan | Pengguna berpindah ke panel diagram batang total dokumen |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel diagram batang total dokumen |
| ***Skenario 3*** | ***Pengguna memilih panel Diagram Batang Total Dokumen dalam Scopus*** |
| Masukan | Memilih tombol *panel diagram total dokumen dalam Scopus* pada dunia virtual |
| Keluaran yang diharapkan | Pengguna berpindah ke panel diagram batang total dokumen dalam scopus |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel diagram batang total dokumen dalam scopus |
| ***Skenario 4*** | ***Pengguna memilih panel Diagram Batang Total Sitasi*** |
| Masukan | Memilih tombol *panel diagram total sitasi* pada dunia virtual |
| Keluaran yang diharapkan | Pengguna berpindah ke panel diagram batang total sitasi |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel diagram batang total sitasi |

### 5.1.2.3 Uji Coba pada Daftar Peneliti

Pada bagian ini akan dijelaskan mengenai pengujian terhadap fitur dan tombol yang terdapat pada daftar peneliti dengan tujuan apakah fitur dan tombol pada daftar peneliti berfungsi dengan baik. Pengujian ini dilakukan dengan mendeskripsikan masukan, keluaran, hasil yang dicapai dan kondisi akhir.

Pada daftar peneliti, terdapat beberapa fitur yang akan diuji yaitu panel fitur filter berdasarkan keahlian, filter berdasarkan subjek penelitian, filter berdasarkan afiliasi, urutkan berdasarkan nama, urutkan berdasarkan h-index, urutkan berdasarkan total dokumen, urutkan berdasarkan total dokumen scopus, urutkan berdasarkan total sitasi, fitur cari menggunakan perintah suara dan . Skenario uji dapat dilihat pada Tabel 5.5.

*Tabel 5.5 Hasil Uji Coba pada Daftar Peneliti*

| **ID** | **UF-003** |
|---|---|
| Nama | Uji coba pada daftar peneliti |
| Tujuan uji coba | Pengguna mengetahui fungsionalitas fitur dan tombol yang ada pada daftar peneliti |
| Kondisi awal | Pengguna berada pada panel Daftar Peneliti |
| ***Skenario 1*** | ***Pengguna memilih fitur Filter berdasarkan Keahlian*** |
| Masukan | Memilih *dropdown* filter keahlian dan memilih tombol tampilkan hasil |
| Keluaran yang diharapkan | Daftar peneliti terfilter sesuai dengan keahlian yang dipilih |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel daftar peneliti dengan peneliti terfilter sesuai keahlian |
| ***Skenario 2*** | ***Pengguna memilih fitur Filter berdasarkan Subjek Penelitian*** |
| Masukan | Memilih *dropdown* filter subjek penelitian dan memilih tombol tampilkan hasil |
| Keluaran yang diharapkan | Daftar peneliti terfilter sesuai dengan subjek penelitian yang dipilih |
| Hasil uji coba | Berhasil |

| | |
|---|---|
| Kondisi Akhir | Pengguna berada pada panel daftar peneliti dengan peneliti terfilter sesuai subjek penelitian |
| ***Skenario 3*** | ***Pengguna memilih fitur Filter berdasarkan Afiliasi*** |
| Masukan | Memilih *dropdown* filter afiliasi dan memilih tombol tampilkan hasil |
| Keluaran yang diharapkan | Daftar peneliti terfilter sesuai dengan subjek penelitian yang dipilih |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel daftar peneliti dengan peneliti terfilter sesuai afiliasi |
| ***Skenario 4*** | ***Pengguna memilih fitur Urutkan berdasarkan Nama*** |
| Masukan | Memilih *dropdown* urutkan nama |
| Keluaran yang diharapkan | Daftar peneliti terurutkan sesuai dengan nama |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel daftar peneliti dengan peneliti terurutkan sesuai dengan nama |
| ***Skenario 5*** | ***Pengguna memilih fitur Urutkan berdasarkan H-Index*** |
| Masukan | Memilih *dropdown* urutkan h-index |
| Keluaran yang diharapkan | Daftar peneliti terurutkan sesuai dengan h-index |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel daftar peneliti dengan peneliti terurutkan sesuai dengan h-index |
| ***Skenario 6*** | ***Pengguna memilih fitur Urutkan berdasarkan Total Dokumen*** |
| Masukan | Memilih *dropdown* urutkan total dokumen |
| Keluaran yang diharapkan | Daftar peneliti terurutkan sesuai dengan total dokumen |
| Hasil uji coba | Berhasil |

| | |
|---|---|
| Kondisi Akhir | Pengguna berada pada panel daftar peneliti dengan peneliti terurutkan sesuai dengan total dokumen |
| ***Skenario 7*** | ***Pengguna memilih fitur Urutkan berdasarkan Total Dokumen Scopus*** |
| Masukan | Memilih *dropdown* urutkan Total Dokumen Scopus |
| Keluaran yang diharapkan | Daftar peneliti terurutkan sesuai dengan Total Dokumen Scopus |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel daftar peneliti dengan peneliti terurutkan sesuai dengan total dokumen scopus |
| ***Skenario 8*** | ***Pengguna memilih fitur Urutkan berdasarkan Total Sitasi*** |
| Masukan | Memilih *dropdown* urutkan Total Sitasi |
| Keluaran yang diharapkan | Daftar peneliti terurutkan sesuai dengan Total Sitasi |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel daftar peneliti dengan peneliti terurutkan sesuai dengan total sitasi |
| ***Skenario 9*** | ***Pengguna melihat Detail Peneliti*** |
| Masukan | Memilih salah satu peneliti |
| Keluaran yang diharapkan | Menampilkan detail peneliti |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel detail peneliti |
| ***Skenario 10*** | ***Pengguna mencari peneliti dengan perintah suara*** |
| Masukan | Memilih tombol mikrofon, berbicara dan stop mikrofon |
| Keluaran yang diharapkan | Menampilkan peneliti sesuai nama |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Menampilkan nama peneliti sesuai yang dicari |

### 5.1.2.4 Uji Coba pada Daftar Subjek Penelitian

Pada bagian ini akan dijelaskan mengenai pengujian terhadap tombol yang terdapat pada daftar subjek penelitian dengan tujuan apakah tombol pada daftar subjek penelitian berfungsi dengan baik. Pengujian ini dilakukan dengan mendeskripsikan masukan, keluaran, hasil yang dicapai dan kondisi akhir.

Pada daftar subjek penelitian, terdapat beberapa tombol untuk daftar subjek penelitian. Daftar inilah yang akan diuji dengan skenario uji yang dapat dilihat pada Tabel 5.6.

*Tabel 5.6 Hasil Uji Coba pada Daftar Subjek Penelitian*

| **ID** | **UF-004** |
|---|---|
| Nama | Uji coba pada daftar subjek penelitian |
| Tujuan uji coba | Pengguna mengetahui fungsionalitas yang ada pada daftar subjek penelitian |
| Kondisi awal | Pengguna berada pada panel Daftar Subjek Penelitian |
| ***Skenario 1*** | ***Pengguna memilih salah satu subjek penelitian*** |
| Masukan | Memilih salah satu subjek penelitian |
| Keluaran yang diharapkan | Menampilkan detail subjek penelitian yang dipilih |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel detail subjek penelitian sesuai dengan subjek yang dipilih |

### 5.1.2.5 Uji Coba pada Detail Subjek Penelitian

Pada bagian ini akan dijelaskan mengenai pengujian terhadap tombol yang terdapat pada detail subjek penelitian dengan tujuan apakah tombol pada detail subjek penelitian berfungsi

dengan baik. Pengujian ini dilakukan dengan mendeskripsikan masukan, keluaran, hasil yang dicapai dan kondisi akhir.

Pada detail subjek penelitian, mememiliki tombol lihat peneliti. Tombol inilah yang akan diuji dengan skenario uji yang dapat dilihat pada Tabel 5.7.

*Tabel 5.7 Hasil Uji Coba pada Detail Subjek Penelitian*

| **ID** | **UF-005** |
|---|---|
| Nama | Uji coba pada detail subjek penelitian |
| Tujuan uji coba | Pengguna mengetahui fungsionalitas yang ada pada detail subjek penelitian |
| Kondisi awal | Pengguna berada pada panel Detail Subjek Penelitian |
| ***Skenario 1*** | ***Pengguna memilih tombol Lihat Peneliti*** |
| Masukan | Memilih tombol lihat peneliti |
| Keluaran yang diharapkan | Menampilkan daftar peneliti sesuai subjek penelitian yang dipilih |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel daftar peneliti dengan menampilkan peneliti sesuai dengan subjek yang dipilih |

### 5.1.2.6 Uji Coba pada Daftar Afiliasi

Pada bagian ini akan dijelaskan mengenai pengujian terhadap tombol yang terdapat pada daftar afiliasi dengan tujuan apakah tombol pada daftar afiliasi berfungsi dengan baik. Pengujian ini dilakukan dengan mendeskripsikan masukan, keluaran, hasil yang dicapai dan kondisi akhir.

Pada daftar afiliasi, terdapat beberapa tombol untuk daftar afiliasi. Daftar inilah yang akan diuji dengan skenario uji yang dapat dilihat pada Tabel 5.8.

*Tabel 5.8 Hasil Uji Coba pada Daftar Afiliasi*

| ID | UF-006 |
|---|---|
| Nama | Uji coba pada daftar afiliasi |
| Tujuan uji coba | Pengguna mengetahui fungsionalitas yang ada pada daftar afiliasi |
| Kondisi awal | Pengguna berada pada panel Daftar Afiliasi |
| ***Skenario 1*** | ***Pengguna memilih salah satu afiliasi*** |
| Masukan | Memilih salah satu afiliasi |
| Keluaran yang diharapkan | Menampilkan detail afiliasi yang dipilih |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel detail afiliasi sesuai dengan afiliasi yang dipilih |

### 5.1.2.7 Uji Coba pada Detail Afiliasi

Pada bagian ini akan dijelaskan mengenai pengujian terhadap tombol yang terdapat pada detail afiliasi dengan tujuan apakah tombol pada detail afiliasi berfungsi dengan baik. Pengujian ini dilakukan dengan mendeskripsikan masukan, keluaran, hasil yang dicapai dan kondisi akhir.

Pada detail afiliasi, mememiliki tombol lihat peneliti. Tombol inilah yang akan diuji dengan skenario uji yang dapat dilihat pada Tabel 5.9.

*Tabel 5.9 Hasil Uji Coba pada Detail Afiliasi*

| ID | UF-007 |
|---|---|
| Nama | Uji coba pada detail afiliasi |
| Tujuan uji coba | Pengguna mengetahui fungsionalitas yang ada pada detail afiliasi |
| Kondisi awal | Pengguna berada pada panel Detail Afiliasi |
| ***Skenario 1*** | ***Pengguna memilih tombol Lihat Peneliti*** |
| Masukan | Memilih tombol lihat peneliti |
| Keluaran yang diharapkan | Menampilkan daftar peneliti sesuai afiliasi yang dipilih |
| Hasil uji coba | Berhasil |

| | |
|---|---|
| Kondisi Akhir | Pengguna berada pada panel daftar peneliti dengan menampilkan peneliti sesuai dengan afiliasi yang dipilih |

### 5.1.2.8 Uji Coba pada Diagram *Scatter*

Pada bagian ini akan dijelaskan mengenai pengujian terhadap tombol yang terdapat pada diagram *scatter* dengan tujuan apakah tombol pada diagram *scatter* berfungsi dengan baik. Pengujian ini dilakukan dengan mendeskripsikan masukan, keluaran, hasil yang dicapai dan kondisi akhir.

Pada diagram *scatter*, mememiliki tombol lihat peneliti. Tombol inilah yang akan diuji dengan skenario uji yang dapat dilihat pada Tabel 5.10.

*Tabel 5.10 Hasil Uji Coba pada Diagram Scatter*

| **ID** | **UF-008** |
|---|---|
| Nama | Uji coba pada diagram *scatter* |
| Tujuan uji coba | Pengguna mengetahui fungsionalitas yang ada pada diagram *scatter* |
| Kondisi awal | Pengguna berada pada panel Diagram *Scatter* |
| ***Skenario 1*** | ***Pengguna memilih tombol Lihat Versi 3D*** |
| Masukan | Memilih tombol lihat versi 3D |
| Keluaran yang diharapkan | Menampilkan diagram *scatter* dengan format 3-dimensi |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel diagram *scatter* dengan format 3-dimensi |
| ***Skenario 2*** | ***Pengguna memilih tombol Lihat Versi 2D*** |
| Masukan | Memilih tombol lihat versi 2D |
| Keluaran yang diharapkan | Menampilkan diagram *scatter* dengan format 2-dimensi |
| Hasil uji coba | Berhasil |
| Kondisi Akhir | Pengguna berada pada panel diagram *scatter* dengan format 3-dimensi |

## 5.2 Pengujian Pengguna

Aplikasi Realitas Virtual ini tidak hanya diuji pada fungsionalitas aplikasi yang dimiliki, tetapi juga dilakukan pengujian terhadap pengguna. Pengujian terhadap pengguna ini ditujukan untuk untuk menguji apakah pengguna memahami antarmuka pada aplikasi ini. Selain itu pengguna juga akan diminta untuk berpendapat mengenai aplikasi Realitas Virtual ini.

### 5.2.1 Daftar Responden

Untuk pengujian terhadap aplikasi Realitas Virtual ini melibatkan mahasiswa dengan total responden sebanyak 10 mahasiswa. Pengguna akan diminta untuk mencoba aplikasi Realitas Virtual dengan memberi kesempatan kepada responden untuk memahami lingkungan aplikasi terlebih dahulu. Daftar responden ditunjukkan pada Tabel 5.11.

*Tabel 5.11 Daftar Responden*

| ID | Nama | Pekerjaan |
|---|---|---|
| R-01 | Putu Gayatri | Mahasiswa Sistem Informasi ITS 2017 |
| R-02 | Aisyah Muswar | Mahasiswa Teknik Informatika ITS 2016 |
| R-03 | Tiara Bening Salsabila | Mahasiswa Teknik Komputer ITS 2018 |
| R-04 | Ubut Eka Putra | Mahasiswa Teknik Informatika ITS 2016 |
| R-05 | Ariq Sudibyo | Mahasiswa Teknik Informatika ITS 2016 |
| R-06 | Gani Wijaya | Mahasiswa Teknik Informatika ITS 2016 |
| R-07 | Moh. Nafis Naufaly | Mahasiswa Teknik Informatika ITS 2016 |
| R-08 | Robby F.Q. | Mahasiswa Fisika ITS 2017 |
| R-09 | Naufal Rais Nada | Mahasiswa Teknik Material dan Metalurgi ITS 2018 |

| ID | Nama | Pekerjaan |
|---|---|---|
| R-10 | Dr.Eng. Nanik Suciati, S.Kom, M.Kom. | Dosen Teknik Informatika ITS |
| R-11 | Dr. Diana Purwitasari, S.Kom., M.Sc. | Dosen Teknik Informatika ITS |

### 5.2.2 Skenario Pengujian Pengguna

Pengujian terhadap pengguna dilakukan dengan memberikan penugasan aksi terhadap aplikasi tersebut. Penugasan tersebut akan dinilai menggunakan metode *Time Completion* dengan tujuan apakah pengguna paham dengan antarmuka pada aplikasi Realitas Virtual sesuai dengan apa yang ditugaskan. Pemahaman terhadap antarmuka pengguna akan dinilai dengan parameter waktu respon pengguna terhadap penugasan tersebut. Penugasan tersebut ditunjukkan pada Tabel 5.12.

*Tabel 5.12 Penugasan Pengguna*

| ID | Penugasan Pengguna |
|---|---|
| P-01 | Perlihatkan diagram batang rata-rata h-index |
| P-02 | Perlihatkan diagram batang total dokumen |
| P-03 | Bukalah tab peneliti |
| P-04 | Urutkan peneliti berdasarkan h-index |
| P-05 | Urutkan peneliti berdasarkan nama |
| P-06 | Tampilkan detail peneliti |
| P-07 | Carilah nama peneliti atas nama Bambang |
| P-08 | Lakukan buka detail peneliti atas nama Bambang |
| P-09 | Lakukan filter peneliti berdasarkan keahlian |
| P-10 | Lakukan filter peneliti berdasarkan subjek penelitian |
| P-11 | Lakukan filter peneliti berdasarkan afiliasi |
| P-12 | Bukalah tab subjek penelitian |
| P-13 | Bukalah salah satu subjek penelitian |
| P-14 | Sebutkan rata-rata h-index pada subjek tersebut |
| P-15 | Sebutkan total dokumen pada subjek |
| P-16 | Perlihatkan daftar peneliti berdasarkan subjek tersebut |
| P-17 | Bukalah tab afiliasi |
| P-18 | Bukalah salah satu afiliasi |

| ID | Penugasan Pengguna |
|---|---|
| P-19 | Sebutkan rata-rata h-index pada afiliasi tersebut |
| P-20 | Tunjukkan total dokumen pada afiliasi tersebut |
| P-21 | Tampilkan diagram *scatter* |
| P-22 | Tampilkan diagram *scatter* 3D |

Selain itu pengguna juga diminta untuk memberikan tanggapan mengenai aplikasi Realitas Virtual ini melalui *interview*. Pada saat *interview*, pengguna diminta untuk menjawab pertanyaan untuk memberikan tanggapan terhadap aplikasi. Pada interview ini memiliki dua jenis pertanyaan yaitu pertanyaan yang diminta untuk mengemukakan pendapat yang dapat dilihat pada Tabel 5.13 pertanyaan iya dan tidak yang dapat dilihat pada Tabel 5.14, serta pertanyaan berskala yang dapat dilihat pada Tabel 5.15. Pertanyaan berskala *interview* ini memiliki parameter nilai yang ditunjukkan pada Tabel 5.16.

*Tabel 5.13 Pertanyaan Mengemukakan Pendapat*

| No. | Pertanyaan |
|---|---|
| 1 | Bagaimana pendapat anda tentang antarmuka pada aplikasi VR ini? |
| 2 | Apakah anda setuju bahwa aplikasi VR ini adalah salah satu inovasi yang baik untuk visualisasi data? |
| 3 | Jelaskan kesulitan anda disaat menggunakan aplikasi VR ini |

*Tabel 5.14 Pertanyaan Iya dan Tidak*

| No. | Pertanyaan |
|---|---|
| 1 | Apakah anda setuju bahwa aplikasi VR ini adalah salah satu inovasi yang baik untuk visualisasi data? |
| 2 | Apakah anda menemui kesulitan disaat menggunakan aplikasi VR ini? |

*Tabel 5.15 Pertanyaan Berskala*

| No. | Pertanyaan |
|---|---|
| 1 | Dalam skala 1-6, apakah informasi yang dimuat aplikasi VR sudah jelas? (semakin tinggi semakin jelas) |
| 2 | Dalam skala 1-6, apakah untuk mengakses informasi pada aplikasi VR ini mudah? |

*Tabel 5.16 Parameter Nilai Pertanyaan Berskala*

| No | Keterangan | Nilai |
|---|---|---|
| 1 | Sangat Kurang (SK) | 1 |
| 2 | Kurang (K) | 2 |
| 3 | Cukup (C) | 3 |
| 4 | Baik (B) | 4 |
| 5 | Sangat Baik (SB) | 5 |
| 6 | Sempurna (S) | 6 |

### 5.2.3 Hasil Pengujian Pengguna

Hasil pengujian pengguna menggunakan dua metode yaitu menggunakan metode penugasan yang dilaksanakan telah dilaksanakan oleh responden serta melakukan *interview* untuk dimintai pendapat aplikasi Realitas Virtual. Hasil pengujian terhadap pengguna disajikan secara lengkap pada subbab ini. Hasil pengujian penugasan terhadap pengguna pengguna dapat dilihat pada Tabel 5.17, Tabel 5.18, Tabel

*Tabel 5.17 Waktu Respon Penguji Terhadap Setiap Penugasan*

| ID | R-01 | R-02 | R-03 | R-04 | R-05 | R-06 | R-07 | R-08 | R-09 | R-10 | R-11 | Rata-rata/ Detik |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| P-01 | 4 | 7 | 15 | 6 | 6 | 4 | 5 | 5 | 5 | 15 | 3 | 7.5 |
| P-02 | 5 | 21 | 5 | 8 | 10 | 7 | 5 | 6 | 4 | 29 | 2 | 10.2 |
| P-03 | 5 | 4 | 6 | 8 | 3 | 6 | 6 | 5 | 4 | 14 | 3 | 6.4 |
| P-04 | 7 | 20 | 6 | 5 | 5 | 6 | 35 | 4 | 5 | 15 | 3 | 11.1 |
| P-05 | 5 | 4 | 6 | 5 | 4 | 6 | 7 | 4 | 7 | 15 | 3 | 6.6 |
| P-06 | 7 | 3 | 9 | 6 | 3 | 4 | 5 | 4 | 3 | 22 | 6 | 7.2 |
| P-07 | 77 | 33 | 36 | 73 | 21 | 100 | 40 | 17 | 30 | 32 | - | 45.9 |
| P-08 | 6 | 3 | 3 | - | 6 | 20 | 4 | 3 | 2 | 4 | - | 3.9 |
| P-09 | 13 | 5 | 26 | 32 | 10 | 13 | 17 | 15 | 17 | 198 | 25 | 37.1 |
| P-010 | 10 | 9 | 15 | 7 | 11 | 7 | 29 | 10 | 18 | 30 | 10 | 15.6 |
| P-11 | 10 | 8 | 11 | 20 | 7 | 7 | 25 | 8 | 17 | 24 | 3 | 14 |
| P-12 | 5 | 2 | 4 | 5 | 4 | 3 | 4 | 4 | 3 | 14 | 4 | 5.2 |
| P-13 | 20 | 3 | 2 | 9 | 5 | 2 | 6 | 5 | 4 | 13 | 2 | 7.1 |
| P-14 | 3 | 5 | 2 | 8 | 2 | 4 | 2 | 2 | 3 | 3 | 2 | 3.6 |

| ID | R-01 | R-02 | R-03 | R-04 | R-05 | R-06 | R-07 | R-08 | R-09 | R-10 | R-11 | Rata-rata/ Detik |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| P-15 | 4 | 2 | 2 | 4 | 2 | 6 | 2 | 2 | 4 | 2 | 3 | 3.3 |
| P-16 | 7 | 6 | 7 | 10 | 5 | 7 | 6 | 5 | 3 | 8 | 4 | 6.8 |
| P-17 | 19 | 7 | 6 | 5 | 5 | 7 | 4 | 5 | 4 | 14 | 4 | 8 |
| P-18 | 4 | 2 | 5 | 4 | 4 | 7 | 4 | 3 | 4 | 5 | 2 | 4.4 |
| P-19 | 3 | 2 | 2 | 4 | 2 | 2 | 2 | 2 | 2 | 9 | 2 | 3.2 |
| P-20 | 2 | 3 | 2 | 3 | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 2.3 |
| P-21 | 2 | 3 | 5 | 7 | 4 | 16 | 5 | 5 | 4 | 33 | 2 | 8.6 |
| P-22 | 2 | 4 | 4 | 9 | 7 | 5 | 4 | 3 | 3 | 5 | 3 | 4.9 |

*Tabel 5.18 Pendapat Penguji terhadap Antarmuka Aplikasi Realitas Virtual*

| ID Penguji | Bagaimana pendapat anda tentang antarmuka pada aplikasi Realitas Virtual ini? |
|---|---|
| R-01 | Aplikasi Realitas Virtual ini sangat terkesan futuristik |
| R-02 | Keren |
| R-03 | Aplikasi Realitasi Virtual pertama yang pernah saya gunakan lumayan menarik, cukup informatif, tertata dengan bagus, rapi serta cukup mudah dijalankan. |
| R-04 | Bagus dan mudah dipahami. |
| R-05 | Intuitif dan mudah dipahami. |
| R-06 | Bagus dan memiliki tampilan dengan sudut yang berbeda. Menurut saya aplikasi Realitas Virtual ini merupakan sesuatu yang baru serta pemilihan menu sudah jelas |
| R-07 | Sudah sangat jelas dan disediakan tab untuk menampilkan fitur-fitur utama dari aplikasi ini. Cara penampilan datanya cukup menarik khususnya yang di versi 3-dimensi untuk diagram *scatter*. |
| R-08 | Antarmuka aplikasi Realitas Virtual ini cukup mudah untuk dipahami. Tulisan pada aplikasi VR ini cukup besar dan mudah untuk dilihat. |
| R-09 | Aplikasi Realitas Virtual ini memuat informasi yang mudah dipahami. Huruf dan angkanya cukup terlihat dengan jelas serta memiliki antarmuka yang cukup bagus |
| R-10 | Memiliki antarmuka yang cukup mudah dimengerti, memiliki visualisasi data yang berbeda dari biasanya sehingga memberikan pengalaman yang baru. |
| R-11 | Bagus |

*Tabel 5.19 Penilaian Penguji Terhadap Aplikasi Realitas Virtual*

| Pertanyaan | Penilaian | | | | | | Rata-rata |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | |
| Apakah informasi yang dimuat aplikasi VR sudah jelas? | 0 | 0 | 0 | 1 | 6 | 4 | 5.27 |
| Apakah untuk mengakses informasi pada aplikasi VR ini mudah? | 0 | 0 | 0 | 2 | 6 | 3 | 5.09 |
| Apakah pengaturan tata letak sudah baik? | 0 | 0 | 0 | 2 | 5 | 4 | 5.18 |

*Tabel 5.20 Pendapat Penguji terhadap Inovasi Visualisasi Data menggunakan Aplikasi Realitas Virtual*

| ID Penguji | Apakah anda setuju bahwa aplikasi VR ini adalah salah satu inovasi yang baik untuk visualisasi data? |
|---|---|
| R-01 | Setuju |
| R-02 | Setuju |
| R-03 | Setuju |
| R-04 | Setuju |
| R-05 | Setuju |
| R-06 | Setuju |
| R-07 | Setuju |
| R-08 | Setuju |
| R-09 | Setuju |
| R-10 | Setuju |
| R-11 | Setuju |

*Tabel 5.21 Penemuan Kesulitan oleh Penguji terhadap Aplikasi Realitas Virtual*

| ID Penguji | Apakah anda menemui kesulitan disaat menggunakan aplikasi VR ini? |
|---|---|
| R-01 | Iya |
| R-02 | Iya |
| R-03 | Iya |
| R-04 | Tidak |
| R-05 | Tidak |
| R-06 | Iya |
| R-07 | Iya |

| | |
|---|---|
| R-08 | Iya |
| R-09 | Tidak |
| R-10 | Iya |
| R-11 | Tidak |

*Tabel 5.22 Penjelasan Penguji terkait Kesulitan terhadap Aplikasi Realitas Virtual*

| ID Penguji | Jelaskan kesulitan anda disaat menggunakan aplikasi VR ini |
|---|---|
| R-01 | Pencarian terlalu tersembunyi dikarenakan terdapat ruang diantara fitur pencarian dan panel peneliti. |
| R-02 | Terdapat kesulitan pada pencari dalam diagram scatter. |
| R-03 | Koneksi internet yang kurang baik serta terdapat data yang sama sehingga membingungkan pengguna. |
| R-04 | (Tidak ada) |
| R-05 | (Tidak ada) |
| R-06 | Terdapat kesulitan yaitu penamaan dikarenakan terdapat diagram *pie* dan saya mengira diagram tersebut merupakan diagram *scatter* tetapi ternyata diagram *scatter* merupakan nama tab. Kemudian disaat *loading* data, aplikasi ini sedikit kurang lancar. Tetapi setelah data dimunculkan, aplikasi ini lancar kembali. |
| R-07 | Sebelum bereksplorasi lebih lanjut mengenai aplikasi VR ini terkadang sedikit membingungkan bagaimana cara menggunakan tombol yang ada pada antarmuka. |
| R-08 | Ketika membuka detail peneliti, tidak dapat kembali ke list peneliti, tetapi harus memilih tab peneliti. |
| R-09 | (Tidak ada) |
| R-10 | (Tidak ada) |
| R-11 | (Tidak ada) |

*Tabel 5.23 Saran Penguji terhadap Antarmuka Aplikasi Realitas Virtual*

| ID Penguji | Apa saran anda mengenai aplikasi VR ini? |
|---|---|
| R-01 | Transparansi icon mohon dibuat lebih jelas dikarenakan terdapat logo ada yang tidak terlihat. Fitur pencarian terlalu tersembunyi. Maksud dan tujuan diagram *scatter* diperjelas tanpa harus melihat keterangan. |
| R-02 | Diagram *scatter* terlalu menyatu dengan dengan warna tema, akan lebih baik diganti dengan warna lain. Icon disamakan dikarenakan terdapat dua tipe gambar yaitu PNG dan JPG. Diagram batang terdapat warna yang hampir sama. Diagram *Scatter* lebih baik bisa disesuaikan ukurannya. Fitur pencarian terlalu tersembunyi serta warna *scroll* dibedakan. |
| R-03 | Dapat ditambah *range* pengambilan data tahun untuk peneliti |
| R-04 | Dapat ditambah jenis diagram atau grafik. |
| R-05 | Keterangan pada diagram *scatter* lebih diperjelas. |
| R-06 | Penataan jangan terlalu mepet sehingga membingungkan karena tidak ada pemisahnya. |
| R-07 | Fitur dalam diagram *scatter* versi 2-dimensi agar tiap titik diberikan representasi tentang siapa |
| R-08 | Tambahkan beberapa tombol pendukung dan tingkat akurasi perintah suara lebih ditingkatkan |
| R-09 | Mungkin dapat dikurangi dalam efek seperti cahaya dikarenakan mungkin untuk beberapa perangkat yang sudah usang ditakutkan tidak memberikan pengalaman penuh. |
| R-10 | Diterapkan yang mungkin benar-benar imersif menggunakan perangkat seperti Oculus serta menggunakan *controller* yang lebih luas sehingga dapat menjangkau objek yang jauh. Kemudian untuk *background* agar bisa disesuaikan atau diganti sesuai selera. |
| R-11 | Untuk saran pengembangan kedepannya bisa diterapkan *controller* yang menggunakan tangan ibarat mengambil awan pada diagram scatter. |

## 5.3 Evaluasi Pengujian

Pada subbab ini akan disajikan rangkuman data hasil pengujian baik dari pengujian fungsional maupun hasil pengujian terhadap pengguna yang telah dilakukan sebelumnya untuk aplikasi Realitas Virtual ini.

### 5.3.1 Evaluasi Hasil Pengujian Fungsional

Evaluasi hasil pengujian fungsional akan dilakukan dengan menyajikan rangkuman data pengujian fungsional pada pengujian sebelumnya. Rangkuman data pengujian fungsional dapat dilihat pada Tabel 5.24. Dari data yang terdapat pada tabel tersebut, dapat diketahui bahwa aplikasi yang telah dibuat telah memenuhi berbagai kasus yang telah ditulis pada skenario pengujian fungsionalitas.

*Tabel 5.24 Rangkuman Hasil Pengujian Fungsional*

| ID | Deskripsi | Skenario | Perilaku Terlaksana |
|---|---|---|---|
| UF-001 | Uji coba pada panel navigasi | Skenario 1 | Ya |
| | | Skenario 2 | Ya |
| | | Skenario 3 | Ya |
| | | Skenario 4 | Ya |
| | | Skenario 5 | Ya |
| UF-002 | Uji coba pada beranda | Skenario 1 | Ya |
| | | Skenario 2 | Ya |
| | | Skenario 3 | Ya |
| | | Skenario 4 | Ya |
| UF-003 | Uji coba pada daftar peneliti | Skenario 1 | Ya |
| | | Skenario 2 | Ya |
| | | Skenario 3 | Ya |
| | | Skenario 4 | Ya |
| | | Skenario 5 | Ya |
| | | Skenario 6 | Ya |
| | | Skenario 7 | Ya |
| | | Skenario 8 | Ya |
| | | Skenario 9 | Ya |

| | | | |
|---|---|---|---|
| | | Skenario 10 | Ya |
| UF-004 | Uji coba pada daftar subjek penelitian | Skenario 1 | Ya |
| UF-005 | Uji coba pada detail subjek penelitian | Skenario 1 | Ya |
| UF-006 | Uji coba pada daftar afiliasi | Skenario 1 | Ya |
| UF-007 | Uji coba pada detail afiliasi | Skenario 1 | Ya |
| UF-008 | Uji coba pada diagram *scatter* | Skenario 1 | Ya |
| | | Skenario 2 | Ya |

### 5.3.2 Evaluasi Hasil Pengujian Pengguna

Evaluasi hasil pengujian pengguna akan dilakukan dengan menyajikan rangkuman data pengujian terhadap pengguna yang telah dilakukan sebelumnya. Rangkuman data pengujian pengguna meliputi seberapa paham pengguna dapat berinteraksi dengan antarmuka yang diukur dalam satuan detik. Rangkuman tersebut dapat dilihat pada Tabel 5.25. Dari data yang terdapat pada tabel tersebut, dapat diketahui bahwa aplikasi Realitas Virtual ini masih memiliki beberapa kelemahan salah satunya adalah pada fitur pencarian peneliti dimana mendapatkan respon waktu yang paling lambat diantara yang lain.

*Tabel 5.25 Rangkuman Waktu Respon Penguji terhadap Aplikasi Realitas Virtual*

| ID | Penugasan Pengguna | Rata-rata/Detik |
|---|---|---|
| P-01 | Perlihatkan diagram batang rata-rata h-index | 7.5 |
| P-02 | Perlihatkan diagram batang total dokumen | 10.2 |
| P-03 | Bukalah tab peneliti | 6.4 |
| P-04 | Urutkan peneliti berdasarkan h-index | 11.1 |
| P-05 | Urutkan peneliti berdasarkan nama | 6.6 |
| P-06 | Tampilkan detail peneliti | 7.2 |
| P-07 | Carilah nama peneliti atas nama Bambang | 45.9 |

| ID | Penugasan Pengguna | Rata-rata/Detik |
|---|---|---|
| P-08 | Lakukan buka detail peneliti atas nama Bambang | 3.9 |
| P-09 | Lakukan filter peneliti berdasarkan keahlian | 37.1 |
| P-010 | Lakukan filter peneliti berdasarkan subjek penelitian | 15.6 |
| P-11 | Lakukan filter peneliti berdasarkan afiliasi | 14 |
| P-12 | Bukalah tab subjek penelitian | 5.2 |
| P-13 | Bukalah salah satu subjek penelitian | 7.1 |
| P-14 | Sebutkan rata-rata h-index pada subjek tersebut | 3.6 |
| P-15 | Sebutkan total dokumen pada subjek | 3.3 |
| P-16 | Perlihatkan daftar peneliti berdasarkan subjek tersebut | 6.8 |
| P-17 | Bukalah tab afiliasi | 8 |
| P-18 | Bukalah salah satu afiliasi | 4.4 |
| P-19 | Sebutkan rata-rata h-index pada afiliasi tersebut | 3.2 |
| P-20 | Tunjukkan total dokumen pada afiliasi tersebut | 2.3 |
| P-21 | Tampilkan diagram *scatter* | 8.6 |
| P-22 | Tampilkan diagram *scatter* 3D | 4.9 |
| | **Rata-rata Total** | |
| | 10.13 | |

Selain respon pengguna terhadap waktu, pengguna menilai dalam skala 1-6 bahwa aplikasi Realitas Virtual ini memuat informasi yang jelas dengan skor 5.27, kemudahan mengakses informasi dengan skor 5.09 dan pengaturan tata letak yang sudah baik dengan skor 5.18 yang dapat dilihat pada Tabel 5.26.

*Tabel 5.26 Penilaian Penguji terhadap Aplikasi Realitas Virtual*

| Pertanyaan | Rata-rata |
|---|---|
| Apakah informasi yang dimuat aplikasi VR sudah jelas? | 5.27 |
| Apakah untuk mengakses informasi pada aplikasi VR ini mudah? | 5.09 |
| Apakah pengaturan tata letak sudah baik? | 5.18 |

Dalam penggunaan aplikasi Realitas Virtual, pengguna masih memiliki kesulitan. Jumlah pengguna yang masih memiliki kesulitan dalam aplikasi Realitas Virtual dengan presentase sebesar 63,63% dari jumlah total. Presentase tersebut ditunjukkan pada Tabel 5.27.

*Tabel 5.27 Persentase Jumlah Penguji yang Mengalami Kesulitan*

| Pertanyaan | Iya | Tidak |
|---|---|---|
| Apakah anda menemui kesulitan disaat menggunakan aplikasi VR ini? | 7 | 4 |
| **Persentase** | | |
| **Iya** | **Tidak** | |
| 63,63% | 36,36% | |

Dalam penggunaan aplikasi Realitas Virtual, 100% pengguna memiliki kesetujuan dalam penerapan aplikasi Realitas Virtual untuk visualisasi data peneliti. Presentase tersebut ditunjukkan pada Tabel 5.27.

*Tabel 5.28 Persentase Jumlah Penguji yang Setuju Invoasi Visualisasi Data menggunakan Realitas Virtual*

| Pertanyaan | Setuju | Tidak |
|---|---|---|
| Apakah anda setuju bahwa aplikasi VR ini adalah salah satu inovasi yang baik untuk visualisasi data? | 11 | 0 |
| **Persentase** | | |
| **Iya** | **Tidak** | |
| 100% | 0% | |

# BAB VI
# PENUTUP

Bab membahas tentang kesimpulan yang didapatkan dari hasil uji terhadap penerapan aplikasi Realitas Virtual untuk visualisasi data peneliti dari hasil uji coba yang telah dilakukan. Selain kesimpulan, terdapat pula saran-saran untuk pengembangan perangkat lunak lebih lanjut.

## 6.1 Kesimpulan

Berikut ini adalah kesimpulan dari pengerjaan Tugas Akhir dengan judul Penerapan Teknologi Realitas Virtual Imersif dan Interaktif untuk Visualisasi Data Peneliti:

1. Pengambilan dan penampilan data untuk Tugas Akhir yang berjudul Penerapan Teknologi Realitas Virtual Imersif dan Interaktif untuk Visualisasi Data Peneliti berhasil diterapkan menggunakan Unity3D dan Flask *micro web framework* dengan arsitektur *client-server* dan menggunakan basis data MySQL dalam penyimpanan data.
2. Berdasarkan hasil pengujian pada pengguna, pengaturan antarmuka untuk Tugas Akhir yang berjudul Penerapan Teknologi Realitas Virtual Imersif dan Interaktif untuk Visualisasi Data Peneliti mendapatkan nilai 5.27 atau dalam persentase sebesar 87,83% untuk kejelasan informasi, 5.09 atau dalam persentase sebesar 84,83% untuk kemudahan akses serta 5.18 atau dalam presentase sebesar 86,33% untuk tata letak pada antarmuka.
3. Berdasarkan hasil pengujian pada pengguna, pengguna memiliki waktu respon terhadap aplikasi Realitas Virtual dangan total rata-rata setiap penugasan sebesar 10,13 detik.

## 6.2 Kritik dan Saran

Dalam pengerjaan aplikasi Realitas Virtual, terdapat hal-hal yang perlu diperbaiki untuk pengembangan lebih lanjut. Berikut merupakan kritik dan saran pengguna untuk pengembangan lebih lanjut mengenai aplikasi Realitas Virtual untuk visualisasi data peneliti.

1. Transparansi *icon* dibuat lebih jelas serta penyamaan jenis gambar untuk *icon.*
2. Maksud dan tujuan diagram *scatter* diperjelas, bisa disesuaikan ukurannya serta versi 2-dimensi tiap titik agar diberikan representasi tentang siapa.
3. Fitur pencarian terlalu tersembunyi dan akurasi perintah suara lebih ditingkatkan.
4. Warna *scroll* dan diagram batang lebih dibedakan serta ditambah jenis diagram atau grafik.
5. Ditambah range pengambilan data tahun dari tiap peneliti.
6. Pemberian jarak antarmuka dan tombol pendukung seperti kembali.
7. Pengurangan efek agar memberi pengalaman maksimal.

# DAFTAR PUSTAKA


[1] Cambridge, "Meaning of Immersive in English," Cambrigde, [Online]. Available: https://dictionary.cambridge.org/dictionary/english/immersive.

[2] S. G. D. A. C. A. W. J. Z. E. L. S. Y. A. M. M. G. A. D. Ciro Donalek, "Immersive and Collaborative Data Visualization Using Virtual Reality Platforms," 2014.

[3] K. G. D. Herlangga, codepolitan.com, 07 Maret 2016. [Online]. Available: https://www.codepolitan.com/virtual-reality-dan-perkembangannya. [Diakses 19 Oktober 2019].

[4] wahyupjl, "Apa itu Unity 3D," Eventkampus, 12 Juli 2018. [Online]. Available: https://eventkampus.com/blog/detail/1474/apa-itu-unity-3d. [Diakses 20 November 2019].

[5] T. Maya, "PEMODELAN MODEL OBJEK 3D," [Online]. Available: http://www.teorimaya.com/2018/09/pemodelan-model-objek-3d.html. [Diakses 20 November 2019].

[6] Codepolitan, "Pengenalan Bahasa Pemrograman C#," codepolitan.com, 18 Januari 2017. [Online]. Available: https://www.codepolitan.com/pengenalan-bahasa-pemrograman-c-587effa1cb95b. [Diakses 20 November 2019].

[7] Python, "What is Python? Executive Summary," [Online]. Available: https://www.python.org/doc/essays/blurb/. [Diakses 26 02 2020].

[8] PhpMyAdmin, "About PhpMyAdmin," [Online]. Available: https://www.phpmyadmin.net/. [Diakses 05 03 2020].

[9] icloudhost, “Mengenal Apa Itu Pengertian XAMPP,” [Online]. Available: https://idcloudhost.com/kamus-hosting/xampp/. [Diakses 17 04 2020].

[10] P. Project, “Flask,” [Online]. Available: https://palletsprojects.com/p/flask/. [Diakses 17 03 2020].

# LAMPIRAN

## A. Link Pengujian Pengguna

*Tabel A-1 Pengujian Pengguna*

| No. | Nama | Pekerjaan | URL Penugasan | URL Interview |
|---|---|---|---|---|
| **1** | Putu Gayatri | Mahasiswa Sistem Informasi ITS 2017 | https://youtu.be/OZ2wroJYUOg | https://youtu.be/Kli7GnzVbf4 |
| **2** | Aisyah Muswar | Mahasiswa Teknik Informatika ITS 2016 | https://youtu.be/VR1VTtpGU1I | https://youtu.be/iFEPS50y5Hs |
| **3** | Tiara Bening Salsabila | Mahasiswa Teknik Komputer ITS 2018 | https://youtu.be/EJaDg87u3JM | https://youtu.be/w4ATNXzTjbY |
| **4** | Ubut Eka Putra | Mahasiswa Teknik Informatika ITS 2016 | https://youtu.be/hK9D6ycj7dQ | https://youtu.be/tQ1GupXFs4M |
| **5** | Ariq Sudibyo | Mahasiswa Teknik Informatika ITS 2016 | https://youtu.be/Z5jnGtY_N-U | https://youtu.be/hl9Df42E_U |
| **6** | Gani Wijaya | Mahasiswa Teknik Informatika ITS 2016 | https://youtu.be/JKYeNm0I4ao | https://youtu.be/eYvaB36HPKk |
| **7** | Moh. Nafis Naufaly | Mahasiswa Teknik Informatika ITS 2016 | https://youtu.be/st57VDcRNW4 | https://youtu.be/LhdR-kHKfhg |
| **8** | Robby F.Q. | Mahasiswa Fisika ITS 2016 | https://youtu.be/cN5a9DOzeA0 | https://youtu.be/lIzuwM46j2c |

| | | | | |
|---|---|---|---|---|
| **9** | Naufal Rais Nada | Mahasiswa Teknik Material dan Metalurgi ITS 2018 | https://youtu.be/UmhZCa7TQgs | https://youtu.be/eTZwzrTPLZc |
| **10** | Dr.Eng. Nanik Suciati, S.Kom, M.Kom. | Dosen Teknik Informatika ITS | https://youtu.be/fUk2ggGH6bs | https://youtu.be/HguaCvtmuJY |
| **11** | Dr. Diana Purwitasari, S.Kom., M.Sc. | Dosen Teknik Informatika ITS | https://youtu.be/5QYddyo_hX8 | https://youtu.be/T_7OqDXdXnc |

## B. Contoh Data

*Tabel A. 1 Contoh Data Peneliti Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

| ID | Kode Peneliti | Nama Peneliti | dou men | Dokumen Scopus | H Index |
|---|---|---|---|---|---|
| 1 | 106418368 0 | Subchan, Subchan | 42 | 22 | 5 |
| 2 | 127619146 0 | Pramujati, Bambang | 86 | 44 | 7 |
| 3 | 127748308 0 | Pramono, Agus Sigit | 30 | 15 | 3 |
| 4 | 132054138 0 | Arifin, Achmad | 40 | 21 | 5 |
| 5 | 150263000 0 | Er, Mahendrawathi | 28 | 16 | 5 |
| 6 | 150726261 0 | Ardhyananta, Hosta | 40 | 21 | 6 |
| 7 | 160261562 0 | Wulandari, Diah Puspito | 20 | 12 | 4 |
| 8 | 160416278 0 | Alasiry, Ali Husein | 22 | 12 | 2 |
| 9 | 163030059 0 | Ariana, I. Made | 16 | 8 | 1 |
| 10 | 165507923 0 | Sardjono, Tri Arief | 50 | 33 | 5 |

| Total Sitasi | X | Y | Keyword | Show Centroid | ID Departemen | ID Afiliasi |
|---|---|---|---|---|---|---|
| 155 | 11.08 | - | estimation,con | 0 | 1 | 1 |
| 139 | 11.74 | - | controller,part, | 1 | 2 | 1 |
| 36 | 14.90 | - | study,simulati | 0 | 2 | 1 |
| 82 | 4.607 | - | processing,cha | 0 | 3 | 1 |
| 71 | 1.502 | - | movement,ana | 0 | 4 | 1 |
| 194 | 16.17 | 7.87763 | performance,e | 1 | 5 | 1 |
| 34 | 2.676 | - | gamelan,music | 0 | 6 | 1 |
| 16 | 13.18 | - | implementatio | 0 | 28 | 3 |
| 3 | 19.56 | - | water,plate,col | 0 | 7 | 1 |
| 91 | - | - | curvature,deter | 0 | 3 | 1 |

*Tabel B. 2 Contoh Data Departemen Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

| ID | Nama Departemen |
|---|---|
| 1 | S1 MATEMATIKA |
| 2 | S1 TEKNIK MESIN |

| | |
|---|---|
| 3 | S1 TEKNIK BIOMEDIK |
| 4 | S1 SISTEM INFORMASI |
| 5 | S1 TEKNIK MATERIAL |
| 6 | S1 TEKNIK KOMPUTER |
| 7 | S1 TEKNIK SISTEM PERKAPALAN |
| 8 | S1 TEKNIK FISIKA |
| 9 | S1 TEKNIK KIMIA |
| 10 | S1 STATISTIKA |

*Tabel C. 3 Contoh Data Afiliasi Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

| ID | Nama Afiliasi | Kota |
|---|---|---|
| 1 | Institut Teknologi Sepuluh Nopember | Surabaya |
| 2 | Telkom University | Bandung West Java |
| 3 | Politeknik Elektronika Negeri Surabaya | Surabaya |
| 4 | Universitas Airlangga | Surabaya |
| 5 | Universitas Islam Darul 'Ulum | |
| 6 | Institut Teknologi Sepuluh November (ITS) | |
| 7 | Institut Tehnologi Sepuluh Nopember | Surabaya |
| 8 | Gadjah Mada University | Yogyakarta |
| 9 | Universitas Nahdlatul Ulama | |
| 10 | Faculty of Mathematics Computation and Data Science | Surabaya |

| Negara | Logo |
|---|---|
| Indonesia | ./logo_universitas/its.png |
| Indonesia | ./logo_universitas/telu.png |
| Indonesia | ./logo_universitas/pens.png |
| Indonesia | ./logo_universitas/unair.png |
| Indonesia | ./logo_universitas/unisda.jpg |
| Indonesia | ./logo_universitas/its.png |
| Indonesia | ./logo_universitas/its.png |
| Indonesia | ./logo_universitas/ugm.png |
| Indonesia | ./logo_universitas/unu.jpg |
| Indonesia | ./logo_universitas/kemenag.pn |

*Tabel D. 4 Contoh Data Fakultas Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

| ID | Nama Fakultas |
|---|---|
| 1 | FAKULTAS MATEMATIKA, KOMPUTASI, DAN SAINS DATA |
| 2 | FAKULTAS TEKNOLOGI INDUSTRI |
| 3 | FAKULTAS TEKNOLOGI ELEKTRO |
| 4 | FAKULTAS TEKNOLOGI INFORMASI DAN KOMUNIKASI |
| 5 | FAKULTAS TEKNOLOGI KELAUTAN |
| 6 | FAKULTAS TEKNIK SIPIL, LINGKUNGAN, DAN KEBUMIAN |
| 7 | FAKULTAS SAINS |
| 8 | FAKULTAS VOKASI |
| 9 | FAKULTAS BISNIS DAN MANAJEMEN TEKNOLOGI |
| 10 | Tidak Diketahui |

*Tabel E. 5 Contoh Data Subjek Area Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

| ID | Subjek Area |
|---|---|
| 1 | Engineering |
| 2 | Mathematics |
| 3 | Physics and Astronomy |
| 4 | Computer Science |
| 5 | Decision Sciences |
| 6 | Materials Science |
| 7 | Chemical Engineering |
| 8 | Social Sciences |
| 9 | Energy |
| 10 | Business, Management and Accounting |

*Tabel F. 6 Contoh Data Detail Subjek Area Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

| ID | ID Subjek Area | ID Peneliti |
|---|---|---|
| 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 3 | 1 |
| 4 | 4 | 1 |
| 5 | 5 | 1 |
| 6 | 1 | 2 |
| 7 | 6 | 2 |
| 8 | 3 | 2 |
| 9 | 4 | 2 |
| 10 | 7 | 2 |

*Tabel G. 7 Contoh Data Subjek Topik Tier 1 Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

| ID | Subjek Topik Tier 1 |
|---|---|
| 1 | Health Science |
| 2 | Life Science |
| 3 | Physical Science |
| 4 | Social Science |

*Tabel H. 8 Contoh Data Subjek Topik Tier 2 Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

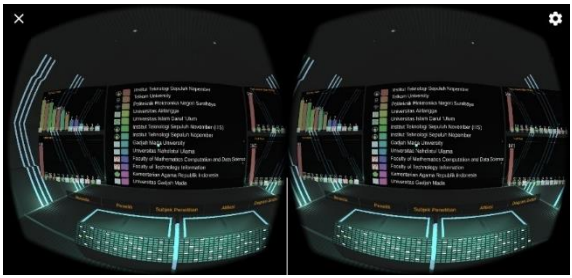
| ID | ID Subjek Topik Tier 2 | ID Subjek Topik Tier 1 |
|---|---|---|
| 1 | MEDI | 1 |
| 2 | NURS | 1 |
| 3 | DENT | 1 |
| 4 | VETE | 1 |
| 5 | HEAL | 1 |
| 6 | AGRI | 2 |
| 7 | NEUR | 2 |
| 8 | IMMU | 2 |
| 9 | BIOC | 2 |
| 10 | PHAR | 2 |

*Tabel I. 9 Contoh Data Keahlian Aplikasi Realitas Virtual untuk Visualisasi Data Peneliti*

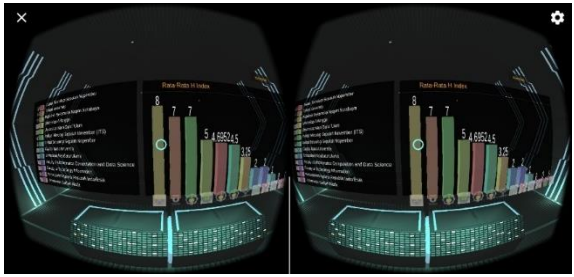
| ID | ID Subjek Topik Tier 2 | ID Peneliti | Jumlah Dokumen |
|---|---|---|---|
| 1 | 17 | 1 | 13 |
| 2 | 17 | 2 | 16 |
| 3 | 19 | 2 | 16 |
| 4 | 19 | 3 | 11 |

| ID | ID Subjek Topik Tier 2 | ID Peneliti | Jumlah Dokumen |
|---|---|---|---|
| 5 | 17 | 4 | 12 |
| 6 | 24 | 5 | 9 |
| 7 | 14 | 6 | 6 |
| 8 | 11 | 6 | 6 |
| 9 | 17 | 7 | 5 |
| 10 | 17 | 8 | 11 |

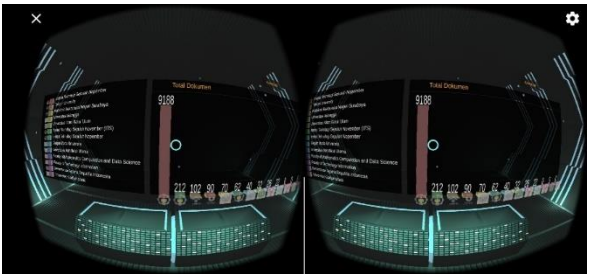
## C. Gambar Pengujian Fungsional



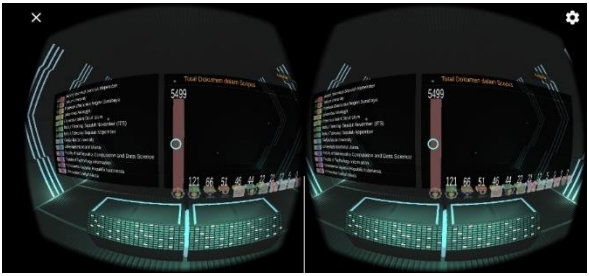
*Gambar C. 1 Pengujian Beranda*



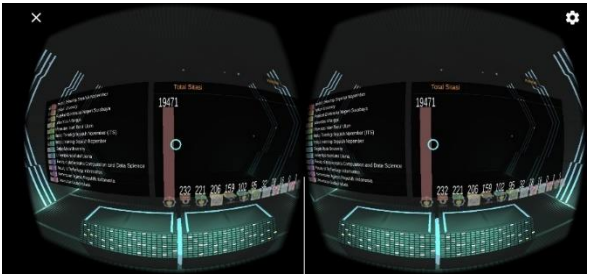
*Gambar C. 2 Pengujian Diagram Batang Rata-rata H-Index*



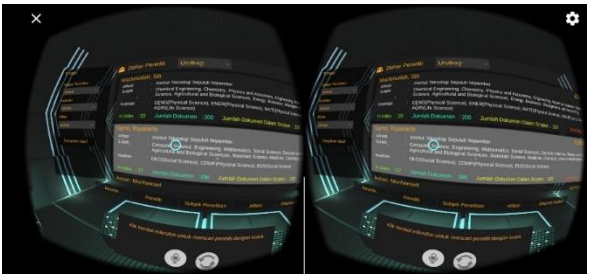

*Gambar C. 3  Pengujian Diagram Batang Total Dokumen*




*Gambar C. 4  Pengujian Diagram Batang Total Dokumen dalam Scopus*



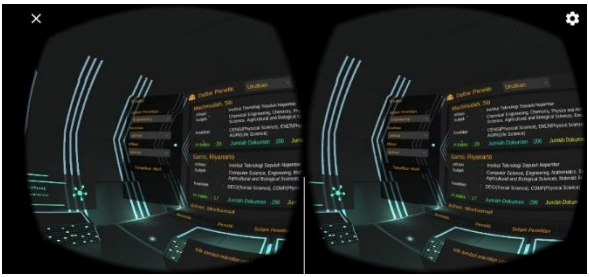

*Gambar C. 5  Pengujian Diagram Batang Total Sitasi*



*Gambar C. 6  Pengujian Daftar Peneliti*

*Gambar C. 7  Pengujian Filter Keahlian Daftar Peneliti*

*Gambar C. 8  Pengujian Filter Afiliasi Daftar Peneliti*



*Gambar C. 9  Pengujian Filter Subjek Penelitian Daftar Peneliti*

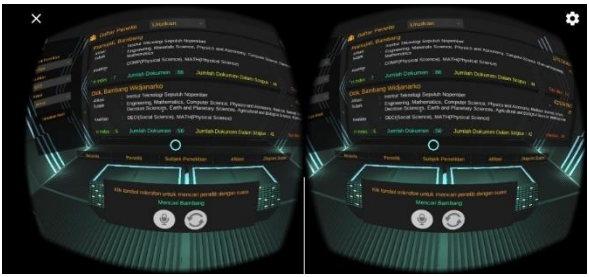
*Gambar C. 10  Pengujian Urutkan Nama Daftar Peneliti*

*Gambar C. 11  Pengujian Urutkan H-Index Daftar Peneliti*

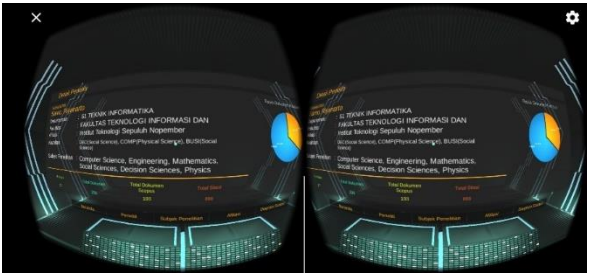
*Gambar C. 12  Pengujian Urutkan Dokumen Daftar Peneliti*

*Gambar C. 13  Pengujian Urutkan Dokumen Scopus Daftar Peneliti*

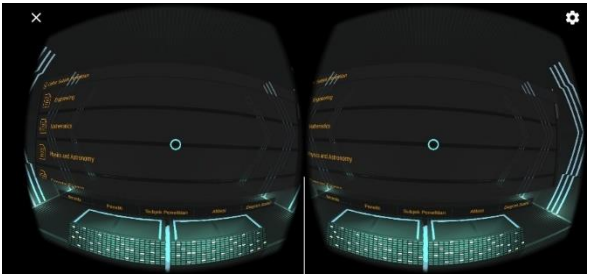
*Gambar C. 14  Pengujian Urutkan Total Sitasi Daftar Peneliti*



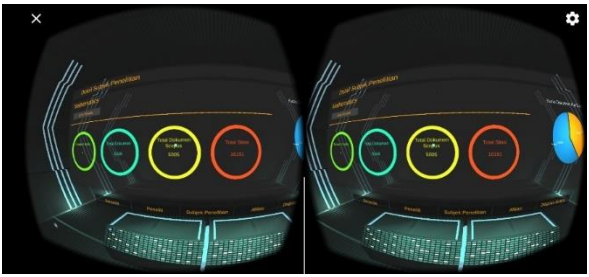
*Gambar C. 15  Pengujian Pencarian Daftar Peneliti*



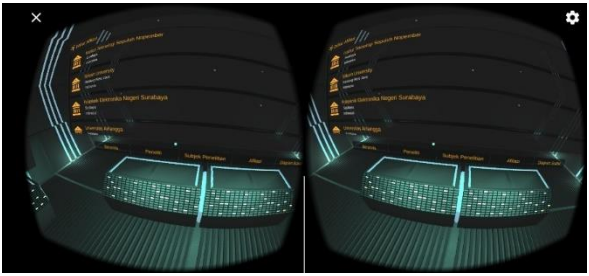
*Gambar C. 16  Pengujian Detail Peneliti*



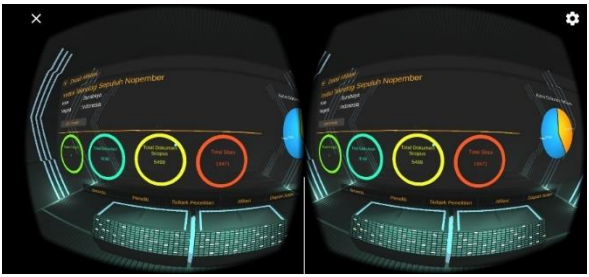
*Gambar C. 17  Pengujian Daftar Subjek Penelitian*



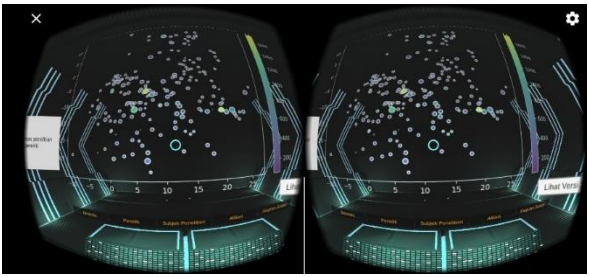
*Gambar C. 18  Pengujian Detail  Subjek Penelitian*



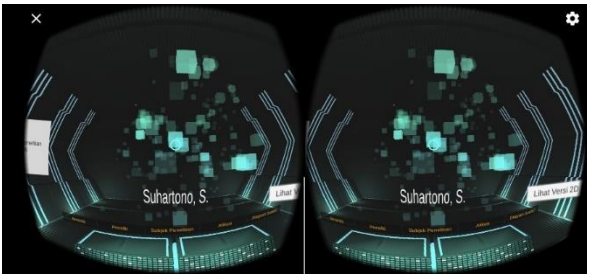
*Gambar C. 19  Pengujian Daftar Afiliasi*



*Gambar C. 20  Pengujian Detail Afiliasi*



*Gambar C. 21  Pengujian Diagram Scatter*




*Gambar C. 22  Pengujian Diagram Scatter 3-dimensi*

# BIODATA PENULIS

Fariz Ardin Adhiyaksa, lahir di Bojonegoro pada tanggal 24 Februari tahun 1998. Menempuh pendidikan di SDIT Al-Uswah Tuban, SMPN 3 Tuban. Lulus dari SMAN 1 Bojonegoro pada tahun 2016 dan melanjutkan studinya di Departemen Informatika Institut Teknologi Sepuluh Nopember Surabaya.

Aktif mengikuti organisasi dan kepanitiaan antara lain staf Departemen Dalam Negeri Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) 2017-2018, Staf Medfo IFLS 2017, Staf Medfo IFLS 2018 staf Publikasi dan Dokumentasi INOCHI 2017, staf Publikasi dan Dokumentasi K-Fest 2017, staf 3D Schematics 2017, Staff 3D Schematics 2018 Koordinator Dokumentasi INOCHI 2018, Koordinator Publikasi K-Fest 2018, staf 3D Schematics 2017, Staff 3D Schematics 2018, staf IM BEM FTIK 2018. Hobi dalam bermusik dengan bermain alat musik gitar, drum, bass dan keyboard.

Dalam menyelesaikan Pendidikan sarjana, penulis mengambil bidang minat Interaksi, Grafika, dan Seni (IGS). Penulis dapat dihubungi melalui alamat *e-mail*: fariz.ardin@gmail.com.