



TUGAS AKHIR-KI091391

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK REKOMENDASI PEMILIHAN *REAL ESTATE* DENGAN METODE *MASHUP*

MOCHAMAD RIZKY ILMAWAN

NRP 5108 100 082

Dosen Pembimbing I

Dwi Sunaryono, S.Kom., M.Kom.

Dosen Pembimbing II

Isye Ariesianti, S.Kom., M.Phil.

JURUSAN TEKNIK INFORMATIKA

Fakultas Teknologi Informasi

Institut Teknologi Sepuluh Nopember

Surabaya 2013



FINAL PROJECT-KI091391

DESIGN AND DEVELOPMENT OF REAL ESTATE RECOMMENDATION APPLICATION USING MASHUP METHOD

MOCHAMAD RIZKY ILMAWAN
NRP 5108 100 082

Advisor I
Dwi Sunaryono, S.Kom., M.Kom.

Advisor II
Isye Ariesanti, S.Kom., M.Phil.

DEPARTMENT OF INFORMATICS
Faculty of Information Technology
Sepuluh Nopember Institute of Technology
Surabaya 2013

LEMBAR PENGESAHAN

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK REKOMENDASI PEMILIHAN *REAL* *ESTATE* DENGAN METODE *MASHUP*

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada

Bidang Studi Rekayasa Perangkat Lunak
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:

MOCHAMAD RIZKY ILMAWAN

NRP 5108 100 082

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Dwi Sunaryono, S.Kom., M.Kom.
NIP. 197205281997021001 (Pembimbing I)
2. Isye Ariesianti, S.Kom., M.
NIP. 197804122006042001 (Pembimbing II)



**SURABAYA
FEBRUARI, 2013**

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK REKOMENDASI PEMILIHAN REAL ESTATE DENGAN METODE MASHUP

Nama Mahasiswa : Mochamad Rizky Ilmawan
NRP : 5108 100 082
Jurusan : Teknik Informatika FTIF-ITS
Dosen Pembimbing I : Dwi Sunaryono, S.Kom., M.Kom.
Dosen Pembimbing II : Isye Ariesianti, S.Kom., M.Phil.

Abstrak

Real estate merupakan kebutuhan yang sangat penting untuk menunjang aktivitas dan kehidupan manusia. Selain itu, real estate merupakan salah satu lahan bisnis yang paling menjanjikan untuk digeluti saat ini. Oleh karena itu, banyak sekali usahawan bidang real estate bermunculan untuk dapat meraih keuntungan dalam bidang ini. Sehingga peran media informasi sangat dibutuhkan dalam menunjang kegiatan tersebut. Karena pentingnya informasi mengenai real estate ini, banyak bermunculan media informasi yang menghadirkan informasi mengenai real estate. Namun semua media informasi tersebut hanya menyediakan informasi karakteristik real estate dan tidak menyajikan informasi mengenai akses dengan fasilitas umum dan kondisi lingkungan. Ini membuat pembeli atau pengusaha tidak bisa optimal dalam memilih real estate karena informasi yang kurang lengkap.

Perangkat lunak ini dibuat untuk menyajikan informasi mengenai real estate secara lengkap baik informasi karakteristik maupun informasi akses dengan fasilitas umum. Informasi akan diambil dari media informasi web yang sudah tersedia. Informasi yang didapatkan tersebut akan diolah dan digabungkan dengan layanan web Google Maps API sehingga akan didapat informasi mengenai akses dengan fasilitas umum. Perangkat ini dibuat dalam bentuk sistem informasi web agar mudah diakses oleh banyak orang.

Diharapkan dengan sistem perangkat lunak ini, pembeli dan pengusaha real estate dapat memperoleh informasi lengkap serta optimal sehingga dapat memberikan keuntungan yang besar. Disamping itu, dengan adanya aplikasi ini pengguna akan dimudahkan dalam mencari real estate dengan fasilitas Google Maps.

Kata kunci: Mashup, Pencarian, Real Estate, Rekomendasi.

DESIGN AND DEVELOPMENT OF REAL ESTATE RECOMMENDATION APPLICATION USING MASHUP METHOD

Student Name : Mochamad Rizky Ilmawan
Student ID : 5108 100 082
Department : Informatics FTIF-ITS
Advisor I : Dwi Sunaryono, S.Kom., M.Kom.
Advisor II : Isye Ariesianti, S.Kom., M.Phil.

Abstract

Real estate is very important to support human activity and human life. More over, real estate is one of the most promising business at this time. Therefore, several real estate businessmen able to make profits in this field. Thus, media role is very important to support their activity. Because of the importance of real estate information, severals media exist to supply information about real estate. But the media that presents real estate information only provide characteristics of real estate. But it does not provide information about access to public facillities and environmental conditions. This condition makes real estate buyer or bussinessman can not choose real estate optimally.

This final project aims to design software that provides information about real estate, both the characteristics of information and access information to public facilities. Information is retrieved from the web that is available. The obtained information will be processed and combined with Google Maps API web service to gain access to public facilities information. The software is developed in the form of web information system.

The usage of this software is expected to assist buyers and real estate entrepreneurs to obtain complete information. In addition, with this application the user will find real estate with Google Map facility easily.

Keyword: Mashup, Real Estate, Recommendation, Searching.

KATA PENGANTAR

Segala puji dan syukur kehadiran ALLAH SWT yang telah memberikan rahmat dan hidayah-nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul perancangan dan pembuatan perangkat lunak rekomendasi pemilihan *real estate* dengan metode *mashup*.

Dalam penyusunan Tugas Akhir, penulis berusaha untuk menerapkan ilmu yang telah didapat selama menempuh perkuliahan dengan tidak terlepas dari bimbingan, petunjuk, dan bantuan dari berbagai pihak.

Pada kesempatan ini penulis hendak menyampaikan penghormatan dan terimakasih yang sebesar-besarnya kepada pihak-pihak yang telah memberi bantuan moril maupun materi secara langsung atau tidak langsung kepada pihak-pihak sebagai berikut.

1. Ayah, ibu, dan kakak tercinta, yang telah dengan sabar membesarkan, mendidik, dan menyayangi penulis sejak lahir hingga saat ini.
2. Bapak Dwi Sunaryono dan Ibu Isye Ariesianti, selaku dosen pembimbing yang telah memberikan banyak arahan dalam menyelesaikan Tugas Akhir dan pengalaman yang sangat berharga bagi penulis untuk kehidupan masa depan.
3. Ibu Umi Laili Yuhana selaku dosen wali penulis.
4. Teman-teman laboratorium GCL dan RPL yang telah mendukung pembuatan Tugas Akhir ini.
5. Teman-teman UKM *Taekwondo* yang selalu memberikan semangat dalam menyelesaikan studi di ITS.

Tiada untaian kata yang cukup yang dapat penulis sampaikan sebagai balas atas jasa yang penulis terima melainkan hanya harapan semoga ALLAH SWT membalas semua amal tersebut. Penulis menyadari bahwa Tugas Akhir ini masih jauh dari

kesempurnaan sehingga dengan segala kerendahan hati penulis mengharapkan segala saran dan kritik yang membangun dari pembaca.

Surabaya, Februari 2013
MOCHAMAD RIZKY ILMAWAN

DAFTAR ISI

LEMBAR PENGESAHAN.....	vii
Abstrak	ix
Abstract	xi
KATA PENGANTAR.....	xiii
DAFTAR ISI	xv
DAFTAR GAMBAR.....	xxi
DAFTAR TABEL	xxv
DAFTAR KODE SUMBER.....	xxvii
BAB I PENDAHULUAN	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah.....	3
1.4 Tujuan.....	3
1.5 Metodologi Pembuatan Tugas Akhir.....	3
1.6 Sistematika Penulisan	5
BAB II TINJAUAN PUSTAKA	7
2.1 <i>Mashup</i>	7
2.1.1 Definisi <i>mashup</i>	7
2.1.2 Komponen <i>mashup</i>	8
2.1.2.1 Isi dan Sumber Data	8
2.1.2.2 Algoritma atau Proses <i>Mashup</i>	9
2.1.2.3 <i>Platform</i> Presentasi.....	9
2.1.3 <i>Mashup</i> di <i>Web</i>	9
2.1.4 Pengembangan <i>Mashup</i>	10
2.1.4.1 <i>Web Mashup</i>	11
2.1.4.2 Pendekatan Pengembangan <i>Mashup</i>	11
2.1.4.3 Pengembangan <i>Mashup</i> Manual.....	12
2.1.4.4 Alat Bantu Pengembangan <i>Mashup</i>	13
2.1.5 Lingkungan Pengembangan <i>Mashup</i>	13
2.1.5.1 Paradigma Lingkungan Antarmuka dan Target Pengguna	13
2.1.5.2 Kebutuhan Sistem.....	14
2.1.6 Lingkungan Waktu Dijalankan.....	15

2.1.6.1 Model Pengembangan	15
2.1.6.2 Lokasi Waktu Berjalan	15
2.1.6.3 Kebutuhan Sistem.....	15
2.1.6.4 Kemampuan Sistem.....	15
2.2 Google Maps API.....	17
2.2.1 Google Maps JavaScript API.....	17
2.2.2 Layanan Web Google Maps.....	18
2.2.2.1 <i>Direction</i> API	19
2.2.2.2 Distance Matrix API.....	19
2.2.2.3 <i>Elevation</i> API	20
2.2.2.4 <i>Geocoding</i> API	21
2.2.3 Google Place API.....	21
2.3 Web Crawler.....	22
2.4 SQL Server	24
BAB III ANALISIS DAN PERANCANGAN	27
3.1 Analisis Sistem	27
3.1.1 Deskripsi Umum Perangkat Lunak.....	27
3.1.2 Arsitektur Sistem	29
3.1.2.1 Lapisan Model	30
3.1.2.2 Lapisan Tampilan (<i>View</i>).....	30
3.1.2.3 Lapisan Pengendali (<i>Controller</i>)	31
3.1.3 Spesifikasi Kebutuhan Perangkat Lunak	31
3.1.4 Analisis Aktor.....	33
3.1.5 Skenario Kasus Penggunaan.....	34
3.1.6 Kelas Analisis.....	36
3.1.6.1 Kelas Analisis Proses Merayapi Informasi.....	36
3.1.6.2 Kelas Analisis Pengurai Dokumen	39
3.1.6.3 Kelas Analisis Proses <i>Geocoding</i>	44
3.1.6.4 Kelas Analisis Pemberi Rekomendasi	50
3.1.6.5 Analisis Kelas Pemberi Informasi <i>Real Estate</i>	56
3.1.6.6 Analisis Kelas Pemberi Informasi Fasilitas Umum	63
3.1.7 Spesifikasi Kasus Penggunaan	68
3.1.7.1 Kasus Penggunaan Pencarian Rekomendasi <i>Real Estate</i>	69

3.1.7.2 Kasus Penggunaan Memilih <i>Real Estate</i>	71
3.1.7.3 Kasus Penggunaan Melihat Informasi <i>Real Estate</i>	73
3.1.7.4 Kasus Penggunaan Melihat Informasi Fasilitas Umum	76
3.2 Tahap Perancangan	79
3.2.1 Perancangan Data	79
3.2.1.1 Data Masukan	80
3.2.1.2 Data Proses	80
3.2.1.3 Data Keluaran	80
3.2.1.4 Data Basis Data	80
3.2.2 Perancangan Proses <i>Server</i>	87
3.2.2.1 Proses Merayapi (<i>Crawling</i>) HTML	87
3.2.2.2 Proses Penguraian (<i>Parser</i>)	90
3.2.2.3 Proses <i>Geocoding</i>	93
3.2.2.4 Proses Pencarian Fasilitas Umum	96
3.2.2.5 Proses Pemberian Rekomendasi	99
3.2.3 Perancangan Proses Klien	100
3.2.3.1 Proses Menampilkan Hasil Rekomendasi	101
3.2.3.2 Proses Menampilkan Informasi <i>Real Estate</i>	102
3.2.4 Perancangan Diagram Kelas	103
3.2.5 Perancangan Antarmuka	104
3.2.5.1 Kriteria Pengguna	105
3.2.5.2 Peta	106
3.2.5.3 Daftar <i>Real Estate</i>	106
3.2.5.4 Informasi	106
3.2.5.5 Gambar	106
BAB IV IMPLEMENTASI	107
4.1 Lingkungan Pembangunan Perangkat Lunak	107
4.2 Implementasi Antarmuka Halaman Utama	107
4.3 Implementasi Pencarian Rekomendasi <i>Real Estate</i> ...	109
4.3.1 Implementasi Antarmuka Pencarian Rekomendasi <i>Real Estate</i>	109
4.3.2 Implementasi Kode Program Pencarian Rekomendasi <i>Real Estate</i>	112

4.3.2.1	Implementasi View Kode Program Pencarian Rekomendasi <i>Real Estate</i>	112
4.3.2.2	Implementasi Controller Kode Program Pencarian Rekomendasi <i>RealEstate</i>	117
4.3.2.3	Implementasi Model Kode Program Pencarian Rekomendasi <i>Real Estate</i>	117
4.4	Implementasi Memilih <i>Real Estate</i>	120
4.4.1	Implementasi Antarmuka Memilih <i>Real Estate</i>	120
4.4.2	Implementasi Kode Program Memilih <i>Real Estate</i>	121
4.4.2.1	Implementasi View Kode Program Pencarian Rekomendasi <i>Real Estate</i>	121
4.4.2.2	Implementasi Controller Kode Program Pencarian Rekomendasi <i>Real Estate</i>	123
4.4.2.3	Implementasi Model Kode Program Pencarian Rekomendasi <i>Real Estate</i>	124
4.5	Implementasi Melihat Informasi <i>Real Estate</i>	126
4.5.1	Implementasi Antarmuka Melihat Informasi <i>Real Estate</i>	126
4.5.2	Implementasi Kode Program Melihat Informasi <i>Real Estate</i>	127
4.5.2.1	Implementasi View Kode Program Melihat Informasi <i>Real Estate</i>	127
4.5.2.2	Implementasi Controller Kode Melihat Informasi <i>Real Estate</i>	128
4.5.2.3	Implementasi Model Kode Program Melihat Informasi <i>Real Estate</i>	130
4.6	Melihat Informasi Fasilitas Umum.....	131
4.6.1	Implementasi Antarmuka Melihat Informasi Fasilitas Umum.....	131
4.6.2	Implementasi Kode Program Melihat Informasi Fasilitas Umum.....	133
4.6.2.1	Implementasi View Kode Program Melihat Informasi Fasilitas Umum	133

4.6.2.2	Implementasi <i>Controller</i> Kode Program Melihat Informasi Fasilitas Umum.....	134
4.6.2.3	Implementasi <i>Controller</i> Kode Program Melihat Informasi Fasilitas Umum.....	136
4.7	Implementasi Proses dalam Sistem	137
4.7.1	Kelas DownloadData.....	138
4.7.2	Kelas Parser	138
4.7.3	Kelas DBConnection	138
4.7.4	Kelas InfrastrukturFinder	138
4.7.5	Kelas GMGeoCode	139
4.7.6	Kelas Rekomendasi	139
BAB V	UJI COBA PERANGKAT LUNAK.....	141
5.1	Lingkungan Uji Coba	141
5.2	Skenario Uji Coba	142
5.3	Uji Coba Kebutuhan Fungsional Klien	143
5.3.1	Uji Coba Meminta Rekomendasi	143
5.3.2	Uji Coba Memilih Hasil Rekomendasi.....	146
5.3.3	Uji Coba Menampilkan Informasi <i>Real Estate</i> ..	149
5.3.4	Uji Coba Memunculkan Fasilitas Umum	151
5.4	Uji Coba Proses <i>Server</i>	153
5.4.1	Uji Coba Mengunduh Informasi HTML dari Situs Sumber	153
5.4.2	Uji Coba Mengunduh dan Mengolah Data XML dari <i>Google Maps</i> API.....	156
5.3	Uji Penggunaan	159
5.3.1	Skenario Uji Penggunaan	159
5.3.2	Evaluasi Pengujian Penggunaan.....	160
5.4	Evaluasi Uji Coba.....	164
BAB VI	PENUTUP.....	165
6.1	Kesimpulan.....	165
6.2	Saran.....	166
DAFTAR	PUSTAKA.....	167
LAMPIRAN A	ATRIBUT DAN FUNGSI KELAS	169
LAMPIRAN B	PROFIL PENGISI ANGKET	175
LAMPIRAN C	KODE SUMBER.....	177

C.1 Kelas DownloadData	177
C.2 Kelas GetData	178
C.3 Kelas DBConnection	181
C.4 Kelas InfrastrukturFinder	185
C.5 Kelas GMGeoCod	188
C.6 Kelas Rekomendasi	189
BIODATA PENULIS	197

DAFTAR GAMBAR

Gambar 2.1 Situs Housingmap.com	10
Gambar 2.2 Arsitektur <i>Web Crawler</i>	23
Gambar 3.1 Ilustrasi Sistem	29
Gambar 3.2 Pola Arsitektur Situs <i>Web</i>	30
Gambar 3.3 Diagram Kasus Penggunaan	35
Gambar 3.4 Diagram Aktivitas Pencarian Rekomendasi <i>Real Estate</i>	70
Gambar 3.5 Diagram Urutan Pencarian Rekomendasi <i>Real Estate</i>	71
Gambar 3.6 Diagram Aktivitas Memilih <i>Real Estate</i>	72
Gambar 3.7 Diagram Urutan Memilih <i>Real Estate</i>	73
Gambar 3.8 Diagram Aktivitas Melihat Informasi <i>Real Estate</i> ..	75
Gambar 3.9 Diagram Urutan Melihat Informasi <i>Real Estate</i>	76
Gambar 3.10 Diagram Aktivitas Melihat Informasi Fasilitas Umum	78
Gambar 3.11 Diagram Urutan Melihat Informasi Fasilitas Umum	79
Gambar 3.12 <i>Conceptual Data Model</i>	81
Gambar 3.13 <i>Physical Data Model</i>	82
Gambar 3.14 Diagram Alir Proses <i>Crawling</i> HTML	88
Gambar 3.15 Diagram Aktivitas Proses <i>Crawling</i> HTML	89
Gambar 3.16 Diagram Urutan Proses <i>Crawling</i> HTML	90
Gambar 3.17 Diagram Alir Proses Penguraian	91
Gambar 3.18 Diagram Aktivitas Proses Penguraian	92
Gambar 3.19 Diagram Urutan Proses Penguraian	93
Gambar 3.20 Diagram Alir Proses <i>Geocoding</i>	94
Gambar 3.21 Diagram Aktivitas Proses <i>Geocoding</i>	95
Gambar 3.22 Diagram Urutan Proses <i>Geocoding</i>	96
Gambar 3.23 Diagram Alir Proses Pencarian Fasilitas Umum ...	97
Gambar 3.24 Diagram Aktivitas Proses Pencarian Fasilitas Umum	98
Gambar 3.25 Diagram Urutan Proses Pencarian Fasilitas Umum	99
Gambar 3.26 Diagram Alir Proses Pemberian Rekomendasi....	100

Gambar 3.27 Diagram Alir Proses Menampilkan Hasil Rekomendasi	101
Gambar 3.28 Diagram Alir Proses Menampilkan Informasi <i>Real Estate</i>	102
Gambar 3.29 Diagram Kelas	103
Gambar 3.30 Rancangan Antarmuka Perangkat Lunak	105
Gambar 4.1 Antarmuka Halaman Utama Perangkat Lunak	108
Gambar 4.2 Antarmuka Memilih Karakteristik Perangkat Lunak	109
Gambar 4.3 Antarmuka Memasukkan Jarak Fasilitas Umum ...	110
Gambar 4.4 Antarmuka Memasukkan Jarak Sekolah	110
Gambar 4.5 Antarmuka Daftar <i>Real Estate</i>	111
Gambar 4.6 Antarmuka Tabel <i>Real Estate</i>	111
Gambar 4.7 Antarmuka Informasi Peta	112
Gambar 4.8 Kode Program Memasukkan Karakteristik.....	113
Gambar 4.9 Kode Program Memasukkan Jarak dengan Fasilitas Umum	113
Gambar 4.10 Kode Program untuk Memasukkan Jarak dengan Sekolah	114
Gambar 4.11 Kode Program untuk Memasukkan Nilai Kriteria	114
Gambar 4.12 Kode Program Menampilkan Daftar Rekomendasi <i>Real Estate</i>	115
Gambar 4.13 Kode Program Menampilkan Rekomendasi pada Tabel	115
Gambar 4.14 Kode Program Menampilkan Rekomendasi <i>Real Estate</i> pada Peta	116
Gambar 4.15 Kode <i>JavaScript</i> untuk Menghubungkan dengan Google Maps API	116
Gambar 4.16 Kode Kelas Rekomendasi	117
Gambar 4.17 Kode Program Kelas <i>PropertyLogic</i>	118
Gambar 4.18 Kode Program Kelas <i>RealEstate</i>	119
Gambar 4.19 Antarmuka Pengguna Memilih <i>Real Estate</i>	120
Gambar 4.20 Antarmuka Sistem Memberikan Informasi <i>Real Estate</i>	121
Gambar 4.21 Kode Program Memilih <i>Real Estate</i>	122

Gambar 4.22 Kode Program Menampilkan <i>Real Estate</i>	122
Gambar 4.23 Kode Program Menampilkan Pilihan <i>Real Estate</i>	123
Gambar 4.24 Kode Program Menampilkan Informasi <i>Real Estate</i>	123
Gambar 4.25 Kode Program Kelas <i>PropertyLogic</i>	124
Gambar 4.26 Kode Program Kelas <i>Property</i>	125
Gambar 4.27 Antarmuka Melihat <i>Real Estate</i>	126
Gambar 4.28 Antarmuka Menampilkan Informasi <i>Real Estate</i>	127
Gambar 4.29 Kode Program untuk Melihat Informasi Detail <i>Real Estate</i>	128
Gambar 4.30 Kode Program Menampilkan Informasi <i>Real Estate</i>	128
Gambar 4.31 Kode Program Melihat Informasi <i>Real Estate</i>	129
Gambar 4.32 Kode Program Menampilkan Informasi <i>Real Estate</i>	129
Gambar 4.33 Kode Program Kelas <i>RealEstate</i>	130
Gambar 4.34 Antarmuka Memilih Fasilitas Umum	131
Gambar 4.35 Antarmuka Melihat Daftar Fasilitas Umum	132
Gambar 4.36 Antarmuka Menampilkan Informasi Fasilitas Umum	132
Gambar 4.37 Kode Program Memilih Fasilitas Umum.....	133
Gambar 4.38 Kode Program Melihat Tabel Fasilitas Umum....	134
Gambar 4.39 Kode Program Menampilkan Informasi Fasilitas Umum.....	134
Gambar 4.40 Kode Program Menampilkan Tabel Fasilitas Umum	135
Gambar 4.41 Kode Program Menampilkan Informasi Fasilitas Umum.....	135
Gambar 4.42 Kode Program Kelas <i>InfrastrukturSearch</i>	136
Gambar 4.43 Kode Program Kelas <i>Infrastruktur</i>	137
Gambar 5.1 Meminta Rekomendasi Tanpa Memasukkan Kriteria	145
Gambar 5.2 Meminta Rekomendasi dengan Memasukkan Kriteria	146

Gambar 5.3 Memilih Hasil Rekomendasi	148
Gambar 5.4 Memilih Menuju Situs Sumber.....	148
Gambar 5.5 Halaman Informasi <i>Real Estate</i>	150
Gambar 5.6 Memunculkan Rumah Sakit	152
Gambar 5.7 Memunculkan Semua Fasilitas Umum	153
Gambar 5.8 Halaman yang Akan Diunduh	155
Gambar 5.9 Data Awal Sebelum Diperbarui	155
Gambar 5.10 Data Setelah Diperbarui.....	156
Gambar 5.11 Data dalam Basis Data.....	157
Gambar 5.12 Data Sebelum Dimasukkan ke Basis Data.....	158
Gambar 5.13 Rekapitulasi Pertanyaan 1.....	161
Gambar 5.14 Rekapitulasi Pertanyaan 2.....	161
Gambar 5.15 Rekapitulasi Pertanyaan 3.....	162
Gambar 5.16 Rekapitulasi Pertanyaan 4.....	162
Gambar 5.17 Rekapitulasi Pertanyaan 5.....	163
Gambar 5.18 Rekapitulasi Pertanyaan 6.....	163

DAFTAR TABEL

Tabel 3.1 Tabel Aktor Aplikasi	34
Tabel 3.2 Deskripsi Kasus Penggunaan	35
Tabel 3.3 Kandidat Kelas untuk Proses Merayapi Informasi	37
Tabel 3.4 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Proses Merayapi Informasi	39
Tabel 3.5 Kandidat Kelas untuk Proses Pengurai Dokumen	40
Tabel 3.6 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Proses Penguraian Dokumen	43
Tabel 3.7 Kandidat Kelas untuk Proses <i>Geocoding</i>	45
Tabel 3.8 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Proses <i>Geocoding</i>	49
Tabel 3.9 Kandidat Kelas untuk Pemberi Rekomendasi	51
Tabel 3.10 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Pemberi Rekomendasi	54
Tabel 3.11 Kandidat Kelas untuk Pemberi Informasi <i>Real Estate</i>	57
Tabel 3.12 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Pemberi Informasi <i>Real Estate</i>	61
Tabel 3.13 Kandidat Kelas untuk Pemberi Informasi Fasilitas Umum	64
Tabel 3.14 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Pemberi Informasi Fasilitas Umum	67
Tabel 3.15 Spesifikasi Kasus Penggunaan Pencarian Rekomendasi <i>Real Estate</i>	69
Tabel 3.16 Spesifikasi Kasus Penggunaan Memilih <i>Real Estate</i>	71
Tabel 3.17 Spesifikasi Kasus Penggunaan Melihat Informasi <i>Real</i> <i>Estate</i>	73
Tabel 3.18 Spesifikasi Kasus Penggunaan Melihat Informasi Fasilitas Umum	76
Tabel 3.19 Penjelasan Atribut Tabel RealEstate	83
Tabel 3.20 Penjelasan Atribut Tabel PublicInfrastructur	84
Tabel 3.21 Penjelasan Atribut Tabel RealEstateProperties	85

Tabel 3.22 Penjelasan Atribut Tabel <i>InfrastrukturType</i>	86
Tabel 3.23 Penjelasan Atribut Tabel <i>City</i>	86
Tabel 3.24 Penjelasan Atribut Tabel <i>Area</i>	86
Tabel 3.25 Penjelasan Atribut Tabel <i>UpdateTime</i>	87
Tabel 5.1 Uji Coba Meminta Rekomendasi	143
Tabel 5.2 Uji Coba Memilih Hasil Rekomendasi.....	146
Tabel 5.3 Uji Coba Menampilkan Informasi <i>Real Estate</i>	149
Tabel 5.4 Uji Coba Memunculkan Fasilitas Umum	151
Tabel 5.5 Uji Coba Mengunduh Informasi HTML dari Situs Sumber.....	154
Tabel 5.6 Mengunduh dan Mengolah Data XML dari Google <i>Maps</i> API.....	156
Tabel B.1 Profil Pengguna Penguji Perangkat Lunak	175

DAFTAR KODE SUMBER

Kode Sumber C.1 Kelas DownloadData	178
Kode Sumber C.2 Kelas GetData	181
Kode Sumber C.3 Kelas DBConnection	185
Kode Sumber C.4 Kelas InfrastrukturFinder	187
Kode Sumber C.5 Kelas GMGeoCod.....	188
Kode Sumber C.6 Kelas Rekomendasi.....	195

[Halaman ini sengaja dikosongkan]

BAB I

PENDAHULUAN

Pada bab ini akan dipaparkan garis besar Tugas Akhir yang meliputi latar belakang, tujuan, rumusan masalah, batasan permasalahan, metodologi pembuatan Tugas Akhir, dan sistematika penulisan.

1.1 Latar Belakang

Seiring dengan pesatnya laju pertumbuhan ekonomi di kota besar, menyebabkan meningkatnya kebutuhan akan *real estate* (tempat tinggal, pertokoan, dan gedung perkantoran) di kota tersebut. Meningkatnya kebutuhan *real estate* di perkotaan merupakan salah satu aspek ekonomi yang sangat menguntungkan dan banyak diminati investor saat ini. Evaluasi dalam hal aset dan fasilitas pada *real estate* merupakan salah satu titik fokus yang harus diperhatikan dalam berinvestasi pada bidang ini. Hal tersebut bertujuan agar *real estate* tersebut dapat berkembang dan memberikan keuntungan yang besar bagi investor maupun pembeli.

Dalam memilih *real estate* yang baik dan menguntungkan, banyak sekali aspek yang harus diperhatikan antara lain: karakteristik *real estate* (harga, luas tanah, luas bangunan, dan semua yang dimiliki oleh *real estate* tersebut), akses dengan fasilitas umum (sekolah, rumah sakit, pasar, stasiun, dan fasilitas penunjang lain), dan kondisi lingkungan (keamanan, kesehatan, hubungan sosial). Dalam sebuah survei yang dilakukan oleh Giudice dari Universitas Naples menyimpulkan bahwa aspek yang paling diperhatikan oleh seseorang dalam membeli *real estate* adalah akses dengan fasilitas umum dan kondisi lingkungan [1].

Saat ini telah banyak tersedia sistem informasi berupa situs *web* yang menyediakan informasi penjualan *real estate* di internet. Namun semua situs *web* tersebut hanya menyediakan informasi mengenai karakteristik *real estate* dan tidak menyediakan informasi mengenai akses dengan fasilitas umum dan kondisi lingkungan. Sehingga pembeli ataupun investor tidak dapat secara optimal memilih *real estate* yang ideal. Agar pembeli dapat memilih *real estate* yang ideal, dibutuhkan sebuah sistem informasi yang menyediakan informasi secara lengkap mengenai *real estate* dengan informasi berupa akses dengan fasilitas umum dan kondisi lingkungan, sehingga dapat memberikan rekomendasi pada pengguna mengenai *real estate* yang sesuai. Hal tersebut dapat dilakukan dengan memberikan informasi berupa peta lokasi yang disertai dengan informasi fasilitas umum dan karakteristik *real estate*.

Sistem informasi tersebut dapat dibuat dengan cara menggabungkan informasi dari situs *web* penyedia informasi *real estate* dengan layanan *web* Google Maps API. Penggabungan dua informasi dari situs *web* tersebut lebih dikenal dengan istilah *mashup*.

1.2 Rumusan Masalah

Beberapa masalah yang terdapat pada pembuatan perangkat lunak ini adalah bagaimana memenuhi kebutuhan pengguna untuk memilih *real estate* dengan menggunakan sistem informasi rekomendasi pemilihan *real estate* yang dibuat dengan menggunakan metode *mashup*. Rincian permasalahan yang dihadapi adalah sebagai berikut:

1. Bagaimana menganalisis kebutuhan sebuah sistem informasi rekomendasi pemilihan *real estate*?
2. Bagaimana perangkat lunak memberikan rekomendasi *real estate*?

3. Bagaimana membuat perangkat lunak rekomendasi pemilihan *real estate* yang mudah digunakan dan dapat diakses semua orang?
4. Bagaimana membuat perangkat lunak rekomendasi pemilihan *real estate* dengan melakukan *mashup* Google Maps API dan situs web?
5. Bagaimana mengimplementasikan pola arsitektur MVC untuk perangkat lunak rekomendasi pemilihan *real estate* dengan menggunakan ASP.NET?

1.3 Batasan Masalah

Beberapa masalah dalam pembuatan perangkat lunak ini dibatasi ruang lingkup pembahasannya antara lain:

1. Rekomendasi *real estate* yang disediakan hanya terbatas pada *real estate* yang terdapat di kota Surabaya.
2. Perangkat lunak yang dibuat terbatas pada perangkat lunak berbasis web.
3. Bahasa pemrograman yang digunakan adalah C# dengan kerangka kerja .NET 4.0.
4. Fasilitas umum yang dijadikan sebagai acuan merupakan fasilitas umum yang terdapat pada Google Map API.

1.4 Tujuan

Tujuan dari Tugas Akhir ini adalah mengaplikasikan metode *mashup* pada sistem informasi rekomendasi *real estate* sehingga dapat memberikan rekomendasi *real estate* yang sesuai dengan permintaan pengguna.

1.5 Metodologi Pembuatan Tugas Akhir

- a. Studi Referensi dan Pemahaman Sistem
Pada tahap ini dilakukan pencarian dan pemahaman referensi untuk pembuatan perangkat lunak. Adapun tahapan-tahapan dalam studi referensi sebagai berikut:

- Mempelajari referensi mengenai cara melakukan *mashup* dalam pembuatan perangkat lunak.
- Mempelajari cara melakukan *crawler* pada suatu sistem informasi berbasis *web*.
- Mempelajari ASP.NET AJAX dengan C# 4.0.
- Mempelajari penggunaan dan komunikasi dengan layanan *web* Google Maps.
- Mempelajari fitur-fitur layanan *web* Google Maps yang dapat digunakan dalam membantu memberikan rekomendasi.
- Mempelajari penggunaan Google Maps API v3 menggunakan *JavaScript*.

b. Perancangan Perangkat Lunak

Pada tahap ini dilakukan analisis awal dan mendefinisikan kebutuhan sistem untuk mengetahui masalah yang sedang dihadapi. Kemudian dari hasil analisis dilakukan pembuatan arsitektur perangkat lunak yang akan dikembangkan. Urutan perancangan perangkat lunak adalah sebagai berikut:

- Menentukan dan menganalisis permasalahan.
- Merancang diagram kasus penggunaan yang merupakan analisis kebutuhan pada aplikasi yang akan dibangun.
- Analisis kebutuhan fungsional dan non-fungsional.
- Menentukan arsitektur perangkat lunak yang digunakan.
- Membuat rancangan relasi basis data.
- Merancang perangkat lunak.

c. Pembuatan Perangkat Lunak

Pada tahap ini dilakukan pembuatan perangkat lunak yang merupakan implementasi dari rancangan yang telah dibuat sebelumnya. Perincian tahap ini adalah sebagai berikut:

- Membuat perangkat lunak untuk mengambil data dari *web* dengan menggunakan bahasa C# dalam kerangka kerja .NET 4.0.

- Melakukan penggabungan antara informasi pada layanan *web* Google Maps dengan informasi dari *web* dengan menggunakan metode *mashup*.
 - Membuat sistem untuk memberi rekomendasi pada pengguna menggunakan ASP.NET 4.0.
 - Membuat sistem informasi *web* yang digunakan pengguna untuk memperoleh rekomendasi *real estate* menggunakan ASP.NET 4.0 dan Google Maps JavaScript API V3.
- d. Uji Coba dan Evaluasi
- Pada tahap ini akan dilakukan pengujian terhadap perangkat lunak menggunakan skenario yang telah dipersiapkan sebelumnya. Uji coba dan evaluasi perangkat lunak dilakukan untuk mencari masalah yang mungkin timbul, mengevaluasi jalannya program, dan mengadakan perbaikan jika ada kekurangan.
- e. Penyusunan Buku Tugas Akhir
- Pada tahap ini dilakukan pendokumentasian dan pelaporan dari seluruh konsep, dasar teori, implementasi, proses yang telah dilakukan, dan hasil-hasil yang telah didapatkan selama pengerjaan Tugas Akhir. Buku Tugas Akhir ini dibuat dengan tujuan untuk memberikan gambaran dari pengerjaan Tugas Akhir ini dan diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut.

1.6 Sistematika Penulisan

Secara garis besar, buku Tugas Akhir nantinya terdiri atas beberapa bagian yaitu:

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah,

metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

Bab II Tinjauan Pustaka

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini.

Bab III Analisis dan Perancangan

Bab ini membahas mengenai analisis dan perancangan perangkat lunak. Perancangan perangkat lunak meliputi perancangan data, arsitektur, proses, dan implementasi dari perancangan sistem yang dilakukan pada tahap perancangan.

Bab IV Implementasi

Bab ini membahas implementasi dari rancangan sistem yang dilakukan pada tahap perancangan.

Bab V Uji Coba Perangkat Lunak

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

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil uji coba yang dilakukan.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan Tugas Akhir.

Lampiran

Merupakan bab tambahan berupa lampiran informasi yang penting pada aplikasi ini.

BAB II

TINJAUAN PUSTAKA

Bab ini berisi teori-teori yang mendukung pembuatan Tugas Akhir ini beserta teknologi yang digunakan.

2.1 *Mashup*

Mashup adalah suatu situs *web* atau aplikasi *web* yang mengombinasikan isi lebih dari satu sumber untuk memproduksi sesuatu yang benar-benar baru. Hasil situs *web* yang diciptakan proses *mashup* menyediakan sesuatu yang baru dan lebih bermanfaat dibandingkan dengan isi dari masing-masing sumber.

2.1.1 Definisi *mashup*

Kata *mashup* sesungguhnya berasal dari industri musik yaitu mengacu pada pencampuran dua atau banyak lagu dari rekaman musik atau instrumen musik dalam menciptakan lagu yang benar-benar baru [2]. Konsep *mashup* pada *web* bukanlah hal baru. Sejak permulaan *web*, orang-orang telah mengumpulkan informasi dari berbagai sumber dan menggabungkannya bersama dengan kode dari *Hypertext Transfer Protocol* (selanjutnya akan disebut dengan HTTP) untuk menyajikan bentuknya yang lebih baru dan lebih berguna.

Selama tahapan awal penggabungan isi *web*, pengembang perlu secara manual mengumpulkan dan mengodekan informasi ke dalam halaman HTML. *Mashup* mengakses data atau informasi secara langsung dari situs *web* sumber dan secara programatik, kemudian menciptakan situs *web* atau aplikasi *web* yang dinamis dan baru, dan menampilkan data mereka lebih terintegrasi atau lebih bermanfaat. Sekali proses *mashup* telah didirikan, akan secara otomatis membaca, memproses, menampilkan, dan memperbarui isi dan data yang dikumpulkan dari sumber-sumber

yang berbeda untuk pengguna akhir dalam bentuk suatu situs *web* [2]. Dibandingkan dengan tahap awal integrasi isi *web*, tidak ada campur tangan secara manual pada tahap apapun.

2.1.2 Komponen *mashup*

Untuk memulai menciptakan *mashup*, ada tiga komponen penting, yaitu isi dan sumber data, algoritma atau proses *mashup*, dan *platform* presentasi. Berikut penjelasan lebih lanjut tentang tiga komponen penting dalam menciptakan *mashup*.

2.1.2.1 Isi dan Sumber Data

Isi dan sumber data adalah pondasi dari *mashup* manapun. Tanpa isi dan data, suatu *mashup* tidak berarti. Isi dan data diperoleh melalui API, *web feeds*, atau teknik *crawling*. Sekarang ini banyak penyedia informasi telah mengambil langkah untuk membuat isi dan data mereka lebih mudah diakses. Beberapa telah mengembangkan API (*Application Programming Interfaces*) untuk anggota pengembang akses kepada isi dan data melalui protokol *web* seperti *REST* (*Representational State Transfer*) dan juga melalui layanan *web*. Selain itu juga terdapat layanan *web feeds* (dokumen XML sederhana untuk *content syndication*) dalam format populer seperti RSS atau *Atom* sebagai alat pengaksesan isi dan data mereka. Melalui penggunaan API dan *web feeds*, pengembang diberikan hak untuk mengakses isi dan data dari sistem yang sudah ada dan dapat berkumpul menerima data ini secara programatik [2]. Tetapi tidak semua sumber data menyediakan alat seperti itu. Untuk mendapatkan isi dari penyedia ini, suatu teknik yang dikenal dengan *crawling* mesti dilakukan. *Crawling* adalah proses penerimaan serangkaian campuran data kasar dan mengekstraksi data tersebut, kemudian membentuknya ke suatu struktur data pengembang. Secara perbandingan, API dan *web feeds* menyediakan sumber informasi yang lebih handal seperti yang disediakan langsung oleh data asli.

2.1.2.2 Algoritma atau Proses *Mashup*

Untuk menjalankan algoritma dan proses *mashup* merupakan suatu hal yang cukup mudah. Selama langkah perancangan, isi dan data dari sumber yang berbeda diintegrasikan. Proses *mashup* mungkin berada pada *server*, klien atau kombinasi keduanya. Pada sisi *server*, *mashup* dapat dikembangkan menggunakan teknik *server side scripting* seperti CGI, PHP, ASP, dan lain-lain. Pada sisi klien, *mashup* dapat dihasilkan langsung dari *browser* pengguna akhir melalui teknologi *scripting* seperti *JavaScript*.

2.1.2.3 Platform Presentasi

Bagian ini adalah bagian dimana pengguna akhir akan melihat produk akhir *mashup* dan berinteraksi dengan isi *mashup*. Platform presentasi akan berinteraksi dengan pengguna akhir, mengeksekusi proses *mashup*, lalu menghasilkan isi *mashup*.

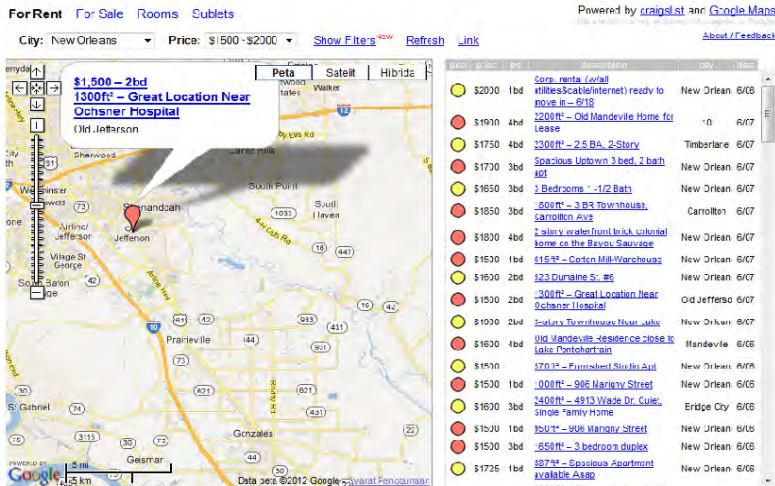
2.1.3 *Mashup* di Web

Dengan pesatnya pertumbuhan *mashup* di internet, maka *mashup* dapat diklasifikasikan ke dalam tipe berikut yaitu:

- *Map mashup*.
- *Photo and video mashup*.
- *Search mashup*.
- *News mashup*.

Jumlah terbanyak dari *mashup* saat ini adalah *map mashup*. Ini dikarenakan tersedianya fasilitas API dan akses API ke penyedia layanan peta daring seperti *Google Maps*, *Yahoo Maps*, *Microsoft Virtual Earth*, dan *AOL MapQuest*. Pengembang *map mashup* harus terlebih dahulu memperoleh informasi geografis seperti koordinat lintang dan bujur dalam *dataset* mereka, lalu mengemas data ke dalam peta daring yang sudah dirancang sebelumnya. Gambar 2.1 merupakan satu contoh pemetaan *mashup* skala besar

yaitu housingmaps.com dimana *web* mengemas daftar *real estate* dari *Craigslist* dengan menggunakan *Google Maps*.



Gambar 2.1 Situs Housingmaps.com

Ketika Google pertama kali meluncurkan layanan pemetaan mereka, tidak ada API untuk menangkap datanya. Kemudian ketika layanan API sudah ditambahkan, layanan daring lainnya ikut menyusul. Beberapa pengembang melakukan *mashup* dengan foto API dan video API dengan informasi foto dan video. Ada pula yang melakukan *mashup* dengan mencari API atau *consumer market place API*, *news source web feeds*, dan lebih banyak lagi.

2.1.4 Pengembangan Mashup

Pengembangan *mashup* yang akan dibahas di subbab ini meliputi *web mashup*, pendekatan pengembangan *mashup*, pengembangan *mashup* manual, alat bantu pengembangan *mashup*, mencirikan pendekatan *mashup*, model komponen, model penggabungan, lingkungan pengembangan, serta lingkungan waktu dijalankan.

2.1.4.1 *Web Mashup*

Web mashup adalah aplikasi *web* yang dihasilkan dengan menggabungkan isi, presentasi, atau fungsionalitas aplikasi dari sumber *web* berlainan. Tujuannya adalah menggabungkan sumber-sumber ini untuk menciptakan aplikasi atau layanan baru. Isi dan presentasi bisanya dalam bentuk RSS atau *Atom feeds*, XML, HTML, *Shock Wave Flash* (SWF), atau elemen grafik lainnya [2]. Ketersediaan API umumnya (contohnya *JavaScript*) menyediakan fungsionalitas untuk aplikasi. Isi, fungsionalitas, dan presentasi dilekatkan bersama dalam cara yang berbeda melalui *JavaScript* di dalam *browser*, dengan bahasa *server side scripting* seperti *Hypertext Preprocessor* (PHP), *Ruby*, ataupun bahasa tradisional seperti Java atau C#. *Mashup* telah menjadi satu dari sekian kata kunci paling hangat dalam area aplikasi *web*. Banyak perusahaan dan institusi bergegas menyediakan solusi *mashup*.

2.1.4.2 Pendekatan Pengembangan *Mashup*

Pengembangan *mashup* berbeda dari pengembangan aplikasi tradisional berbasis komponen, karena *mashup* melayani kebutuhan situasional, spesifik, dan terdiri dari teknologi *web* yang paling baru saat ini dan mudah digunakan seperti *Representational State Transfer* (*RESTful*), layanan *web*, RSS atau *Atom feeds*. Dengan situasi demikian, *web* adalah lingkungan yang cocok untuk melakukan pengembangan mereka. Aplikasi *The HousingMaps* (www.housingmaps.com) di Gambar 2.1 adalah contoh keberhasilan *mashup*. Aplikasi tersebut menggabungkan daftar-daftar *real estate* dari *Craigslist* dengan data peta dari *Google Maps* untuk membantu orang berpindah dari satu kota ke kota lainnya dan mencari tempat tinggal. Ketika seseorang melakukan pencarian melalui daftar *real estate*, alamat *real estate* tidak memberikan cukup informasi jika mereka belum mengenal tempat baru. *HousingMaps* memberikan daftar *real*

estate, informasi *real estate*, dan lokasi pada peta menggunakan *popup cloud*.

2.1.4.3 Pengembangan *Mashup* Manual

Aplikasi seperti *mashup* dapat secara manual dikembangkan menggunakan teknologi pemrograman *web* konvensional. Namun dengan menggunakan alat *mashup* yang tepat, dapat menguntungkan pengembangan atau bahkan membuat pengguna akhir dapat merancang *mashup* mereka sendiri. Umumnya penggabungan aplikasi dan data *enterprise* ke dalam suatu aplikasi memerlukan kemampuan pemrograman dan pengetahuan tentang skema dan semantik sumber data atau konvensi protokol bisnis untuk pertukaran pesan. Namun dengan teknologi baru seperti AJAX, layanan *RESTful*, RSS, dan *Atom* telah menyederhanakan pengembangan *mashup*. Tambahan *intelligent source components* membantu penggabungan isi, logika aplikasi, dan antarmuka [2]. Untuk pengembangan *mashup* seperti *HousingMap* secara manual dibutuhkan dua sumber aplikasi (yaitu *Craigslist* dan *Google Maps*) dan mengidentifikasi bagaimana menggunakan kembali ataupun mengekstrak data dari dua situs tersebut. *Google Maps* menyediakan *JavaScript* API sedangkan *Craigslist* menyediakan daftarnya melalui RSS. Untuk mengekstrak data *real estate* dan alamat, harus mengubah dan menginterpretasikan *RSS feed* dari *Craigslist*. Untuk mengonfigurasi penanda yang dapat dipilih yang akan menampilkan informasi *real estate* dalam suatu *popup cloud window* saat dipilih, *HousingMap* harus berinteraksi dengan *Google Maps JavaScript* API. Untuk memunculkan *automatic popup*, dibutuhkan fungsi *JavaScript* yang menanggapi pemilihan *real estate* dan bereaksi dengan cara menarik *Google Maps* API untuk memilih penanda yang dimaksud. Meskipun *Google Maps* memiliki antarmuka sendiri dan membiarkan pemakai menyeleksi *real estate* yang dibungkus oleh *Craigslist*, namun pengembang harus mampu menentukan format yang tepat. Terakhir,

pengembang harus menata dua komponen sedemikian rupa untuk membentuk antarmuka aplikasi gabungan.

2.1.4.4 Alat Bantu Pengembangan *Mashup*

Untuk mempercepat keseluruhan proses pengembangan *mashup* API yang memungkinkan pengguna akhir yang masih awam dapat menggabungkan aplikasi *web* mereka sendiri, alat-alat pengembangan yang menjurus ke *mashup* dan kerangka kerjanya telah banyak bermunculan yaitu: Yahoo Pipes, Microsoft Popfly, Intel Mash Maker, Google Mashup Editor, dan QedWiki. Alat-alat ini datang dengan aneka ragam fitur dan campuran dari pendekatan gabungan [3].

2.1.5 Lingkungan Pengembangan *Mashup*

Karakteristik lingkungan pengembangan alat *mashup* mempengaruhi efisiensi pengembangan *mashup* dan menentukan keberhasilan alat tersebut. Alat *mashup* sangat berperan dalam dukungan kepada pengguna. Beberapa alat ditujukan untuk pengembang, sedangkan lainnya lebih mengarah kepada pengguna akhir. Beberapa properti mencirikan lingkungan pengembangan *mashup* dijelaskan pada subbab dibawah ini.

2.1.5.1 Paradigma Lingkungan Antarmuka dan Target Pengguna

Alat-alat *mashup* dapat mendukung perancangan melalui antarmuka yang berbeda atau paradigma pemodelan seperti fitur visual *drag-and-drop*, *textual editor*, atau gabungan keduanya. Antarmuka dapat menargetkan rata-rata pengguna *web*, pengguna mahir, atau pemrogram.

2.1.5.2 Kebutuhan Sistem

Eksekusi alat-alat *mashup* membutuhkan modul tambahan spesifik, *plugins*, atau fitur *browser* yang ketidakhadirannya dapat menghambat penggunaan instrumen. *Yahoo Pipes* menyediakan visual editor *drag-and-drop* AJAX yang ditargetkan kepada pengguna dengan kemampuan pemrograman dasar. Editor dieksekusi dalam standar *web browser* dengan dukungan untuk XML objek *HttpRequest JavaScript*. *Google Mashup Editor browser* yang berbasis editor teks AJAX dengan sintaksis *highlighting* dan *tag completion* yang otomatis ditargetkan kepada pemrogram. *Google Mashup Editor* dapat dieksekusi dalam standar *web browser*. *Microsoft Popfly* menawarkan gambar dan editor teks dengan dukungan *drag-and-drop* kepada pengguna *web*. *Microsoft Popfly* didasarkan pada teknologi *Microsoft Silverlight*, *mandatory browser plugin*. *Intel Mash Maker* mendukung antarmuka pengguna *point-and-click* yang membolehkan pengguna mahir dan pemrogram menganotasikan halaman dan pengguna *web* awam mengekstrak dan menyatukan data melalui *copy-and-paste*. *Intel Mash Maker* membutuhkan *dedicated plugin* untuk memuat *browser* dengan fitur *mashup*. *QedWiki* datang dengan antarmuka *drag-and-drop* yang mudah digunakan kepada pengguna mahir. Dalam antarmuka ini, komponen segera divisualisasikan. *QedWiki* berjalan dalam standar *web browser* dan tidak membutuhkan *plugins* apapun [3]. Tidak seperti perlengkapan *mashup* ini, teknologi penggabungan tradisional menawarkan pengembangan aplikasi *desktop* dibandingkan dengan berbasis *browser*. Teknologi tersebut membutuhkan pembelajaran lebih lanjut karena teknologi itu lebih menakutkan dan kaya fitur. Pendekatan tradisional, yang selalu menargetkan pemrogram, menawarkan tidak adanya penggabungan data pengguna ataupun penggabungan aplikasi.

2.1.6 Lingkungan Waktu Dijalankan

Setiap alat *mashup* juga menyediakan lingkungan waktu dijalankan yang memungkinkan eksekusi alat *mashup* dan menentukan bagaimana informasi akan dikirim kepada pengguna. Ada empat aspek yang perlu diperhatikan dalam lingkungan waktu *mashup* dijalankan yaitu model pengembangan, lokasi waktu berjalan, kebutuhan sistem, dan kemampuan sistem.

2.1.6.1 Model Pengembangan

Sama seperti aplikasi *web* konvensional, aplikasi *mashup* dapat disebarkan sendirian pada *web server* apapun yang dikelola oleh pengembang *mashup* atau melalui *web server* pihak ketiga.

2.1.6.2 Lokasi Waktu Berjalan

Mashup dapat dirakit pada sisi *server* (sebagai contoh: PHP atau Ruby), sisi klien (sebagai contoh: di dalam *web browser* melalui JavaScript), atau keduanya.

2.1.6.3 Kebutuhan Sistem

Eksekusi *mashup* dapat bergantung pada ketersediaan tambahan *browser plugins* atau *extensions*. Ketika program *mashup* menyediakan informasi berupa video, maka *browser* juga harus menyediakan *plugin* untuk dapat membuka video tersebut.

2.1.6.4 Kemampuan Sistem

Pada kemampuan sistem terdapat tiga perspektif yaitu:

1. Jumlah sumber data.
2. Jumlah model (gabungan).
3. Jumlah pengguna.

Secara umum, pendekatan sisi klien tidak luput dari masalah kemampuan sistem. *Mashup* dieksekusi pada sisi klien, sehingga tidak ada masalah yang muncul kecuali dari kelebihan muatan pada sumber data itu sendiri. Tetapi hal tersebut diluar dari kendali *mashup*. Di sini masalah kemampuan sistem berhubungan dengan jumlah *instance*, jumlah pengguna, dan kompleksitas *mashup* (jumlah sumber dan hubungan pemrosesan data). Dalam semua kasus, pendekatan sisi klien menggunakan teknik dengan kemampuan sistem yang sama sebagaimana yang dilakukan penggabungan tradisional atau aplikasi *web* untuk aplikasi *web* berbasis *runtimes*.

Berbagai macam layanan *mashup* memiliki standar tersendiri dalam menyediakan kebutuhan sistem. Komposisi Yahoo *Pipes* yang ditaruh pada Yahoo *server* akan dikomputasikan dan dirakit pada sisi *server* (secara *engine-based*), sehingga mengeksekusi suatu *pipe* tidak membutuhkan persyaratan sistem apapun pada klien. Namun, *server side engine* yang mengeksekusi *pipes* mungkin akan terbebani jika banyak *pipes* sedang berjalan, jika banyak pengguna mengakses *pipe* yang sama, atau jika *pipe* terdiri dari ratusan sumber. Google *mashup Editor* ditaruh pada Google *server*. *Mashup* dieksekusi pada sisi *server* dan tidak memerlukan persyaratan sistem apapun. Sistem mengompilasi *mashup* ke dalam aplikasi *web* konvensional. *Popfly mashup* ditaruh pada Microsoft *server*. Eksekusi dari aplikasi *Popfly* muncul pada sisi klien dan bisaanya membutuhkan ketersediaan dari *Silverlight plugin*. Eksekusi sisi klien memfasilitasi *scalability* karena penggabungan multi sumber terjadi kebanyakan pada sisi klien. Intel *Mash Maker mashup* ditaruh pada komputer klien dan dieksekusi di dalam *web browser* menggunakan *Mash Maker plugin*. Meskipun hal tersebut kelihatannya tidak seperti kebanyakan jumlah sumber yang akan dilakukan *mashup*, *Mash Maker* seharusnya dapat menyekalakan secara tepat. Halaman-halaman *QedWiki* ditaruh pada IBM *server*. *Mashup* dieksekusi sebagian besar pada sisi klien dan standar *web browser* dapat

mengeksekusi halaman *QedWiki*. *Wiki engine* mungkin akan menghitung beberapa kesulitan yang terjadi jika *wiki engine* tersebut mengintegrasikan sebagian besar sumber. Mendistribusikan *workload*(entitas-entitas berbeda yang saling berhubungan) yang kelebihan mesin-mesin, dapat menjamin kemampuan sistem.

2.2 Google Maps API

Google *Maps* merupakan salah satu layanan yang diberikan oleh Google untuk membantu pengguna internet dalam hal yang berhubungan dengan peta lokasi. Jasa layanan yang dapat digunakan langsung oleh pengguna dapat diakses dalam situs <http://maps.google.com>. Data pada Google *Maps* diambil dari satelit yang berada di seluruh dunia.

Maps pertama dibuat sebagai perangkat lunak dengan bahasa C++ yang dibangun oleh Danish bersaudara Lars dan Jens Rasmussen di Sydney pada perusahaan *Where 2 Technologis*. Kemudian program tersebut diubah menjadi perangkat lunak berbasis *web* sebagai bagian dari pengelola Google. Pada tahun 2004 perusahaan dibeli oleh Google dan program tersebut berubah menjadi perangkat lunak berbasis *web* Google *Maps*.

2.2.1 Google Maps JavaScript API

Google *Maps JavaScript* API merupakan sebuah layanan yang disediakan oleh Google *Maps* dimana pengguna dapat menambahkan Google *Maps* pada situs *web* mereka. Google *Maps JavaScript* API V3 dirancang untuk dapat berjalan lebih cepat dari V2 dan dapat digunakan pada piranti *mobile*. Sesuai dengan namanya, untuk dapat melakukan akses pada layanan ini, digunakan pemrograman *JavaScript* [4].

Layanan ini bersifat gratis bagi *web* yang tidak berorientasi pada keuntungan. Namun untuk *web* yang berorientasi pada keuntungan diberikan batasan 25.000 kali akses peta dan selebihnya pemilik *web* harus membayar sesuai dengan ketentuan yang diberlakukan *Google Maps*.

Layanan API ini memberikan banyak fungsi untuk melakukan manipulasi pada peta yang akan digunakan seperti menambahkan gambar pada peta dan memberikan tanda pada peta. Selain itu pengguna juga dapat menambahkan peta dengan berbagai macam layanan yang telah disediakan, sehingga dapat memperkaya dan membuat situs *web* yang pengguna buat menjadi lebih baik.

2.2.2 Layanan Web Google Maps

Layanan *web Google Maps* merupakan layanan yang diberikan oleh *Google Maps* sebagai antarmuka yang dapat digunakan oleh pengguna untuk meminta berbagai data mengenai *map API*. Contoh data yang dapat diambil antara lain: koordinat tempat, jarak satu tempat dengan tempat lainnya, petunjuk jalan, dan mencari kemiringan antara dua tempat. Data tersebut nantinya dapat digunakan untuk aplikasi peta yang dimiliki oleh pengguna. Layanan *web Google Maps* dibagi menjadi beberapa layanan *web* sesuai dengan penggunaannya antara lain:

- *Direction API*
- *Distance Matrix API*
- *Elevation API*
- *Geocoding API*

Untuk dapat mengakses layanan *web* ini digunakan permintaan HTML menuju URL tertentu. URL digunakan sebagai parameter untuk meminta jenis layanan dan informasi yang diinginkan. Kemudian *Google Maps* akan mengembalikan data berupa JSON atau XML yang kemudian dapat diolah dan dimanfaatkan oleh pengembang dalam perangkat lunak yang dikembangkan.

2.2.2.1 *Direction API*

Google *Directions API* merupakan layanan untuk menghitung arah antara beberapa lokasi dengan menggunakan permintaan HTTP. Arah menggambarkan asal, tujuan, dan titik perantara yang dinyatakan dalam *string* ataupun dalam koordinat lintang bujur. *Direction API* dapat memberikan nilai kembalian lebih dari satu titik perantara dengan menggunakan rangkaian titik perantara.

Layanan ini secara umum dirancang untuk menghitung jarak tempat statis, dimana alamat tempat yang dicari merupakan bagian dari peta. Layanan ini tidak dirancang untuk merespon secara langsung masukan dari pengguna. Untuk pelayanan secara langsung, Google *Maps* sudah menyediakan Google *Maps JavaScript API V3*.

Banyak jenis informasi yang diberikan oleh *Direction API* ini seperti jalan yang harus ditempuh untuk dapat menuju suatu tempat, waktu tempuh, dan jarak tempuh. Selain itu kita juga dapat memilih cara yang akan digunakan untuk menuju tempat yang dituju. Pilihan yang diberikan oleh *Derection API* adalah dengan berjalan, bersepeda, mengendarai mobil, atau menggunakan perahu, sehingga memudahkan pengguna untuk menentukan pilihan yang tepat [5].

2.2.2.2 *Distance Matrix API*

Google *Distance Matrix API* merupakan layanan yang menyediakan informasi jarak perjalanan dan waktu perjalanan dari kumpulan asal dan tujuan suatu tempat. Nilai yang dikembalikan oleh API ini didasarkan pada rute yang direkomendasikan mulai dari titik awal hingga titik akhir yang telah dihitung oleh Google *Maps API*. Nilai kembalian ini terdiri dari baris yang berisi nilai waktu tempuh dan jarak tempuh.

Berbeda dengan *Direction API*, *Distance Matrix API* tidak memberikan informasi detail rute namun hanya nilai dari jarak tempuh dan waktu tempuh.

Google *Maps* memberi batasan untuk dapat mengakses layanan ini. Kriteria batasan yang diperbolehkan oleh Google *Maps* sebagai berikut:

- a. 100 elemen tiap *query*.
- b. 100 elemen tiap 10 detik.
- c. 2500 elements tiap periode 24 jam.

Selain itu, URL yang dikirimkan pada *Distance Matrix API* terbatas hanya 2048 karakter [6].

2.2.2.3 Elevation API

Google *Elevation API* merupakan layanan yang menyediakan informasi elevasi (ketinggian) untuk semua lokasi di permukaan bumi, termasuk kedalaman dari suatu lokasi di lautan. Untuk membedakan ketinggian dan kedalaman, ketinggian dinyatakan dengan nilai positif dan kedalaman dinyatakan dalam nilai negatif. Ketika Google tidak dapat mengukur ketinggian suatu tempat tertentu, maka layanan akan mengembalikan nilai dengan cara menambahkan dan mengembalikan rata-rata nilai dari empat lokasi terdekat.

Dengan menggunakan *Elevation API*, dapat dikembangkan aplikasi untuk pendakian, aplikasi penyelaman, dan aplikasi untuk bersepeda gunung.

Selain menggunakan layanan *web Elevation API*, Google *Maps* juga menyediakan cara untuk mengakses *Elvation API* secara langsung menggunakan Google *JavaScript API V3*. Informasi ketinggian dapat secara langsung didapat dengan menggunakan objek `ElevationService()` [7].

2.2.2.4 Geocoding API

Geocoding adalah proses mengubah suatu alamat (seperti “Jalan Karang Menjangan, Surabaya, Indonesia”) menjadi informasi geografis (seperti koordinat lintang -7.2739630 dan bujur 112.7481850) yang nantinya dapat digunakan untuk mencari berbagai macam informasi atau memberikan tanda pada peta. Google *Geocoding* API menyediakan layanan untuk mengakses koordinat secara langsung melalui permintaan HTTP. Selain itu, API juga dapat melakukan proses kebalikannya yaitu mengubah data koordinat menjadi data tempat yang disebut *reverse geocoding* [8].

2.2.3 Google Place API

Google *Place* API adalah layanan yang memberikan informasi mengenai sebuah tempat seperti perkantoran, lokasi geografis, atau tempat-tempat yang menarik dengan menggunakan permintaan HTTP. Permintaan yang dikirimkan pada Google *Place* mengenai suatu lokasi harus berupa koordinat lintang bujur.

Layanan pada Google *Place* API dibagi menjadi beberapa jenis sesuai dengan fungsinya antara lain:

- Layanan *Place Searches* berfungsi mengembalikan daftar beberapa tempat yang dekat dengan lokasi yang diminta oleh pengguna.
- Layanan *Place Details* mengembalikan informasi secara detail mengenai tempat tertentu.
- Layanan *Place Check-ins* membuat pengguna dapat melaporkan bahwa pengguna sudah berada pada tempat tersebut. *Check-ins* digunakan untuk memberikan nilai popularitas pada suatu tempat.

- Layanan *Place Reports* membuat pengguna dapat menambahkan tempat baru pada Google *Place API* dan menghapus tempat yang pernah ditambahkan pengguna.

Google *Place API* biasanya digunakan untuk mencari tempat tinggal ideal dengan memanfaatkan layanan *Place Searches*. Selain itu API ini juga sangat populer digunakan untuk mendukung sebuah acara dengan memanfaatkan layanan *Place Report*. Dengan menggunakan layanan ini, acara dapat dipromosikan melalui Google *Maps* [9].

2.3 *Web Crawler*

Web crawling adalah proses mengambil kumpulan halaman dari sebuah *web* untuk dilakukan pengindeksan sehingga mendukung kinerja dari aplikasi rekomendasi pemilihan *real estate* ini. Pada Gambar 2.2 dijelaskan bagaimana aplikasi mengambil data tentang penjualan *real estate* dari sebuah situs *web* penjualan *real estate*. Aplikasi ini akan berjalan menelusuri halaman *web* dan mengumpulkan dokumen-dokumen atau data-data mengenai informasi penjualan *real estate* yang ada di dalamnya. Selanjutnya aplikasi *web crawler* akan membangun sebuah indeks untuk memudahkan proses pencarian. Kemudian aplikasi *web crawler* menganalisis halaman *web* yang telah tersimpan sebelumnya dengan cara mengindeks setiap kemungkinan pola yang terdapat di dalamnya. Data term yang ditemukan disimpan dalam sebuah indeks basis data untuk digunakan dalam pencarian selanjutnya. Kemudian sistem akan mengumpulkan, memilah, dan menyimpan data untuk memberikan kemudahan dalam pengaksesan informasi secara tepat dan akurat. Tujuan dari penyimpanan data berupa indeks adalah untuk kemampuan dan kecepatan dalam menemukan informasi yang baik dan relevan berdasarkan masukan pengguna. Pada Gambar 2.2 ditunjukkan bagaimana arsitektur dari sebuah *web crawler*.

Crawler diawali dengan adanya daftar URL yang akan dikunjungi (*seeds*). Setelah *crawler* mengunjungi URL tersebut, kemudian mengidentifikasi semua *hyperlink* dari halaman itu dan menambahkan kembali ke dalam *seeds* (*crawl frontier*). Setelah *web crawler* mengunjungi halaman-halaman *web* yang ditentukan di dalam *seeds*, *web crawler* membawa data-data yang dicari oleh pengguna kemudian menyimpannya pada basis data. *Web crawler* dimulai dengan sekumpulan URL, kemudian mengunduh setiap halamannya, mendapatkan alamat dari setiap halaman yang dikunjungi, kemudian mengulangi kembali proses *crawling* pada setiap alamat halaman tersebut [10].



Gambar 2.2 Arsitektur Web Crawler

2.4 SQL Server

Microsoft SQL Server adalah sebuah sistem manajemen basis data relasional (RDBMS) produk Microsoft. Bahasa *query* utamanya adalah Transact-SQL yang merupakan implementasi dari SQL standar ANSI/ISO yang digunakan oleh Microsoft dan Sybase. Umumnya SQL Server digunakan di dunia bisnis yang memiliki basis data berskala kecil sampai dengan menengah. Tetapi kemudian berkembang dengan digunakannya SQL Server pada basis data besar.

Microsoft SQL Server dan Sybase/ASE dapat berkomunikasi lewat jaringan dengan menggunakan protokol TDS (*Tabular Data Stream*). Selain itu, Microsoft SQL Server juga mendukung ODBC (*Open Database Connectivity*) dan mempunyai *driver* JDBC untuk bahasa pemrograman Java. Fitur yang lain dari SQL Server ini adalah kemampuannya untuk membuat basis data *mirroring* dan *clustering*.

SQL Server terbagi dalam dua bagian besar, yaitu DDL (*Data Definition Language*) dan DML (*Data Manipulation Language*).

- DDL mendefinisikan struktur basis data seperti: pembuatan basis data, dan pembuatan tabel.
- DML merupakan bagian untuk memanipulasi basis data, seperti pengaksesan data, penghapusan, penambahan, dan pengubahan data.

Sampai saat ini, SQL Server menyediakan dua pelayanan utama yaitu:

- Pelayanan penyimpanan data (*data storage service*)
Merupakan mesin *relational* basis data yang modern memiliki skala dan kehandalan yang tinggi mendukung grup basis data *server*, kapasitas penyimpanan data mampu mendukung ukuran hingga *terabyte*, dan bisa diakses oleh ribuan pengguna dalam waktu yang bersamaan.

Mesin ini mempunyai sistem pengamanan yang tinggi. *Login authentication* dapat diintegrasikan dengan *Windows authentication*. Oleh karena itu, tidak ada kata sandi yang tersimpan dalam *SQL server* dan tidak ada kata sandi yang dikirim antar jaringan sehingga kata sandi ini tidak bisa dibaca oleh *shiffer*.

Mesin ini dilengkapi dengan *setup C2-level* yang akan mengaudit semua pengguna yang sedang mengakses basis data. Mesin ini menggunakan *secure socket layer (SSL) encryption*, yang berfungsi untuk melakukan *encrypt* semua data yang ditransfer dari aplikasi ke basis data.

- Pelayanan analisis data (*analysis data service*)
Layanan analisis Microsoft SQL Server 2000 menyediakan perlengkapan untuk menganalisis *data warehouses* dan *data mart*. Proses analisis seperti membuat rangkuman laporan dapat diambil dari OLTP sistem yang secara periodik membuat laporan dan menyimpan *data warehouse* atau *data mart*.

Adapun fitur-fitur yang telah disediakan di dalam Microsoft SQL Server 2000 antara lain:

- Kemampuan dan ketersediaan
SQL Server 2000 dapat digunakan untuk berbagai macam *platform* Microsoft Windows, mulai dari komputer yang menggunakan sistem operasi Windows 98 sampai dengan *server multiprocessor* menggunakan sistem operasi Windows 2000 Data Center Edition.
- *Enterprise level database features*
Berbeda dengan versi sebelumnya, SQL Server 2000 dapat dihubungkan dengan lebih dari satu basis data. RDBMS ini mampu mengatur ribuan pengguna yang sedang mengakses basis data dalam waktu yang bersamaan.

[Halaman ini sengaja dikosongkan]

BAB III

ANALISIS DAN PERANCANGAN

Bab ini membahas tahap analisis sistem dan perancangan perangkat lunak dari sistem yang dibangun. Pendekatan yang digunakan dalam perancangan perangkat lunak ini adalah pendekatan rancangan berorientasi objek yang direpresentasikan dengan menggunakan *Unified Modelling Language* (yang selanjutnya disebut UML).

3.1 Analisis Sistem

Pada subbab ini akan dilakukan beberapa analisis terhadap sistem yang nantinya akan dibuat.

3.1.1 Deskripsi Umum Perangkat Lunak

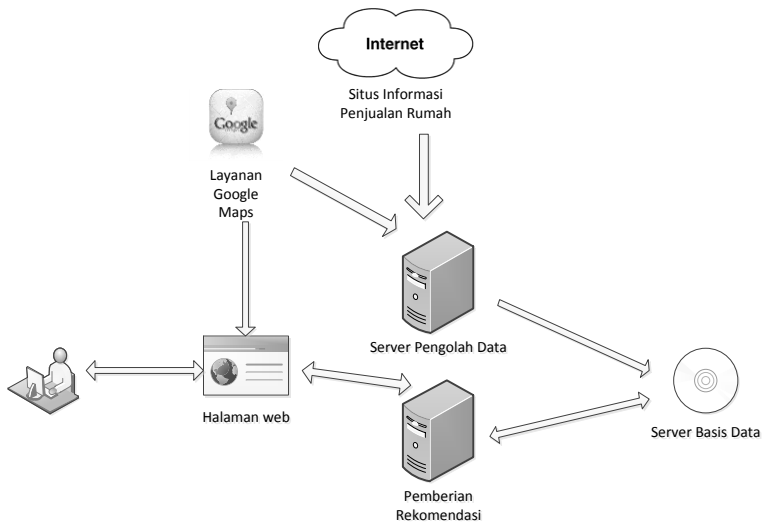
Dalam Tugas Akhir ini dibuat perangkat lunak rekomendasi pemilihan *real estate*. Aplikasi yang dibangun merupakan aplikasi berbasis *web* agar dapat digunakan dan diakses oleh semua orang yang terhubung dengan jaringan internet.

Perangkat lunak ini memberikan informasi dan rekomendasi pada pengguna mengenai *real estate* sesuai dengan kriteria yang diminta oleh pengguna. Dengan perangkat lunak ini pengguna tidak perlu susah-susah menghitung ataupun mengira-ngira tempat yang strategis untuk membeli *real estate*. Untuk mendapat rekomendasi dan informasi, pengguna cukup memberikan masukan berupa kondisi *real estate* sesuai dengan yang diinginkan atau memilih kondisi sesuai dengan kondisi ideal pada umumnya. Dalam penentuan rekomendasi *real estate* yang ideal, dilakukan *query* sesuai dengan masukan pengguna atau nilai yang sudah ditentukan oleh sistem secara umum.

Untuk dapat menyajikan informasi yang baik mengenai *real estate*, maka dalam pembuatan perangkat lunak ini digunakan metode *mashup*. *Mashup* dalam kasus ini merupakan metode yang menggabungkan dua atau lebih sumber informasi baik dari basis data maupun dari situs internet. Dengan menggabungkan banyak informasi, maka akan terbentuk informasi baru yang lebih berguna. Pada perangkat lunak ini akan digabungkan antara informasi yang terdapat pada Google Maps API dan RSS yang menyediakan informasi mengenai *real estate*. Contoh penggunaan metode *mashup* penyedia informasi *real estate* adalah housingmap.com. Namun situs *web* tersebut belum memberikan informasi akses pada fasilitas umum.

Data yang disediakan oleh perangkat lunak ini berasal dari data yang diambil dari situs *web* rumah123.com dan Google Maps API. Data dari situs *web* diambil dengan menggunakan metode *web crawler* dan data tersebut kemudian dikombinasikan dengan data dari layanan *web* Google Maps API. Kemudian data yang telah dikombinasikan tersebut akan disimpan ke dalam basis data untuk nantinya digunakan sebagai data sumber pemberian rekomendasi *real estate* pada pengguna. Perangkat lunak basis data yang digunakan adalah SQL Server 2008. Sedangkan untuk mengembangkan perangkat lunak yang dibuat digunakan ASP.NET 4.0 dengan bahasa yang digunakan C#.

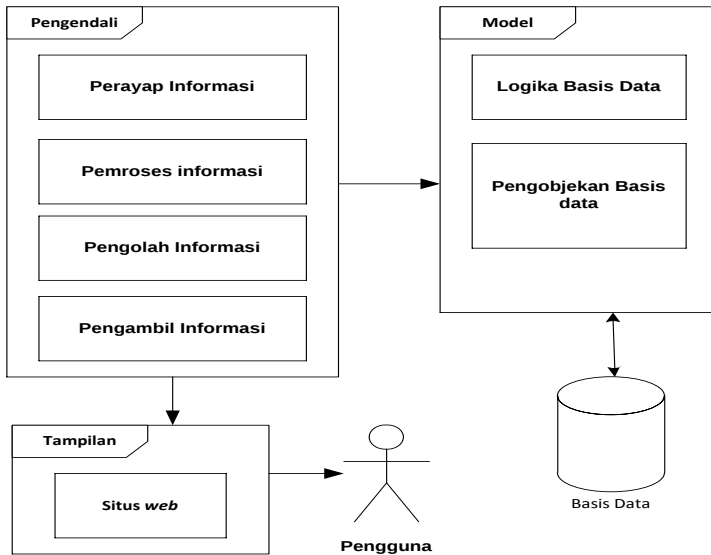
Gambar 3.1 merupakan ilustrasi dari sistem yang akan dibangun. Data-data yang terdapat pada perangkat lunak ini berasal dari situs informasi penjualan rumah dan layanan Google Maps API. Kemudian data-data tersebut diolah dan dimasukkan ke dalam *server* basis data. Setelah itu, informasi tersebut akan diambil dan ditampilkan kepada pengguna dalam bentuk sistem informasi *web*.



Gambar 3.1 Ilustrasi Sistem

3.1.2 Arsitektur Sistem

Pola arsitektur yang digunakan dalam pembangunan perangkat lunak ini adalah MVC (*Model View Controller*) yang merupakan arsitektur tiga lapis yang terdiri dari lapisan model, *view*, dan *controller* [11]. Pada Gambar 3.2 terdapat pola arsitektur yang dibentuk untuk membangun perangkat lunak ini. Pada arsitektur ini, HTML merupakan lapisan terluar yang digunakan oleh pengguna untuk memperoleh informasi dan memasukkan masukan. Semua *controller* atau operasi akan diubah menjadi kelas-kelas yang kemudian akan memproses masukan dari pengguna untuk menyajikan informasi dan mengatur tampilan *view*. Bagian model memberikan informasi untuk melakukan akses pada basis data.



Gambar 3.2 Pola Arsitektur Situs Web

3.1.2.1 Lapisan Model

Lapisan model berinteraksi langsung dengan basis data. Lapisan ini dalam sistem dibagi menjadi dua bagian yaitu pengobjekan basis data dan logika basis data. Pengobjekan basis data berisi kelas-kelas yang merupakan tabel-tabel dalam basis data. Atribut dalam tabel dijadikan atribut dalam kelas. Logika basis data berisi kelas koneksi ke basis data dan kelas-kelas yang memiliki fungsi-fungsi berisi permintaan (*query*) ke basis data.

3.1.2.2 Lapisan Tampilan (View)

Lapisan tampilan dalam sistem berbeda *platform* dengan lapisan model dan *controller*. Lapisan ini ditampilkan pada *browser* yang digunakan oleh pengguna. *Browser* akan mengakses data dalam

bentuk HTML yang berisi informasi hasil olahan dari lapisan pengendali. Lapisan tampilan melakukan penayangan tampilan (*presentation logic*) kepada pengguna dan sebagai antarmuka pengguna.

3.1.2.3 Lapisan Pengendali (*Controller*)

Lapisan pengendali pada sistem dibagi menjadi empat bagian besar, yaitu perayap informasi, pemrosesan informasi, pengelolaan informasi, dan pengolah informasi. Lapisan ini berisi kelas-kelas dimana satu kelas merupakan implementasi dari satu proses. Lapisan pengendali menerima masukan dari logika basis data yang berupa objek basis data. Setelah sistem melakukan proses, keluaran dari lapisan ini juga berupa objek basis data yang akan disimpan ke dalam basis data melalui logika basis data.

3.1.3 Spesifikasi Kebutuhan Perangkat Lunak

Berdasarkan hasil analisis kebutuhan yang telah dilakukan, dirumuskan beberapa fungsi untuk membuat perangkat lunak tersebut. Fungsi-fungsi yang hendak dipenuhi untuk membuat perangkat lunak ini sebagai berikut:

1. Perayap informasi

Perayap informasi mempunyai tugas untuk merayapi data yang tersedia di situs *web* informasi *real estate*. Perayap akan menyimpan URL yang menyimpan informasi *real estate*. Kemudian semua alamat URL tersebut akan dibuka untuk diambil informasinya. Informasi yang dihasilkan berupa dokumen HTML dan gambar *real estate*.

2. Pengurai dokumen

Pengurai dokumen mempunyai tugas untuk menguraikan dokumen dan mengambil isinya. Awalnya dokumen akan diunduh terlebih dahulu dari situs sumber. Dokumen dalam bentuk HTML

tersebut kemudian diambil bagian-bagian yang dianggap penting sebagai sumber informasi. Data tersebut kemudian akan disimpan di dalam basis data. Selain data dalam bentuk HTML, pengurai dokumen juga menguraikan data XML yang didapat dari proses *geocoding*.

3. Proses *geocoding*

Proses *geocoding* ini merupakan proses pencarian informasi mengenai *real estate* yang berhubungan dengan koordinat tempat di peta. Data alamat yang tersimpan dalam basis data akan digunakan sebagai masukan untuk dikirim pada layanan Google *Maps* dalam bentuk alamat URL. Kemudian layanan Google *Maps* akan memberikan keluaran berupa dokumen XML. Informasi koordinat ini digunakan untuk mencari jarak antar tempat atau mencari fasilitas umum terdekat dengan titik tersebut.

4. Pencari fasilitas umum

Proses pencarian fasilitas umum merupakan proses mencari fasilitas umum tertentu yang berada di sekitar *real estate*. Untuk melakukan proses ini digunakan bantuan layanan *web* Google *Place*. Sama seperti proses *geocoding*, pencarian fasilitas umum dilakukan dengan cara mengirimkan URL tertentu menuju ke Google *Place* dengan nilai kembalian berupa XML atau JSON. Data tersebut akan disimpan di dalam basis data yang nantinya berguna untuk memberikan rekomendasi bagi pengguna yang ingin mencari rekomendasi *real estate* yang berdekatan dengan fasilitas umum.

5. Pemberi rekomendasi

Pemberi rekomendasi berfungsi memberikan rekomendasi *real estate* yang sesuai dengan keinginan pengguna. Hasil rekomendasi yang dihasilkan akan ditampilkan pada HTML dalam bentuk tabel. Hasil rekomendasi juga akan ditampilkan dalam bentuk peta dengan memanfaatkan Google *Maps JavaScript API*.

6. Pemberi informasi *real estate*

Pemberi informasi memiliki tugas untuk memberikan informasi lengkap kepada pengguna mengenai *real estate* yang dipilih oleh pengguna. Informasi akan ditampilkan pada pengguna adalah daftar rekomendasi *real estate* dalam bentuk halaman HTML. Informasi yang didapat bersumberkan pada basis data, layanan *web Google Maps API*, dan situs *web* sumber. Selain itu juga akan ditampilkan informasi berupa peta dengan memanfaatkan *Google Maps JavaScript API*.

7. Pemberi informasi fasilitas umum

Pemberi informasi fasilitas umum ini memiliki tugas untuk memberikan informasi lengkap mengenai daftar fasilitas umum yang terdapat di sekitar titik tertentu atau di sekitar *real estate* tertentu. Informasi akan ditampilkan oleh sistem dalam bentuk halaman daftar fasilitas umum dan peta lokasi fasilitas umum pada halaman HTML dengan memanfaatkan layanan *Google Maps JavaScript API*.

3.1.4 Analisis Aktor

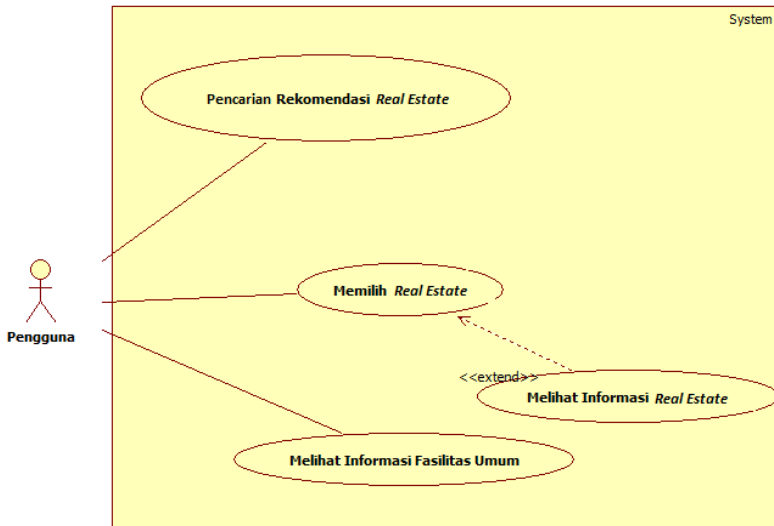
Berdasarkan spesifikasi kebutuhan perangkat lunak di atas, sistem ini hanya mempunyai satu jenis aktor yaitu pengguna. Pengguna yang dimaksud adalah pelaku yang mengakses situs ini dan melakukan pencarian untuk mendapatkan rekomendasi *real estate*. Pengguna dapat memanfaatkan semua fitur dan mengakses semua fasilitas yang terdapat pada situs ini. Untuk penjelasan mengenai aktor yang lebih detail, akan dijelaskan pada Tabel 3.1.

Tabel 3.1 Tabel Aktor Aplikasi

No.	Aktor	Deskripsi
1	Pengguna	Pengguna adalah aktor utama dan satu-satunya pada sistem rekomendasi <i>real estate</i> ini. Aktor ini adalah pengguna yang ingin mencari dan mendapat rekomendasi <i>real estate</i> melalui situs <i>web</i> ini. Akses yang dapat dilakukan oleh aktor ini adalah sebagai berikut: 1. Mencari <i>real estate</i> berdasarkan karakteristiknya. 2. Mencari <i>real estate</i> berdasarkan kondisi lingkungan dan karakteristiknya. 3. Melihat informasi mengenai <i>real estate</i> yang telah direkomendasikan. 4. Melihat lokasi <i>real estate</i> pada <i>Google Maps</i>.

3.1.5 Skenario Kasus Penggunaan

Terdapat empat kasus penggunaan berada pada sistem rekomendasi *real estate*. Secara garis besar, skenario kasus penggunaan sistem ini dapat direpresentasikan oleh diagram kasus penggunaan sebagaimana tampak pada Gambar 3.3. Sedangkan uraian dari kasus penggunaan Gambar 3.3 ditunjukkan pada Tabel 3.2. Tabel 3.2 secara lengkap menunjukkan daftar kasus penggunaan pada sistem. Tiap kasus penggunaan memiliki kode yang ditulis dengan format UC-xxx dimana xxx merupakan urutan kasus penggunaan. Pada tabel juga diberi keterangan mengenai masing-masing kasus penggunaan.



Gambar 3.3 Diagram Kasus Penggunaan

Tabel 3.2 Deskripsi Kasus Penggunaan

No	Kode Kasus Penggunaan	Nama Kasus Penggunaan	Keterangan
1	UC-001	Kasus penggunaan pencarian rekomendasi <i>real estate</i> .	Pengguna memperoleh rekomendasi mengenai <i>real estate</i> yang cocok untuk pengguna dari sistem setelah memasukkan kriteria <i>real estate</i> pada sistem.
2	UC-002	Kasus penggunaan melihat informasi <i>real estate</i> .	Pengguna dapat melihat gambar dan informasi mengenai <i>real estate</i> secara lengkap dengan cara memilih <i>real estate</i>

No	Kode Kasus Penggunaan	Nama Kasus Penggunaan	Keterangan
			yang telah direkomendasikan.
3	UC-003	Kasus penggunaan memilih <i>real estate</i> .	Pengguna dapat memilih <i>real estate</i> yang ingin dilihat informasinya dan dapat menuju situs <i>web</i> sumber.
4	UC-004	Kasus penggunaan melihat informasi fasilitas umum.	Pengguna dapat mengetahui informasi dan lokasi fasilitas umum yang terdapat di sekitar <i>real estate</i> yang dipilih.

Selain spesifikasi kasus penggunaan dan tabel kasus penggunaan, disertakan pula diagram aktivitas yang merupakan ilustrasi alur normal dari kasus penggunaan yang bersangkutan dan juga disediakan diagram urutan yang menunjukkan waktu hidup dari suatu objek selama kasus penggunaan yang bersangkutan berjalan.

3.1.6 Kelas Analisis

Dalam bagian ini akan ditentukan kelas yang akan digunakan dalam pembuatan perangkat lunak. Pemilihan kelas yang digunakan dalam sistem dilakukan dengan melakukan pendataan pada seluruh kata benda dan kata sifat di dalam rincian spesifikasi kebutuhan sistem.

3.1.6.1 Kelas Analisis Proses Merayapi Informasi

Proses perayapan ini digunakan untuk mengambil berbagai informasi *real estate* yang terdapat pada situs *web* sumber. Perayap akan mencari informasi dalam situs *web* sumber dengan

cara mencari alamat-alamat URL yang dianggap penting kemudian dimasukkan ke dalam daftar alamat. Dari alamat-alamat URL tersebut akan diunduh data dalam bentuk data HTML dan data gambar dengan pengunduh *file*. Berdasarkan analisis tersebut, kandidat kelas dapat dilihat pada Tabel 3.3.

Tabel 3.3 Kandidat Kelas untuk Proses Merayapi Informasi

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas?	Alasan
1	Perayapan	Proses merayapi situs <i>web</i> informasi <i>real estate</i> .	Ya	Proses yang merayapi situs <i>web</i> informasi <i>real estate</i> untuk mencari informasi dan URL terkait.
2	Informasi <i>real estate</i>	Informasi yang dicari dari situs sumber.	Bukan	Hasil yang diharapkan dari proses pengunduhan.
3	Situs <i>web</i> sumber	Tempat perayapan dilakukan.	Bukan	<i>Web</i> yang merupakan sistem luar yang menjadi masukan untuk perayapan.
4	Alamat URL	Alamat yang harus dirayapi atau diunduh.	Bukan	Atribut dalam kelas perayapan yang menyimpan alamat dokumen.
5	Daftar Alamat	Daftar alamat situs yang berisi alamat untuk	Bukan	Merupakan atribut yang terdapat pada kelas perayapan.

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas?	Alasan
		diunduh atau dirayapi.		
6	Data HTML	Hasil dari pengunduhan .	Bukan	Merupakan sebuah atribut yang disimpan dalam sistem.
7	Data gambar	Hasil dari pengunduhan yang berbentuk gambar.	Bukan	Merupakan sebuah atribut yang disimpan dalam sistem.
8	Pengunduh data	Tempat proses mengunduh data HTML dan gambar.	Ya	Proses untuk mengunduh data HTML untuk diambil alamatnya ataupun mengunduh gambar.

Dari hasil analisis kandidat kelas pada Tabel 3.3, penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas dibuat. Penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas ditunjukkan pada Tabel 3.4.

Tabel 3.4 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Proses Merayapi Informasi

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
Perayapan (<i>Crawler</i>)	Merepresentasikan proses perayapan situs <i>web</i> informasi <i>real estate</i> .	Kelas menerima masukan berupa URL <i>web</i> informasi <i>real estate</i> , kemudian menjelajahi <i>web</i> dan alamat URL yang berkaitan.	 <i>Control</i>
Pengunduh data (<i>Downloader</i>)	Merepresentasikan proses mengunduh data dari situs <i>web</i> sumber.	Kelas menerima alamat URL untuk diunduh datanya.	 <i>Control</i>

3.1.6.2 Kelas Analisis Pengurai Dokumen

Penguraian dimulai ketika *server* sudah mendapatkan daftar alamat dokumen. Proses penguraian data bertugas untuk menguraikan isi dokumen yang telah diunduh oleh pengunduh dari situs sumber menjadi informasi yang bermanfaat. Pengurai dokumen akan mengolah data dalam bentuk HTML dan XML untuk disimpan dalam beberapa objek *real estate*. Untuk mengurai data dalam HTML digunakan pustaka *RegularExpression* dan untuk menguraikan data XML digunakan pustaka *XMLParser*. Objek tersebut kemudian akan disimpan ke dalam basis data melalui logika dokumen. Berdasarkan penjelasan

di atas, didapatkan kandidat kelas yang ditunjukkan pada Tabel 3.5.

Tabel 3.5 Kandidat Kelas untuk Proses Pengurai Dokumen

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
1	Penguraian	Proses yang menguraikan isi dokumen yang telah diunduh.	Ya	Proses ini melakukan penguraian dokumen yang telah diunduh untuk mengambil isi dokumen.
2	<i>Server</i>	Tempat penguraian dilakukan.	Bukan	Merupakan tempat jalannya proses.
3	Daftar alamat dokumen	Daftar yang berisi tempat penyimpanan dokumen.	Bukan	Atribut yang merupakan masukan dalam penguraian dokumen.
4	Objek <i>real estate</i>	Objek <i>real estate</i> berisi informasi mengenai <i>real estate</i> .	Ya	<i>Real estate</i> memiliki atribut yaitu IdRealEstate, IdArea, Alamat, Harga, Koordinat, Fasilitas, Gambar, JumlahKamar, LuasTanah, dan

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
				LuasBangunan.
5	Alamat dokumen	Tempat penyimpanan dokumen.	Bukan	Atribut yang berada dalam daftar alamat dokumen yang merupakan tempat penyimpanan dokumen.
6	Isi dokumen	Teks yang berada pada badan dokumen.	Bukan	Atribut yang merupakan isi dokumen yang akan diuraikan dan diambil informasinya.
7	Situs sumber	Tempat dokumen diunduh.	Bukan	Merupakan sistem luar yang menjadi masukan untuk penguraian.
8	HTML	Merupakan data HTML yang berisi informasi <i>real estate</i> yang diuraikan.	Bukan	Atribut berisi data HTML yang akan diuraikan.
9	XML	Merupakan data XML yang berisi informasi <i>real estate</i> yang akan	Bukan	Atribut berisi data XML yang akan diuraikan.

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
		diuraikan.		
10	Pustaka <i>RegularExpression</i>	Membantu untuk menguraikan data HTML dan XML.	Bukan	Fungsi dalam pustaka <i>RegularExpression</i> membantu menguraikan data HTML.
11	Pustaka <i>XMLParser</i>	Membantu untuk menguraikan data HTML.	Bukan	Fungsi dalam pustaka <i>XMLParser</i> membantu untuk menguraikan data XML yang berasal dari layanan Google Maps API.
12	Basis data	Tempat untuk menyimpan informasi <i>real estate</i> .	Bukan	Merupakan sistem luar yang berfungsi menyimpan informasi.
13	Logika dokumen	Pembantu dokumen dalam pengaksesan ke basis data.	Ya	Digunakan sebagai tempat kode pengaksesan dokumen ke basis data.
14	Pengunduh	Melakukan pengunduhan data untuk diambil informasinya.	Ya	Merupakan bagian dari sistem untuk mengunduh data.

Dari hasil analisis kandidat kelas pada Tabel 3.5, penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas dibuat. Penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas ditunjukkan pada Tabel 3.6.

Tabel 3.6 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Proses Penguraian Dokumen

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
Penguraian (<i>Parser</i>)	Merepresentasikan proses penguraian data yang telah diunduh dari situs sumber.	Kelas menerima masukan berupa URL <i>web</i> informasi <i>real estate</i> , kemudian menjelajahi <i>web</i> dan alamat URL berkaitan.	 <i>Control</i>
Pengunduh data (<i>Downloader</i>)	Merepresentasikan proses mengunduh data dari situs <i>web</i> sumber.	Kelas menerima alamat URL untuk diunduh datanya.	 <i>Control</i>
Objek <i>real estate</i> (<i>Realestate</i>)	Merepresentasikan objek yang digunakan untuk menyimpan data dan atribut <i>real estate</i> .	Kelas memodelkan <i>real estate</i> sebagai objek <i>real estate</i> dengan atribut yaitu: <i>idRealEstate</i> , <i>idArea</i> , Alamat, Harga,	 <i>Model</i>

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
		Koordinat, Fasilitas, Gambar, JumlahKamar, LuasTanah, dan LuasBangunan.	
Logika <i>real estate</i> (<i>DataBase</i>)	Merepresentasikan proses penyimpanan objek <i>real estate</i> dalam basis data.	Kelas menerima dan menghubungkan program dengan basis data untuk dapat menyimpan objek ke dalam basis data.	 <i>Control</i>

3.1.6.3 Kelas Analisis Proses *Geocoding*

Proses *geocoding* pada dasarnya merupakan proses untuk mengubah alamat suatu tempat menjadi sebuah koordinat lintang bujur. Selain untuk mendapat informasi koordinat lintang bujur, proses ini berguna untuk mendapatkan informasi mengenai lokasi fasilitas umum, jarak antara dua tempat, dan akses sebuah *real estate* dengan fasilitas umum. Semua informasi yang berhubungan dengan fasilitas umum akan disimpan dalam objek fasilitas umum.

Proses *geocoding* dimulai ketika sistem mendapatkan informasi mengenai alamat *real estate*. Informasi alamat itu akan dikirimkan menuju layanan Google *Geocoding* untuk mendapatkan koordinat alamat *real estate* yang dimaksud. Setelah koordinat didapat, koordinat tersebut digunakan untuk

mendapatkan informasi mengenai fasilitas umum yang terletak di sekitar *real estate* dengan menggunakan layanan Google *Place*. Koordinat *real estate* juga digunakan untuk mencari jarak antara *real estate* dengan fasilitas umum dengan menggunakan layanan Google *Matrix Distance*. Data hasil keluaran dari proses ini akan disimpan ke dalam basis data melalui logika *real estate*. Dari penjelasan di atas didapatkan kandidat kelas yang dijelaskan pada Tabel 3.7.

Tabel 3.7 Kandidat Kelas untuk Proses *Geocoding*

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
1	<i>Geocoding</i>	Proses yang mencari informasi tentang peta <i>real estate</i> .	Ya	Proses ini melakukan pencarian informasi <i>real estate</i> dan fasilitas umum pada peta menggunakan fasilitas layanan Google <i>Place API</i> .
2	Alamat tempat	Merupakan alamat <i>real estate</i> berada.	Bukan	Atribut yang merupakan masukan untuk melakukan proses <i>Geocoding</i> .
3	Koordinat lintang bujur	Merupakan koordinat lintang bujur suatu lokasi.	Bukan	Atribut dalam kelas <i>geocoding</i>

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
				yang disimpan dan sebagai masukan untuk mencari jarak dua tempat dan fasilitas umum.
4	Lokasi fasilitas umum	Informasi mengenai fasilitas umum yang berada di sekitar <i>real estate</i> .	Bukan	Atribut dalam kelas <i>geocoding</i> yang merupakan hasil keluaran proses <i>geocoding</i> dengan memanfaatkan layanan <i>Google Place</i> .
5	Jarak dua tempat	Jarak antara <i>real estate</i> dengan tempat tertentu.	Bukan	Atribut dalam kelas <i>geocoding</i> yang merupakan hasil keluaran proses <i>geocoding</i> dengan memanfaatkan layanan <i>Google Matrix Distance</i> .

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
6	Akses <i>real estate</i> ke fasilitas umum	Akses dan lama waktu yang digunakan untuk menuju fasilitas umum tertentu.	Bukan	Atribut yang merupakan hasil keluaran dari layanan Google <i>Place</i> dan layanan Google <i>Direction</i> .
7	Objek fasilitas umum	Objek fasilitas umum berisi informasi mengenai fasilitas umum.	Ya	Fasilitas umum memiliki atribut berupa nama, alamat, koordinat, dan jenis.
8	Layanan Google <i>Geocoding</i>	Layanan yang digunakan untuk membantu melakukan proses <i>geocoding</i> .	Bukan	Fasilitas dari luar sistem yang digunakan untuk mengubah alamat menjadi koordinat lintang bujur.
9	Layanan Google <i>Place</i>	Layanan dari Google untuk menemukan tempat tertentu.	Bukan	Fasilitas dari luar sistem yang digunakan untuk menemukan fasilitas umum di sekitar <i>real</i>

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
				<i>estate.</i>
10	Layanan Google <i>Matrix Distance</i>	Layanan dari Google untuk mengetahui jarak antara dua tempat berbeda.	Bukan	Fasilitas dari luar sistem yang digunakan untuk menemukan jarak dan waktu yang ditempuh untuk menuju fasilitas umum.
11	Logika <i>real estate</i>	Pembantu <i>real estate</i> dalam pengaksesan ke basis data.	Ya	Digunakan sebagai tempat kode pengaksesan <i>real estate</i> ke basis data.
12	Basis data	Tempat untuk menyimpan informasi <i>real estate.</i>	Bukan	Merupakan sistem luar yang berfungsi menyimpan informasi.

Dari hasil analisis kandidat kelas pada Tabel 3.7, penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas dibuat. Penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas ditunjukkan pada Tabel 3.8.

Tabel 3.8 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Proses *Geocoding*

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
<i>Geocoding</i> (<i>Geocoding</i>)	Merepresentasikan proses pencarian informasi <i>real estate</i> dari layanan <i>Google maps</i> .	Kelas menerima masukan berupa alamat <i>real estate</i> , kemudian mengubahnya menjadi koordinat lintang bujur dan informasi mengenai fasilitas umum.	 <i>Control</i>
Objek <i>real estate</i> (<i>RealEstate</i>)	Merepresentasikan objek yang digunakan untuk menyimpan data dan atribut <i>real estate</i> .	Kelas memodelkan <i>real estate</i> sebagai objek <i>real estate</i> dengan atribut yaitu <i>idRealEstate</i> , <i>idArea</i> , Alamat, Harga, Koordinat, Fasilitas, Gambar, JumlahKamar, LuasTanah, dan LuasBangunan.	 <i>Model</i>
Logika <i>real estate</i> (<i>LogikaPrope</i>)	Merepresentasikan proses penyimpanan	Kelas menerima dan menghubungkan	 <i>Control</i>

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
<i>rti</i>)	objek <i>real estate</i> dalam basis data.	program dengan basis data untuk dapat menyimpan objek ke dalam basis data.	

3.1.6.4 Kelas Analisis Pemberi Rekomendasi

Pemberi rekomendasi *real estate* merupakan proses utama untuk mencari dan menentukan *real estate* yang harus direkomendasikan pada pengguna. Proses pemberian rekomendasi dimulai ketika sistem menerima masukan dari pengguna berupa kriteria *real estate* yang diinginkan. Kriteria tersebut dimasukkan oleh pengguna melalui halaman utama sistem informasi. Dari masukan kriteria tersebut sistem akan mencari *real estate* dalam basis data *real estate* yang sesuai dengan kriteria yang diinginkan pengguna melalui logika rekomendasi kemudian dijadikan sebuah objek rekomendasi *real estate*. Hasil keluaran tersebut akan disimpan dalam daftar *real estate*. Kemudian daftar *real estate* tersebut ditampilkan pada halaman HTML rekomendasi sebagai informasi rekomendasi *real estate* kepada pengguna. Selain itu, daftar *real estate* tersebut akan ditampilkan pada halaman HTML rekomendasi dalam bentuk peta dengan bantuan layanan Google Maps JavaScript API. Dari penjelasan tersebut didapatkan kandidat kelas yang dijelaskan pada Tabel 3.9.

Tabel 3.9 Kandidat Kelas untuk Pemberi Rekomendasi

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
1	Pemberi rekomendasi	Proses yang memberikan rekomendasi pada pengguna <i>real estate</i> yang sesuai dengan kriteria pengguna.	Ya	Proses ini melakukan proses pencarian <i>real estate</i> dan menyimpannya sebagai daftar <i>real estate</i> untuk ditampilkan pada pengguna.
2	Pengguna	Merupakan subjek yang menggunakan sistem informasi ini.	Bukan	Pengguna bukan merupakan bagian dari sistem informasi pembuatan perangkat lunak ini.
3	Kriteria <i>real estate</i>	Informasi yang dimasukkan oleh pengguna sebagai ciri-ciri <i>real estate</i> yang diinginkan.	Bukan	Atribut dalam kelas pemberi rekomendasi untuk melakukan pencarian dan pemilihan <i>real estate</i> .

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
4	Halaman utama	Tempat pertama yang berhubungan dengan pengguna ketika pengguna membuka situs pencarian rekomendasi <i>real estate</i> ini.	Ya	Halaman ini berfungsi menerima masukan dari pengguna selain itu juga menampilkan informasi <i>real estate</i> secara umum sebelum dilakukan pemberian rekomendasi.
5	Sistem	Sebuah lingkungan dimana sebuah proses berjalan.	Bukan	Merupakan tempat dimana kelas-kelas berada dan semua proses bekerja.
6	Logika rekomendasi	Pembantu sistem dalam pengaksesan ke basis data dalam mencari rekomendasi <i>real estate</i> .	Ya	Digunakan sebagai tempat kode pengaksesan data untuk mencari data rekomendasi <i>real estate</i> dari basis data.

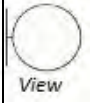
No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
7	Objek rekomendasi <i>real estate</i>	Objek <i>real estate</i> berisi informasi mengenai <i>real estate</i> yang direkomendasikan.	Ya	<i>Real estate</i> memiliki atribut yaitu idRealEstate, idArea, Alamat, Harga, Koordinat, Fasilitas, Gambar, JumlahKamar, LuasTanah, dan LuasBangunan.
8	Basis data <i>real estate</i>	Tempat menyimpan informasi <i>real estate</i> .	Bukan	Merupakan sistem luar yang berfungsi menyimpan informasi.
9	Daftar <i>real estate</i>	Kumpulan <i>real estate</i> yang sesuai dengan kriteria yang diminta oleh pengguna.	Bukan	Atribut yang disimpan dalam kelas pemberi rekomendasi sebagai keluaran proses pemberian rekomendasi.
10	Halaman HTML rekomendasi	Tempat sistem menampilkan hasil rekomendasi pada pengguna.	Ya	Halaman ini melakukan beberapa fungsi untuk dapat menampilkan daftar <i>real</i>

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
				<i>estate</i> pada pengguna.
11	Google Maps JavaScript API	Merupakan API yang bertugas membantu menampilkan informasi <i>real estate</i> dalam peta.	Bukan	Merupakan fungsi di luar sistem.

Dari hasil analisis kandidat kelas pada Tabel 3.9, penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas dibuat. Penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas ditunjukkan pada Tabel 3.10.

Tabel 3.10 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Pemberi Rekomendasi

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
Pemberi rekomendasi (<i>Recomendation</i>)	Merepresentasikan proses pemberian rekomendasi <i>real estate</i> sesuai dengan kriteria yang dimasukkan pengguna.	Kelas menerima masukan dari pengguna berupa kriteria <i>real estate</i> , kemudian melalui logika rekomendasi akan dicari daftar <i>real estate</i> yang	 <i>Control</i>

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
		direkomendasikan.	
Halaman utama (<i>Home</i>)	Merepresentasikan sebuah kelas sebagai antarmuka sebagai halaman awal yang dijumpai oleh pengguna.	Memodelkan sebuah antarmuka yang menampilkan pilihan kriteria dan <i>real estate</i> secara umum.	
Logika rekomendasi (<i>recomSearch</i>)	Merepresentasikan kelas yang digunakan untuk mengambil data <i>real estate</i> berdasarkan proses rekomendasi yang telah dilakukan.	Kelas menerima dan menghubungkan program dengan basis data untuk mengambil data <i>real estate</i> sesuai dengan hasil proses rekomendasi.	
Objek rekomendasi <i>real estate</i> (<i>RecomRealEstate</i>)	Merepresentasikan objek yang digunakan untuk menyimpan data dan atribut <i>real estate</i> hasil.	Kelas memodelkan <i>real estate</i> sebagai objek <i>real estate</i> dengan atribut yaitu <i>idRealEstate</i> , <i>idArea</i> ,	

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
		Alamat, Harga, Koordinat, Fasilitas, Gambar, JumlahKamar, LuasTanah, dan LuasBanguna n.	
Halaman HTML rekomendasi	Merepresentasikan sebuah kelas sebagai antarmuka yang menampilkan pada pengguna <i>real estate</i> hasil dari proses rekomendasi.	Memodelkan sebuah antarmuka yang menerima masukan informasi mengenai <i>real estate</i> hasil rekomendasi kemudian menampilkan pada pengguna.	

3.1.6.5 Analisis Kelas Pemberi Informasi *Real Estate*

Pemberi informasi memiliki tugas untuk memberikan detail informasi mengenai *real estate* yang dipilih oleh pengguna. Proses dimulai ketika pengguna memilih salah satu *real estate* dari daftar *real estate* yang terdapat di halaman rekomendasi atau di halaman utama. Melalui logika *real estate*, kemudian diambil informasi detail *real estate* dari basis data dan dimasukkan ke

dalam objek *real estate*. Kemudian hasil keluaran dari proses pengambilan informasi detail *real estate* ditampilkan pada halaman HTML detail *real estate*. Kemudian dengan menggunakan bantuan *Google Maps JavaScript API* akan ditampilkan juga informasi dalam bentuk peta.

Dari penjelasan di atas didapatkan kandidat kelas yang dijelaskan pada Tabel 3.11. Kandidat kelas tersebut yang akan dianalisis untuk dijadikan sebuah kelas.

Tabel 3.11 Kandidat Kelas untuk Pemberi Informasi *Real Estate*

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
1	Pemberi informasi	Proses yang memberikan informasi mengenai <i>real estate</i> secara detail.	Ya	Proses ini mengambil data mengenai sebuah <i>real estate</i> dan informasi yang berhubungan dengan <i>real estate</i> tersebut.
2	Pengguna	Merupakan subjek yang menggunakan sistem informasi ini.	Bukan	Pengguna bukan merupakan bagian dari sistem informasi pembuatan perangkat

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
				lunak ini.
3	<i>Real estate</i>	Merupakan <i>real estate</i> yang ditampilkan dalam halaman rekomendasi rumah.	Bukan	Atribut dalam kelas pemberi informasi untuk melakukan pencarian informasi suatu <i>real estate</i> dalam basis data.
4	Halaman rekomendasi	Halaman yang menampilkan daftar <i>real estate</i> hasil rekomendasi pada pengguna.	Ya	Halaman ini berfungsi menerima masukan <i>real estate</i> yang dipilih dari daftar rekomendasi <i>real estate</i> oleh pengguna untuk proses pemberian informasi <i>real estate</i> .
5	Halaman utama	Tempat pertama yang berhubungan dengan pengguna ketika pengguna	Ya	Halaman ini berfungsi menerima masukan <i>real estate</i> yang dipilih dari daftar

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
		membuka situs pencarian rekomendasi <i>real estate</i> ini.		rekomendasi <i>real estate</i> oleh pengguna untuk proses pemberian informasi <i>real estate</i> .
6	Logika informasi <i>real estate</i>	Pembantu sistem dalam pengaksesan ke basis data dalam mencari informasi <i>real estate</i> .	Ya	Digunakan sebagai tempat kode pengaksesan data untuk mencari data informasi <i>real estate</i> dari basis data.
7	Informasi detail <i>real estate</i>	Merupakan informasi tentang detail <i>real estate</i> yang disediakan untuk pengguna.	Bukan	Merupakan informasi yang dicari dari basis data dan yang ditampilkan pada halaman detail informasi.
8	Objek <i>real estate</i>	Objek <i>real estate</i> berisi informasi mengenai <i>real estate</i> .	Ya	<i>Real estate</i> memiliki atribut yaitu idRealEstate, idArea, Alamat, Harga,

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
				Koordinat, Fasilitas, Gambar, JumlahKamar, LuasTanah, dan LuasBanguna n.
9	Basis data <i>real estate</i>	Tempat menyimpan informasi <i>real estate</i> .	Bukan	Merupakan sistem luar yang berfungsi menyimpan informasi.
10	Halaman HTML detail <i>real estate</i>	Tempat sistem menampilkan informasi detail <i>real estate</i> .	Ya	Halaman ini melakukan beberapa fungsi untuk dapat menampilkan informasi detail mengenai <i>real estate</i> .
11	Google Maps JavaScript API	Merupakan API yang bertugas membantu menampilkan informasi <i>real estate</i> dalam peta.	Bukan	Merupakan fungsi di luar sistem.

Dari hasil analisis kandidat kelas pada Tabel 3.11, penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas dibuat. Penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas ditunjukkan pada Tabel 3.12.

Tabel 3.12 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Pemberi Informasi *Real Estate*

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
Pemberi informasi (<i>GetInformation</i>)	Merepresentasikan proses pemberian informasi detail <i>real estate</i> sesuai dengan masukan <i>real estate</i> yang ingin dicari pengguna.	Kelas menerima masukan berupa <i>real estate</i> yang terdapat pada daftar rekomendasi, kemudian mengambil informasi detail pada basis data melalui logika informasi <i>real estate</i> .	 Control
Halaman utama (<i>Home</i>)	Merepresentasikan sebuah kelas antarmuka sebagai halaman awal yang dijumpai oleh pengguna dan menampilkan <i>real estate</i> terbaru.	Memodelkan sebuah antarmuka yang menampilkan daftar <i>real estate</i> terbaru untuk dipilih oleh pengguna.	 View

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
Halaman rekomendasi (<i>recommendation</i>)	Merepresentasikan sebuah kelas sebagai antarmuka yang menampilkan pada pengguna <i>real estate</i> hasil dari proses rekomendasi.	Memodelkan sebuah antarmuka yang menampilkan hasil rekomendasi <i>real estate</i> untuk dipilih oleh pengguna.	 View
Logika informasi <i>real estate</i> (<i>RealestateLogic</i>)	Merepresentasikan kelas yang digunakan untuk mengambil data detail <i>real estate</i> berdasarkan masukan pengguna.	Kelas menghubungkan sistem dengan basis data untuk mengambil informasi detail <i>real estate</i> .	 Control
Objek <i>Real estate</i> (<i>Realestate</i>)	Merepresentasikan objek yang digunakan untuk menyimpan data dan atribut <i>real estate</i> .	Memodelkan <i>real estate</i> sebagai objek <i>real estate</i> dengan atribut yaitu idRealEstate, idArea, Alamat, Harga, Koordinat, fasilitas, Gambar, JumlahKamar, LuasTanah, dan	 Model

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
		LuasBangunan.	
Halaman HTML detail <i>real estate</i>	Merepresentasikan sebuah kelas sebagai antarmuka yang menampilkan pada pengguna informasi detail <i>real estate</i> .	Memodelkan antarmuka yang menerima masukan berupa informasi mengenai detail <i>real estate</i> kemudian menampilkann ya pada pengguna.	

3.1.6.6 Analisis Kelas Pemberi Informasi Fasilitas Umum

Pemberi informasi fasilitas umum ini memiliki tugas untuk memberikan informasi lengkap mengenai fasilitas umum yang terdapat di sekitar titik tertentu. Proses ini akan dimulai ketika pengguna ingin melihat informasi tentang fasilitas umum yang berada di sekitar *real estate* yang terdapat pada halaman HTML detail *real estate*. Kemudian sistem akan mencari fasilitas umum yang berkaitan dalam basis data dengan bantuan logika fasilitas umum. Hasil dari pencarian akan ditampung sebagai objek fasilitas umum dan disimpan dalam daftar fasilitas umum. Daftar fasilitas umum tersebut akan ditampilkan dalam halaman HTML informasi fasilitas umum. Selain itu informasi fasilitas umum ditampilkan dalam peta dengan menggunakan *Google Maps JavaScript API*.

Dari penjelasan di atas didapatkan kandidat kelas yang dijelaskan pada Tabel 3.13. Kandidat kelas tersebut yang akan dianalisis untuk dijadikan sebuah kelas.

Tabel 3.13 Kandidat Kelas untuk Pemberi Informasi Fasilitas Umum

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
1	Pemberi informasi fasilitas umum	Proses yang memberikan informasi mengenai informasi fasilitas umum.	Ya	Proses ini mencari daftar data mengenai fasilitas umum yang berada di sekitar titik atau <i>real estate</i> tertentu.
2	Fasilitas umum	Merupakan informasi yang dicari dalam proses pemberi informasi <i>real estate</i> .	Bukan	Informasi yang dicari dalam proses ini dan bukan merupakan objek yang digunakan untuk menyimpan atribut fasilitas umum.
3	<i>Real estate</i>	Merupakan <i>real estate</i> yang ditampilkan dalam halaman rekomendasi rumah.	Bukan	Atribut dalam kelas pemberi informasi untuk melakukan pencarian informasi suatu <i>real estate</i> dalam basis data.
4	Halaman	Tempat sistem	Ya	Halaman ini

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
	HTML detail <i>real estate</i>	menampilkan informasi detail <i>real estate</i> .		melakukan beberapa fungsi untuk dapat menampilkan informasi detail mengenai <i>real estate</i> .
5	Basis data <i>real estate</i>	Tempat menyimpan informasi <i>real estate</i> .	Bukan	Merupakan sistem luar yang berfungsi menyimpan informasi.
6	Logika fasilitas umum	Pembantu sistem dalam pengaksesan ke basis data dalam mencari informasi fasilitas umum.	Ya	Digunakan sebagai tempat kode pengaksesan data untuk mencari data informasi <i>real estate</i> dari basis data.
7	Objek fasilitas umum	Objek fasilitas umum berisi informasi mengenai fasilitas umum.	Ya	Fasilitas umum memiliki atribut berupa nama, alamat, koordinat, dan jenis.
8	Daftar fasilitas umum	Kumpulan fasilitas umum yang berada di	Bukan	Atribut yang disimpan dalam kelas

No.	Kandidat	Pengertian dalam Logika Bisnis	Kelas ?	Alasan
		sekitar <i>real estate</i> atau titik tertentu.		pemberi informasi fasilitas umum sebagai keluaran proses pemberi informasi fasilitas umum.
9	Halaman HTML informasi fasilitas umum	Tempat sistem menampilkan informasi mengenai fasilitas umum.	Ya	Halaman ini melakukan beberapa fungsi untuk dapat menampilkan informasi mengenai fasilitas umum.
10	Google Maps JavaScript API	Merupakan API yang bertugas membantu menampilkan informasi <i>real estate</i> dalam peta.	Bukan	Merupakan fungsi di luar sistem.

Dari hasil analisis kandidat kelas pada Tabel 3.13, penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas dibuat. Penjelasan rinci tentang deskripsi dan tanggung jawab setiap kandidat kelas ditunjukkan pada Tabel 3.14.

Tabel 3.14 Penjelasan Rinci Deskripsi Kelas dan Tanggung Jawab Kandidat Kelas untuk Pemberi Informasi Fasilitas Umum

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
Pemberi informasi fasilitas umum (<i>Infrastruktur Search</i>)	Merepresentasikan proses pemberian informasi fasilitas umum yang berada di sekitar <i>real estate</i> .	Pengguna menginginkan untuk menampilkan fasilitas umum kemudian melalui logika fasilitas umum dicari beberapa fasilitas umum yang berkaitan untuk ditampilkan pada pengguna.	 <i>Control</i>
Halaman HTML detail <i>real estate</i>	Merepresentasikan sebuah kelas sebagai antarmuka yang menampilkan pada pengguna informasi detail <i>real estate</i> .	Memodelkan sebuah antarmuka yang menerima masukan berupa informasi mengenai detail <i>real estate</i> kemudian menampilkannya pada pengguna.	 <i>View</i>
Logika fasilitas umum (<i>infrastructure Search</i>)	Merepresentasikan kelas yang digunakan untuk mengambil data fasilitas umum	Kelas menghubungkan sistem dengan basis data untuk mengambil	 <i>Control</i>

Nama Kelas (Alias Kelas)	Deskripsi Kelas	Tanggung Jawab Kelas	Jenis Kelas
	yang terdapat di sekitar <i>real estate</i> tertentu.	informasi fasilitas umum.	
Objek fasilitas umum (<i>Infrastruktur</i>)	Merepresentasikan objek yang digunakan untuk menyimpan data dan atribut <i>real estate</i> .	Memodelkan <i>real estate</i> sebagai objek <i>real estate</i> dengan atribut yaitu idRealEstate, idArea, Alamat, Harga, Koordinat, Fasilitas, Gambar, JumlahKamar, LuasTanah, dan LuasBangunan.	 Model
Halaman HTML informasi fasilitas umum	Merepresentasikan sebuah kelas sebagai antarmuka yang menampilkan pada pengguna informasi fasilitas umum.	Memodelkan sebuah antarmuka yang menerima masukan berupa informasi mengenai informasi fasilitas umum, kemudian menampilkannya pada pengguna	 View

3.1.7 Spesifikasi Kasus Penggunaan

Pada subbab ini akan dijelaskan tentang spesifikasi kasus penggunaan beserta penggambarannya dalam bentuk diagram

aktivitas yang merupakan ilustrasi alur normal dan alur alternatif kasus penggunaan tersebut dan diagram urutan yang menggambarkan daur hidup objek selama kasus penggunaan tersebut berjalan.

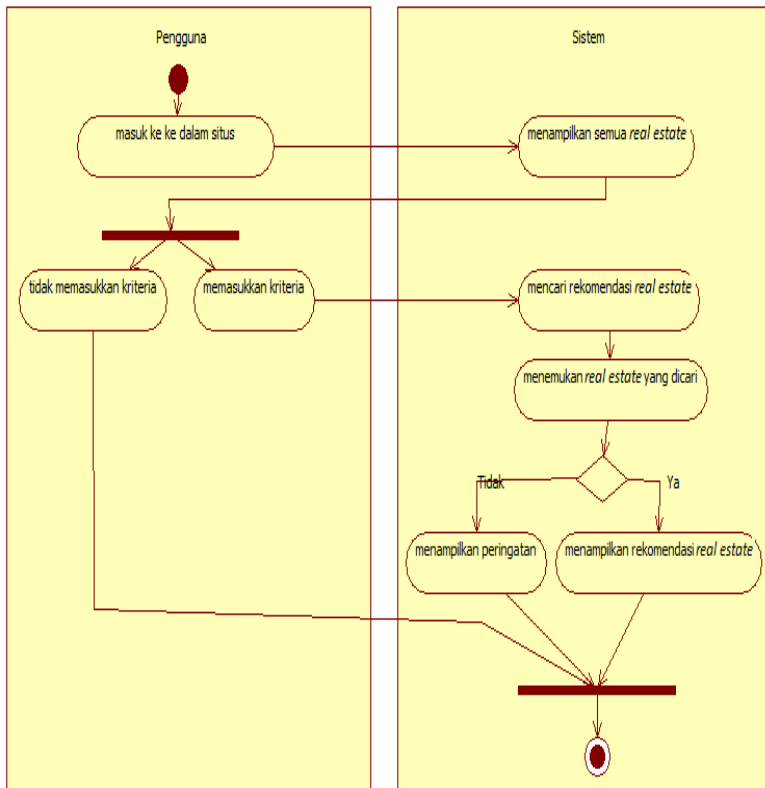
3.1.7.1 Kasus Penggunaan Pencarian Rekomendasi *Real Estate*

Spesifikasi kasus penggunaan pencarian rekomendasi *real estate* dijelaskan pada Tabel 3.15. Sedangkan aktivitas yang dilakukan untuk memenuhi kasus penggunaan ini tergambar dalam diagram aktivitas pada Gambar 3.4. Sementara itu komunikasi antar model yang terjadi saat berlangsungnya kasus penggunaan ini digambarkan dalam diagram urutan pada Gambar 3.5.

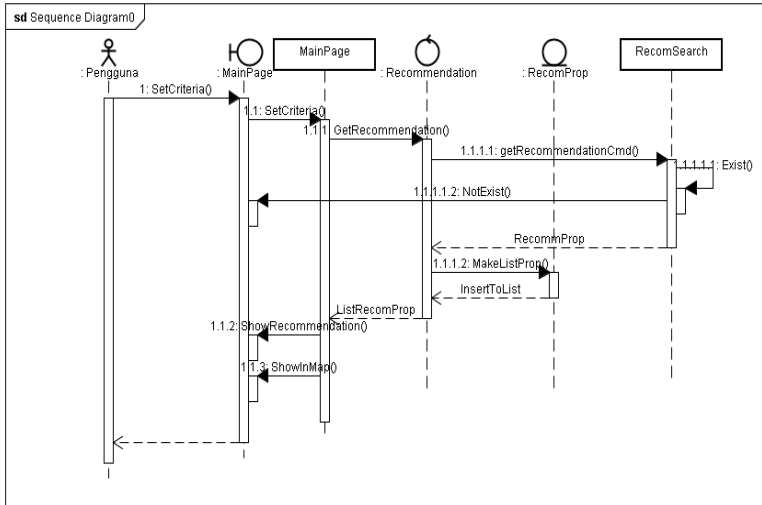
Tabel 3.15 Spesifikasi Kasus Penggunaan Pencarian Rekomendasi *Real Estate*

Nama Kasus Penggunaan	Mencari Rekomendasi <i>Real Estate</i> .
Nomor	UC-001.
Deskripsi	Pengguna memperoleh rekomendasi mengenai <i>real estate</i> yang cocok untuk pengguna dari sistem setelah memasukkan kriteria <i>real estate</i> pada sistem.
Aktor	Pengguna.
Kondisi Awal	Seluruh <i>real estate</i> akan muncul sesuai dengan area yang muncul di halaman awal atau halaman utama.
Kondisi Akhir	Terdapat <i>real estate</i> yang direkomendasikan oleh sistem.
Alur Normal	1. Pengguna memilih karakteristik <i>real estate</i>. 2. Sistem memberikan rekomendasi <i>real</i>

	<i>estate</i> sesuai dengan karakteristik yang dipilih pengguna.
Alur Alternatif	-



Gambar 3.4 Diagram Aktivitas Pencarian Rekomendasi *Real Estate*



Gambar 3.5 Diagram Urutan Pencarian Rekomendasi *Real Estate*

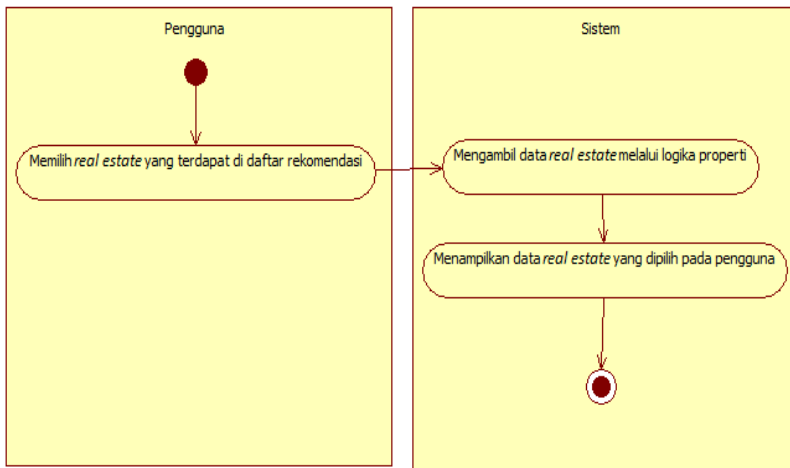
3.1.7.2 Kasus Penggunaan Memilih *Real Estate*

Spesifikasi kasus penggunaan memilih *real estate* dijelaskan pada Tabel 3.16. Sedangkan aktivitas yang dilakukan untuk memenuhi kasus penggunaan ini tergambar dalam diagram aktivitas pada Gambar 3.6. Sementara itu komunikasi antar model yang terjadi saat berlangsungnya kasus penggunaan ini digambarkan dalam diagram urutan pada Gambar 3.7.

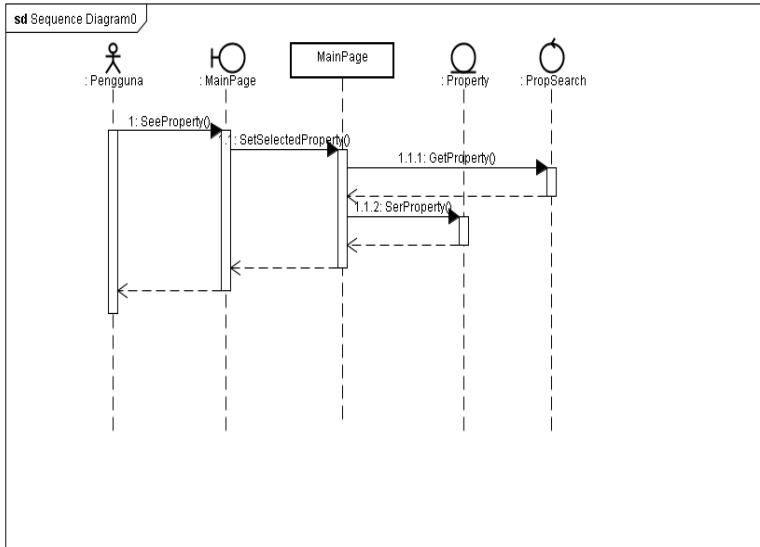
Tabel 3.16 Spesifikasi Kasus Penggunaan Memilih *Real Estate*

Nama Kasus Penggunaan	Memilih <i>Real Estate</i> .
Nomor	UC-002.
Deskripsi	Pengguna dapat memilih <i>real estate</i> yang ingin

	dilihat informasinya dan dapat menuju situs <i>web</i> sumber.
Aktor	Pengguna.
Kondisi Awal	Tidak ada data mengenai <i>real estate</i> .
Kondisi Akhir	Muncul data mengenai <i>real estate</i> .
Alur Normal	1. Pengguna memilih <i>real estate</i>. 2. Sistem mengambil data <i>real estate</i> pada basis data. 3. Sistem menampilkan data <i>real estate</i> pada halaman HTML.
Alur Alternatif	-



Gambar 3.6 Diagram Aktivitas Memilih *Real Estate*



Gambar 3.7 Diagram Urutan Memilih *Real Estate*

3.1.7.3 Kasus Penggunaan Melihat Informasi *Real Estate*

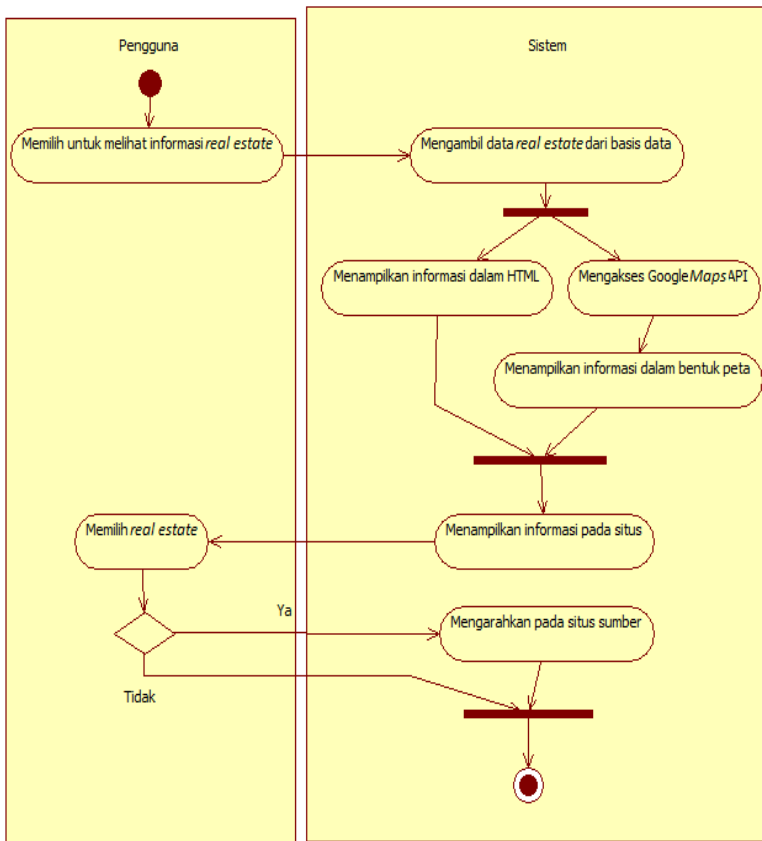
Spesifikasi kasus penggunaan melihat informasi *real estate* dijelaskan pada Tabel 3.17.

Tabel 3.17 Spesifikasi Kasus Penggunaan Melihat Informasi *Real Estate*

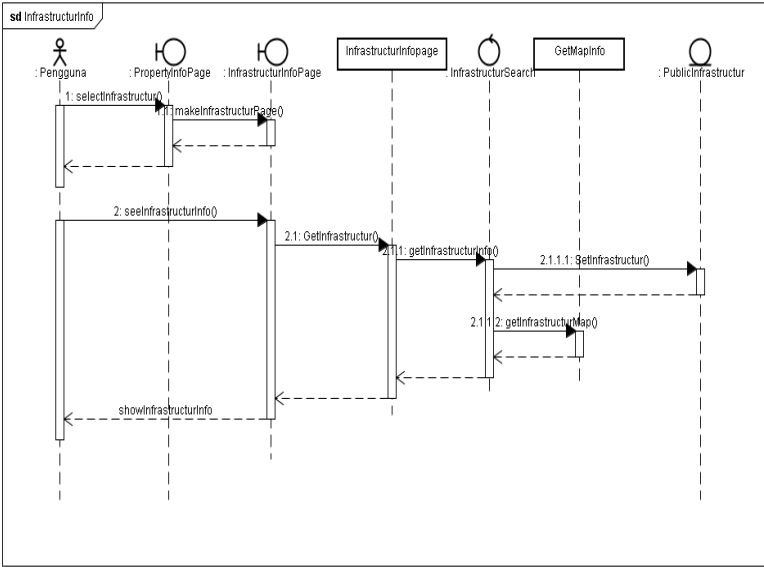
Nama Kasus Penggunaan	Melihat Informasi <i>Real Estate</i> .
Nomor	UC-003.
Deskripsi	Pengguna dapat melihat gambar dan informasi mengenai <i>real estate</i> secara lengkap dengan cara memilih <i>real estate</i> yang telah direkomendasikan dan

	ditampilkan.
Aktor	Pengguna.
Kondisi Awal	Tidak terdapat informasi mengenai sebuah <i>real estate</i> secara detail.
Kondisi Akhir	Muncul gambar dan informasi mengenai sebuah <i>real estate</i> secara detail.
Alur Normal	1. Pengguna memilih <i>real estate</i> untuk mengetahui detail informasi mengenai <i>real estate</i> yang dipilih. 2. Sistem meminta data pada basis data. 3. Sistem meminta data pada <i>Google Maps API</i> untuk menampilkan informasi <i>real estate</i> dalam bentuk peta. 4. Sistem menampilkan informasi mengenai <i>real estate</i> yang dipilih dalam bentuk HTML. 5. Sistem menampilkan informasi mengenai <i>real estate</i> kepada pengguna dalam bentuk peta.
Alur Alternatif	-

Sedangkan aktivitas yang dilakukan untuk memenuhi kasus penggunaan ini ditunjukkan dalam diagram aktivitas pada Gambar 3.8. Sementara itu, komunikasi antar model yang terjadi saat berlangsungnya proses pada kasus penggunaan ini digambarkan dalam diagram urutan pada Gambar 3.9.



Gambar 3.8 Diagram Aktivitas Melihat Informasi *Real Estate*



Gambar 3.9 Diagram Urutan Melihat Informasi *Real Estate*

3.1.7.4 Kasus Penggunaan Melihat Informasi Fasilitas Umum

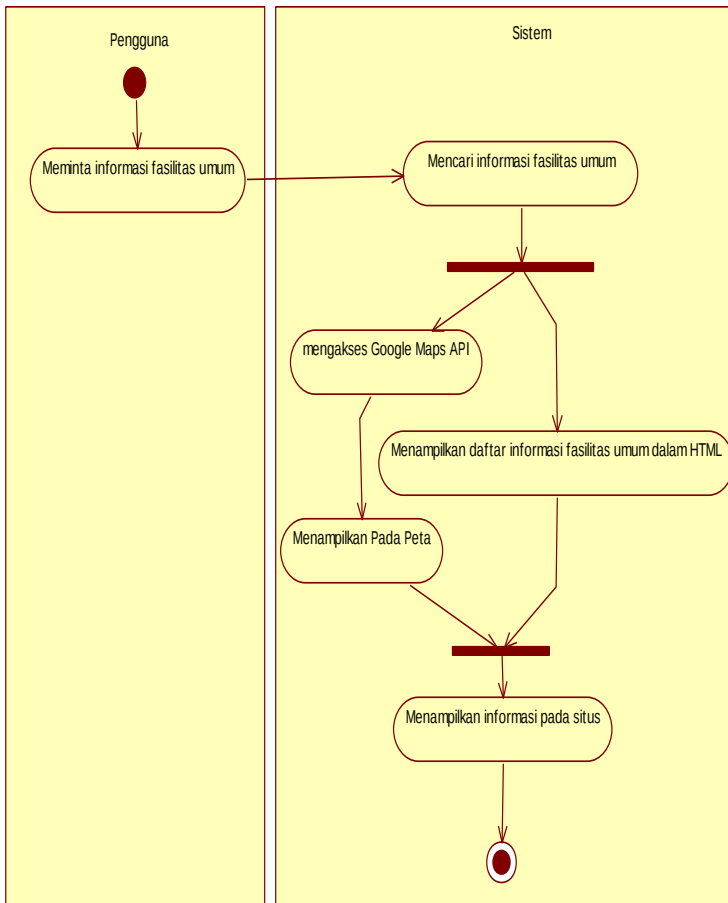
Pada bagian kasus penggunaan ini disajikan spesifikasi kasus penggunaan melihat informasi fasilitas umum yang dijelaskan pada Tabel 3.18.

Tabel 3.18 Spesifikasi Kasus Penggunaan Melihat Informasi Fasilitas Umum

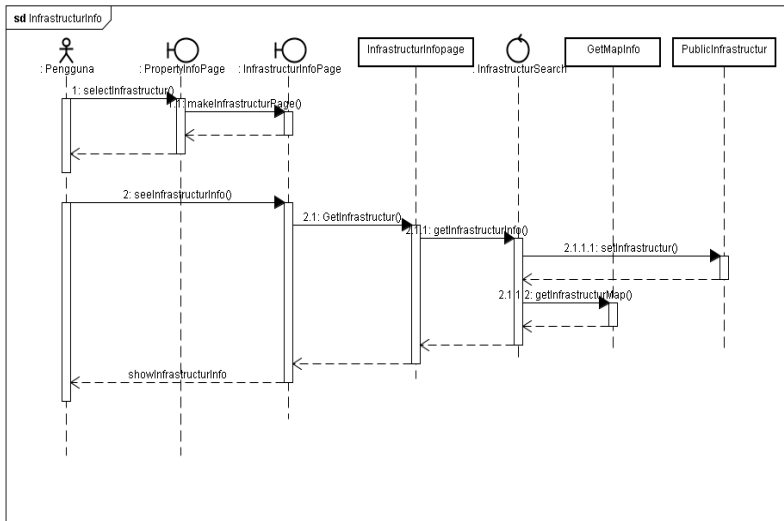
Nama Kasus Penggunaan	Melihat Informasi Fasilitas Umum.
Nomor	UC-004.
Deskripsi	Pengguna dapat mengetahui informasi dan lokasi fasilitas umum yang terdapat di sekitar

	<i>real estate</i> .
Aktor	Pengguna.
Kondisi Awal	Tidak terdapat informasi dan lokasi mengenai fasilitas umum dalam bentuk daftar fasilitas umum dan dalam bentuk peta.
Kondisi Akhir	Terdapat informasi dan lokasi mengenai fasilitas umum dalam bentuk daftar fasilitas umum dan dalam bentuk peta.
Alur Normal	1. Pengguna meminta rekomendasi <i>real estate</i> sesuai dengan kriteria yang diinginkan pengguna. 2. Sistem menampilkan kepada pengguna informasi berupa peta mengenai lokasi <i>real estate</i> berada. 3. Sistem menampilkan kepada pengguna informasi berupa peta mengenai fasilitas umum yang terdapat di sekitar <i>real estate</i>.
Alur Alternatif	-

Sedangkan aktivitas yang dilakukan untuk memenuhi kasus penggunaan ini tergambar dalam diagram aktivitas yang ditunjukkan pada Gambar 3.10. Sementara itu komunikasi antar model yang terjadi saat berlangsungnya kasus penggunaan ini digambarkan dalam diagram urutan yang ditunjukkan pada Gambar 3.11.



Gambar 3.10 Diagram Aktivitas Melihat Informasi Fasilitas Umum



Gambar 3.11 Diagram Urutan Melihat Informasi Fasilitas Umum

3.2 Tahap Perancangan

Dari analisis sistem yang telah dijelaskan sebelumnya, maka dibutuhkan perancangan yang bisa memenuhi kebutuhan fungsional yang ada. Perancangan perangkat lunak yang ada meliputi perancangan data, proses, diagram kelas, arsitektur perangkat lunak, dan perancangan antarmuka pengguna.

3.2.1 Perancangan Data

Pada bagian ini diberikan gambaran mengenai rancangan basis data yang digunakan dalam perangkat lunak ini. Data yang digunakan dalam sistem ini terbagi menjadi empat, yaitu data masukan, data proses, data keluaran, dan basis data. Data masukan adalah data yang diambil oleh sistem dari *web* informasi *real estate*. Data proses adalah data yang digunakan ketika

pengolahan dokumen sedang berlangsung. Data keluaran adalah data hasil proses dari sistem yang akan ditampilkan kepada pengguna. Basis data adalah suatu media penyimpanan dan pengolahan yang digunakan terhadap ketiga jenis data sebelumnya.

3.2.1.1 Data Masukan

Terdapat tiga data yang menjadi data masukan pada sistem. Data pertama adalah data *real estate* dari situs *real estate* sebagai data masukan. Data kedua adalah data XML yang berasal dari layanan *Google Maps* yang berisi informasi peta *real estate* dan informasi peta fasilitas umum. Data ketiga adalah waktu untuk menjalankan sistem secara berkala.

3.2.1.2 Data Proses

Data yang digunakan saat proses berlangsung adalah data kriteria *real estate*. Data kriteria digunakan untuk mencari *real estate* yang diinginkan pengguna.

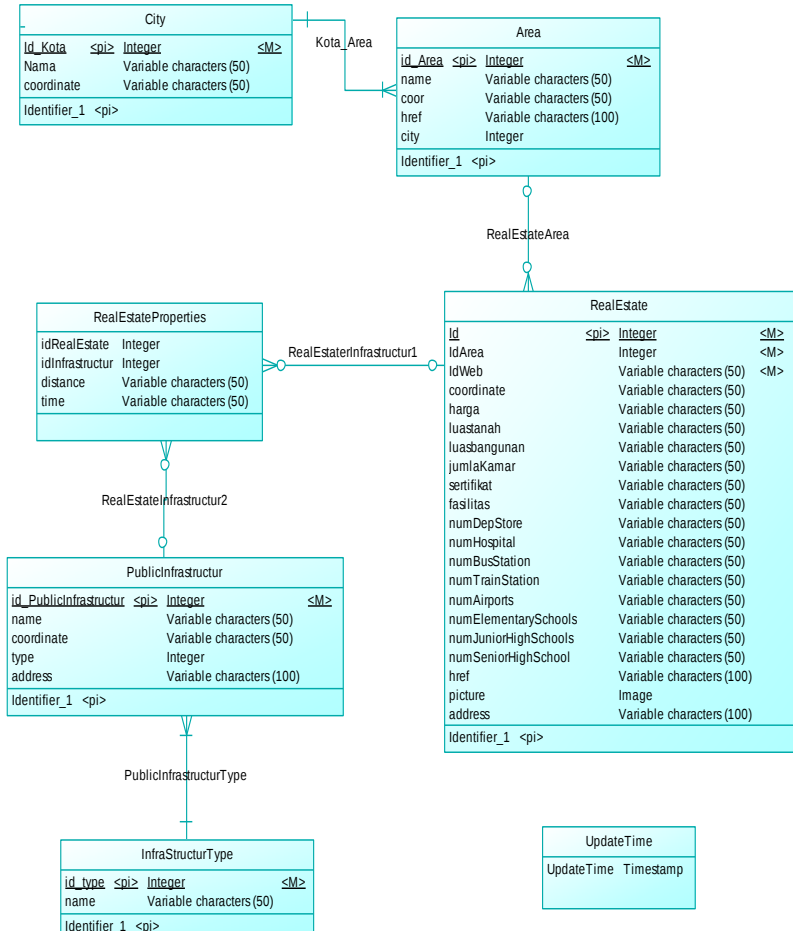
3.2.1.3 Data Keluaran

Proses utama sistem adalah pengambilan data dari situs sumber dan pemberian daftar rekomendasi *real estate*. Jadi, data yang ditampilkan pada pengguna adalah daftar rekomendasi *real estate* dan informasi yang berhubungan dengan rekomendasi *real estate*.

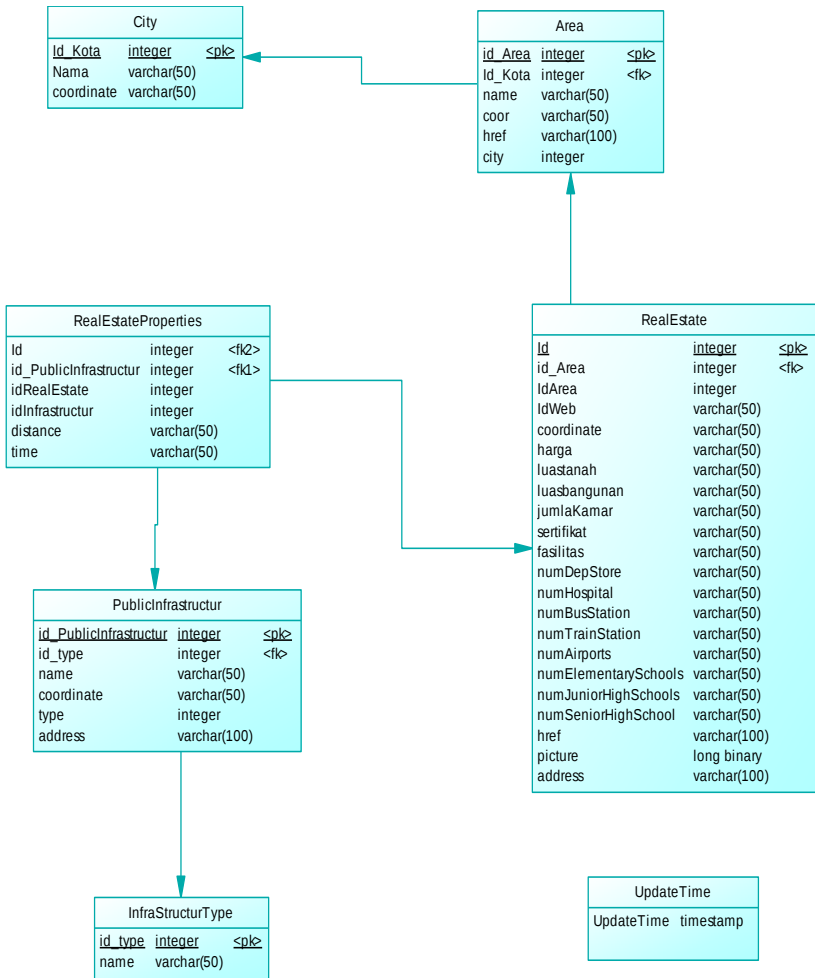
3.2.1.4 Data Basis Data

Penjelasan mengenai basis data digambarkan berupa struktur *Conceptual Data Model* (CDM). CDM dapat dilihat pada Gambar 3.12. Selain itu ditampilkan juga *Physical Data Model* (PDM) yang ditampilkan pada Gambar 3.13. Data yang diambil untuk membuat perangkat lunak ini adalah data HTML dan data XML.

HTML merupakan data dari sistem informasi *real estate* dan kemudian data tersebut disimpan dalam basis data. Sedangkan data XML merupakan data yang didapat dari layanan *web* Google *Maps* API.



Gambar 3.12 Conceptual Data Model



Gambar 3.13 Physical Data Model

Dari PDM dan CDM pada Gambar 3.12 dan Gambar 3.13 terdapat tujuh tabel yang digunakan dalam sistem. Ketujuh tabel tersebut antara lain:

1. Tabel RealEstate

Tabel ini menyimpan data berupa *real estate* beserta atribut-atributnya yang diambil dari proses *crawling* dari situs *real estate* dan layanan *Google Maps*. Tabel ini digunakan sebagai sumber informasi dalam memberikan rekomendasi pada pengguna. Penjelasan tabel RealEstate dapat dilihat pada Tabel 3.19

Tabel 3.19 Penjelasan Atribut Tabel RealEstate

Nama Atribut	Tipe Data	Keterangan
Id	<i>Integer</i>	Sebagai kunci utama dalam tabel RealEstate.
Id_Area	<i>Integer</i>	Menyimpan kunci asing dari tabel Area yang menunjukkan area <i>real estate</i> berada.
IdWeb	<i>Varchar</i>	Menyimpan id <i>real estate</i> yang berasal dari situs sumber.
Coordinat	<i>varchar</i>	Menyimpan koordinat lintang bujur <i>real estate</i> .
Harga	<i>Varchar</i>	Menyimpan harga <i>real estate</i> .
LuasTanah	<i>Varchar</i>	Menyimpan ukuran luas tanah <i>real estate</i> .
LuasBangunan	<i>Varchar</i>	Menyimpan ukuran luas bangunan <i>real estate</i> .
Sertifikat	<i>Varchar</i>	Menyimpan sertifikat <i>real estate</i> .
Fasilitas	<i>Varchar</i>	Menyimpan informasi fasilitas yang dimiliki <i>real estate</i> .
NumDepStore	<i>Varchar</i>	Jumlah pasar yang berada di sekitar <i>real estate</i> .
NumHospital	<i>Varchar</i>	Jumlah rumah sakit yang berada di sekitar <i>real estate</i> .
NumBusStation	<i>Varchar</i>	Jumlah terminal bus yang berada di sekitar <i>real estate</i> .
NumTrainStation	<i>Varchar</i>	Jumlah stasiun kereta yang

Nama Atribut	Tipe Data	Keterangan
		berada di sekitar <i>real estate</i> .
NumAirport	Varchar	Jumlah bandara yang berada di sekitar <i>real estate</i> .
NumElemSchool	Varchar	Jumlah sekolah dasar yang berada di sekitar <i>real estate</i> .
NumJuniorSchool	Varchar	Jumlah SMP yang berada di sekitar <i>real estate</i> .
NumSeniorSchool	Varchar	Jumlah SMA yang berada di sekitar <i>real estate</i> .
Href	Varchar	Menyimpan alamat URL situs sumber <i>real estate</i> .
Picture	Long binary	Menyimpan gambar <i>real estate</i> .
Address	Varchar	Menyimpan alamat <i>real estate</i> .

2. Tabel PublicInfrastruktur

Tabel ini menyimpan data berupa fasilitas umum beserta atribut-atributnya yang diambil dari layanan *web Google Maps*. Tabel ini akan digunakan sebagai sumber informasi mengenai fasilitas umum yang terdapat di sekitar *real estate*. Penjelasan mengenai tabel PublicInfrastruktur dapat dilihat pada Tabel 3.20

Tabel 3.20 Penjelasan Atribut Tabel PublicInfrastruktur

Nama Atribut	Tipe Data	Keterangan
Id_PublicInfrastruktur	Integer	Sebagai kunci utama dalam tabel PublicInfrastruktur.
Id_Type	Integer	Menyimpan kunci asing dari tabel InfrastrukturType menunjukkan jenis fasilitas umum.
Name	Varchar	Menyimpan nama fasilitas umum.
Coordinat	Varchar	Menyimpan koordinat.

Nama Atribut	Tipe Data	Keterangan
Type	<i>Varchar</i>	Menyimpan tipe fasilitas umum.
Address	<i>Varchar</i>	Menyimpan alamat fasilitas umum.

3. Tabel RealEstateProperties

Tabel ini menyimpan data yang menjelaskan fasilitas umum yang terdapat disekitar *real estate*. Tabel ini menghubungkan antara tabel RealEstate dan tabel PublicInfrastructur yang berhubungan banyak ke banyak (*many to many*). Penjelasan mengenai tabel RealEstateProperties dapat dilihat pada Tabel 3.21.

Tabel 3.21 Penjelasan Atribut Tabel RealEstateProperties

Nama Atribut	Tipe Data	Keterangan
Id	<i>Integer</i>	Sebagai kunci utama dalam tabel RealEstate.
Id_PublicInfrastructur	<i>Integer</i>	Menyimpan kunci asing dari tabel PublicInfrastructur yang menunjukkan fasilitas umum.
IdRealEstate	<i>Integer</i>	Menyimpan kunci asing dari tabel RealEstate yang menunjukkan <i>real estate</i> .
Distance	<i>Varchar</i>	Menyimpan jarak antara <i>real estate</i> dengan fasilitas umum.
Time	<i>Varchar</i>	Menyimpan waktu tempuh <i>real estate</i> dengan fasilitas umum.

4. Tabel InfrastructurType

Tabel ini menyimpan data tipe fasilitas umum yang terdapat pada tabel PublicInfrastructur. Penjelasan mengenai tabel InfrastructurType dapat dilihat pada Tabel 3.22.

Tabel 3.22 Penjelasan Atribut Tabel InfrastrukturType

Nama Atribut	Tipe Data	Keterangan
Id_Type	<i>Integer</i>	Sebagai kunci utama dalam tabel InfrastrukturType.
Name	<i>Varchar</i>	Menyimpan jenis fasilitas umum.

5. Tabel City

Tabel ini menyimpan daftar kota yang menyatakan tempat suatu area berada. Tabel ini memiliki hubungan banyak ke satu (*many to one*) dengan tabel Area. Penjelasan mengenai tabel City dapat dilihat pada Tabel 3.23.

Tabel 3.23 Penjelasan Atribut Tabel City

Nama Atribut	Tipe Data	Keterangan
Id_Kota	<i>Integer</i>	Sebagai kunci utama dalam tabel RealEstate.
Name	<i>Varchar</i>	Menyimpan nama kota.
Coordinat	<i>Varchar</i>	Menyimpan koordinat kota.

6. Tabel Area

Tabel ini menyimpan data area yang menyatakan area suatu *real estate* berada. Tabel ini memiliki hubungan banyak ke satu (*many to one*) dengan tabel RealEstate. Penjelasan mengenai tabel City ditunjukkan Tabel 3.24.

Tabel 3.24 Penjelasan Atribut Tabel Area

Nama Atribut	Tipe Data	Keterangan
Id_Kota	<i>Integer</i>	Sebagai kunci utama tabel Area.
Name	<i>Varchar</i>	Menyimpan nama area.

Nama Atribut	Tipe Data	Keterangan
Coordinat	<i>Varchar</i>	Menyimpan koordinat area.

7. Table UpdateTime

Tabel ini menyimpan data kapan waktu terakhir basis data melakukan pembaruan. Tabel UpdateTime berfungsi agar sistem tahu kapan saatnya basis data perlu diperbarui. Penjelasan mengenai tabel UpdateTime dapat dilihat pada Tabel 3.24.

Tabel 3.25 Penjelasan Atribut Tabel UpdateTime

Nama Atribut	Tipe Data	Keterangan
updateTime	<i>Timestamp</i>	Menyimpan waktu terakhir pembaruan data.

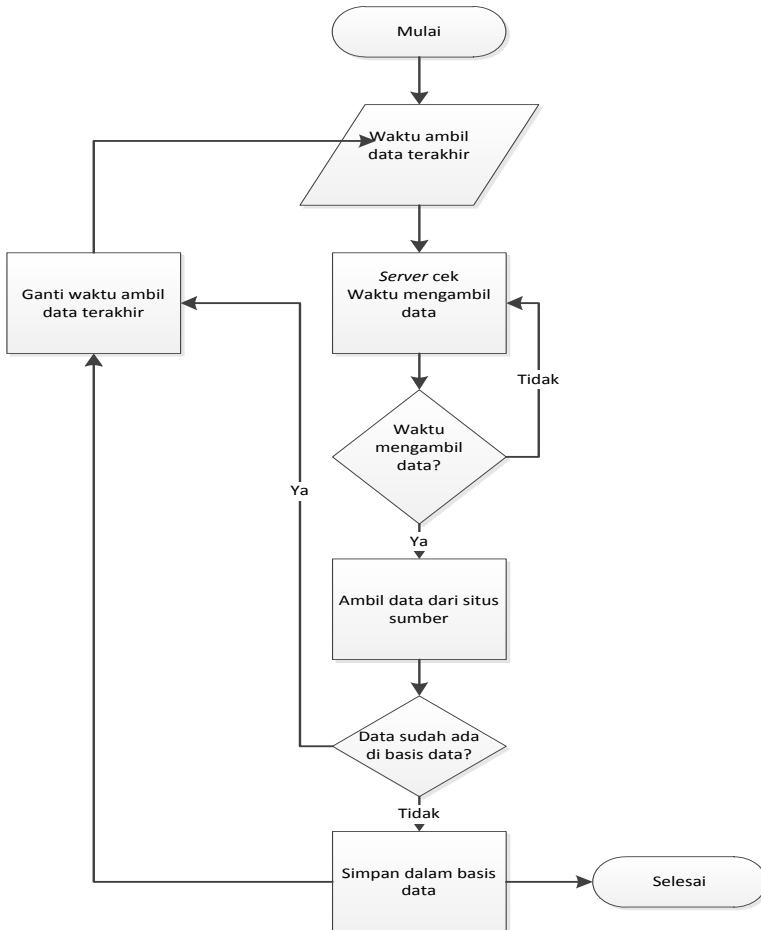
3.2.2 Perancangan Proses Server

Pada bagian ini dijelaskan tentang proses-proses apa saja yang berjalan dan bagaimana proses tersebut berjalan pada sisi *server*. Setiap proses akan dijelaskan dengan beberapa gambar berupa diagram alur, diagram aktivitas, dan diagram urutan.

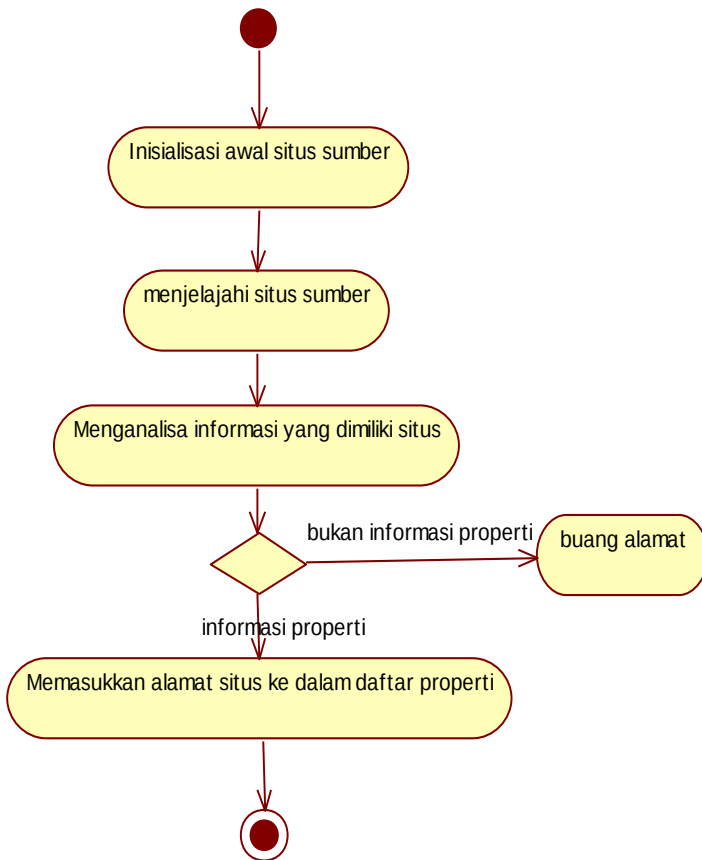
3.2.2.1 Proses Merayapi (*Crawling*) HTML

Proses *crawling* HTML merupakan tahap awal untuk mendapatkan informasi mengenai *real estate* yang saat ini sedang dijual. Proses ini dilakukan dengan cara mengunduh data HTML kemudian diproses dengan menggunakan metode *crawling* sehingga menghasilkan informasi pokok mengenai *real estate* yang kemudian akan dimasukkan ke dalam basis data. Masukan awal proses ini berupa URL dari sebuah situs yang akan diunduh. Proses pengunduhan akan dilakukan secara terus menerus dengan periode waktu tertentu. Proses ini digambarkan diagram alir pada Gambar 3.14. Sedangkan aktivitas yang dilakukan untuk

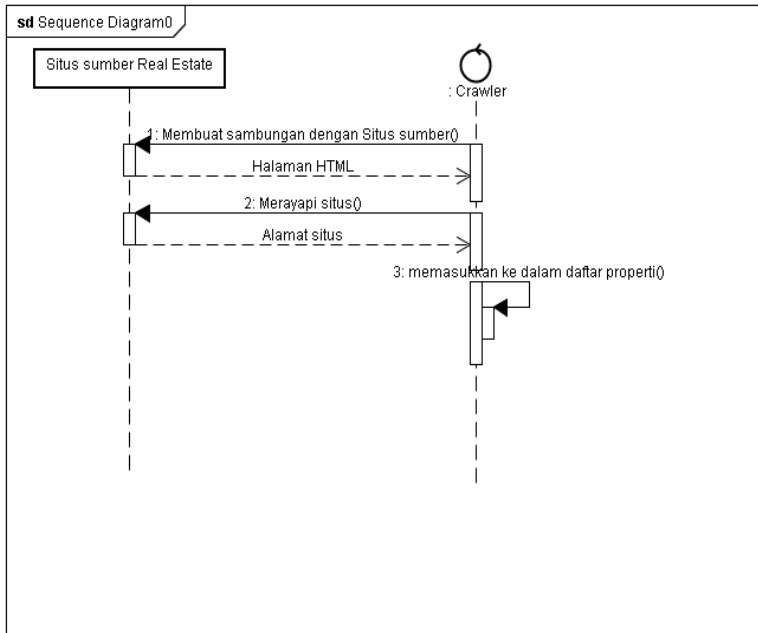
memenuhi proses ini tergambar dalam diagram aktivitas pada Gambar 3.15. Sementara itu komunikasi antar model yang terjadi saat berlangsungnya kasus penggunaan ini digambarkan dalam diagram urutan pada Gambar 3.16.



Gambar 3.14 Diagram Alir Proses *Crawling* HTML



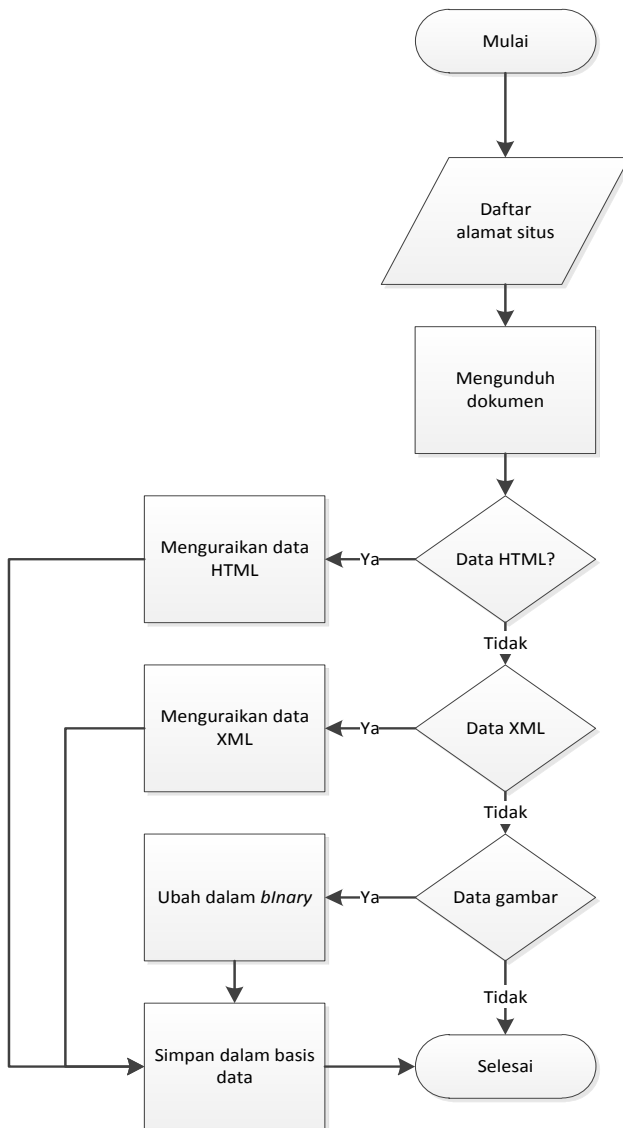
Gambar 3.15 Diagram Aktivitas Proses *Crawling* HTML



Gambar 3.16 Diagram Urutan Proses *Crawling* HTML

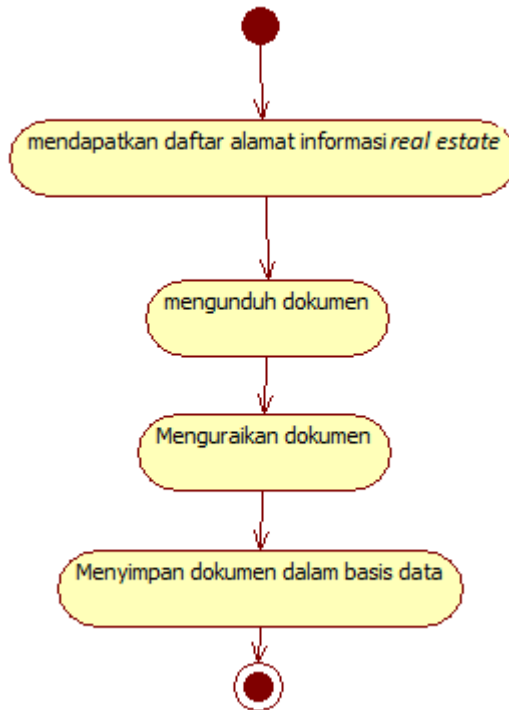
3.2.2.2 Proses Penguraian (*Parser*)

Pengurai dokumen mempunyai tugas untuk menguraikan dokumen dan mengambil isinya. Awalnya dokumen akan diunduh terlebih dahulu dari situs sumber. Dokumen dalam bentuk HTML tersebut kemudian diambil bagian-bagian yang dianggap penting sebagai sumber informasi. Data tersebut kemudian akan disimpan di dalam basis data. Selain data dalam bentuk HTML, pengurai dokumen juga menguraikan data XML yang didapat dari proses *geocoding*. Proses ini digambarkan diagram alir pada Gambar 3.17

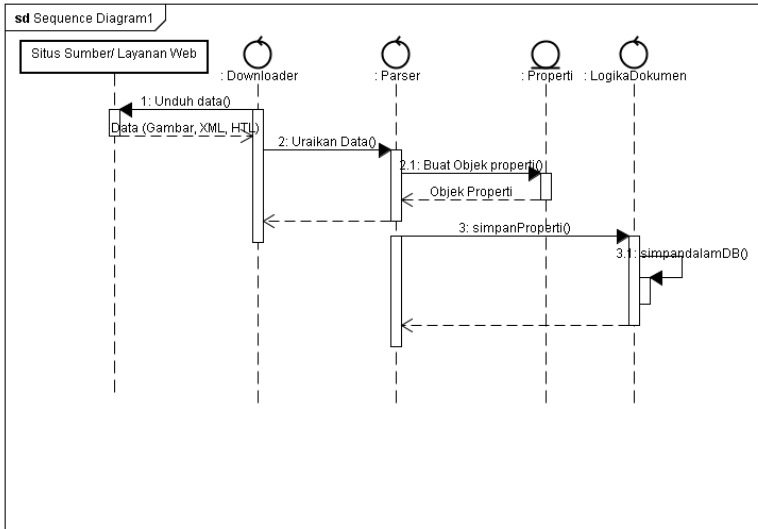


Gambar 3.17 Diagram Alir Proses Penguraian

Sedangkan aktivitas yang dilakukan untuk memenuhi proses ini tergambar dalam diagram aktivitas pada Gambar 3.18. Sementara itu, komunikasi antar model yang terjadi saat berlangsungnya proses ini digambarkan dalam diagram urutan pada Gambar 3.19.



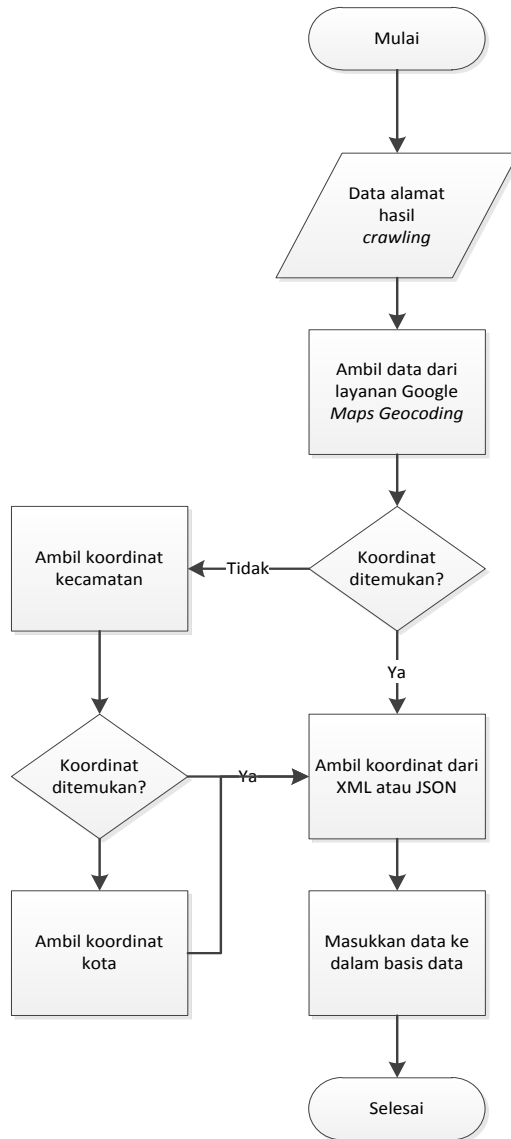
Gambar 3.18 Diagram Aktivitas Proses Penguraian



Gambar 3.19 Diagram Urutan Proses Penguraian

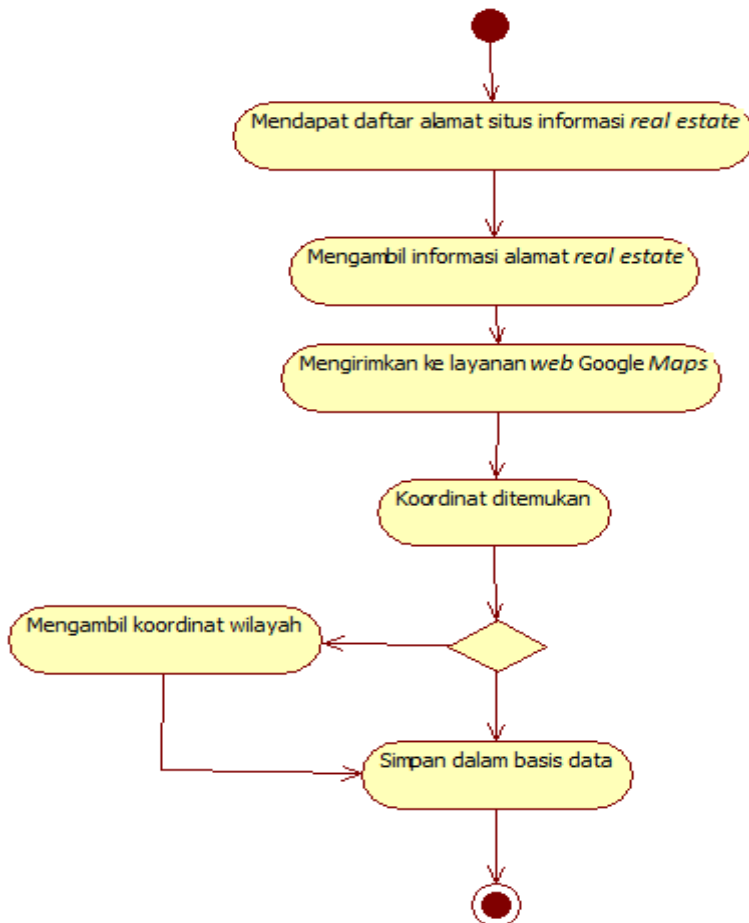
3.2.2.3 Proses *Geocoding*

Proses *geocoding* merupakan tahap mengubah alamat menjadi koordinat lintang bujur. Pada *server*, *geocoding* digunakan untuk mengetahui koordinat sebuah *real estate* sehingga data tersebut nantinya dapat dilakukan *mashup* dengan layanan *web* Google Maps. Proses *geocoding* ini dilakukan dengan cara mengirimkan URL tertentu menuju Google Maps yang nantinya akan diproses oleh layanan *web* Google Maps. Hasil yang didapat merupakan sebuah data berbentuk JSON atau XML dan informasi yang penting akan diambil dan dimasukkan ke dalam basis data. Penjelasan lebih detail akan dijelaskan diagram alir pada Gambar 3.20.

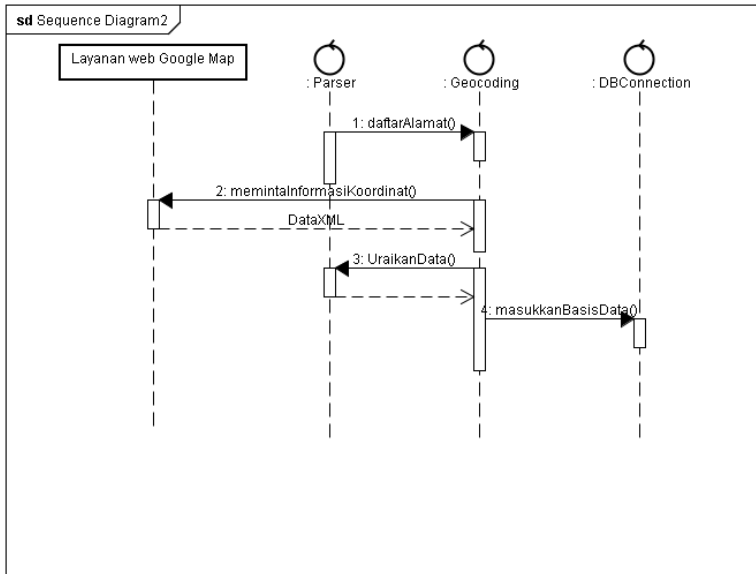


Gambar 3.20 Diagram Alir Proses *Geocoding*

Sedangkan aktivitas yang dilakukan untuk memenuhi proses ini tergambar dalam diagram aktivitas pada Gambar 3.21. Sementara itu, komunikasi antar model yang terjadi saat berlangsungnya proses ini digambarkan dalam diagram urutan pada Gambar 3.22.



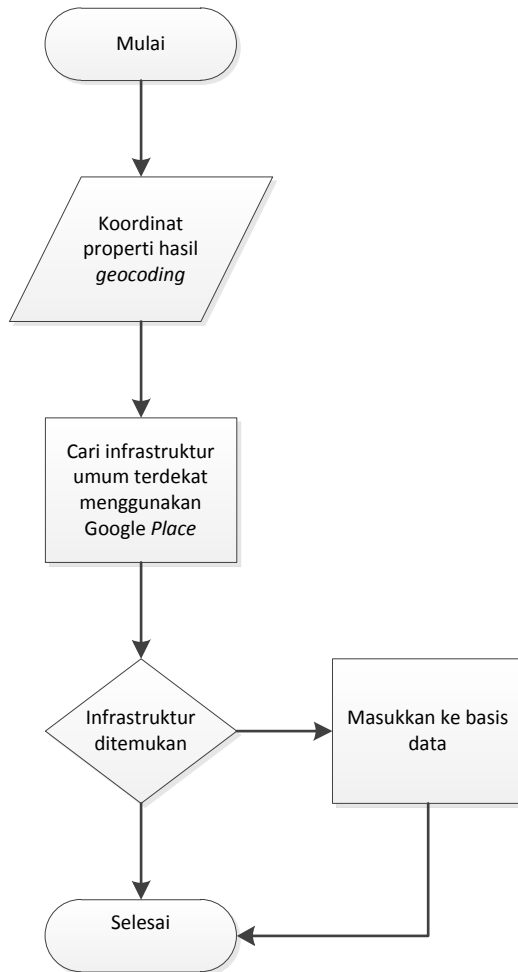
Gambar 3.21 Diagram Aktivitas Proses *Geocoding*



Gambar 3.22 Diagram Urutan Proses Geocoding

3.2.2.4 Proses Pencarian Fasilitas Umum

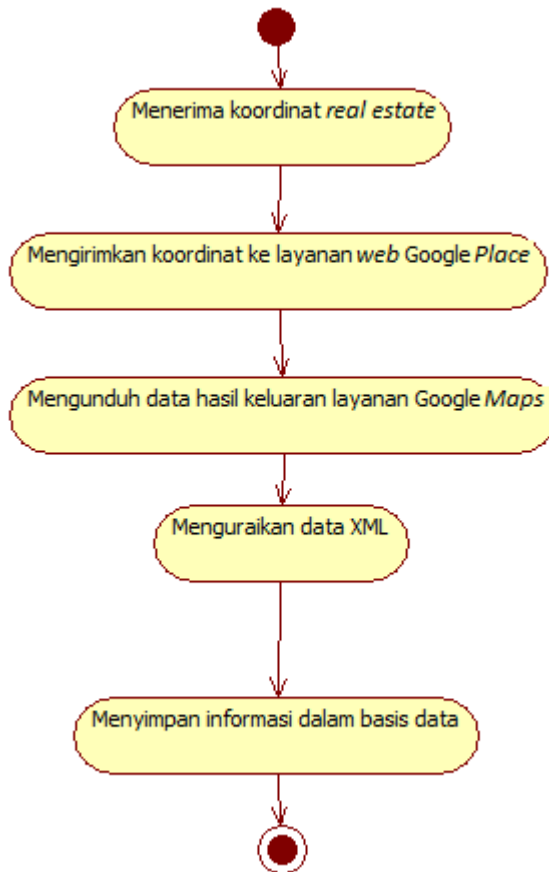
Proses pencarian fasilitas umum merupakan proses mencari infrastruktur tertentu yang berada di sekitar *real estate*. Untuk melakukan proses ini digunakan bantuan layanan *web* Google *Place*. Sama seperti proses *Geocoding* di atas, pencarian infrastruktur dilakukan dengan cara mengirimkan URL tertentu menuju ke Google *Place* dengan nilai kembalian berupa XML atau JSON. Data tersebut akan disimpan di dalam basis data yang nantinya berguna untuk memberikan rekomendasi bagi pengguna. Penjelasan alur proses pencarian fasilitas umum akan dijelaskan dalam diagram alir pada Gambar 3.23.



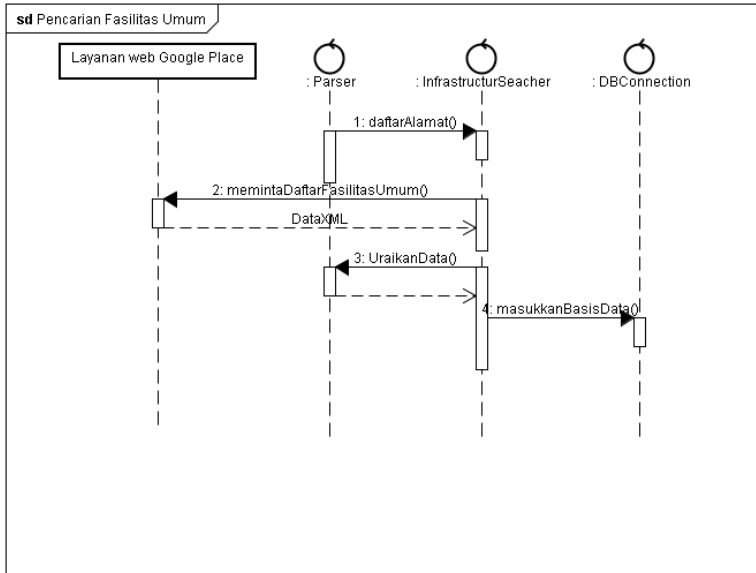
Gambar 3.23 Diagram Alir Proses Pencarian Fasilitas Umum

Sedangkan aktivitas yang dilakukan untuk memenuhi proses ini tergambar dalam diagram aktivitas pada Gambar 3.24. Sementara itu komunikasi antar model yang terjadi saat berlangsungnya

kasus penggunaan ini digambarkan dalam diagram urutan pada Gambar 3.25.



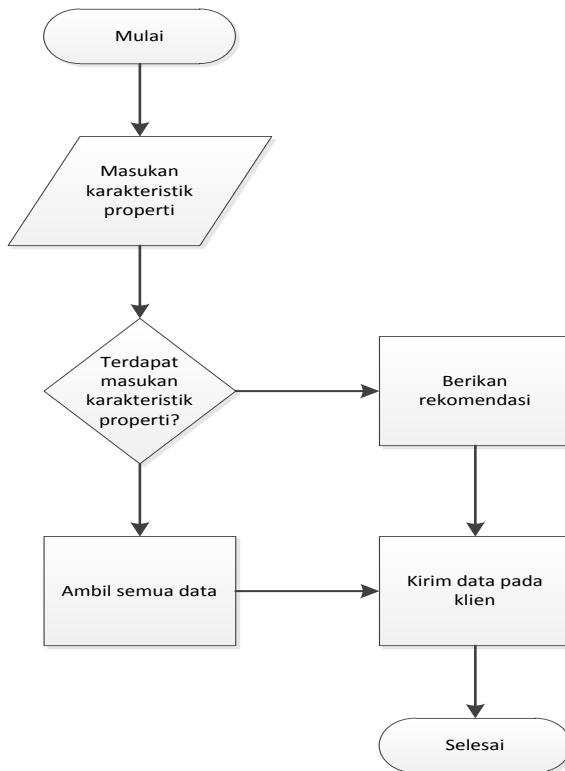
Gambar 3.24 Diagram Aktivitas Proses Pencarian Fasilitas Umum



Gambar 3.25 Diagram Urutan Proses Pencarian Fasilitas Umum

3.2.2.5 Proses Pemberian Rekomendasi

Proses pemberian rekomendasi pada klien merupakan proses pokok dalam pembuatan perangkat lunak ini. Pemberian rekomendasi *real estate* pada perangkat lunak ini dilakukan dengan cara mengambil data masukan dari pengguna mengenai kriteria *real estate* yang diinginkan. Kemudian sistem akan melakukan *query* sesuai dengan masukan pengguna. Hasil yang diperoleh nanti akan dilakukan proses *mashup* kembali, kemudian akan ditampilkan pada pengguna dalam bentuk informasi situs *web* dan peta. Penjelasan alur proses pemberian rekomendasi akan dijelaskan dalam diagram alir pada Gambar 3.26.



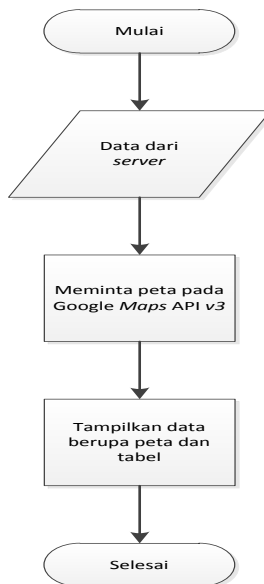
Gambar 3.26 Diagram Alir Proses Pemberian Rekomendasi

3.2.3 Perancangan Proses Klien

Pada bagian ini akan dijelaskan tentang proses-proses apa saja yang berjalan pada sisi klien. Setiap proses akan dijelaskan dengan sebuah gambar berupa diagram alir beserta penjelasannya.

3.2.3.1 Proses Menampilkan Hasil Rekomendasi

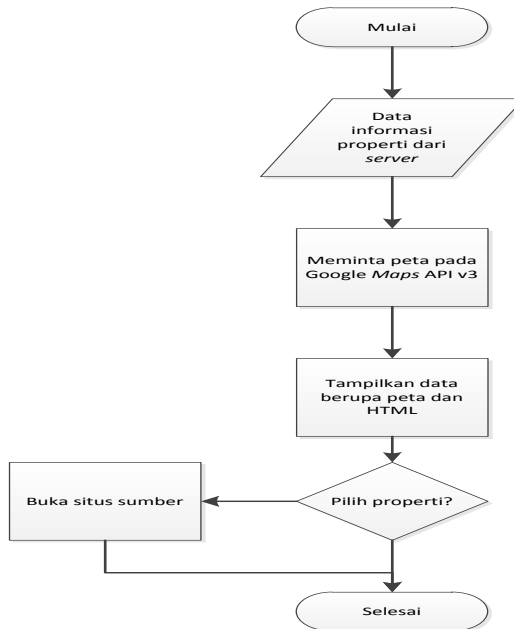
Pada proses ini data dari *server* akan diterjemahkan oleh klien dan ditampilkan dalam bentuk HTML. Klien akan meminta data rekomendasi dari *server* sesuai dengan masukan dari pengguna. Data yang diberikan oleh *server* akan dibagi lagi menjadi beberapa *real estate* dengan nilai rekomendasi yang berbeda yaitu rekomendasi tinggi, rekomendasi sedang, dan rekomendasi rendah. Selain ditampilkan secara langsung dalam bentuk HTML, data akan diproses lagi pada sisi klien untuk dapat menampilkan informasi peta dengan menggunakan bantuan *Google Maps API V3*. Penjelasan alur proses menampilkan hasil rekomendasi akan dijelaskan dalam diagram alir pada Gambar 3.27.



Gambar 3.27 Diagram Alir Proses Menampilkan Hasil Rekomendasi

3.2.3.2 Proses Menampilkan Informasi *Real Estate*

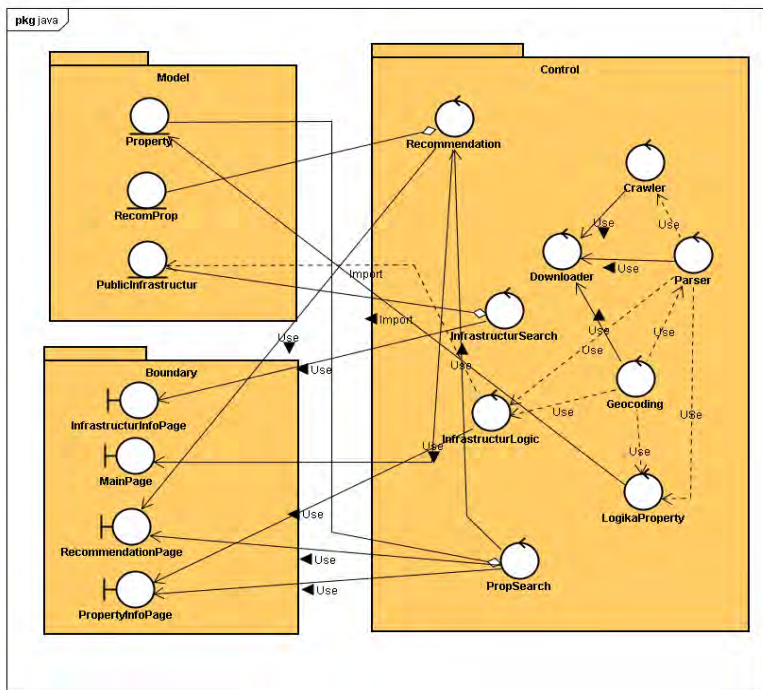
Pada proses ini data dari *server* akan diterjemahkan oleh klien dan ditampilkan dalam bentuk HTML. Klien akan meminta data informasi sebuah *real estate* dari *server* sesuai dengan masukan dari pengguna. Data yang diberikan oleh *server* berupa informasi detail mengenai *real estate* tersebut termasuk informasi mengenai fasilitas umum yang berada di sekitar *real estate*. Data yang didapat akan diolah pada sisi klien dan ditampilkan dalam bentuk HTML dan informasi peta dengan menggunakan *Google Maps API V3*. Penjelasan alur proses menampilkan informasi *real estate* akan dijelaskan dalam diagram alir pada Gambar 3.28.



Gambar 3.28 Diagram Alir Proses Menampilkan Informasi *Real Estate*

3.2.4 Perancangan Diagram Kelas

Diagram kelas digunakan untuk menggambarkan hubungan antara kelas dan hubungannya dalam sebuah sistem, termasuk juga menggambarkan masing-masing atribut dan operasi yang dimiliki oleh sebuah kelas. Pada subbab ini akan dijelaskan hubungan antar kelas yang terdapat pada perangkat lunak ini. Kelas-kelas yang terdapat pada diagram kelas ini merupakan kelas-kelas hasil dari proses analisis kelas yang telah dilakukan sebelumnya. Perancangan diagram kelas dapat dilihat pada Gambar 3.29.



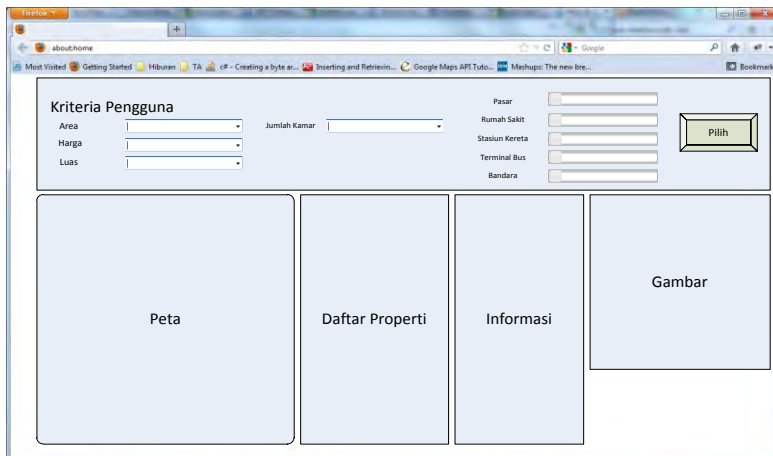
Gambar 3.29 Diagram Kelas

Pada gambar tersebut garis panah putus-putus mengandung arti hubungan *independent* antara kedua kelas tersebut, garis panah terang menggambarkan hubungan asosiasi antar kelas, sedangkan garis dengan tanda wajik di ujung menunjukkan hubungan *aggregation* antar kelas. Pada gambar diagram kelas di atas, hubungan antara kelas *view* dan *model* dilakukan melalui perantara *controller*. Hal tersebut merupakan ciri-ciri arsitektur MVC. Keterangan rincian setiap kelas akan ditampilkan pada Lampiran A.

3.2.5 Perancangan Antarmuka

Pada subbab ini akan dijelaskan mengenai rancangan antarmuka grafis yang akan menghubungkan antara pengguna dengan sistem. Rancangan yang dibahas meliputi kontrol pada antarmuka dan ketentuan masukan serta navigasi pada kontrol tersebut. Pada masing-masing subbab akan dijelaskan dan diberi gambaran tentang antarmuka pengguna sesuai dengan halaman-halaman yang ada pada aplikasi.

Perangkat lunak rekomendasi *real estate* ini hanya memiliki satu jenis halaman. Ini bertujuan agar pengguna mudah menggunakan perangkat lunak ini. Semua hasil rekomendasi dan informasi *real estate* yang diminta oleh pengguna akan muncul di halaman ini. Dengan hanya ada satu halaman informasi rekomendasi yang ada akan lebih mudah dimengerti oleh pengguna. Gambaran mengenai rancangan antarmuka terdapat pada Gambar 3.30.



Gambar 3.30 Rancangan Antarmuka Perangkat Lunak

Pada gambar di atas terdapat lima elemen pokok antarmuka perangkat lunak yaitu:

- Kriteria pengguna.
- Peta.
- Daftar *real estate*.
- Informasi.
- Gambar.

3.2.5.1 Kriteria Pengguna

Elemen ini merupakan antarmuka yang berfungsi sebagai masukan kriteria *real estate* yang diinginkan oleh pengguna. Pada bagian ini terdapat dua jenis masukan yang dapat dimanfaatkan pengguna untuk memasukkan kriteria yaitu *drop down list* dan *slide bar*. Masukan *drop down list* merupakan masukan berupa karakteristik *real estate* yang ingin dicari seperti harga, area, luas tanah, dan jumlah kamar. Sedangkan masukan *slide bar* merupakan masukan berupa akses dengan fasilitas umum seperti pasar, rumah sakit, stasiun kereta, terminal bus, dan bandara.

Tombol pilih berfungsi untuk memulai meminta rekomendasi pada sistem dan memunculkan informasi rekomendasi.

3.2.5.2 Peta

Elemen ini merupakan antarmuka yang berfungsi menampilkan informasi peta mengenai lokasi *real estate* yang direkomendasikan dan infrastruktur yang ada disekitar *real estate* yang dipilih. *Real estate* dan infrastruktur yang telah direkomendasikan akan muncul pada bagian ini dalam bentuk gambar yang menandakan lokasi keberadaan objek tersebut. Elemen ini memanfaatkan fitur dari Google Map API V3 yang dibuat dengan menggunakan *JavaScript*.

3.2.5.3 Daftar *Real Estate*

Elemen ini menampilkan beberapa *real estate* hasil rekomendasi yang dilakukan oleh sistem dalam bentuk tabel. Hasil rekomendasi yang terdapat dalam tabel ini dapat dipilih oleh pengguna untuk menampilkan informasi secara lengkap mengenai suatu *real estate*.

3.2.5.4 Informasi

Elemen ini menampilkan informasi secara lengkap mengenai karakteristik *real estate* yang telah dipilih. Informasi disajikan dalam bentuk tabel. Pada bagian ini pengguna juga dapat melakukan hubungan dengan penjual dengan cara memilih sambungan dengan situs *web* sumber.

3.2.5.5 Gambar

Elemen ini menampilkan gambar *real estate* yang telah dipilih. Dengan adanya elemen ini, pengguna dapat mengetahui kondisi rumah secara visual.

BAB IV IMPLEMENTASI

Bab ini membahas implementasi perancangan perangkat lunak dari aplikasi ini yang meliputi perancangan antarmuka yang mengacu pada analisis dan perancangan yang telah dibahas sebelumnya. Selain itu, subbab ini juga membahas lingkungan pembangunan perangkat lunak yang menjelaskan spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam pembangunan sistem.

4.1 Lingkungan Pembangunan Perangkat Lunak

Aplikasi dikembangkan dalam lingkungan pemrograman dengan spesifikasi berikut ini:

1. Spesifikasi komputer yang digunakan untuk pengembangan dan pengujian aplikasi adalah prosesor Intel Pentium Core (TM) 2 Duo @1.8Ghz, RAM 3 GB dengan sistem operasi Windows 7 Ultimate.
2. Visual Studio 2010 sebagai IDE pengembangan aplikasi.
3. Adobe Photoshop 7.0 dan CorelDRAW X5 sebagai perkakas untuk pembuatan gambar yang dibutuhkan di dalam aplikasi Rekomendasi *Real Estate* ini.
4. Star UML dan Jude alat bantu untuk pembuatan rancangan UML.
5. Mozilla Firefox 13 sebagai *browser* internet untuk pendaftaran dan pengujian aplikasi.

4.2 Implementasi Antarmuka Halaman Utama

Implementasi antarmuka halaman utama pengguna ditunjukkan oleh Gambar 4.1. Antarmuka utama terdiri dari beberapa *panel* yaitu *panel* untuk mengatur karakteristik *real estate*, *panel* untuk menampilkan daftar *real estate* yang berisi informasi singkat

mengenai *real estate*, dan panel untuk menampilkan informasi *real estate* dan fasilitas umum di dalam peta. Pada antarmuka ini juga terdapat beberapa navigasi untuk meminta rekomendasi, menampilkan fasilitas umum, dan menampilkan tabel daftar fasilitas umum.

REKOMENDASI REAL ESTATE

Cari berdasar Karakteristik

Karakteristik Property		Akses Fasilitas Umum (Menit)		Akses Sekolah (Menit)	
Area:	Harga:	Jumlah Kamar:	Luas Tana		
Gubeng	---	---	---		

Cari Properti	
Area	Harga
Jumlah Kamar	Luas Tanah
Pusat Perbelanjaan	Rumah Sakit
Terminal Bus	Stasiun Kereta



Daftar Real Estate



Alamat
Jl Barata Jaya IX
Gubeng
Harga
Rp. 450000000



Alamat
GUBENG Gubeng
Harga
Rp. 17500000000



Alamat
Kertajaya Gubeng
Harga
Rp. 4000000000

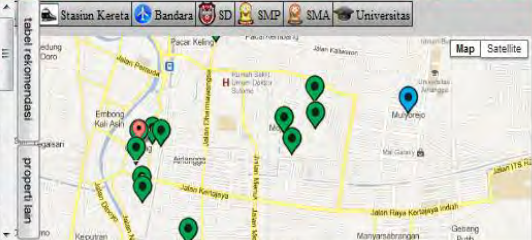
SOLD

Alamat
Jl Lombok Gubeng
Harga
Rp. 8000000000

Informasi Peta dan Tabel Real Estate

REKOMENDASI REAL ESTATE

no	harga	alamat
Pilih 0	638000000	Jl Jojoran Gubeng
Pilih 1	12000000000	Jl Sumbawa Gubeng
Pilih 2	16000000000	Jalan Blitton 11 Guba
Pilih 3	11000000000	Galaxy Royal Palace Gubeng
Pilih 4	1000000000	Jl Jojoran 1 Gubeng
Pilih 5	9800000000	Mojo Klangro Kidul Gubeng



Gambar 4.1 Antarmuka Halaman Utama Perangkat Lunak

4.3 Implementasi Pencarian Rekomendasi *Real Estate*

Pada subbab ini akan dijelaskan tentang implementasi antarmuka pengguna dan kode program untuk membuat fungsi mencari rekomendasi *real estate*.

4.3.1 Implementasi Antarmuka Pencarian Rekomendasi *Real Estate*

Antarmuka untuk pencarian rekomendasi *real estate* ditunjukkan oleh Gambar 4.2, Gambar 4.3, dan Gambar 4.4. Pada Gambar 4.2 terdapat pilihan menu untuk memilih karakter *real estate* yang diinginkan. Pada Gambar 4.3 terdapat pilihan menu untuk memilih jarak *real estate* dengan fasilitas umum. Sedangkan pada Gambar 4.4 terdapat pilihan menu untuk memilih jarak *real estate* dengan sekolah terdekat.

Cari berdasar Karakteristik					
Karakteristik Property		Akses Fasilitas Umum (Menit)		Akses Sekolah (Menit)	
☐ Pusat Perbelanjaan:		☐ Rumah Sakit:		☐ Terminal Bis:	
Pusat Perbelanjaan	---	Rumah Sakit	---		
Terminal Bus	---	Stasiun Kereta	---		
Bandara	---	SD	---		
SMP	---	SMA	---		

Gambar 4.2 Antarmuka Memilih Karakteristik Perangkat Lunak

Cari berdasar Karakteristik

Karakteristik Property Akses Fasilitas Umum (Menit) Akses Sekolah (Menit)

☐ Pusat Perbelanjaan:

☐ Rumah Sakit:

☐ Terminal Bis:

Gambar 4.3 Antarmuka Memasukkan Jarak Fasilitas Umum

Cari berdasar Karakteristik

Karakteristik Property Akses Fasilitas Umum (Menit) Akses Sekolah (Menit)

☐ Sekolah dasar:

☐ Sekolah Menengah Pertama:

☐ Sekolah Menengah:

Gambar 4.4 Antarmuka Memasukkan Jarak Sekolah

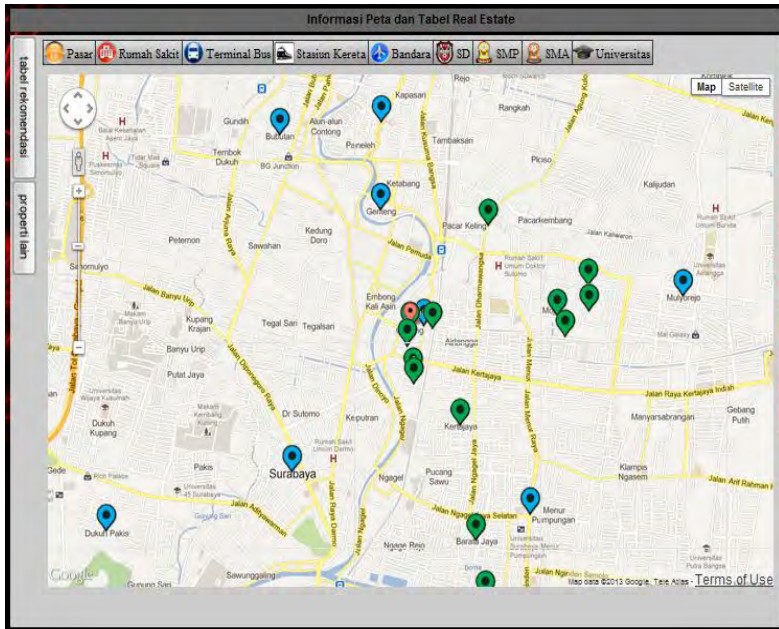
Ketika pengguna sudah memasukkan karakteristik dan meminta rekomendasi *real estate*, maka sistem akan melakukan proses untuk memberikan rekomendasi. Hasil rekomendasi akan ditampilkan pada halaman rekomendasi dalam berbagai macam cara yaitu dalam bentuk daftar seperti yang ditampilkan pada Gambar 4.5, dalam bentuk tabel seperti yang ditampilkan pada Gambar 4.6, dan dalam bentuk peta seperti yang ditampilkan pada Gambar 4.7.



Gambar 4.5 Antarmuka Daftar *Real Estate*



Gambar 4.6 Antarmuka Tabel *Real Estate*



Gambar 4.7 Antarmuka Informasi Peta

4.3.2 Implementasi Kode Program Pencarian Rekomendasi *Real Estate*

Pada implementasi kode program ini, untuk memenuhi arsitektur MVC maka implementasi kode program dibedakan menjadi tiga yaitu implementasi *view*, implementasi *controller*, dan implementasi *model*.

4.3.2.1 Implementasi *View* Kode Program Pencarian Rekomendasi *Real Estate*

Untuk kode program pada *view* pencarian rekomendasi *real estate*, ditangani oleh kelas `HalamanUtama`. Kelas ini merupakan penghubung kelas `Rekomendasi` dan antarmuka

HalamanUtama. Melalui kelas ini, pengguna meminta data dari *server* kemudian memberikan data pada antarmuka untuk ditampilkan. Detail implementasi kelas ini ditunjukkan pada Gambar 4.8 yaitu untuk memasukkan karakteristik, Gambar 4.9 untuk memasukkan jarak dengan fasilitas umum, Gambar 4.10 untuk memasukkan jarak dengan sekolah. Untuk menjalankannya ditangani oleh fungsi-fungsi untuk memasukkan nilai kriteria pada Gambar 4.11.

```
<div class="menuWrapper">
    <div id="menuTabs" style="border-style: none; border-
width: 0px; background: #333333;"> <ul >
        <li><a href="#tabs-1">Karakteristik
Property</a></li>
        <li><a href="#tabs-2">Akses Fasilitas Umum
(Menit)</a></li>
        <li><a href="#tabs-3">Akses Sekolah
(Menit)</a></li>
    </ul>
<div id="tab-1">...</div>
<div id="tab-2">...</div>
<div id="tab-3">...</div>
```

Gambar 4.8 Kode Program Memasukkan Karakteristik

```
<div id="tabs-2" style="background-color:#CCCCC; height: 100px;
overflow-x: scroll; overflow-y: hidden">
<div style="display: inline-block; width:1500px" >
<div class="menu2"><input id="depstoreCB" type="checkbox"
onclick="enableDepStore()"/>Pusat Perbelanjaan: <br />
<div id="sliderdepstore" style="width:200px; "
class="slider"></div>
<asp:TextBox ID="depstoreVal" runat="server" AutoPostBack="True"
Height="22px" Width="35px"
CssClass="sliderContainer" Enabled="false"/>
</div>
```

Gambar 4.9 Kode Program Memasukkan Jarak dengan Fasilitas Umum

```

<div id="tabs-3" style="background-color:#CCCCC; height: 100px;
overflow-x: scroll; overflow-y: hidden">
<div style="display: inline-block; width:1500px" >
<div class="menu2">
    <input id="elementaryCB" type="checkbox"
onclick="enableElementary()"/>
        Sekolah dasar:<br />
        <div id="slide-elementary"
style="width:200px; " class="slider"></div>
            <asp:TextBox ID="elementaryVal"
runat="server" AutoPostBack="True" Height="22px"
Width="35px"
CssClass="sliderContainer" Enabled="false"/>
        </div>

```

Gambar 4.10 Kode Program untuk Memasukkan Jarak dengan Sekolah

```

protected void setMapCondition(object sender, EventArgs e)
{
    Session["area"] = DropDownList1.SelectedValue.ToString();
    Session["Price"] = DropDownList2.SelectedValue.ToString();
    Session["room"] = DropDownList3.SelectedValue.ToString();
    Session["volArea"] =
DropDownList4.SelectedValue.ToString();
    Session["depstore"] = depstoreVal.Text;
    Session["hospital"] = hospitalVal.Text;
    Session["bus"] = busVal.Text;
    Session["train"] = trainVal.Text;
    Session["airport"] = airportVal.Text;
    Session["elementary"] = elementaryVal.Text;
    Session["junior"] = juniorVal.Text;
    Session["senior"] = highVal.Text;
    Session["university"] = universityVal.Text;
}

```

Gambar 4.11 Kode Program untuk Memasukkan Nilai Kriteria

Untuk menampilkan hasil rekomendasi, akan ditangani oleh kelas `HalamanRekomendasi` dan dibantu oleh kode *JavaScript* untuk menampilkan peta dari *Google Maps*. Detail implementasi kelas ini ditunjukkan pada Gambar 4.12 untuk menampilkan

rekomendasi pada tabel. Gambar 4.13 adalah implementasi kode program untuk menampilkan daftar rekomendasi *real estate*. Sedangkan Gambar 4.14 adalah implementasi menampilkan rekomendasi pada peta dengan dibantu kode *JavaScript* yang ditunjukkan pada Gambar 4.15.

```
public void getRecom(string max, string min)
{
    if (Session["listRecom"] == null)
    {
        Session["listRecom"] = "SELECT
recom.Id, recom.harga, recom.Address, recom.Rank " +
        "FROM (SELECT distinct re.Id, re.harga,
re.Address, ROW_NUMBER() OVER (ORDER BY re.Id DESC) As \"Rank\" \"
+
        "FROM RealEstate re)recom ";
    }
    RecomLogic rl = new
RecomLogic(Session["listRecom"].ToString(), max, min);
    propertyys = rl.getRecomendation();
    indexMax = propertyys.Count().ToString();
}
```

Gambar 4.12 Kode Program Menampilkan Daftar Rekomendasi *Real Estate*

```
public void getRecom(string max, string min)
{
    RealEstate.SelectCommand =
re.getRecomendationCommandStr("SELECT distinct re.Id, re.harga,
re.Address FROM RealEstate re ", Session["depstore"].ToString(),
Session["hospital"].ToString(), RealEstate3.SelectCommand =
re.getRecomendationCommandStr("SELECT distinct Id, harga, Address,
coordinate from RealEstate where Id not in (SELECT re.Id FROM
...
    if (RealEstate != null)
    {
        GridView1.DataSourceID = RealEstate.ID;
        GridView1.DataBind();
    }
    if (RealEstate3 != null)
    {
        GridView2.DataSourceID = RealEstate3.ID;
        GridView2.DataBind();
    }
}
```

Gambar 4.13 Kode Program Menampilkan Rekomendasi pada Tabel

```

public string getCoorRealEstate()
{
    string result = "";
    this.area = DropDownList1.SelectedValue.ToString();
    string queryString = re.getRecomendationCommandStr("SELECT re.Id,
re.coordinate FROM RealEstate re ",
Session["depstore"].ToString(), Session["hospital"].ToString(),
Session["bus"].ToString(), Session["train"].ToString(),
Session["airport"].ToString(), Session["area"].ToString(),
Session["Price"].ToString, string coorArea =
conn.selectData(queryString);
        result = coorArea;
    }
    return result;
}

```

**Gambar 4.14 Kode Program Menampilkan Rekomendasi
Real Estate pada Peta**

```

function showMap(centerCoor) {
    var lat; var lng;
    centerCoor = centerCoor.replace("&", "");
    latLng = centerCoor.split(",");
    lat = getLat(latLng[0]); lng = getLng(latLng[1]);
    var myLatitude = new google.maps.LatLng(lat, lng);
    var myOptions = {
        zoom: 14,
        center: myLatitude,
        mapTypeId: google.maps.MapTypeId.ROADMAP};
    map = new
    google.maps.Map(document.getElementById("map_canvas"), myOptions);
    var mymarker = new google.maps.Marker(
        { position: myLatitude,
          title: "Center",
          clickable: true,
          map: map });
    var infowindow = new google.maps.InfoWindow({
        content: 'Tengah Area'});
    google.maps.event.addListener(mymarker, 'click',
function () { infowindow.open(map, mymarker); });
}

```

**Gambar 4.15 Kode JavaScript untuk Menghubungkan dengan
Google Maps API**

4.3.2.2 Implementasi *Controller* Kode Program Pencarian Rekomendasi *RealEstate*

Untuk kode program pada *controller*, pencarian rekomendasi *real estate* ditangani oleh kelas Rekomendasi dan PropertyLogic. Kelas Rekomendasi ini digunakan untuk mencari rekomendasi *real estate* sesuai dengan kriteria yang dimasukkan pengguna. Sedangkan kelas PropertyLogic digunakan untuk menghubungkan dengan kelas model. Melalui kelas ini data dari pengguna akan diolah untuk dihubungkan dengan model guna memperoleh rekomendasi data. Detail implementasi kelas Rekomendasi ini ditunjukkan pada Gambar 4.16.

```
public class Rekomendasi
{
    DBConnection conn;
public Rekomendasi()
{
    conn = new DBConnection();
}
    public DataSet getRekomendasi(string area, string harga,
string room, string volume){
...}
public string getRekomendasiCommandStr(string rec, string
market, string hospital, string bus, string train, string airport,
string area, string harga, string room, string volume){
...}
public string getSubCmdStr(string market, string hospital, string
bus, string train, string airport)
}}
```

Gambar 4.16 Kode Kelas Rekomendasi

4.3.2.3 Implementasi Model Kode Program Pencarian Rekomendasi *Real Estate*

Pada implementasi kode program model dalam kasus pencarian rekomendasi *real estate* ini, terdapat dua kelas yang dapat

melakukan akses langsung dengan basis data yaitu kelas `RecomLogic` dan kelas `RealEstate`. Kelas `RecomLogic` dapat mengambil langsung tabel *real estate* sebagai daftar rekomendasi dari basis data. Sedangkan kelas `RealEstate` dapat mengakses informasi sebuah entitas *real estate*. Detail implementasi kelas `RecomLogic` ini ditunjukkan oleh Gambar 4.17. Detail implementasi kelas `RealEstate` ini ditunjukkan pada Gambar 4.18.

```
public class RecomLogic
{
    List<Real Estate> Recom;
    private SqlConnection conn;
    private string query;
    public string Query
    {get { return query; }
     set { query = value; }}
    public RecomLogic(string query, string max, string min)
    {
        this.conn = new SqlConnection("Data Source=KIKI-
PC;Initial Catalog=RealEstateDB;Integrated Security=True");
        public List<Real Estate> getRecomendation()
        {
            conn.Open();
            SqlDataReader reader = null;
            SqlCommand command = new SqlCommand(query, conn);
            reader
            command.ExecuteReader(System.Data.CommandBehavior.CloseConnection)
            ;
            while (reader.Read())
            {
                string id = ""; int harga = 0; string Address= "";
                int index = 0; int lim = reader.FieldCount;
                id += reader.GetValue(0).ToString();
                ...
                return Recom;
            }
        }
    }
}
```

Gambar 4.17 Kode Program Kelas PropertyLogic

```

public class Real Estate
{
...
    public Real Estate(string id, string address, string
price, int index)
    {
        this.id = id;
        this.address = address;
        this.price = price;
        this.index = index;
    }

    public Real Estate()
    {
        // TODO: Complete member initialization
    }

    public void InformasiProperty(string id, string address,
string harga, string Gambar, string koordinat, string stfkt,
string luasTanah, string luasBangunan, string jumlahKamar, string
href)
    {
        this.Id = id;
        this.Address = address;
        this.Price = harga;
        this.Gambar = Gambar;
        this.Koordinat = koordinat;
        this.Sertifikat = stfkt;
        this.LuasTanah = luasTanah;
        this.LuasBangunan = LuasBangunan;
        this.JumlahKamar = jumlahKamar;
        this.Href = href;
        conn.Open();
        SqlDataReader reader = null;
        SqlCommand command = new SqlCommand(query, conn);
        reader =
command.ExecuteReader(Sistem.Data.CommandBehavior.CloseConnection)
;
    }
}

```

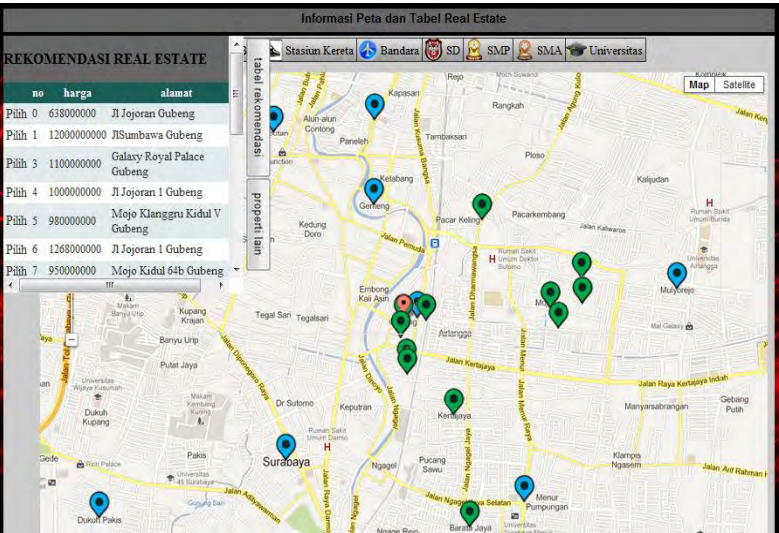
Gambar 4.18 Kode Program Kelas RealEstate

4.4 Implementasi Memilih Real Estate

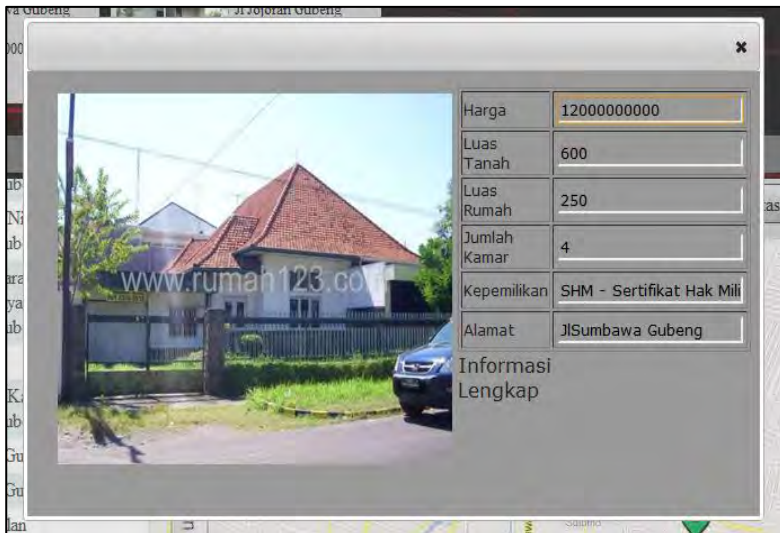
Pada subbab ini akan dijelaskan tentang implementasi antarmuka pengguna dan kode program untuk memenuhi kasus penggunaan memilih *real estate*.

4.4.1 Implementasi Antarmuka Memilih Real Estate

Antarmuka untuk memilih *real estate* ditunjukkan oleh Gambar 4.19. Pengguna dapat memilih *real estate* dengan cara memilih *real estate* yang tersedia baik yang terdapat pada peta, daftar *real estate*, maupun tabel *real estate*. Sistem akan mengirimkan hasil berupa informasi *real estate* yang dipilih seperti yang terdapat pada Gambar 4.20.



Gambar 4.19 Antarmuka Pengguna Memilih Real Estate



Gambar 4.20 Antarmuka Sistem Memberikan Informasi *Real Estate*

4.4.2 Implementasi Kode Program Memilih *Real Estate*

Pada implementasi kode program memilih *real estate*, untuk memenuhi arsitektur MVC, maka implementasi kode program dibedakan menjadi tiga yaitu implementasi *view*, implementasi *controller*, dan implementasi model.

4.4.2.1 Implementasi *View* Kode Program Pencarian Rekomendasi *Real Estate*

Untuk kode program pada *view* pencarian rekomendasi *real estate* ditangani oleh kelas `HalamanRekomendasi`. Detail implementasi kelas ini ditunjukkan oleh Gambar 4.21 yaitu untuk memilih *real estate*. Sedangkan informasi yang ditampilkan oleh

sistem ditangani juga oleh kelas `HalamanRekomendasi` yang ditunjukkan pada Gambar 4.22.

```
<div class="infoWrap">
  
  <div id="propertyInforWrap" class="floatleft">
    <%
      string q = "fasfafa";
      q = Request.QueryString["q"];
      string q2 = "afasfsafa";
      q2 = Request.QueryString["q2"];
      if (q != "" && q2 != "" && q != null && q2 != null)
      {
        getRecom(q, q2);
      }

      for (int i = 0; i < propertys.Count; i++)
      {
        <div class="propertiInfo"
id="<%=propertys[i].Id.ToString()%">
onclick="divInfo('<%=propertys[i].Id.ToString()%">')">
          <img id="gmbr"<%=i.ToString()%">
class="listPropertyImg"
src="Handler1.ashx?Id=<%=propertys[i].Id.ToString()%">
width="100px" height="100px"/>
          <div class="infoDetail">Alamat<br
/><%=propertys[i].Address%> <br />Harga <br />Rp.
<%=propertys[i].Price.ToString()%"></div>
          </div>
          <%
        }
      <%>
    </div>
```

Gambar 4.21 Kode Program Memilih *Real Estate*

```
<div id="dialog-modal">
  <div id="infoContainer" title="Informasi Properti">
    
  </div>
  <div class="infoDetail"> ... </div>
```

Gambar 4.22 Kode Program Menampilkan *Real Estate*

4.4.2.2 Implementasi *Controller* Kode Program Pencarian Rekomendasi *Real Estate*

Pada kode program *controller* untuk pengguna dapat memilih *real estate*, ditangani oleh kelas *HalamanRekomendasi* yang ditunjukkan pada potongan kode Gambar 4.23. Untuk sistem yang dapat menampilkan informasi *real estate*, juga ditangani oleh kelas *HalamanRekomendasi* yang ditunjukkan pada potongan kode Gambar 4.24.

```
public void getRecom(string max, string min)
{
    if (Session["listRecom"] == null)
    {
        Session["listRecom"] = "SELECT
recom.Id, recom.harga, recom.Address, recom.Rank " +
        "FROM (SELECT distinct re.Id, re.harga,
re.Address, ROW_NUMBER() OVER (ORDER BY re.Id DESC) As \"Rank\" \"
+
        "FROM RealEstate re)recom ";
    }
    RecomLogic rl = new
    RecomLogic(Session["listRecom"].ToString(), max, min);
    propertyys = rl.getRecomendation();
    indexMax = propertyys.Count().ToString();
}
```

Gambar 4.23 Kode Program Menampilkan Pilihan *Real Estate*

```
public string getRealEstateInfo()
{
    string result = "";
    result = conn.selectData("SELECT Id, coordinate,
harga, luastanah, luasbangunan, jumlahKamar, Sertifikat, Address,
Href FROM RealEstate WHERE Id = '" +
Session["IdRealEstate"].ToString() + "'");
    return result; }
```

Gambar 4.24 Kode Program Menampilkan Informasi *Real Estate*

4.4.2.3 Implementasi Model Kode Program Pencarian Rekomendasi *Real Estate*

Pada implementasi kode program model dalam kasus pencarian rekomendasi *real estate* ini terdapat dua kelas yang dapat melakukan akses langsung dengan basis data yaitu kelas *PropertyLogic* dan kelas *Property*. Kelas *PropertyLogic* mengambil daftar *real estate* yang digunakan sebagai pilihan oleh pengguna. Sedangkan kelas *Property* digunakan untuk mengambil informasi secara lengkap mengenai *Property*. Potongan kode program kelas *PropertyLogic* ditampilkan pada Gambar 4.25. Sedangkan potongan kode program kelas *RealEstate* ditampilkan pada Gambar 4.26.

```
public class LogikaProperty
{
    private SqlConnection conn;
    private Real Estate real estate = new Real Estate();
    public LogikaProperty()
    {
        this.conn = new SqlConnection("Data Source=KIKI-PC;Initial Catalog=RealEstateDB;Integrated Security=True");
    }
    public Real Estate getInformation(string id)
    {
        string queryProperty = "SELECT Id, Address, harga, coordinate, Sertifikat, luastanah, luasbangunan, jumlahKamar, Href FROM RealEstate WHERE Id='" + id + "'"; conn.Open();
        SqlDataReader reader = null;
        SqlCommand command = new SqlCommand(queryProperty, conn);
        reader = command.ExecuteReader(System.Data.CommandBehavior.CloseConnection)
        while (reader.Read())
        {
            address = reader.GetValue(1).ToString();
            ...
            real estate.InformasiProperty(id, address, harga, Gambar, koordinat, stfkt, luasTanah, luasBangunan, jumlahKamar, href);
        }
        return real estate;
    }
}
```

Gambar 4.25 Kode Program Kelas *PropertyLogic*

```

public class Real Estate
{
...
    public Real Estate(string id, string address, string
price, int index)
    {
        this.id = id;
        this.address = address;
        this.price = price;
        this.index = index;
    }
    public Real Estate()
    {
        // TODO: Complete member initialization
    }
    public void InformasiProperty(string id, string address,
string harga, string Gambar, string koordinat, string stfkt,
string luasTanah, string luasBangunan, string jumlahKamar, string
href)
    {
        this.Id = id;
        this.Address = address;
        this.Price = harga;
        this.Gambar = Gambar;
        this.Koordinat = koordinat;
        this.Sertifikat = stfkt;
        this.LuasTanah = luasTanah;
        this.LuasBangunan = LuasBangunan;
        this.JumlahKamar = jumlahKamar;
        this.Href = href;
        conn.Open();
        SqlDataReader reader = null;
        SqlCommand command = new SqlCommand(query, conn);
        reader =
command.ExecuteReader(Sistem.Data.CommandBehavior.CloseConnection)
;
    }
}

```

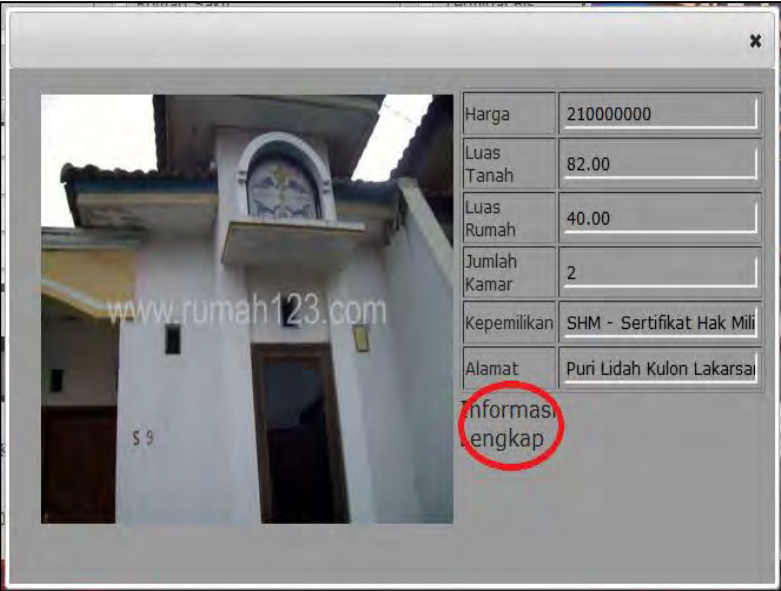
Gambar 4.26 Kode Program Kelas Property

4.5 Implementasi Melihat Informasi *Real Estate*

Pada subbab ini akan dijelaskan tentang implementasi antarmuka pengguna dan kode program untuk memenuhi kasus penggunaan melihat informasi *real estate*.

4.5.1 Implementasi Antarmuka Melihat Informasi *Real Estate*

Antarmuka untuk dapat melihat *real estate* ditunjukkan oleh Gambar 4.27. Pengguna dapat melihat *real estate* dengan cara memilih *real estate*, kemudian menekan tombol informasi selengkapnya. Sistem akan memproses aktivitas pengguna, kemudian menampilkan halaman yang ditunjukkan Gambar 4.28.



Harga	210000000
Luas Tanah	82.00
Luas Rumah	40.00
Jumlah Kamar	2
Kepemilikan	SHM - Sertifikat Hak Mili
Alamat	Puri Lidah Kulon Lakarsa
Informasi selengkapnya	

Gambar 4.27 Antarmuka Melihat *Real Estate*

The screenshot displays a web application interface for real estate recommendations. At the top, the title 'REKOMENDASI REAL ESTATE' is prominently displayed. Below the title, there is a large image of a house with the watermark 'www.rumah123.com'. To the right of the image, a table provides details about the property:

Alamat	Jl Nias Gubeng
Harga	2950000000
Luas Tanah	570 m2
Luas Rumah	m2
Jumlah Kamar	4
Kepemilikan	SHM - Sertifikat Hak Milik

On the right side of the interface, there is a vertical menu with icons and labels for various locations: Pasar, Rumah Sakit, Terminal Bus, Stasiun Kereta, Bandara, SD, SMP, and SMA. Below the property details, there is a table listing recommended properties:

no	nama	alamat
Pilih 0	Universitas Airlangga Mojo 1	Mojo, Surabaya
Pilih 43	Universitas Katolik Widya Mandala	Jl. Dinoyo No. 44 Keputran Tegalsari Surabaya Jawa Timur
Pilih 44	Universitas Widya Mandala	Jalan Mojopahit, Surabaya
Pilih 45	Universitas Teknologi Surabaya	Jalan Ngagel, Surabaya
Pilih 46	Universitas	

At the bottom of the interface, there is a map showing the location of the property and surrounding areas. The map includes labels for various landmarks and roads, such as 'Jalan Nias Gubeng', 'Jalan Mojopahit', and 'Jalan Ngagel'.

Gambar 4.28 Antarmuka Menampilkan Informasi *Real Estate*

4.5.2 Implementasi Kode Program Melihat Informasi *Real Estate*

Pada implementasi kode program melihat informasi *real estate*, untuk memenuhi arsitektur MVC, maka implementasi kode program dibedakan menjadi tiga, yaitu implementasi *view*, implementasi *controller*, dan implementasi model.

4.5.2.1 Implementasi *View* Kode Program Melihat Informasi *Real Estate*

Untuk kode program pada *view* melihat informasi *real estate*, ditangani oleh antarmuka `HalamanRekomendasi`. Potongan kode program halaman ini ditunjukkan pada Gambar 4.29, yaitu tampilan untuk dapat melihat informasi *real estate*. Sedangkan informasi yang ditampilkan oleh sistem, ditangani oleh antarmuka

HalamanInfoProperty yang ditunjukkan pada potongan kode program Gambar 4.30.

```
<div id="dialog-modal">
  <div id="infoContainer" title="Informasi Properti">
    
  </div>
  <div class="infoDetail">
    <table class="sample">...</table>
    <a id="infoLengkap" href="#"> Informasi Lengkap</a>
  </div>
</div>
```

**Gambar 4.29 Kode Program untuk Melihat Informasi Detail
*Real Estate***

```
<div class="titleWrap">
  
  <a href="Default.aspx">
</div>
<div class="headertheme" style="height: 320px">
  <div id="infoContainer" title="Informasi Properti" style="height:300px; width:380px">
    
  </div>
  <div class="infoDetail">...</div>
  <div style="display: block; float: right; background-color: #FFFFFF; margin-top: 9px; margin-right: 10px;">
    <div>
      <div id="InfoMenuFasilitas">...</div>
    </div>
  </div>
</div>
<div id="mainHome">
  <div id="sukasuka">...</div>
```

Gambar 4.30 Kode Program Menampilkan Informasi *Real Estate*

4.5.2.2 Implementasi *Controller* Kode Melihat Informasi *Real Estate*

Pada kode program *controller*, agar pengguna dapat melihat informasi *real estate* ditangani oleh kelas HalamanRekomendasi yang ditunjukkan oleh potongan kode Gambar 4.31. Sedangkan kode program untuk sistem dapat menampilkan informasi *real estate*, ditangani oleh kelas

HalamanInfoProperty yang ditunjukkan oleh potongan kode pada Gambar 4.32.

```
protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
{
    string selectedId = GridView1.SelectedRow.Cells[1].Text;
    Session["IdRealEstate"] = selectedId;
    string result = "";
    result = conn.selectData("SELECT Id, coordinate, harga, luastanah, luasba
    ScriptManager.RegisterClientScriptBlock(this, this.GetType(), "lakukan",
}

protected void GridView2_SelectedIndexChanged(object sender, EventArgs
{
    string selectedId = GridView2.SelectedRow.Cells[1].Text;
    Session["IdRealEstate"] = selectedId;
    string result = "";
    result = conn.selectData("SELECT Id, coordinate, harga, luastanah, luasba
    ScriptManager.RegisterClientScriptBlock(this, this.GetType(), "lakukan",
}
```

Gambar 4.31 Kode Program Melihat Informasi *Real Estate*

```
private string id = "";
public Property property;
protected void Page_Load(object sender, EventArgs e)
{
    if (!string.IsNullOrEmpty(Request.QueryString["Id"]))
    {
        Session["Id"] = id;
        this.id = Request.QueryString["Id"].ToString();
    }
    LogikaProperty lp = new LogikaProperty();
    property = new Property();
    property = lp.getInformation(id);
    string g = property.Gambar;
    Page.ClientScript.RegisterStartupScript(Page.GetType(), "myKey", "initialize()", true);
}
```

Gambar 4.32 Kode Program Menampilkan Informasi *Real Estate*

4.5.2.3 Implementasi Model Kode Program Melihat Informasi *Real Estate*

Pada implementasi kode program model dalam melihat informasi *real estate*, untuk dapat melihat dan menampilkan informasi *real estate* ditangani oleh satu kelas yaitu kelas `RealEstate`. Hal ini dikarenakan kedua proses tersebut hanya melakukan akses pada satu entitas *real estate* saja. Adapun potongan kode program kelas `RealEstate` ditampilkan pada Gambar 4.33.

```
public class Real Estate
{
    public Real Estate(string id, string address, string
price, int index)
    {
        this.id = id; this.address = address;
        this.price = price; this.index = index;
    }
    public Real Estate()
    {
    }
    public void InformasiProperty(string id, string address,
string harga, string Gambar, string koordinat, string stfkt,
string luasTanah, string luasBangunan, string jumlahKamar, string
href)
    {
        this.LuasBangunan = LuasBangunan;
        this.JumlahKamar = jumlahKamar;
        this.Href = href;
        conn.Open();
        SqlDataReader reader = null;
        SqlCommand command = new SqlCommand(query, conn);
        reader =
command.ExecuteReader(Sistem.Data.CommandBehavior.CloseConnection)
;
    }
}
```

Gambar 4.33 Kode Program Kelas `RealEstate`

4.6 Melihat Informasi Fasilitas Umum

Pada subbab ini akan dijelaskan tentang implementasi antarmuka pengguna dan kode program untuk memenuhi kasus penggunaan melihat informasi fasilitas umum.

4.6.1 Implementasi Antarmuka Melihat Informasi Fasilitas Umum

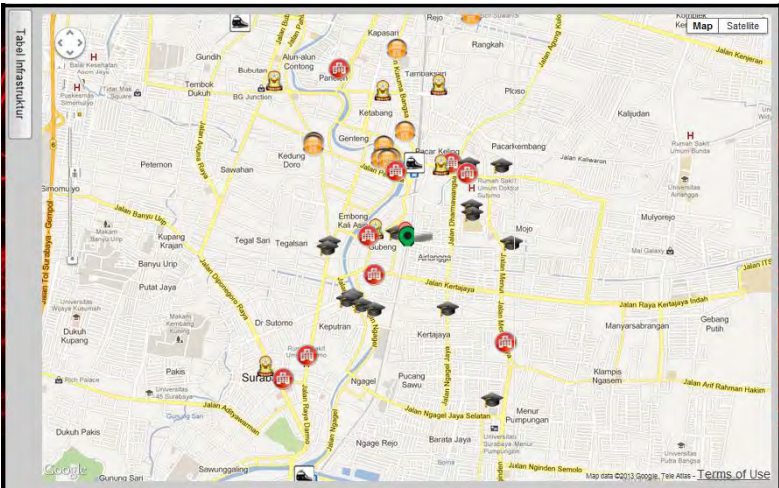
Antarmuka untuk dapat melihat informasi fasilitas umum ditunjukkan oleh Gambar 4.34 dengan cara memilih tipe fasilitas umum pada menu fasilitas umum. Selain itu, pengguna juga dapat melihat daftar fasilitas umum yang terdapat di sekitar *real estate* dalam bentuk tabel dengan cara menekan tombol Tabel Infrastruktur yang ditunjukkan pada Gambar 4.35. Ketika pengguna memilih tipe fasilitas umum, maka sistem akan memberikan informasi mengenai fasilitas umum dalam bentuk peta yang ditunjukkan Gambar 4.36.



Gambar 4.34 Antarmuka Memilih Fasilitas Umum

	no	nama	alamat
Pilih	0	Universitas Airlangga Mojo 1	Mojo, Surabaya
Pilih	43	Universitas Katolik Widya Mandala	Jl. Dinoyo No. 44 Keputran Tegalsari Surabaya Jawa Timur
Pilih	44	Universitas Widya Mandala	Jalan Mojopahit, Surabaya
Pilih	45	Universitas Teknologi Surabaya	Jalan Ngagel, Surabaya

Gambar 4.35 Antarmuka Melihat Daftar Fasilitas Umum



Gambar 4.36 Antarmuka Menampilkan Informasi Fasilitas Umum

4.6.2 Implementasi Kode Program Melihat Informasi Fasilitas Umum

Pada implementasi kode program melihat informasi fasilitas umum untuk memenuhi arsitektur MVC, maka implementasi kode program dibedakan menjadi tiga yaitu implementasi *view*, implementasi *controller*, dan implementasi model.

4.6.2.1 Implementasi View Kode Program Melihat Informasi Fasilitas Umum

Untuk kode program pada *view* memilih informasi fasilitas umum ditangani oleh antarmuka HalamanInfoProperty. Potongan kode halaman ini ditunjukkan pada Gambar 4.37 yaitu, tampilan untuk dapat melihat informasi fasilitas umum. Sedangkan informasi fasilitas umum yang tertera pada tabel dan ditampilkan oleh sistem, ditangani oleh antarmuka HalamanInfoProperty yang ditunjukkan pada potongan kode program Gambar 4.38. Untuk menampilkan informasi berupa peta, ditunjukkan oleh potongan kode Gambar 4.39.

```
<div style="display: block; float: right; background-color: #FFFFFF; margin-top: 9px; margin-right: 10px" >
  <div>
    <div id="InfoMenuFasilitas">
      <div onclick="showMarket()"> <p>Pasar</p> </div>
      <div onclick="showHospital()"> <p>Rumah Sakit</p> </div>
      <div onclick="showBusStation()"> <p>Terminal Bus</p> </div>
      <div onclick="showTrainStation()"> <p>Stasiun Kereta</p> </div>
      <div onclick="showAirport()"> <p>Bandara</p> </div>
      <div onclick="showElementary()"> <p>SD</p> </div>
      <div onclick="showJunior()"> <p>SMP</p> </div>
      <div onclick="showSenior()"> <p>SMA</p> </div>
      <div onclick="showUniversity()"> <p>Universitas</p> </div>
      <div onclick="showUniversity()"> <p>Tampilkan Semua</p> </div>
    </div>
  </div>
</div>
```

Gambar 4.37 Kode Program Memilih Fasilitas Umum

```

<div id="Infrastruktur" class="subSukasuka" >
  <asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False"
    BackColor="White" BorderColor="#999999" BorderStyle="None" BorderWidth="1"
    CellPadding="3" DataSourceID="InfrastrukturList" GridLines="Vertical">
    <AlternatingRowStyle BackColor="#DCDCDC" />
    <Columns>...</Columns>
    <FooterStyle BackColor="CCCCCC" ForeColor="Black" />
    <HeaderStyle BackColor="#000084" Font-Bold="True" ForeColor="White" />
    <PagerStyle BackColor="#999999" ForeColor="Black" HorizontalAlign="Center" />
    <RowStyle BackColor="#EEEEEE" ForeColor="Black" />
    <SelectedRowStyle BackColor="#008A8C" Font-Bold="True" ForeColor="White" />
    <SortedAscendingCellStyle BackColor="#F1F1F1" />
    <SortedAscendingHeaderStyle BackColor="#0000A9" />
    <SortedDescendingCellStyle BackColor="#CAC9C9" />
    <SortedDescendingHeaderStyle BackColor="#000065" />
  </asp:GridView>
  <asp:SqlDataSource ID="InfrastrukturList" ...>...</asp:SqlDataSource>
</div>

```

Gambar 4.38 Kode Program Melihat Tabel Fasilitas Umum

```

function initialize() {
  var coorReal = "<%=property.Koordinat%>".toString();
  alert(coorReal);
  showMap(coorReal);
  showAll();
  var index = 0;
  arrcoorRealEstate = coorReal.split("&");
  while (arrcoorRealEstate[index] != "" && arrcoorRealEstate[index] != null) {
    marker[index] = setMarker(arrcoorRealEstate[index], "images/greenPin.png");
    index++;
  }
}

```

Gambar 4.39 Kode Program Menampilkan Informasi Fasilitas Umum

4.6.2.2 Implementasi *Controller* Kode Program Melihat Informasi Fasilitas Umum

Untuk kode program pada *controller* agar pengguna dapat melihat tabel fasilitas umum, ditangani oleh kelas HalamanInfoProperty yang ditunjukkan oleh potongan kode Gambar 4.40. Sedangkan kode program agar sistem dapat menampilkan informasi *real estate*, ditangani oleh kelas

HalamanInfoProperty yang ditunjukkan oleh potongan kode pada Gambar 4.41.

```

InfrastrukturList.SelectCommandType =qlDataSourceCommandType.Text;
        InfrastrukturList.SelectCommand = "SELECT pubi.id As
'no', pubi.name As 'nama', pubi.address As 'alamat'" +
        " FROM PublicInfrastruktur pubi RIGHT JOIN (select
idInfrastruktur from RealEstateProperties where idRealEstate = '"+
id +"')infra ON pubi.id = infra.idInfrastruktur";
if(id != ""){
GridView1.DataSourceID = InfrastrukturList.ID;
GridView1.DataBind();
}

```

Gambar 4.40 Kode Program Menampilkan Tabel Fasilitas Umum

```

protected void Page_Load(object sender, EventArgs e)...

public string getAllMarket()...
public string getAllHospital()...
public string getAllTrainStation()...
public string getAllBusStation()...
public string getAllAirport()...
public string getAllElementary()...
public string getAllJunior()...
public string getAllSenior()...
public string getAllUniversity()
{
    string result = "";
    try
    {
        string coorArea = conn.selectData("SELECT pubi.name, pubi.coordinate" +
        " FROM PublicInfrastruktur pubi RIGHT JOIN (select idInfrastruktur from
        result = coorArea;
        return result;
    }
    catch
    {
        return "error";
    }
}

```

Gambar 4.41 Kode Program Menampilkan Informasi Fasilitas Umum

4.6.2.3 Implementasi *Controller* Kode Program Melihat Informasi Fasilitas Umum

Pada implementasi kode program model dalam melihat informasi fasilitas umum, untuk dapat melihat dan menampilkan informasi fasilitas umum ditangani oleh dua kelas yaitu kelas *InfrastructurSearch* dan kelas *Infrastructur*. Kelas *InfrastructurSearch* digunakan untuk mencari daftar fasilitas umum yang berada di sekitar *real estate*. Sedangkan kelas *Infrastructur* digunakan untuk dapat mengambil informasi mengenai fasilitas umum itu sendiri. Adapun potongan kode program kelas *InfrastructurSearch* ditampilkan pada Gambar 4.42 dan kelas *Infrastructur* ditampilkan pada Gambar 4.43.

```
public string selectInfrastructur(string selectQuery)
{
    SqlDataReader reader = null;
    string result = "";
    try
    {
        conn.Open();
        SqlCommand command = new SqlCommand(selectQuery, conn);
        reader = command.ExecuteReader(System.Data.CommandBehavior.CloseConnection);

        while (reader.Read())
        {
            int index = 0;
            int lim = reader.FieldCount;
            while(index < lim)
            {
                if(index == 0)
                    result += reader.GetValue(index).ToString();
                else
                    result += ";" + reader.GetValue(index).ToString();
                index++;
            }
            result += "&";
        }
        return result;
    }
    catch (Exception ex)
    {
    }
    finally
    {
    }
}
```

Gambar 4.42 Kode Program Kelas *InfrastructurSearch*

```

namespace RealEstate.Model
{
    public class FasilitasUmum
    {
        private string nama;

        public string Nama[...]
        private string koordinat;

        public string Koordinat[...]
        private string alamat;

        public string Alamat[...]
        private string id;

        public string Id[...]
        private string koordinat;

        public string Koordinat[...]

        public FasilitasUmum(string id, string nama, string koordinat, string alamat)
        {
            this.Id = id;
            this.Nama = nama;
            this.Coordinat = koordinat;
            this.Alamat = alamat;
        }

        public void setInfo()[...]
    }
}

```

Gambar 4.43 Kode Program Kelas Infrastructur

4.7 Implementasi Proses dalam Sistem

Subbab ini menjelaskan mengenai implementasi dari lapisan model dan *controller* yang berjalan pada sistem *server*. Lapisan model sistem *server* ini berisi kelas-kelas yang digunakan untuk merepresentasikan data-data ke dalam basis data sehingga pengaksesan dan penyimpanan data yang berlangsung pada *server* dijalankan menggunakan kelas-kelas dalam lapisan ini. Sedangkan lapisan *controller* sistem *server* digunakan untuk melakukan pembaruan data, proses *mashup*, proses *crawling*, dan proses *parser* dengan data diambil dari situs sumber dan Google Maps.

4.7.1 Kelas DownloadData

Kelas ini berfungsi untuk menangani semua proses yang berhubungan dengan pengambilan data dari situs pusat. Kelas ini berfungsi untuk mengunduh semua data HTML, XML, dan gambar yang dibutuhkan sebagai sumber informasi perangkat lunak ini. Detail implementasi kelas ini ditunjukkan pada Kode Sumber C.1 Lampiran C.

4.7.2 Kelas Parser

Kelas ini berfungsi untuk menangani pengambilan data yang telah diunduh. Data yang telah diunduh akan diambil informasi-informasi yang bermanfaat untuk kemudian disimpan dalam basis data. Detail implementasi kelas ini ditunjukkan pada Kode Sumber C.2 Lampiran C.

4.7.3 Kelas DBConnection

Kelas ini berfungsi untuk menangani penyimpanan dan pengambilan data pada basis data. Kelas ini merupakan satu-satunya kelas yang berhubungan langsung dengan basis data. Semua fungsi yang membutuhkan untuk mengambil data dan menyimpan data harus melalui kelas ini terlebih dahulu. Detail implementasi kelas ini ditunjukkan pada Kode Sumber C.3 Lampiran C.

4.7.4 Kelas InfrastrukturFinder

Kelas ini berfungsi untuk mencari infrastruktur yang berada di sekitar *real estate*. Kelas ini berhubungan dengan kelas `GetData` untuk mengambil data dari *Google Maps* API dan berhubungan dengan `DBConnection` untuk menyimpan hasil pencarian. Detail implementasi kelas ini ditunjukkan pada Kode Sumber C.4 Lampiran C.

4.7.5 Kelas GMGeoCode

Kelas ini berfungsi untuk mengubah data alamat yang berbentuk teks menjadi data koordinat. Kelas ini berhubungan dengan kelas `GetData` untuk mengambil data dari Google *Geocoding* dan berhubungan dengan kelas model untuk menyimpan hasil pencarian. Detail implementasi kelas ini ditunjukkan pada Kode Sumber C.5 Lampiran C.

4.7.6 Kelas Rekomendasi

Kelas ini berfungsi untuk menangani proses pemberian rekomendasi untuk pengguna. Pada kelas ini permintaan dari pengguna akan diproses dengan menggunakan *query* sehingga didapat rekomendasi *real estate*. Detail implementasi kelas ini ditunjukkan pada Kode Sumber C.6 Lampiran C.

[Halaman ini sengaja dikosongkan]

BAB V

UJI COBA PERANGKAT LUNAK

Bab ini membahas uji coba dan evaluasi perangkat lunak rekomendasi pemilihan *real estate*. Perangkat lunak diuji coba dari segi fungsionalitas dan performa dengan berbagai macam skenario. Pengujian dilakukan sesuai dengan perancangan kasus penggunaan dan implementasi yang semuanya telah dijelaskan pada bab 3.

5.1 Lingkungan Uji Coba

Lingkungan uji coba merupakan komputer tempat uji coba sistem dilakukan. Spesifikasi lingkungan uji coba terbagi menjadi dua, yaitu lingkungan perangkat keras dan lingkungan perangkat lunak. Uji coba untuk perangkat lunak ini dilakukan dengan sebuah komputer *server* dan sebuah komputer klien. Spesifikasi masing-masing komputer yang digunakan pada uji coba adalah sebagai berikut:

- **Komputer *server***

Komputer ini berperan sebagai *web server* dan basis data *server*. Spesifikasi komputer *server* adalah sebagai berikut.

- ❖ *Server* basis data.
 - ❖ Spesifikasi perangkat keras.
 - Prosesor Intel Pentium Core 2 Duo @2.80Ghz.
 - RAM 2048MB.
 - ❖ Spesifikasi perangkat lunak.
 - Microsoft *Operating System*.
 - SQL *Server* 2008.
- ❖ *Server* aplikasi
 - ❖ Spesifikasi perangkat keras.
 - Intel Core 2 Duo CPU E7400 @2.80Ghz.
 - RAM 2048MB.

- ❖ Spesifikasi perangkat lunak.
 - Windows *Server* 2008 R2 64-bit.
 - IIS 7.0.
- Komputer pengguna

Komputer ini berperan sebagai pengakses perangkat lunak melalui sebuah perambah *web*. Spesifikasi komputer pengguna adalah sebagai berikut.

 - ❖ Spesifikasi perangkat keras.
 - Prosesor Intel Core 2 Duo 1,8 Ghz.
 - Memori 1 GB.
 - Harddisk 120 GB.
 - ❖ Spesifikasi perangkat lunak.
 - Microsoft Windows 7 SP1.
 - Internet Explorer 7, Mozilla Firefox 2.0.

5.2 Skenario Uji Coba

Pada subbab ini akan dijabarkan beberapa skenario uji coba pada perangkat lunak rekomendasi *real estate* ini. Seperti diketahui bahwa aplikasi perangkat lunak akan diuji pada dua sisi yaitu klien dan *server*. Masing-masing akan dilakukan uji coba secara mandiri berdasarkan metode kotak hitam.

Uji coba pada sisi klien akan dilakukan berdasarkan skenario yang akan dijelaskan pada subbab uji coba kebutuhan fungsional klien yang meliputi pengujian sebagai berikut:

1. Uji coba meminta rekomendasi.
2. Uji coba memilih hasil rekomendasi.
3. Uji coba memunculkan informasi *real estate*.
4. Uji coba menampilkan informasi fasilitas umum.

Uji coba pada sisi *server* difokuskan untuk beberapa proses utama terutama proses yang melibatkan pengambilan informasi.

Beberapa proses yang akan dijelaskan pada subbab uji coba proses *server* meliputi pengujian sebagai berikut:

1. Uji coba mengunduh informasi HTML dari situs sumber.
2. Uji coba mengunduh dan mengolah data XML dari Google *Maps API*.

Untuk mengetahui nilai guna dari aplikasi ini, maka dilakukan pula pengujian pengguna. Pengujian pengguna dilakukan dengan cara memberikan angket pada pengguna aplikasi.

5.3 Uji Coba Kebutuhan Fungsional Klien

Uji coba fungsionalitas dilakukan untuk melihat apakah fungsi-fungsi dasar aplikasi berjalan sebagaimana mestinya. Uji coba fungsionalitas meliputi semua kasus penggunaan yang telah dijelaskan pada Bab 3.

5.3.1 Uji Coba Meminta Rekomendasi

Uji coba meminta rekomendasi adalah pengujian terhadap kemampuan aplikasi Rekomendasi *Real Estate* untuk memberikan rekomendasi *real estate* sesuai dengan kriteria yang dipilih oleh pengguna. Skenario uji coba memilih kategori berita dapat dilihat pada Tabel 5.1.

Tabel 5.1 Uji Coba Meminta Rekomendasi

ID	UJ-001
Nama	Uji Coba Meminta Rekomendasi <i>Real Estate</i> .
Tujuan Uji Coba	Mendapatkan daftar <i>real estate</i> yang direkomendasikan oleh sistem.
Skenario 1	<i>Pengguna meminta hasil rekomendasi tanpa memasukkan karakteristik real estate.</i>
Kondisi Awal	a. Belum ada hasil rekomendasi yang diberikan oleh sistem.

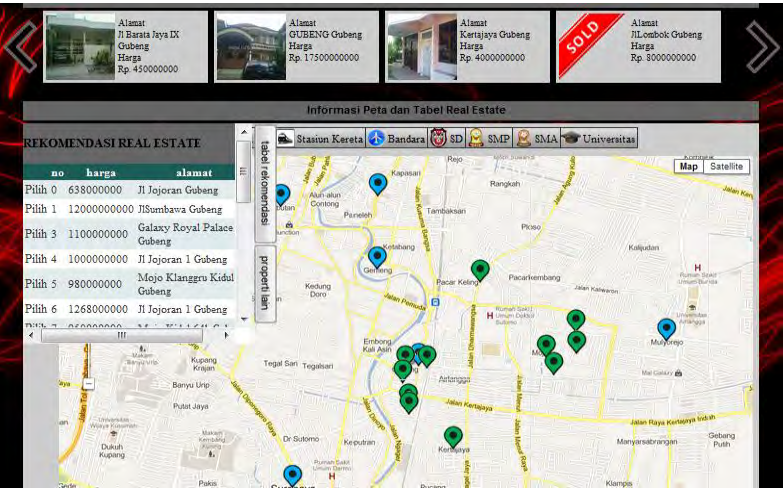
	b. Pengguna pengguna tidak memasukkan nilai karakteristik <i>real estate</i> .
Data Uji	Daftar <i>real estate</i> di Surabaya.
Langkah Pengujian	Pengguna meminta rekomendasi pada sistem.
Hasil Yang Diharapkan	Sistem tidak menampilkan daftar <i>real estate</i> .
Hasil Yang Didapat	Sistem tidak menampilkan daftar <i>real estate</i> .
Hasil Uji Coba	Berhasil.
Kondisi Akhir	Pengguna mendapatkan daftar seluruh <i>real estate</i> yang terdapat di kota Surabaya.
Skenario 2	<i>Pengguna meminta hasil rekomendasi dengan memasukkan karakteristik real estate</i>
Kondisi Awal	a. Pengguna memasukkan daftar karakteristik <i>real estate</i> . b. Pengguna meminta rekomendasi pada sistem.
Data Uji	Daftar <i>real estate</i> di Surabaya.
Langkah Pengujian	1. Pengguna memasukkan kriteria <i>real estate</i> . 2. Pengguna menekan tombol cari properti.
Hasil Yang Diharapkan	a. Sistem memberikan hasil rekomendasi. b. Situs <i>web</i> menampilkan data sesuai dengan rekomendasi.
Hasil Yang Didapat	a. Sistem memberikan hasil rekomendasi. b. Situs <i>web</i> menampilkan data sesuai dengan rekomendasi.
Hasil Uji Coba	Berhasil.
Kondisi Akhir	a. Pengguna masuk ke halaman rekomendasi. b. Pengguna dapat melihat hasil rekomendasi yang diberikan sistem.

Pada uji coba meminta rekomendasi dengan skenario 1 didapatkan hasil yaitu sistem tidak memberikan rekomendasi *real estate* apapun. Hasil pengujian skenario 1 dapat dilihat pada



Pada uji coba meminta rekomendasi dengan skenario 2 didapatkan hasil yaitu sistem akan memberikan rekomendasi *real estate* sesuai dengan kriteria yang dimasukkan oleh pengguna. Pada uji coba skenario 2 ini pengguna memilih karakteristik *real estate* yang berada di wilayah kecamatan Gubeng dan jarak yang ditempuh dari rumah sakit yaitu tujuh menit. Hasil pengujian skenario 2 dapat dilihat pada Gambar 5.2. Pada gambar tersebut muncul informasi *real estate* yang telah direkomendasikan oleh sistem. Pada peta, hasil *real estate* yang direkomendasikan akan muncul dengan bentuk bulatan berwarna hijau. Hasil uji coba menunjukkan, bahwa data yang direkomendasikan sistem sudah

benar yaitu *real estate* yang beradai di wilayah Gubeng dan jarak dengan fasilitas rumah sakit kurang dari tujuh menit.



Gambar 5.2 Meminta Rekomendasi dengan Memasukkan Kriteria

5.3.2 Uji Coba Memilih Hasil Rekomendasi

Uji coba memilih hasil rekomendasi adalah pengujian terhadap kemampuan aplikasi Rekomendasi *Real Estate* untuk memberikan informasi secara lengkap mengenai *real estate* yang dipilih oleh pengguna. Skenario uji coba memilih hasil rekomendasi dapat dilihat pada Tabel 5.2.

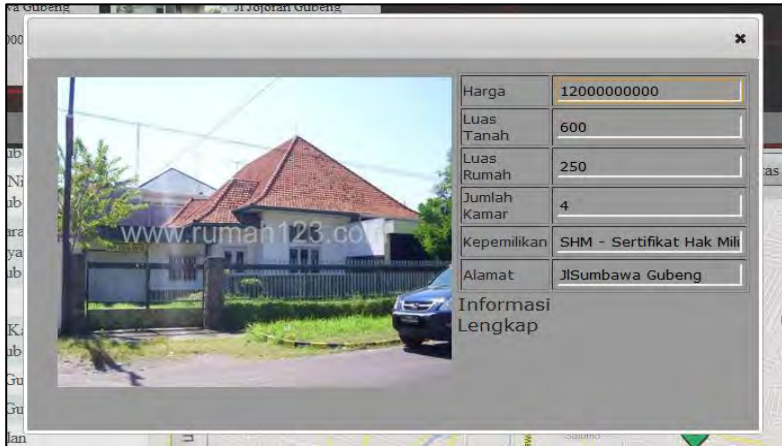
Tabel 5.2 Uji Coba Memilih Hasil Rekomendasi

ID	UJ-002
Nama	Uji Coba Meminta Rekomendasi <i>Real Estate</i> .
Tujuan Uji Coba	Mendapatkan informasi mengenai <i>real</i>

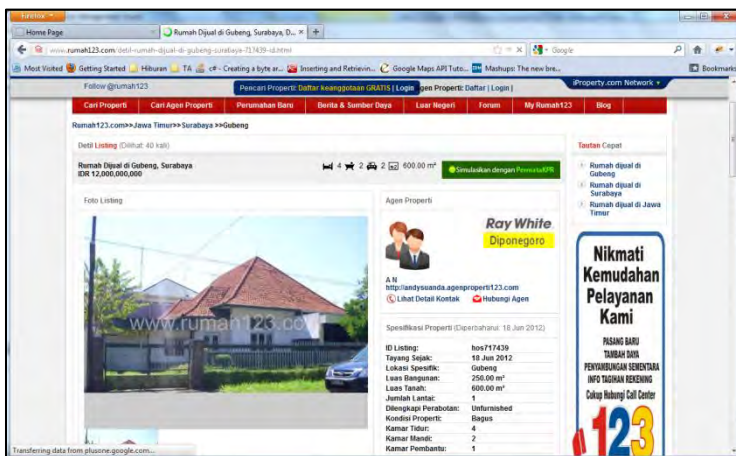
	<i>estate</i> yang dipilih.
Skenario 1	<i>Pengguna meminta informasi hasil rekomendasi salah satu real estate.</i>
Kondisi Awal	a. Pengguna belum memilih <i>real estate</i> yang diinginkan. b. Tidak muncul informasi mengenai <i>real estate</i> .
Data Uji	Daftar <i>real estate</i> di Surabaya.
Langkah Pengujian	Pengguna memilih <i>real estate</i> yang direkomendasikan oleh sistem.
Hasil Yang Diharapkan	Sistem menampilkan informasi <i>real estate</i> .
Hasil Yang Didapat	Sistem menampilkan informasi <i>real estate</i> .
Hasil Uji Coba	Berhasil.
Skenario 2	<i>Pengguna meminta untuk menuju situs sumber.</i>
Kondisi Awal	a. Pengguna masih belum memilih untuk melanjutkan mencari informasi ke situs sumber. b. Situs sumber belum terbuka.
Data Uji	Daftar <i>real estate</i> yang muncul.
Langkah Pengujian	Pengguna memilih untuk membuka situs sumber untuk melanjutkan transaksi lebih lanjut.
Hasil Yang Diharapkan	Sistem mengarahkan pengguna menuju situs sumber.
Hasil Yang Didapat	Sistem mengarahkan pengguna menuju situs sumber.
Hasil Uji Coba	Berhasil.
Kondisi Akhir	Pengguna diarahkan menuju situs sumber dan dapat melakukan transaksi.

Pada uji coba memilih hasil rekomendasi dengan skenario 1 didapatkan hasil yaitu sistem akan memberikan informasi

mengenai *real estate* yang dipilih oleh pengguna. Hasil pengujian skenario 1 dapat dilihat pada Gambar 5.3.



Gambar 5.3 Memilih Hasil Rekomendasi



Gambar 5.4 Memilih Menuju Situs Sumber

Pada uji coba memilih hasil rekomendasi dengan skenario 2 didapatkan hasil yaitu sistem akan membuka situs awal yang sesuai dengan pilihan pengguna. Hasil pengujian skenario 2 dapat dilihat pada Gambar 5.4 di atas. Dengan membandingkan data yang ada pada Gambar 5.4 dan Gambar 5.3 menunjukkan bahwa informasi *real estate* yang dituju sudah benar dan sama.

5.3.3 Uji Coba Menampilkan Informasi *Real Estate*

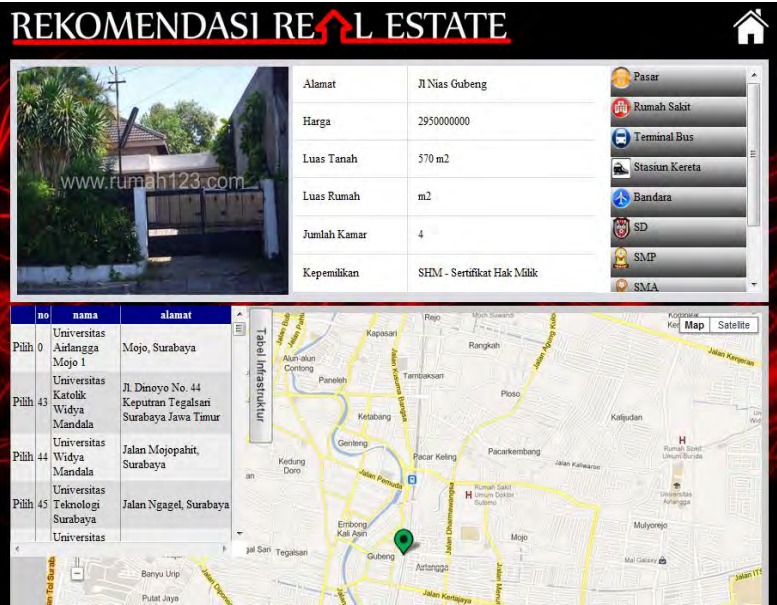
Uji coba menampilkan informasi *real estate* adalah pengujian terhadap kemampuan aplikasi Rekomendasi *Real Estate* untuk menampilkan informasi secara lengkap mengenai *real estate*. Skenario uji coba menampilkan informasi *real estate* dapat dilihat pada Tabel 5.3.

Tabel 5.3 Uji Coba Menampilkan Informasi *Real Estate*

ID	UJ-003
Nama	Uji Coba Menampilkan Informasi <i>Real Estate</i> .
Tujuan Uji Coba	Mendapatkan informasi secara lengkap mengenai karakteristik <i>real estate</i> yang dipilih.
Skenario 1	<i>Pengguna meminta informasi hasil rekomendasi salah satu real estate.</i>
Kondisi Awal	a. Pengguna belum memilih <i>real estate</i> yang diinginkan. a. Tidak muncul halaman informasi mengenai <i>real estate</i> .
Data Uji	Daftar <i>real estate</i> di Surabaya.
Langkah Pengujian	Pengguna memilih untuk melihat informasi <i>real estate</i> yang direkomendasikan oleh sistem.
Hasil Yang Diharapkan	Sistem menampilkan halaman informasi <i>real estate</i> .

Hasil Yang Didapat	Sistem menampilkan halaman informasi <i>real estate</i> .
Hasil Uji Coba	Berhasil.

Pada uji coba menampilkan informasi *real estate* dengan skenario 1 didapatkan hasil yaitu sistem akan membuka halaman informasi *real estate* yang sesuai dengan pilihan pengguna. Pada halaman tersebut akan terdapat berbagai informasi karakteristik *real estate* dan informasi fasilitas umum. Informasi yang disajikan berbentuk peta dan tabel. Hasil pengujian skenario 1 dapat dilihat pada Gambar 5.5.



Gambar 5.5 Halaman Informasi *Real Estate*

5.3.4 Uji Coba Memunculkan Fasilitas Umum

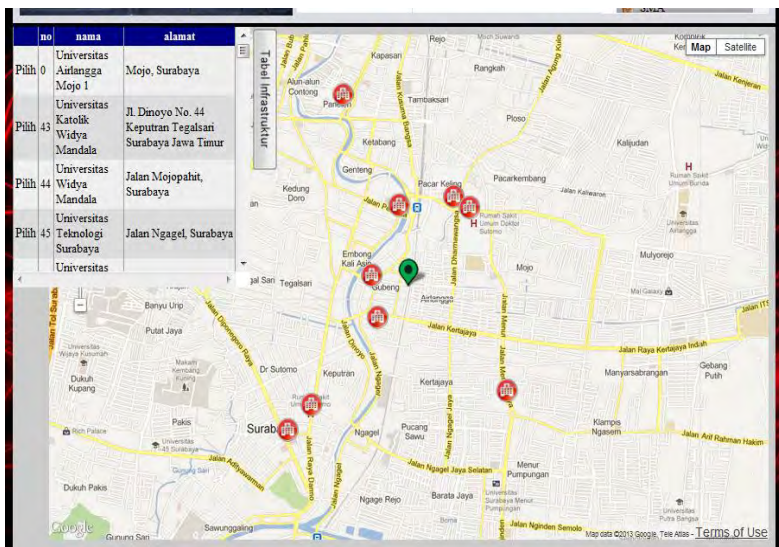
Uji coba memilih hasil rekomendasi adalah pengujian terhadap kemampuan aplikasi rekomendasi *real estate* untuk menampilkan fasilitas umum sesuai dengan tipe fasilitas umum yang dipilih oleh pengguna. Skenario uji coba menampilkan fasilitas umum dapat dilihat pada Tabel 5.4.

Tabel 5.4 Uji Coba Memunculkan Fasilitas Umum

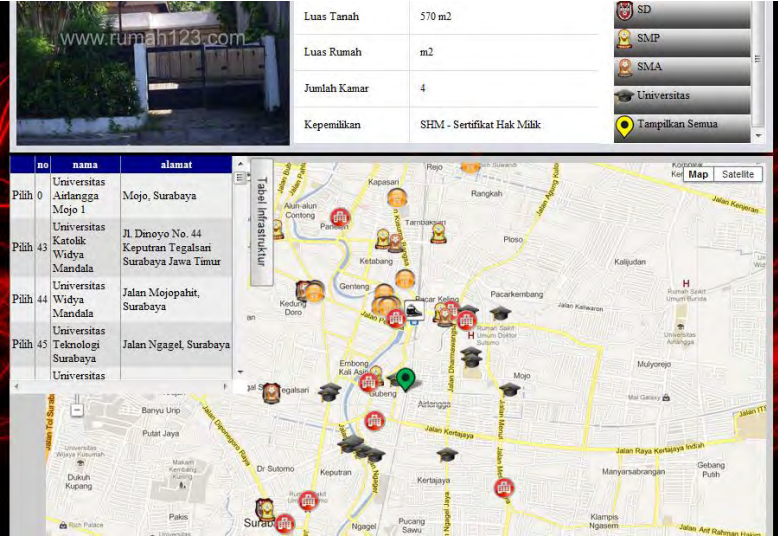
ID	UJ-004
Nama	Uji Coba Memunculkan Fasilitas Umum.
Tujuan Uji Coba	Mendapatkan lokasi dan informasi fasilitas umum.
Skenario	<i>Pengguna meminta menampilkan informasi fasilitas umum.</i>
Kondisi Awal	a. Tidak menampilkan fasilitas umum pada peta. b. Dalam peta tidak terdapat fasilitas umum.
Data Uji	Daftar fasilitas umum di Surabaya.
Langkah Pengujian	Pengguna memilih untuk memunculkan fasilitas umum.
Hasil Yang Diharapkan	Sistem menampilkan fasilitas umum sesuai pilihan pengguna.
Hasil Yang Didapat	Sistem menampilkan fasilitas umum sesuai pilihan pengguna.
Hasil Uji Coba	Berhasil

Pada uji coba ini didapatkan hasil, yaitu fasilitas umum akan muncul pada peta sesuai dengan koordinat yang diberikan oleh sistem. Dengan terdapatnya informasi ini, pengguna dapat memastikan bahwa rekomendasi yang diberikan oleh sistem sudah benar dan sesuai dengan keinginan pengguna. Dari hasil pantauan secara visual, hasil yang ditampilkan pada peta sudah

sesuai dengan kondisi nyata yang ada di Surabaya. Gambar 5.6 menggambarkan bahwa pengguna hanya memilih untuk menampilkan fasilitas umum yang bertipe rumah sakit. Sedangkan pada Gambar 5.7 menggambarkan pengguna memilih untuk menampilkan semua fasilitas umum yang ada di kota Surabaya. Tanda fasilitas umum rumah sakit digambarkan dengan gambar rumah sakit dalam bulatan merah, pasar ditandai gambar troli dalam bulatan kuning, stasiun kereta api ditandai gambar kereta dalam kotak putih, sekolah dasar ditandai logo sekolah dasar, universitas ditandai gambar topi, dan gambar terminal bus ditandai gambar bus dalam lingkaran biru.



Gambar 5.6 Memunculkan Rumah Sakit



Gambar 5.7 Memunculkan Semua Fasilitas Umum

5.4 Uji Coba Proses Server

Uji coba proses pada *server* dilakukan untuk melihat apakah proses yang terjadi pada *server* sudah berjalan sebagaimana mestinya. Uji coba yang dilakukan meliputi mengunduh informasi HTML dari situs sumber, dan mengolah data XML dari Google Maps API.

5.4.1 Uji Coba Mengunduh Informasi HTML dari Situs Sumber

Uji coba mengunduh informasi HTML dari situs sumber adalah pengujian terhadap kemampuan aplikasi Rekomendasi *Real Estate* untuk merawat data yang ada pada basis data, sehingga data yang ada terus terbaru. Skenario uji coba menampilkan fasilitas umum dapat dilihat pada Tabel 5.5.

Tabel 5.5 Uji Coba Mengunduh Informasi HTML dari Situs Sumber

ID	UJ-005
Nama	Uji Coba Mengunduh Informasi HTML dari situs sumber.
Tujuan Uji Coba	Memperbarui data <i>real estate</i> yang terdapat pada basis data.
Kondisi Awal	Data <i>real estate</i> dalam basis data belum terbaru.
Data Uji	URL: http://www.rumah123.com/rumah-dijual-di-surabaya-id.html
Langkah Pengujian	Sistem secara otomatis melakukan perbaruan data pada basis data.
Hasil Yang Diharapkan	Data pada basis data terbaru sesuai dengan data pada URL.
Hasil Yang Didapat	Data pada basis data terbaru sesuai dengan data pada URL.
Hasil Uji Coba	Berhasil.

Gambar 5.8 merupakan informasi terbaru mengenai *real estate* yang ada pada URL <http://www.rumah123.com/rumah-dijual-di-surabaya-id.html>. Data ini yang akan diunduh oleh sistem untuk memperbarui basis data. Gambar 5.9 merupakan data *real estate* yang ada pada basis data sebelum diperbarui. Pada basis data tersebut menunjukkan data terakhir merupakan data ke 70 dengan idWeb yaitu hos704896 sesuai dengan data terbaru kedua pada situs sumber. Gambar 5.10 merupakan gambar data *real estate* yang ada pada basis data setelah sistem melakukan pembaruan pada basis data. Pada gambar tersebut terlihat data pada basis data bertambah satu menjadi 71 dengan IdWeb hos720016 sesuai dengan data yang ada pada situs sumber.

Menampilkan 1 sampai 10 dari 2305 listings 1 2 3 4 5 Berikutnya » Sortir: Listing Terbaru ▾

Rumah dijual di Citraland, Surabaya
IDR 8,500,000,000

Simulasikan dengan PermataKPR



hos720016
4 4 2 375.00 m²
Modern Minimalist Home @ Raya Villa...
PRIME LOCATION!! ...

Lihat Detil

BRIGHTON™
CITRALAND
031 740 2288

Speedy
Lead your life

Tayang premium sejak: 25 Jun 2012 (Diperbaharui: 25 Jun 2012)
Dilihat: 304 kali

☐ Pilih untuk Bertanya

Willy Chandra
Lihat Detail Kontak

Rumah dijual di Citraland, Surabaya
IDR 2,350,000,000

Simulasikan dengan PermataKPR



hos704896
3 2 1 240.00 m²
Dijual Rumah Baru Dengan Cepat ...
Rumah minimalis di kawasan Bukit Golf...

Lihat Detil

BRIGHTON™
CITRALAND
031 740 2288

Speedy
Lead your life

Gambar 5.8 Halaman yang Akan Diunduh

Results		Messages							
	id	idArea	idWeb	coordinate	harga	luasan tanah	luas bangunan	jumlah Kamar	sertifikat
62	6..	2	hos711844	-7.2936997,112.7624886	875000000	144.00 ...	200.00 m&s...	14	SHM - Sertifikat He
63	6..	2	hos712667	-7.1962779,112.6456420	1700000000	350.00 ...	275.00 m&s...	6	SHM - Sertifikat He
64	6..	2	hos712555	-7.2891660,112.7343980	2300000000	200.00 ...	350.00 m&s...	3	SHM - Sertifikat He
65	6..	2	hos711997	-7.2936997,112.7624886	2300000000	200.00 ...	350.00 m&s...	3	SHM - Sertifikat He
66	6..	2	hos648171	-7.2891660,112.7343980	2100000000	364.00 ...	300.00 m&s...	3	SHM - Sertifikat He
67	6..	2	hos635910	-7.1962779,112.6456420	875000000	144.00 ...	220.00 m&s...	13	SHM - Sertifikat He
68	6..	2	hos643377	-7.1962779,112.6456420	1100000000	157.50 ...	140.00 m&s...	3	SHM - Sertifikat He
69	6..	2	hos645072	-7.1361054,112.6008783	1300000000	153.00 ...	300.00 m&s...	4	SHM - Sertifikat He
70	6..	0	hos704896	-7.2891660,112.7343980	2350000000	240.00 ...	200.00 m&s...	3	Lainnya

Query executed successfully.

KIKI-PC (10.0 RTM) | kiki-PC\kiki (58) | master | 00:00:00 | 70 rows

Gambar 5.9 Data Awal Sebelum Diperbarui

Results		Messages							
id	idArea	idWeb	coordinate	harga	luastanah	luasbangunan	jumlahKamar	sertifikat	
63	6..	2	hos712667	-7.1962779,112.6456420	1700000000	350.00 ...	275.00 m&s...	6	SHM - Sertifikat Ha
64	6..	2	hos712555	-7.2891660,112.7343980	2300000000	200.00 ...	350.00 m&s...	3	SHM - Sertifikat Ha
65	6..	2	hos711997	-7.2936997,112.7624886	2300000000	200.00 ...	350.00 m&s...	3	SHM - Sertifikat Ha
66	6..	2	hos648171	-7.2891660,112.7343980	2100000000	364.00 ...	300.00 m&s...	3	SHM - Sertifikat Ha
67	6..	2	hos635910	-7.1962779,112.6456420	875000000	144.00 ...	220.00 m&s...	13	SHM - Sertifikat Ha
68	6..	2	hos643377	-7.1962779,112.6456420	1100000000	157.50 ...	140.00 m&s...	3	SHM - Sertifikat Ha
69	6..	2	hos645072	-7.1361054,112.6008783	1300000000	153.00 ...	300.00 m&s...	4	SHM - Sertifikat Ha
70	6..	0	hos704896	-7.2891660,112.7343980	2350000000	240.00 ...	200.00 m&s...	3	Lainnya
71	7..	0	hos720016	-7.2891660,112.7343980	8500000000	375.00 ...	670.00 m&s...	4	HGB - Hak Guna E

...

Query executed successfully.

KIKI-PC (10.0 RTM)

kiki-PC\kiki (58)

master

00:00:00

71 rows

Gambar 5.10 Data Setelah Diperbarui

Dengan melihat hasil uji coba di atas dapat dikatakan sistem sudah berjalan dengan baik. Sistem sudah dapat melakukan pengunduhan dan menyimpan data tersebut ke dalam basis data dengan benar. Dengan demikian data yang diberikan kepada pengguna merupakan data terbaru dan lengkap.

5.4.2 Uji Coba Mengunduh dan Mengolah Data XML dari Google Maps API.

Uji coba mengunduh dan mengola data XML dari Google Maps API adalah pengujian terhadap kemampuan aplikasi Rekomendasi Real Estate untuk meminta data XML dari layanan Google Map dan menerjemahkannya untuk dapat disimpan dalam basis data. Skenario uji coba menampilkan fasilitas umum dapat dilihat pada Tabel 5.6.

Tabel 5.6 Mengunduh dan Mengolah Data XML dari Google Maps API

ID	UJ-006
Nama	Uji Coba Mengunduh dan Mengolah Data XML dari Google Maps API.
Tujuan Uji Coba	Mengunduh data XML dari Google Maps untuk dibaca dan disimpan dalam basis

	data.
Data Uji	Semua data <i>real estate</i> pada basis data.
Langkah Pengujian	Mencocokkan data XML dengan tabel.
Hasil Yang Diharapkan	Sistem dapat mengunduh XML dengan tepat dan menyimpannya dalam basis data dengan benar.
Hasil Yang Didapat	Sistem dapat mengunduh XML dengan tepat dan menyimpannya dalam basis data dengan benar.
Hasil Uji Coba	Berhasil.

Pada uji coba ini diukur ketepatan sistem dalam meminta data dari layanan Google *Maps*. Data yang didapat dari XML nantinya akan dimasukkan ke dalam basis data. Gambar 5.11 menunjukkan data yang telah di simpan dalam basis data. Data tersebut berasal dari data XML yang merupakan data layanan Google *Place*. Data tersebut sama dengan data yang ditunjukkan pada Gambar 5.12 yang menunjukkan data XML yang didapat dengan cara langsung mengakses layanan *web* Google *Place* dari *browser*.

9	Institut Elektronika Indonesia	-7.2824580, 112.7558480	university	Jl. Pucang Anom Timur 26...
10	Institut Informatika Indonesia	-7.2736800, 112.7498640	university	Jl. Blitton 21-23, Surabaya...
11	Universitas Dharma Cendika	-7.2897800, 112.7720000	university	Jl Deles I 29 Surabaya 601...
12	Universitas Surabaya (UBAYA)	-7.2930860, 112.7611170	university	Jl. Ngagel Jaya Selatan 1...
13	Stasiun Gubeng	-7.2654220, 112.7518890	trainstation	Tambaksari, Surabaya
14	Stasiun Wonokromo	-7.3019750, 112.7389860	trainstation	Jalan Stasiun Wonokromo,...
15	Terminal Bratang	-7.2888980, 112.7618220	busstation	Jalan Manyar, Surabaya
16	Rumah Sakit Umum Husa	-7.2823300, 112.7795820	hospital	Jalan Manyar Kerto Adi, S...
17	Rumah Sakit Umum Doktor Sutomo	-7.2666810, 112.7582210	hospital	Jalan Profesor Doktor Mus...
18	Rumah Sakit Umum Husada Utama	-7.2655990, 112.7505130	hospital	Jl. Margorejo Indah No. 9...
19	Rumah Sakit Jiwa Menur Surabaya	-7.2863890, 112.7626380	hospital	Manyarsabangan, Surabaya
20	Rumah Sakit Umum Mukti Mulya	-7.2664460, 112.7497670	hospital	Genteng, Surabaya
21	Rumah Sakit Bersalin Lombok Dua Dua	-7.2784630, 112.7471860	hospital	Jalan Lombok, Surabaya
22	Surabaya Plaza	-7.2650900, 112.7480000	market	Jl. PEMUDA, Surabaya

Gambar 5.11 Data dalam Basis Data

```

- <result>
  <name>Rumah Sakit Umum Haji</name>
  <vicinity>Jalan Manyar Kerto Adi, Surabaya</vicinity>
  <type>establishment</type>
- <geometry>
  - <location>
    <lat>-7.2823300</lat>
    <lng>112.7795820</lng>
  </location>
  <geometry>
  <rating>4.2</rating>
+ <icon></icon>
+ <reference></reference>
  <id>213509ed9da3b082a75450c4639b14ec4cf871b6</id>
</result>
- <result>
  <name>Rumah Sakit Umum Doktor Sutomo</name>
  <vicinity>Jalan Profesor Doktor Mustopo 6-8, Surabaya</vicinity>
  <type>establishment</type>
- <geometry>
  - <location>
    <lat>-7.2666810</lat>
    <lng>112.7582210</lng>
  </location>
  <geometry>
  <rating>4.0</rating>

```

Gambar 5.12 Data Sebelum Dimasukkan ke Basis Data

Dari kedua gambar di atas menunjukkan bahwa proses untuk mengunduh data dari layanan *web* Google Maps sudah benar. Begitu juga dengan proses mengubah data XML menjadi data masukan pada basis data juga sudah benar. Namun terdapat beberapa data yang didapat dari layanan *web* Google Maps kurang valid. Hal ini disebabkan oleh beberapa faktor antara lain:

- Banyak data berasal dari situs *web* sumber kurang lengkap atributnya, sehingga tidak dapat dicari dengan menggunakan layanan *web* Google Maps.
- Data yang terdapat dalam Google Maps merupakan masukan dari pengguna umum, sehingga data tidak diketahui kebenarannya.

- Di wilayah Surabaya Google *Maps* masih belum bisa mencari alamat sampai jalan-jalan kecil.
- Di kota Surabaya banyak data fasilitas umum dari Google *Maps* yang tidak terkelompokkan dengan baik.

5.3 Uji Penggunaan

Pada subbab ini dijelaskan mengenai mengenai skenario yang digunakan dalam uji penggunaan dan hasil dari pelaksanaan uji penggunaan.

5.3.1 Skenario Uji Penggunaan

Pengujian penggunaan dilakukan untuk memastikan bahwa perangkat lunak yang dibangun sudah benar dan bermanfaat dalam memenuhi kebutuhan pengguna. Pengujian penggunaan dilakukan di lingkungan Teknik Informatika. Perangkat lunak *server* ditempatkan pada komputer *server* laboratorium Rekayasa Perangkat Lunak. Sasaran pengguna untuk pengujian adalah semua pengguna dari berbagai latar belakang yang berbeda.

Alur dari pengujian penggunaan adalah:

1. Pengguna diberikan alamat URL aplikasi *web* perangkat lunak klien.
2. Pengguna melakukan uji coba semua fungsionalitas yang disediakan pada aplikasi klien.
3. Pengguna diberikan alamat URL pengisian angket.
4. Pengguna mengisi pertanyaan berupa pilihan berganda sesuai dengan penilaiannya masing-masing.
5. Hasil penilaian dirangkum memanfaatkan layanan Google *Docs*.

Adapun beberapa pertanyaan yang diajukan pada angket sebagai berikut:

1. *Website* Rekomendasi *Real Estate* tidak ditemukan galat selama digunakan.

2. Tampilan antarmuka dari Rekomendasi *Real Estate* menarik dan mudah digunakan.
3. Fitur pencarian dan filter-filter yang ada pada Rekomendasi *Real Estate* membantu dalam menemukan *real estate* yang diinginkan.
4. Rekomendasi *real estate* yang diberikan Rekomendasi *Real Estate* sesuai dengan karakteristik yang dimasukkan pengguna.
5. Waktu untuk memberi rekomendasi dari Rekomendasi *Real Estate* terbilang cepat.
6. Situs Rekomendasi *Real Estate* sudah siap dikenalkan pada masyarakat luas.

Adapun pilihan jawaban yang diberikan untuk setiap pertanyaan:

1. Sangat tidak setuju.
2. Tidak setuju.
3. Tidak tahu.
4. Setuju.
5. Setuju sekali.

5.3.2 Evaluasi Pengujian Penggunaan

Hasil rekapitulasi angket ditunjukkan pada Gambar 5.13 hingga Gambar 5.18 dengan jumlah responden sebanyak 15 orang. Biodata pengguna yang melakukan pengujian dicantumkan dalam Tabel B.1 pada Lampiran B.



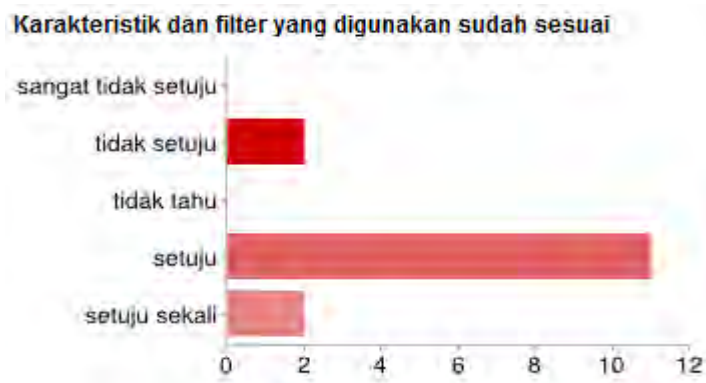
Gambar 5.13 Rekapitulasi Pertanyaan 1

Gambar 5.13 menunjukkan bahwa secara umum tidak ditemukan kesalahan umum dalam perangkat lunak, walau beberapa mengatakan tidak tahu dan tidak setuju.



Gambar 5.14 Rekapitulasi Pertanyaan 2

Gambar 5.14 menunjukkan bahwa tampilan situs *web* sudah menarik, walau beberapa mengatakan tidak tahu dan tidak setuju.



Gambar 5.15 Rekapitulasi Pertanyaan 3

Gambar 5.15 menunjukkan bahwa secara umum karakteristik yang digunakan sebagai filter sudah sesuai, walau beberapa mengatakan tidak tahu dan tidak setuju.



Gambar 5.16 Rekapitulasi Pertanyaan 4

Gambar 5.16 menunjukkan bahwa secara umum program telah memberikan rekomendasi yang tepat, walau beberapa mengatakan tidak tahu dan tidak setuju.



Gambar 5.17 Rekapitulasi Pertanyaan 5

Gambar 5.17 menunjukkan bahwa secara umum rekomendasi yang dilakukan oleh sistem sudah cepat, walau beberapa mengatakan tidak tahu dan tidak setuju.



Gambar 5.18 Rekapitulasi Pertanyaan 6

Gambar 5.18 menunjukkan bahwa pengguna setuju bahwa situs Rekomendasi *Real Estate* siap untuk diperkenalkan ke

masyarakat luas, walau beberapa mengatakan tidak tahu dan tidak setuju.

Secara umum, sebagian besar pengguna mengemukakan pendapat bahwasannya perangkat lunak yang dibangun merupakan perangkat lunak berbasis *web* yang terorganisir dengan baik. Pengguna merasa dimudahkan dengan adanya perangkat lunak yang dapat membantu mereka menemukan *real estate* yang sesuai.

5.4 Evaluasi Uji Coba

Dari keseluruhan uji coba yang dilakukan pada sisi klien didapatkan hasil pengujian kesemuanya berhasil. Sehingga dapat disimpulkan fungsi-fungsi perangkat lunak yang terdapat pada sisi klien sudah dapat digunakan. Pada sisi *server* didapatkan kesimpulan bahwa proses-proses pada *server* sudah berjalan dengan baik. Dari uji penggunaan dapat disimpulkan bahwa pengguna memberikan respon positif terhadap perangkat lunak ini. Dengan semua hasil uji coba ini dapat disimpulkan perangkat lunak Rekomendasi *Real Estate* dapat digunakan dengan baik.

BAB VI PENUTUP

Bab ini membahas kesimpulan yang dapat diambil dari tujuan pembuatan perangkat lunak, serta hasil uji coba yang telah dilakukan. Selain itu, terdapat beberapa saran untuk pengembangan lebih lanjut.

6.1 Kesimpulan

Dari hasil pengamatan selama perancangan, implementasi, dan proses uji coba perangkat lunak yang dilakukan, penulis mengambil kesimpulan sebagai berikut:

- a. Pada uji fungsionalitas yang ditunjukkan pada hasil uji coba skenario UJ-001 hingga skenario UJ-004 aplikasi yang dibuat dalam Tugas Akhir ini telah dapat mengimplementasikan sebuah sistem informasi yang mampu memenuhi kebutuhan fungsionalitas yang dibutuhkan oleh pengguna yang memiliki tujuan untuk mencari *real estate* sesuai dengan kriteria yang diinginkan.
- b. Sesuai dengan hasil rekapitulasi uji pengguna pada grafik yang ditunjukkan Gambar 5.17, dengan perangkat lunak ini proses pencarian rekomendasi *real estate* dapat dilakukan dengan mudah dan cepat.
- c. Perangkat lunak ini membuat pengguna dapat melihat kondisi lingkungan dan melihat infrastruktur yang terdapat di sekitar *real estate* yang telah dipilih. Hal tersebut ditunjukkan pada hasil uji coba skenario UJ-004.
- d. Dari hasil uji pengguna yang ditunjukkan pada grafik Gambar 5.13 sampai Gambar 5.18 dapat disimpulkan bahwa pengguna sudah puas dengan aplikasi ini.

6.2 Saran

Dalam pembuatan Tugas Akhir ini, terdapat beberapa kemungkinan pengembangan aplikasi yang dilakukan, yaitu:

- a. Perangkat lunak ini juga memungkinkan untuk dihubungkan dengan sosial media, sehingga seorang pengusaha yang sering berbisnis *real estate* dapat mengetahui jika ada *real estate* baru yang sesuai dengan kriteria yang diinginkan.
- b. Perangkat lunak ini juga dapat dikembangkan seperti situs penjualan rumah lainnya, sehingga selain dapat mencari informasi *real estate*, pengguna juga dapat mengunggah informasi mengenai *real estate*.

DAFTAR PUSTAKA

- [1] Giudice, Vincenzo (2009), *Decision Support Sistem for Real estate Investment Choice*, University of Naples Italy.
- [2] *Mashup: The new breed of Web app*, (2006) <http://www.ibm.com/developerworks/xml/library/x-mashups/index.html>, diakses tanggal 3 Juni 2012.
- [3] Pahala, Bagindo (2010), *Fenomena Internet Masa Kini Dengan Pengembangan Mashup, Linked Data untuk Membangun Aplikasi Web*, Binus University.
- [4] Google Maps JavaScript API v3, (2012) <https://developers.google.com/maps/documentation/javascript/?hl=id>, diakses tanggal 3 Juni 2012.
- [5] The Google *Directions* API <https://developers.google.com/maps/documentation/Directions/?hl=id>, diakses tanggal 4 Juni 2012.
- [6] The Google *Distance Matrix* API <https://developers.google.com/maps/documentation/distancematrix/?hl=id>, diakses tanggal 4 Juni 2012.
- [7] The Google *Elevation* API <https://developers.google.com/maps/documentation/elevation/?hl=id>, diakses tanggal 4 Juni 2012.
- [8] The Google *Geocoding* API <https://developers.google.com/maps/documentation/geocoding/?hl=id>, diakses tanggal 4 Juni 2012.
- [9] The Google *Places* API, (2012), <https://developers.google.com/places/documentation/>, diakses tanggal 9 Juni 2012.
- [10] Jones, M Tim, *Build a Web spider*, (2006), <http://www.ibm.com/developerworks/linux/library/l-spider/>, diakses tanggal 10 Juni 2012.
- [11] Microsoft ASP.NET Team, *ASP.NET MVC Overview*, (Januari 2009), <http://www.asp.net/mvc/tutorials/older-versions/overview/asp-net-mvc-overview#author-info>, diakses tanggal 2 Juni 2012.

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS



Penulis lahir di Surabaya, 2 Juli 1989. Penulis menempuh pendidikan kelas satu sampai enam di SD Tempeh Tengah 01 Lumajan. Untuk pendidikan menengah, penulis tempuh di SMPN 1 Tempeh. Selanjutnya penulis melanjutkan pendidikan SMA di SMAN 2 Lumajang. Selanjutnya penulis melanjutkan pendidikan sarjana di Jurusan Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya.

Penulis dalam menyelesaikan pendidikan S1 mengambil bidang minat Rekayasa Perangkat Lunak (*Software Engineering*) dan memiliki ketertarikan di bidang *Computer Network*, *Design Algorithm*, Basis data, *web*, *Mobile Application* dan *Software Development*. Penulis dapat dihubungi melalui surel:calonagnia@gmail.com.

[Halaman ini sengaja dikosongkan]

LAMPIRAN A

ATRIBUT DAN FUNGSI KELAS

Crawler
-inputHTML : string
-listURL : string
+getHTML() : string
+getListURL() : string

Gambar A.1 Atribut dan Fungsi Kelas Crawler

Downloader
-stringURL : string
-stringHTML : string
-image : object
-dataXML : string
-proxyString : string
+downloadHTML() : string
+downloadImage() : object
+downloadXML() : string

Gambar A.2 Atribut dan Fungsi Kelas Downloader

RecomProp
-listProp : Property -queryList : string
+getListProperty() : Property +getListRecomProp() : Property

Gambar A.3 Atribut dan Fungsi Kelas **RecomProp**

Parser
-arrInfoProperty : string -arrInfoInfra : string -arrURL : string -arrInfMapProp : string -arrInfMapInfra : string
+getArrayProperty() : string +getArrayInfra() : string +getArrayURL() : string +getArrayMapProperty() : string +getArrayMapInfra()

Gambar A.4 Atribut dan Fungsi Kelas **Parser**

GeoCoding
-coordinatLat : string -coordinatLng : string -infoXML : string
+getCoordinat() : string +getGoogleMap() : string +setURLGoogle() : string

Gambar A.5 Atribut dan Fungsi Kelas Geocoding

Property
-id : string -address : string -harga : long -luasTanah : int -luasBangunan : int -jumlahKamar : int -sertifikat : string -fasilitas : string
+saveToDB() +setProperty() +getPropertyInfor() : string +setInfrastruktur()

Gambar A.6 Atribut dan Fungsi Kelas Property

Infrastruktur
-id : string -name : string -address : string -type : string
+setInfrastruktur() +getInfrastrukturInfo() : string +saveToDB() +setMapInfra()

Gambar A.7 Atribut dan Fungsi Kelas Infrastruktur

Recommendation
-price : string -volumeLand : string -area : string -numRoom : string -hospital : string -market : string -train : string -bus : string -airport : string -elementary : string -junior : string -senior : string -university : string
+getCarcterQuery() : string +gerAccessQuery() : string +bundleQuery() : string

Gambar A.8 Atribut dan Fungsi Kelas Recommendation

InfrastructurSearch
-id : string -listInfrastructur : Infrastructur -queryStr : string -idProperty : string
+getListInfra() : Infrastructur +()

Gambar A.9 Atribut dan Fungsi Kelas
InfrastructurSearch

InfrastructurSearch
-id : string -name : string -address : string -type : string -queryGetList : string
+getById() : string +getByType() : string +getByAddress() : string

Gambar A.10 Atribut dan Fungsi Kelas
InfrastructurSearch

[Halaman ini sengaja dikosongkan]

LAMPIRAN B

PROFIL PENGISI ANGKET

Dalam uji pengguna perangkat lunak ini, untuk menghasilkan data yang tepat, maka pengguna diambil dari orang-orang yang memiliki berbagai macam latar belakang. Dalam Tabel B.1 dijelaskan profil-profil dari pengguna yang melakukan pengujian perangkat lunak ini.

Tabel B.1 Profil Pengguna Penguji Perangkat Lunak

No.	Usia (Tahun)	Jenis Kelamin	Profil
1	22	Laki-laki	Mahasiswa Teknik Informatika ITS bidang minat RPL
2	22	Laki-laki	Mahasiswa Teknik Informatika ITS bidang minat RPL
3	23	Laki-laki	Mahasiswa Teknik Informatika ITS bidang minat RPL
4	22	Laki-laki	Mahasiswa Teknik Informatika ITS bidang minat KCV
5	23	Laki-laki	Mahasiswa Teknik Informatika ITS bidang minat NCC
6	20	Perempuan	Mahasiswa Despro ITS bidang DKV
7	22	Laki-laki	Mahasiswa Elektro ITS
8	29	Laki-laki	Desainer Arsitek
9	21	Laki-laki	Mahasiswa Teknik Mesin ITS
10	19	Perempuan	Mahasiswa Statistika ITS
11	35	Laki-laki	Pegawai PT.KAI kepala stasiun Perak
12	31	Perempuan	Ibu rumah tangga
13	18	Laki-laki	Mahasiswa Kelautan ITS
14	29	Perempuan	Guru SD Kreatif An-Nur
15	29	Laki-laki	Pelatih Taekwondo

[Halaman ini sengaja dikosongkan]

LAMPIRAN C KODE SUMBER

C.1 Kelas DownloadData

```
class DownloadData
{
    private WebClient wc = new WebClient();
    private WebProxy myProxy;

    //fungsi mengambil data string
    public string DownloadDataString(string URL)
    {
        string result = "";
        result = wc.DownloadString(URL);
        return result;
    }

    //fungsi mengambil data gambar
    public byte[] Download(string imageUrl)
    {
        byte[] result = new byte[16 * 1024];
        try {
            Stream stream = wc.OpenRead(imageUrl);
            using (MemoryStream ms = new MemoryStream())
            {
                int read;
                while ((read = stream.Read(result, 0,
result.Length)) > 0)
                {
                    ms.Write(result, 0, read);
                }
                result = ms.ToArray();
                Image Result = Image.FromStream(ms);
                stream.Flush();
                stream.Close();
                wc.Dispose();
                if (result != null)
                {
                    string temp = "";
                    temp =
Sistem.Text.UTF8Encoding.UTF8.GetString(ms.ToArray());
                }
                return ms.ToArray();
            }
        }
    }
}
```

```

    }
    catch (Exception e) {
        Console.WriteLine(e.Message);
    }
    return result;
}
}

```

Kode Sumber C.1 Kelas DownloadData

C.2 Kelas GetData

```

class GetData
{
    public string area = "";
    public string Globalcoor = "";
    public string getFile(string data)
    {
        string file = ""; Match id = Regex.Match(data,
@"(ID\sListing:.*?</div>.*?</div>)", RegexOptions.Singleline);
        file += Regex.Replace(id.Value,
@"\s*ID\sListing:\s*<.*?>\s*<.*?>|</div>", "");
        Match mAd = Regex.Match(data,
@"(<textarea.*Id=\"\"message\"\".*?>.*?</textarea>)",
RegexOptions.Singleline);
        string sAlamat = Regex.Replace(mAd.Value,
@"\s*<.*?>\s*", "");
        sAlamat = Regex.Replace(sAlamat, @"\"..*", "");
        sAlamat = Regex.Replace(sAlamat, @"\"hos[0-9]*\s-\s",
        "");

        file += sAlamat + ";";

        GMGeoCod ggc = new GMGeoCod();
        file += ggc.getCoordinate(sAlamat);

        return file;
    }

    public string getRealEstateInfo(string data)
    {
        string idWeb = "";
        string sAlamat = "";
        string coor = "";
        string picture = "null, ";
    }
}

```

```

        string sPrice = "";
        string luasTanah = "";
        string luasBangunan = "";
        string jumlahKamar = "";
        string sertifikat = "";
        string fasilitas = "";
        string file = "";
        //Get Area
        this.area = Regex.Match(data,
@"<div\sclass=\"listing-detail-header-title\">.*?</div>",
RegexOptions.Singleline).Value;
        this.area = Regex.Replace(area,
@"\s*<.*?>\s*|Rumah\s*Dijual\s*di\s*|,\s*Surabaya.*?IDR\s*|\s&nb
p;","");

        //Get ID Web
        Match id = Regex.Match(data,
@"(ID\sListing:.*?</div>.*?</div>)", RegexOptions.Singleline);
        idWeb = "" + Regex.Replace(id.Value,
@"\s*ID\sListing:\s*<.*?>\s*<.*?>|</div>", "") + "", ";

        DBConnection conn = new DBConnection();
        string checkIdWeb = "";
        checkIdWeb = conn.selectData("select IdWeb from
RealEstate where IdWeb = '" + idWeb + "'");
        if (checkIdWeb != "")
        {
            return "";
        }
        //Get Alamat
        Match mAd = Regex.Match(data,
@"(<textarea.*Id=\"message\".*?>.*?</textarea>)",
RegexOptions.Singleline);
        sAlamat = Regex.Replace(mAd.Value, @"\"s*<.*?>\s*",
        "");
        sAlamat = Regex.Replace(sAlamat, @"\"..*", "");
        sAlamat = Regex.Replace(sAlamat, @"\", \"sSurabaya", "");
        sAlamat = "" + Regex.Replace(sAlamat, @"\"hos[0-9]*\"s-
\s", "") + "", ";

        //GetCoordinate
        GMGeoCod ggc = new GMGeoCod();
        Globalcoor = ggc.getCoordinate(sAlamat);
        coor = "" + Globalcoor + "", ";
        //GetPicture

        //GetPrice

```

```

        Match price = Regex.Match(data,
@"<div\sclass=\"\"listing-detail-header-title\"\">.*?</div>",
RegexOptions.Singleline);
        string p = Regex.Replace(price.Value,
@"\s*<.*?>\s*|Rumah\s*Dijual\s*di.*?Surabaya.*?IDR\s*|\s&nbsp;",
""");
        sPrice = "" + Regex.Replace(p, @",", " ") + ", ";
//GetLuasTanah
        luasTanah = Regex.Match(data,
@"(Luas\sTanah:.*?</div>.*?</div>)",
RegexOptions.Singleline).Value;
        string lt = Regex.Replace(luasTanah, @" m&sup2;", "");
        luasTanah = "" + Regex.Replace(luasTanah,
@"\s*Luas\sTanah:\s*<.*?>\s*<.*?>|</div>", "") + ", ";

//GetLuasBangunan
        luasBangunan = Regex.Match(data,
@"(Luas\sBangunan:.*?</div>.*?</div>)",
RegexOptions.Singleline).Value;
        string lb = Regex.Replace(luasBangunan, @" m&sup2;",
""");
        luasBangunan = "" + Regex.Replace(luasBangunan,
@"\s*Luas\sBangunan:\s*<.*?>\s*<.*?>|</div>", "") + ", ";

//GetNumRoom
        jumlahKamar = Regex.Match(data,
@"(Kamar\sTidur:.*?</div>.*?</div>)",
RegexOptions.Singleline).Value;
        jumlahKamar = "" + Regex.Replace(jumlahKamar,
@"\s*Kamar\sTidur:\s*<.*?>\s*<.*?>|</div>", "") + ", ";

//GetInfoSertifikat
        sertifikat = Regex.Match(data,
@"(Sertifikat:.*?</div>.*?</div>)",
RegexOptions.Singleline).Value;
        sertifikat = "" + Regex.Replace(sertifikat,
@"\s*Sertifikat:\s*<.*?>\s*<.*?>|</div>", "") + ", ";

//Get Info Fasilitas
        fasilitas = Regex.Match(data,
@"(Fasilitas:.*?</div>)", RegexOptions.Singleline).Value;
        fasilitas = "" + Regex.Replace(fasilitas,
@"\s*Fasilitas:\s*<.*?>\s*<.*?>|\s*<.*?>\s*", "") + ", ";

        file += idWeb + sAlamat + coord + sPrice + luasTanah +
luasBangunan + jumlahKamar + sertifikat + fasilitas;
        return file;

```

```

    }

    public string getImageUrl(string data)
    {
        string result = "";
        result = Regex.Match(data, @"<div\sclass=\"\"listing-
detail-photo\"\">(.*?)</div>", RegexOptions.Singleline).Value;
        result = Regex.Match(result,
@"src=\"\"(.*?)\"\"", RegexOptions.Singleline).Value;
        result = Regex.Replace(result, @"src=\"\"|\"\"", "");
        return result;
    }
}

```

Kode Sumber C.2 Kelas GetData

C.3 Kelas DBConnection

```

class DBConnection
{
    public string strConn;
    public string global = "";
    public DBConnection()
    {
        this.strConn = strConn = "Data Source=KIKI-PC;Initial
Catalog=RealEstateDB;Integrated Security=True";
    }

    public string insertToDB(string insertQuery)
    {
        SqlConnection conn = new SqlConnection(strConn);
        try
        {
            conn.Open();
            SqlCommand command = new SqlCommand(insertQuery,
conn);
            command.ExecuteNonQuery();
            return insertQuery + "affected ";
        }
        catch (Exception ex)
        {
            return ex.ToString() + "can't affected";
        }
        finally

```

```

        {
            conn.Close();
        }
    }

    public string insertRealEstate(string insertQuery, string
imageURL)
    {
        SqlConnection conn = new SqlConnection(strConn);
        try
        {
            conn.Open();
            DownloadData dd = new DownloadData();
            byte[] image = dd.Download(imageURL);

            string temp = "";
            if(image != null)
                temp =
Sistem.Text.UTF8Encoding.UTF8.GetString(image);

            Stream s = new MemoryStream(image);
            Image i = Image.FromStream(s);
            i.Save("Berhasil.jpg");

            SqlParameter picparameter = new SqlParameter();
            picparameter.SqlDbType = SqlDbType.Image;
            picparameter.ParameterName = "image";
            picparameter.Value = image;

            SqlCommand command = new SqlCommand(insertQuery +
",@image", conn);
            command.Parameters.Add(picparameter);
            command.ExecuteNonQuery();

            return insertQuery + "affected ";
        }
        catch (Exception ex)
        {
            return "can't affected";
        }
        finally
        {
            conn.Close();
        }
    }

    public string selectData(string selectQuery)
    {

```

```

        SqlDataReader reader = null;
        SqlConnection conn = new SqlConnection(strConn);
        string result = "";
        try
        {
            conn.Open();
            SqlCommand command = new SqlCommand(selectQuery,
conn);
            reader =
command.ExecuteReader(System.Data.CommandBehavior.CloseConnection)
;

            while (reader.Read())
            {
                int index = 0;
                int lim = reader.FieldCount;
                while(index < lim)
                {
                    if(index == 0)
                        result +=
reader.GetValue(index).ToString();
                    else
                        result += ";" +
reader.GetValue(index).ToString();
                    index++;
                }
                result += "&";
            }
            return result;
        }
        catch (Exception ex)
        {
            return null;
        }
        finally
        {
            conn.Close();
        }

        return result;
    }

    public string selectLastRecord(string tableName)
    {
        string result= "0";
        SqlConnection conn = new SqlConnection(strConn);

```

```

        SqlDataReader reader = null;
        try
        {
            conn.Open();
            SqlCommand command = new SqlCommand("select top 1
Id from "+ tableName +" order by Id desc", conn);
            reader =
command.ExecuteReader(Sistem.Data.CommandBehavior.CloseConnection)
;

            while (reader.Read())
            {
                result = reader["Id"].ToString();
                int x = int.Parse(result) + 1;
                result = x.ToString();
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Failed to selecting data from
data base");
        }
        finally
        {
            if (reader != null)
                reader.Close();

            conn.Close();
        }
        return result;
    }

    public string selectIdArea(string area)
    {
        string result = null;
        SqlConnection conn = new SqlConnection(strConn);
        SqlDataReader reader = null;
        try
        {
            conn.Open();
            SqlCommand command = new SqlCommand("select Id
from Area where Name='"+ area + "'", conn);
            reader =
command.ExecuteReader(Sistem.Data.CommandBehavior.CloseConnection)
;

            while (reader.Read())
            {

```

```

        result = reader["Id"].ToString();
    }
    catch
    {
        MessageBox.Show("Failed to selecting data from
data base");
    }
    finally
    {
        if (reader != null)
            reader.Close();

        conn.Close();
    }
    return result;
}
}

```

Kode Sumber C.3 Kelas DBConnection

C.4 Kelas InfrastrukturFinder

```

class InfrastrukturFinder
{
    private WebClient wc = new WebClient();
    private WebProxy myProxy;
    private string latLng;
    private string idRE = "";
    DBConnection conn = new DBConnection();

    public InfrastrukturFinder(string latLng, string idRE)
    {
        this.latLng = latLng;
        this.idRE = idRE;
    }

    //Find market
    public string getMarket()...

    ///Find Hospital
    public string getHospital()...

    //Find Elementary Shools

```

```

        public string getElementarySchools()...

        //Find junior high school
        public string getJuniorHighSchool()...

        //Find senior high school
        public string getSeniorHighSchool()...

        //Find Bus Station
        public string getBusStation()...

        //Find Train Station
        public string getTrainStation()...

        //Find Air Port
        public string getAirPort() ...

        //Find University
        public string getUniversity()...

        //Find certain place
        public string findPlace(string radius, string types,
string name, int treshold, string description)
        {
            string resultLat = "";
            string result = "";
            GMGeoCod geoCod = new GMGeoCod()
            try
            {
                string URL = "";
                if(types != "")
                    URL =
"https://maps.googleapis.com/maps/api/place/search/xml?location="
+ latLng + "&radius=" + radius + "&types=" + types + "&Name="+ name
+ "&sensor=false&key=AiZaSyAZdIJdzCT4RVxT-sZU2D7lZUx08BoJ82Y";
                else
                    URL =
"https://maps.googleapis.com/maps/api/place/search/xml?location="
+ latLng + "&radius=" + radius + "&Name=" + name +
"&sensor=false&key=AiZaSyAZdIJdzCT4RVxT-sZU2D7lZUx08BoJ82Y";

                string stringXml = wc.DownloadString(URL);
                XmlDocument xmldoc = new XmlDocument();
                xmldoc.LoadXml(stringXml);
                XmlNodeList xnl =
xmldoc.GetElementsByTagName("result");

                int index = 0;

```

```

        string[] infName = new string[100];
        string[] infAddress = new string[100];
        foreach (XmlNode xn in xn1)
        {
            string a = xn["Name"].InnerText;
            string[] arrStr = a.Split(' ', '\\t');

            if (arrStr.Length <= threshold)
            {
                string lat = "";
                string lng = "";
                string nama = "";
                string alamat = "";
                infName[index] = "";
                infAddress[index] = "";
                lat =
                xn["geometry"]["location"]["lat"].InnerText.ToString();
                lng =
                xn["geometry"]["location"]["lng"].InnerText.ToString();
                infName[index] =
                xn["Name"].InnerText.ToString();
                infAddress[index] =
                xn["vicinity"].InnerText.ToString();
                nama = xn["Name"].InnerText.ToString();
                alamat =
                xn["vicinity"].InnerText.ToString();

                index++;
                result = index.ToString();
                Infrastruktur inf = new
                Infrastruktur(idRE, nama, latLng, description, lat + "," + lng,
                alamat);

                resultLat += inf.getInformation();
            }
        }
    }
    catch (Exception ex)
    {
        resultLat += "";
    }
    return result;
}
}

```

Kode Sumber C.4 Kelas InfrastrukturFinder

C.5 Kelas GMGeoCod

```

class GMGeoCod
{
    private struct Coordinate
    {
        public string coorX;
        public string coorY;
        public override string ToString()
        {
            return coorX + ", " + coorY;
        }
    }
    private WebClient wc = new WebClient();
    private Coordinate coor;
    private WebProxy myProxy;          public GMGeoCod()...
    public string getCoordinate(string address)
    {
        try
        {
            address = Regex.Replace(address, @" ", "+");
            string stringXml =
wc.DownloadString("http://maps.googleapis.com/maps/api/geocode/xml
?Address="+ address +",+Surabaya,+Indonesia&sensor=false");
            XmlDocument doc = new XmlDocument();
            doc.LoadXml(stringXml);
            XmlNodeList list =
doc.GetElementsByTagName("geometry");
            string xml = "";
            foreach (XmlNode node in list)
            {
                xml += node["location"]["lat"].InnerText + ",";
                xml += node["location"]["lng"].InnerText;
                break;
            }
            return xml;
        }
        catch (Exception ex)
        {
            return ex.ToString();
        }
    }
}

```

Kode Sumber C.5 Kelas GMGeoCod

C.6 Kelas Rekomendasi

```

public class Rekomendasi
{
    DBConnection conn;
    public Rekomendasi()
    {
        conn = new DBConnection();
    }
    public DataSet getRekomendasi(string area, string harga,
string room, string volume)
    {
        DataSet ds = new DataSet();
        DataTable dt = new DataTable();
        if (area != "")
        {
            dt = conn.getDataTable("select id, coordinate,
harga from RealEstate where idArea = '1'");
        }
        ds.Tables.Add(dt);

        return ds;
    }
    public string getRekomendasiCommandStr(string rec,
string university, string hospital, string bus, string train,
string airport, string area, string harga, string room, string
volume)
    {
        string result = "";
        string strCmd = getSubCmdStr(university,hospital, bus,
train, airport);
        result = rec + strCmd;
        string strCmdStr = getCommandString("", area, harga,
room, volume);
        if (strCmd != "")
            result += " )place ";
        result += strCmdStr;
        if (strCmdStr != "" && strCmd != "")
            result += " and re.id = place.idRealEstate order
by re.id";
        else if (strCmd != "")
            result += "where re.id = place.idRealEstate order
by re.id";
        return result;
    }
    public string getCommandString(string resource, string
area, string harga, string room, string volume)

```

```

{
    string result = "";
    int checkFirsVar = 0;

    if (area != "---")
    {
        result += "where idArea = (select id from Area
where name = '" + area + "') ";
        checkFirsVar++;
    }

    //Seleksi Harga
    if (harga != "---")
    {
        if (checkFirsVar == 0)
        {
            if(harga == "Rp 250 juta")
                result += " where cast(harga as float) <
250000000 ";
            else if(harga == "Rp 250 juta - 500 juta")
                result += " where cast(harga as float) >
250000000 and cast(harga as float) < 1000000001 ";
            else if (harga == "Rp 500 juta - 1 milyar")
                result += " where cast(harga as float) >
500000000 and cast(harga as float) < 1000000001 ";
            else if (harga == "Rp 1 milyar - 2 milyar")
                result += " where cast(harga as float) >
1000000000 and cast(harga as float) < 2000000001 ";
            else if (harga == "Rp 2 milyar - 5 milyar")
                result += " where cast(harga as float) >
2000000000 and cast(harga as float) < 5000000001 ";
            else if (harga == ">Rp 5 milyar")
                result += " where cast(harga as float) >
5000000000 ";
        }
        else
        {
            if (harga == " Rp 250 juta ")
                result += " and cast(harga as float) <
250000000 ";
            else if (harga == "Rp 250 juta - 500 juta")
                result += " and cast(harga as float) >
250000000 and cast(harga as float) < 500000001 ";
            else if (harga == "Rp 500 juta - 1 milyar")
                result += " and cast(harga as float) >
500000000 and cast(harga as float) < 1000000001 ";
            else if (harga == "Rp 1 milyar - 2 milyar")

```

```

        result += " and cast(harga as float) >
1000000000 and cast(harga as float) < 2000000001 ";
        else if (harga == "Rp 2 milyar - 5 milyar")
            result += " and cast(harga as float) >
2000000000 and cast(harga as float) < 5000000001 ";
    }
}

//Seleksi Ruangan
if (room != "---")
{
    if (checkFirsVar == 0)
    {
        if (room == "1")
            result += " where jumlaKamar = '1' ";
        else if (room == "2")
            result += " where jumlaKamar = '2' ";
        else if (room == "3")
            result += " where jumlaKamar = '3' ";
        else if (room == "4")
            result += " where jumlaKamar = '4' ";
        else if (room == "5")
            result += " where jumlaKamar = '5' ";
        else if (room == "6")
            result += " where jumlaKamar = '6' ";
        else if (room == "7")
            result += " where jumlaKamar = '7' ";
        else if (room == "8")
            result += " where jumlaKamar = '8' ";
        else if (room == "9")
            result += " where jumlaKamar = '9' ";
        else if (room == "10")
            result += " where jumlaKamar = '10' ";
    }
    else
    {
        if (room == "1")
            result += " and jumlaKamar = '1' ";
        else if (room == "2")
            result += " and jumlaKamar = '2' ";
        else if (room == "3")
            result += " and jumlaKamar = '3' ";
        else if (room == "4")
            result += " and jumlaKamar = '4' ";
        else if (room == "5")
            result += " and jumlaKamar = '5' ";
        else if (room == "6")

```

```

        result += " and jumlaKamar = '6' ";
    else if (room == "7")
        result += " and jumlaKamar = '7' ";
    else if (room == "8")
        result += " and jumlaKamar = '8' ";
    else if (room == "9")
        result += " and jumlaKamar = '9' ";
    else if (room == "10")
        result += " and jumlaKamar = '10' ";
    }
}
if (volume != "---")
{
    if (checkFirsVar == 0)
    {
        if (volume == "0 - 150")
            result += " where cast(luastanah as float)
< 151 ";
        else if (volume == "150 - 350")
            result += " where cast(luastanah as float)
> 150 and cast(luastanah as float) < 351 ";
        else if (volume == "350 - 700")
            result += " where cast(luastanah as float)
> 350 and cast(luastanah as float) < 701 ";
        else if (volume == "700 - 1100")
            result += " where cast(luastanah as float)
> 700 and cast(luastanah as float) < 1101 ";
        else if (volume == ">1100")
            result += " where cast(luastanah as float)
> 2000000000 and cast(luastanah as float) < 5000000001 ";
    }
    else
    {
        if (harga == "Rp 250 juta")
            result += " and cast(harga as float) <
250000000 ";
        else if (harga == "Rp 250 juta - 500 juta")
            result += " and cast(harga as float) >
250000000 and cast(harga as float) < 5000000001 ";
        else if (harga == "Rp 500 juta - 1 milyar")
            result += " and cast(harga as float) >
500000000 and cast(harga as float) < 10000000001 ";
        else if (harga == "Rp 1 milyar - 2 milyar")
            result += " and cast(harga as float) >
1000000000 and cast(harga as float) < 20000000001 ";
        else if (harga == "Rp 2 milyar - 5 milyar")

```

```

        result += " and cast(harga as float) >
2000000000 and cast(harga as float) < 5000000001 ";

    }
}
//result += "and re.id = place.idRealEstate order by
re.id";
    return result;
}
public string getSubCmdStr(string university, string
hospital, string bus, string train, string airport)
{
    string result = "";
    int checkFirsVar = 0;
    if (university != "")
    {
        if(checkFirsVar == 0)
        {
            result += ", (SELECT
PublicInfrastruktur.type, RealEstateReal estatees.idInfrastruktur,
PublicInfrastruktur.id, RealEstateReal estatees.time,
RealEstateReal estatees.idRealEstate " +
" FROM          PublicInfrastruktur INNER JOIN " +
"RealEstateReal estatees ON PublicInfrastruktur.id =
RealEstateReal estatees.idInfrastruktur " +
" WHERE ";
            checkFirsVar++;
        }
        else
            result += " OR ";
        result += " (PublicInfrastruktur.type =
'university') AND (RealEstateReal estatees.time = '"+ university
+"') ";
    }
    if (hospital != "")
    {
        if (checkFirsVar == 0)
        {
            result += ", (SELECT
PublicInfrastruktur.type, RealEstateReal estatees.idInfrastruktur,
PublicInfrastruktur.id, RealEstateReal estatees.time,
RealEstateReal estatees.idRealEstate " +
" FROM          PublicInfrastruktur
INNER JOIN " +
" RealEstateReal estatees ON
PublicInfrastruktur.id = RealEstateReal estatees.idInfrastruktur "
+
" WHERE ";

```

```

        checkFirsVar++;
    }
    else
        result += " OR ";
    result += " (PublicInfrastructur.type =
'hospital') AND (RealEstateReal estatees.time = '" + hospital +
"') ";
    }
    if (bus != "")
    {
        if (checkFirsVar == 0)
        {
            result += ", (SELECT
PublicInfrastructur.type, RealEstateReal estatees.idInfrastructur,
PublicInfrastructur.id, RealEstateReal estatees.time,
RealEstateReal estatees.idRealEstate " +
                                " FROM          PublicInfrastructur
INNER JOIN " +
                                " RealEstateReal estatees ON
PublicInfrastructur.id = RealEstateReal estatees.idInfrastructur "
+
                                " WHERE ";
            checkFirsVar++;
        }
        else
            result += " OR ";
        result += " (PublicInfrastructur.type =
'busstation') AND (RealEstateReal estatees.time = '" + bus + "')
";
    }
    if (train != "")
    {
        if (checkFirsVar == 0)
        {
            result += ", (SELECT
PublicInfrastructur.type, RealEstateReal estatees.idInfrastructur,
PublicInfrastructur.id, RealEstateReal estatees.time,
RealEstateReal estatees.idRealEstate " +
                                " FROM          PublicInfrastructur
INNER JOIN " +
                                " RealEstateReal estatees ON
PublicInfrastructur.id = RealEstateReal estatees.idInfrastructur "
+
                                " WHERE ";
            checkFirsVar++;
        }
        else
            result += " OR ";
    }

```

```

        result += " (PublicInfrastruktur.type =
'trainstation') AND (RealEstateReal estatees.time = '" + train +
"') ";
    }
    if (airport != "")
    {
        if (checkFirsVar == 0)
        {
            result += ", (SELECT
PublicInfrastruktur.type, RealEstateReal estatees.idInfrastruktur,
PublicInfrastruktur.id, RealEstateReal estatees.time,
RealEstateReal estatees.idRealEstate " +
                " FROM          PublicInfrastruktur
INNER JOIN " +
                " RealEstateReal estatees ON
PublicInfrastruktur.id = RealEstateReal estatees.idInfrastruktur "
+
                " WHERE ";
            checkFirsVar++;
        }
        else
            result += " OR ";
        result += " (PublicInfrastruktur.type = 'airport')
AND (RealEstateReal estatees.time = '" + airport + "') ";
    }
    return result;
}
}

```

Kode Sumber C.6 Kelas Rekomendasi

[Halaman ini sengaja dikosongkan]