KERJA PRAKTIK – IF184801

Implementasi Aplikasi Monitoring Kinerja FTEIC ITS berbasis Web

Fakultas Teknologi Elektro dan Informatika Cerdas – ITS Surabaya
Gedung Rektorat Lt.3 Kampus ITS Sukolilo Surabaya

Periode: 1 Agustus 2020 – 31 Oktober 2020

Oleh:
Fandy Kuncoro Adianto 05111740000118

Pembimbing Jurusan
Waskitho Wibisono, S.Kom., M.Eng., Ph.D
Pembimbing Lapangan
Dr. I Ketut Eddy Purnama, S.T., M.T.

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

[Halaman ini sengaja dikosongkan]

KERJA PRAKTIK – IF184801

Implementasi Aplikasi Monitoring Kinerja FTEIC ITS berbasis Web

Fakultas Teknologi Elektro dan Informatika Cerdas – ITS Surabaya
Gedung Rektorat Lt.3 Kampus ITS Sukolilo Surabaya

Periode: 1 Agustus 2020 – 31 Oktober 2020

Oleh:
Fandy Kuncoro Adianto 05111740000118

Pembimbing Jurusan
Waskitho Wibisono, S.Kom., M.Eng., Ph.D
Pembimbing Lapangan
Dr. I Ketut Eddy Purnama, S.T., M.T.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2020

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN
KERJA PRAKTIK

Implementasi Aplikasi Monitoring Kinerja FTEIC ITS berbasis Web

Oleh:
Fandy Kuncoro Adianto 05111740000118

Disetujui oleh Pembimbing Kerja Praktik:

1. Waskitho Wibisono, S.Kom., M.Eng., Ph.D.
NIP. 197512202001122002

(Pembimbing Departemen)

2. Dr. I Ketut Eddy Purnama, S.T., M.T.
NIP. 196907301995121001

(Pembimbing Lapangan)

[Halaman ini sengaja dikosongkan]

Implementasi Aplikasi Monitoring Kinerja FTEIC ITS berbasis Web

Nama Mahasiswa	: Fandy Kuncoro Adianto
NRP	: 05111740000118
Departemen	: Informatika FTEIC-ITS
Pembimbing Jurusan	: Waskitho Wibisono, S.Kom., M.Eng., Ph.D.

ABSTRAK

Dosen serta tenaga kependidikan di Fakultas Teknologi Elektro dan Informatika Cerdas memilki *track record* yang sangat banyak tiap tahunnya. Mulai dari prestasi, kuliah tamu, training, sertifikasi, konferensi dan jurnal. Karena masalah tersebut, diperlukan adanya suatu sistem informasi untuk menampung semua catatan yang ada agar data yang tersimpan dapat digunakan dengan lebih baik.

Sehingga perlu dibangun aplikasi untuk melakukan monitoring terhadap kinerja yang telah dilakukan terhadap semua kegiatan yang terjadi di FTEIC. Aplikasi ini mampu untuk melakukan proses administratif sekaligus melakukan masukan setiap catatan pada kategorinya masing-masing.

Pembuatan aplikasi menggunakan beberapa *framework.* Untuk *frontend* menggunakan React.js. Lalu untuk *backend* menggunakan Django. basis data yang digunakan adalah MySQL. Aplikasi Web didesain menggunakan paradigma *Single Page Application* sehingga tampilan terlihat lebih dinamis dalam satu buah halaman.

Kata kunci:
FTEIC, Monitoring, Single Page Application

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena atas berkat limpahan rahmat dan lindungan-Nya penulis dapat melaksanakan salah satu kewajiban sebagai mahasiswa Teknik Informatika ITS yaitu Kerja Praktik (KP).

Penulis menyadari masih terdapat banyak kekurangan baik dalam pelaksanaan kerja praktik maupun penyusunan buku laporan ini, namun kami berharap buku laporan ini dapat menambah wawasan pembaca dan dapat menjadi sumber referensi. Penulis mengharapkan kritik dan saran yang membangun untuk kesempurnaan penulisan buku laporan ini.

Melalui laporan ini penulis juga ingin menyampaikan rasa terima kasih kepada kepada orang-orang yang telah membantu dalam pelaksanaan kerja praktik hingga penyusunan laporan Kerja praktik baik secara langsung maupun tidak langsung. Orang-orang tersebut antara lain adalah:

1. Orang tua penulis,
2. Bapak Waskitho Wibisono, S.Kom., M.Eng., Ph.D. selaku dosen pembimbing kerja praktik yang telah membimbing penulis selama kerja praktik berlangsung.
3. Bapak Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D selaku koordinator Kerja Praktik.
4. Bapak Dr. I Ketut Eddy Purnama, S.T., M.T. selaku pembimbing lapangan selama kerja praktik yang telah memberikan bimbingan serta ilmunya kepada penulis.
5. Sahabat penulis.

Surabaya, Januari 2021

Penulis

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dari waktu ke waktu, setiap kegiatan yang dilaksanakan di Fakultas Teknologi Elektro dan Informatika Cerdas tidak tercatat dengan baik. Akibatnya, pencarian kegiatan-kegiatan tersebut sulit dilakukan di waktu mendatang.

Masalah tersebut memunculkan ide untuk membangun sebuah aplikasi untuk melakukan pengawasan terhadap seluruh catatan yang dapat disajikan secara terperinci dan terstruktur.

Aplikasi ini akan mengelompokkan kegiatan berdasarkan kategorinya dan untuk setiap kategori dapat diurutkan berdasarkan tahun, departemen, dan detil kegiatan lainnya.

1.2 Tujuan

Tujuan Kerja praktik kali ini adalah melakukan implementasi aplikasi monitoring kinerja FTEIC ITS sekaligus menuntaskan kewajiban kuliah kerja praktik di Institut Teknologi Sepuluh Nopember.

1.3 Manfaat

Manfaat dari implementasi aplikasi ini adalah memudahkan pencarian dan penyortiran catatan kegiatan yang ada pada FTEIC ITS yang akan berguna di waktu yang akan datang.

1.4 Rumusan Permasalahan

Berikut rumusan masalah dalam pelaksanaan kerja praktik implementasi aplikasi monitoring kinerja FTEIC ITS berbasis Web:

- Bagaimana membangun aplikasi monitoring kinerja FTEIC ITS berbasis web?
- Bagaimana mengelola aplikasi monitoring kinerja FTEIC ITS berbasis web agar lebih fleksibel dan optimal?

1.5 Lokasi dan Waktu Kerja Praktik

Kerja praktik ini dilaksanakan pada waktu dan tempat sebagai berikut:

Lokasi	: *Online*
Alamat	: Marina Emas Timur V No. 41 Keputih, Sukolilo, Surabaya
Waktu	: 1 Oktober 2020 – 31 Desember 2020
Hari Kerja	: Setiap Hari
Jam kerja	: Fleksibel

1.6 Metodologi Kerja Praktik

1.6.1 Perumusan Masalah

Untuk mengetahui domain dan fungsionalitas, dijelaskan secara rinci bagaimana sistem yang harus dibuat. Penjelasan oleh pembimbing lapangan kerja praktik kali ini menghasilkan beberapa catatan mengenai gambaran secara garis besar tentang sistem berbasis website yang sebelumnya telah diterapkan. Setelah mendapatkan gambaran sistem, diskusi lebih lanjut dilakukan guna menentukan rancangan serta *tools* pendukung pembuatan sistem.

1.6.2 Studi Literatur

Pada tahap ini, setelah ditentukannya rancangan *database*, bahasa pemrograman sampai dengan teknologi beserta *tools* tambahan yang digunakan, dilakukan studi literatur lanjut mengenai bagaimana penggunaannya dalam membangun sistem sesuai yang diharapkan.

Dikarenakan aplikasi yang akan dibuat merupakan bagian dari sistem yang sudah terbangun, maka secara garis besar tools yang digunakan juga tidak jauh berbeda. Bahasa pemrograman yang digunakan Python untuk *backend* dan *web server*, dengan bantuan kerangka kerja (framework) Django. Kemudian untuk *frontend*, digunakan javascript dengan bantuan kerangka kerja ReactJS.

1.6.3 Analisis dan Perancangan Sistem

Langkah ini meliputi penjelasan awal tentang sistem. Bagaimana

cara kerja sistem dengan skenario tertentu. Dari penjelasan awal telah didapatkan beberapa kebutuhan fungsional secara garis besar. Kemudian dilanjutkan dengan memperjelas dan menspesifikkan kebutuhan- kebutuhan tersebut.

1.6.4 Implementasi Sistem

Implementasi sistem didasarkan oleh perancangan dan analisis sebelumnya. Semua didasari pada rancangan yang sudah ada sebelumnya dan penentuan *tools* yang telah dilakukan sebelumnya. Penentuan tipe data saat pembuatan table baru pada database disesuaikan juga dengan kebutuhan.

Pengerjaan dilakukan dengan progres setiap hari, dengan setiap harinya menargetkan perkembangan dari hari sebelumnya. Progres penyelesaian aplikasi terus dipantau oleh Pembimbing Lapangan.

1.6.5 Pengujian dan Evaluasi

Pengujian dilakukan oleh pembimbing lapangan setiap bagian dari fitur telah selesai dikerjakan untuk memberikan evaluasi ketika ada yang tidak sesuai, dan persetujuan apabila sudah sesuai.

1.7 Sistematika Laporan

Laporan kerja praktik ini terdiri dari 7 bab dengan rincian sebagai berikut:

1.7.1 Bab I Pendahuluan

Bab ini berisi tentang latar belakan masalah, tujuan, manfaat, rumusan masalah, lokasi dan waktu kerja praktik, metodologi, dan sistematika laporan.

1.7.2 Bab II Profil Perusahaan

Bab ini berisi sekilas tentang profil Fakultas Teknologi Elektro dan Informatika Cerdas.

1.7.3 Bab III Tinjauan Pustaka

Dalam bab ini dibahas mengenai konsep-konsep pembuatan

aplikasi, dasar teori, teknologi yang dipakai dalam pembuatan aplikasi.

1.7.4 Bab IV Analisis dan Perancangan Sistem

Pada bab ini, dijelaskan hasil pembelajaran atau analisis terhadap apa saja yang diperlukan dan harus diperhatikan dalam pengembangan aplikasi yang dikerjakan selama KP.

1.7.5 Bab V Implementasi Sistem

Pada bab ini, berisi penjelasan tahap-tahap yang dilakukan untuk proses implementasi aplikasi.

1.7.6 Bab VI Pengujian dan Evaluasi

Pada bab ini, dijelaskan tentang hasil pengujian dan evaluasi dari sistem yang telah dikembangkan selama pelaksanaan KP.

1.7.7 Bab VII Kesimpulan dan Saran

Pada bab ini, dipaparkan kesimpulan yang dapat diambil dan juga saran selama pengerjaan KP.

[Halaman ini sengaja dikosongkan]

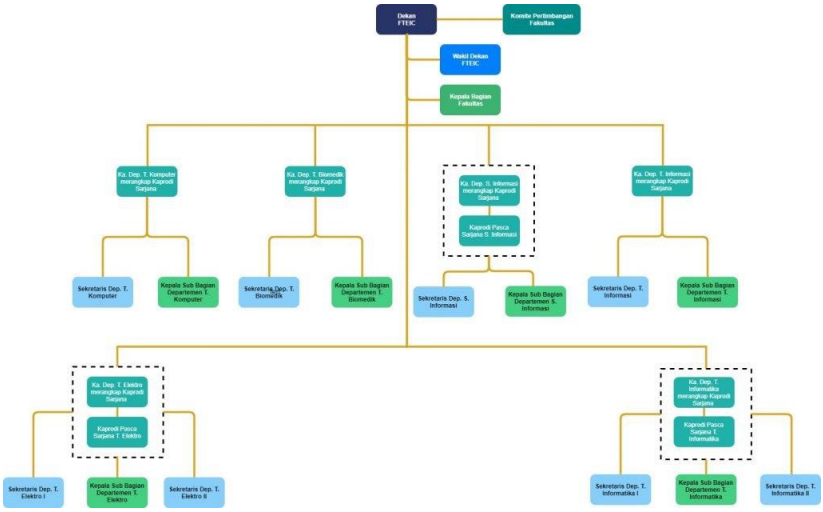
BAB II
PROFIL INSTANSI

2.1 Profil Instansi

Fakultas Teknologi Elektro dan Informatika Cerdas (F-Electics) adalah fakultas unggulan di ITS yang didirikan sebagai gabungan dari Fakultas Teknologi Informasi dan Komunikasi (FTIK) dan Fakultas Teknologi Elektro (FTE) yang terdiri dari 6 Departemen dengan lebih dari 2500 mahasiswa baik di tingkat S1, S2 ataupun S3.F-Electics hadir dengan visi membentuk sumber daya manusia berkarakter, berbudi unggul serta berkelas dunia dalam bidang teknologi elektro, sistem informasi, biomedik, komputer, informatika dan teknologi informasi.

2.2 Struktur Organisasi

Struktur organisasi di FTEIC dapat dilihat pada Gambar 2.1.



Gambar 2.1 Struktur organisasi FTEIC

2.3 Lokasi Instansi

Penampakan depan salah satu departemen yang ada pada FTEIC ITS dapat dilihat pada Gambar 2.2.

Alamat : Jl. Teknik Kimia - Gedung Departemen Teknik Informatika
Kampus Institut Teknologi Sepuluh Nopember Surabaya
Jalan Raya ITS, Sukolilo, Surabaya 60111, Indonesia

Gambar 2.2: Foto Departemen Informatika FTEIC ITS

BAB III
TINJAUAN PUSTAKA

3.1 Aplikasi Web

Dalam rekayasa perangkat lunak, suatu aplikasi web adalah suatu aplikasi yang diakses menggunakan penjelajah web melalui suatu jaringan seperti Internet atau intranet. Ia juga merupakan suatu aplikasi perangkat lunak komputer yang dikodekan dalam bahasa yang didukung penjelajah web dan bergantung pada penjelajah tersebut untuk menampilkan aplikasi.

3.2 Python

Python adalah Bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif. Python juga didukung oleh komunitas yang besar.

Python mendukung multi paradigma pemrograman, utamanya; namun tidak dibatasi; pada pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Salah satu fitur yang tersedia pada python adalah sebagai bahasa pemrograman dinamis yang dilengkapi dengan manajemen memori otomatis. Seperti halnya pada bahasa pemrograman dinamis lainnya, python umumnya digunakan sebagai bahasa skrip meski pada praktiknya penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan dengan menggunakan bahasa skrip. Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi.

3.3 Django

Django adalah web framework Python yang didesain untuk membuat aplikasi web yang dinamis, kaya fitur dan aman. Django yang dikembangkan oleh Django Software Foundation terus mendapatkan

perbaikan sehingga membuat web framework yang satu ini menjadi pilihan utama bagi banyak pengembang aplikasi web.

3.4 JavaScript

JavaScript adalah sebuah script yang memungkinkan kita mengimplementasikan hal-hal kompleks pada halaman web terutama yang bersifat interaktif. Javascript juga dapat digunakan untuk memanipulasi dan mengirim data pada browser pengguna.

3.5 HTML

Hyper Text Markup Language (HTML) adalah sebuah bahasa markah yang digunakan untuk membuat sebuah halaman web, menampilkan berbagai informasi di dalam sebuah penjelajah web Internet dan pemformatan hiperteks sederhana yang ditulis dalam berkas format ASCII agar dapat menghasilkan tampilan wujud yang terintegerasi.

3.6 CSS

Cascading Style Sheets (CSS) merupakan mekanisme sederhana untuk menambahkan *style* (seperti warna tulisan, *font*, jarak tulisan) ke dalam dokumen web. Hampir semua *browser* dan banyak aplikasi yang ada saat ini telah menunjang penggunaan CSS.

3.7 React

React merupakan library javascript yang membantu dalam pembuatan antarmuka pengguna. React memungkinkan penggunanya untuk membuat tampilan antarmuka dari kumpulan bagian kecil yang disebut komponen.

3.8 *Web Server*

Web Server adalah sebuah perangkat lunak server yang berfungsi menerima permintaan HTTP atau HTTPS dari klien yang dikenal dengan *web browser* dan mengirimkan kembali hasilnya dalam halaman-halaman web yang umumnya berbentuk dokumen HTML.

3.9 Visual Studio Code

Visual Studio Code adalah editor kode yang dikembangkan oleh Microsoft untuk Windows, Linux dan macOS. Visual Studio Code memberikan dukungan untuk debugging, kontrol Git tertanam, penyorotan sintaksis, penyelesaian kode cerdas, snippet, dan refactoring kode. Visual Studio Code juga dapat disesuaikan, sehingga pengguna dapat mengubah tema editor, pintasan keyboard, dan preferensi.

[Halaman ini sengaja dikosongkan]

BAB IV
ANALISIS DAN PERANCANGAN SISTEM

4.1 Analisis Sistem

Pada bab ini akan dijelaskan mengenai tahapan dalam membangun Aplikasi Monitoring Kinerja FTEIC berupa analisisi dari sistem yang akan dibangun. Hal tersebut dijelaskan ke dalam dua bagian, yaitu definisi umum aplikasi dan analisis kebutuhan fungsional.

4.1.1 Definisi Umum Aplikasi

Secara umum, Monitoring Kinerja FTEIC merupakan sistem berbasis web yang digunakan untuk memudahkan pendataan terkait beberapa hal yang ada dalam FTEIC seperti pendataan prestasi mahasiswa maupn dosen, lalu pelatihan yang diikuti oleh dosen, kuliah tamu, dan juga jurnal dosen serta konferensi yang diadakan oleh dosen FTEIC. Alasan utama pengembangan aplikasi ini ialah untuk memudahkan pendataan topik terkait yang ada di FTEIC.

Kategori Pengguna	Tugas	Hak Akses ke aplikasi
Admin	Membuat user, memvalidasi data baru, dan menghapus atau merubah data	Modul *Scenario*
User	Menambahkan data baru dan melihat daftar data	Modul *Scenario*

Tabel 4.1 Karakteristik Pengguna

4.1.2 Analisis Kebutuhan

Dalam aplikasi ini, terdapat fungsi-fungsi yang harus dipenuhi oleh sistem. Kebutuhan ini terbagi ke dalam dua jenis, yakni kebutuhan fungsional dan kebutuhan non- fungsional.

4.1.2.1 Kebutuhan Fungsional

Kebutuhan fungsional pada aplikasi ini menjelaskan bagaimana sistem ini bekerja. Kebutuhan fungsional dari Aplikasi Monitoring Kinerja FTEIC dijelaskan pada Tabel 4.2.

Kode Kebutuhan	Deskripsi kebutuhan	Modul
F01	Membuka Halaman Awal	Scenario
F02	Melakukan Login	Scenario
F03	Admin dan user dapat memasukan data baru	Scenario
F04	Admin dapat membuat user baru	Scenario
F05	Admin dapat memvalidasi data baru	Scenario

Tabel 4.2 Kebutuhan Fungsional

4.1.2.2 Kebutuhan Non-Fungsional

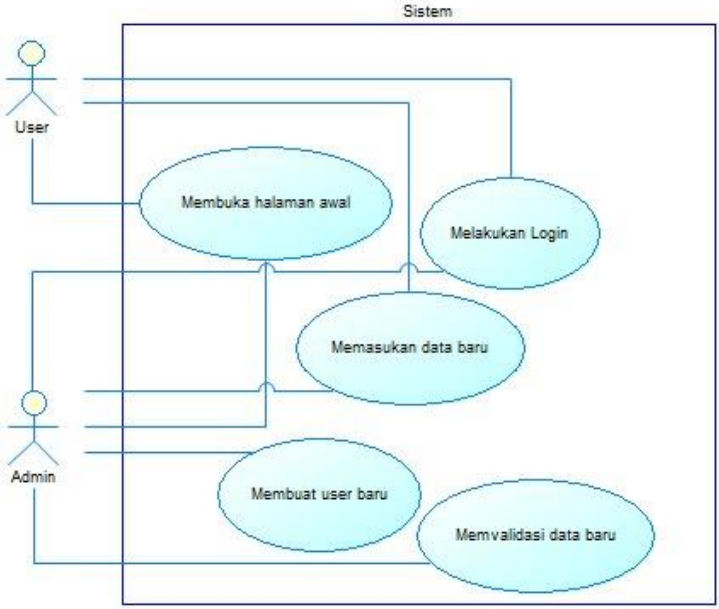
Kebutuhan non-fungsional adalah kebutuhan pengguna untuk mendefinisikan bagaimana Batasan dan karakteristik dari sebuah sistem yang dibangun. Kebutuhan non-fungsional dari aplikasi Monitoring Kinerja FTEIC secara umum serta khusus untuk modul Scenario terdapat pada Tabel 4.3.

Kode Kebutuhan	Deskripsi kebutuhan	Kualitas
NF01	Hanya akun pengguna yang teridentifikasi dan terotorisasi yang dapat menjalankan aplikasi	Security
NF02	Sistem dapat diakses 24 jam	Portability
NF03	Sistem dibuat dalam bentuk aplikasi berbasis web	Portability
NF04	Sistem hanya menampilkan data yang sudah tervalidasi	Reliability
NF05	Data hanya dapat diubah oleh admin	Security

Tabel 4.3 Kebutuhan Non-Fungsional

4.2 Diagram Kasus Penggunaan

Daftar kebutuhan fungsional dapat direpresentasikan menjadi diagram kasus penggunaan (Use Case Diagram) sehingga memudahkan untuk dipahami. Use Case Diagram yang telah dibuat dapat dilihat pada Gambar 4.1.



Gambar 4.1 Diagram Use Case Modul Scenario

4.3 Spesifikasi Kasus Penggunaan Aplikasi E-Learning

4.3.1 Membuka Halaman Awal

Tabel 4.4 berikut merupakan tabel use case dari Aplikasi Monitoring Kinerja FTEIC membuka halaman awal.

Nama	Membuka homepage
Kode	UC001
Deskripsi	Aktor dapat membuka homepage
Tipe	Fungsional

Pemicu	Aktor membuka homepage
Aktor	Seluruh pengguna (admin, user)
Kondisi Awal	-
Kondisi Akhir	Aktor dapat membuka homepage
Alur Kejadian Secara Normal	1. Aktor membuka homepage 2. Sistem menampilkan homepage
Alur Kejadian Alternatif	-
Pengecualian	-

Tabel 4.4 Tabel Use Case Membuka Halaman Awal

4.3.2 Melakukan Login

Tabel 4.5 berikut merupakan tabel use case dari Aplikasi Monitoring Kinerja FTEIC melakukan login.

Nama	Melakukan login
Kode	UC002
Deskripsi	Aktor dapat masuk ke halaman awal dengan akun sesuai role yang dimaksud
Tipe	Fungsional
Pemicu	Aktor menekan tombol ‘sign in’ setelah mengisi email dan password akun untuk admin dan user
Aktor	Seluruh pengguna (admin, user)
Kondisi Awal	Form login ditampilkan
Kondisi Akhir	Aktor memasuki halaman awal (homepage)
Alur Kejadian Secara Normal	1. Aktor mengisi form login 2. Sistem memeriksa field kosong pada form login 3. Sistem mencocokkan data login dengan database 4. Sistem menampilkan halaman awal
Alur Kejadian Alternatif	1. Sistem menampilkan pesan email atau password salah 2. Sistem menampilkan form login 3. Aktor mengisi kembali form login

	4. Sistem mencocokkan data dengan data pengguna pada basis data 5. Jika cocok, sistem menampilkan halaman awal untuk pengguna
Pengecualian	-

Tabel 4.5 Tabel Use Case Melakukan Login

4.3.3 Memasukan data baru

Tabel 4.6 berikut merupakan tabel use case dari Aplikasi Monitoring Kinerja FTEIC memasukan data baru.

Nama	Memasukan Data Baru
Kode	UC003
Deskripsi	Aktor dapat menambahkan data baru
Tipe	Fungsional
Pemicu	Aktor menekan tombol '*Tambah'* di salah satu pilihan halaman, mengisi form dan menekan tombol '*Submit*
Aktor	Seluruh pengguna (admin, user)
Kondisi Awal	Halaman Daftar Data ditampilkan
Kondisi Akhir	Aktor dapat menambahkan data baru
Alur Kejadian Secara Normal	1. Aktor menekan tombol '*Tambah*' di salah satu pilihan halaman 2. Sistem akan menampilkan form data baru 3. Aktor mengisi form data baru 4. Aktor menekan tombol '*Submit*' 5. Sistem memeriksa field kosong pada form data baru 6. Sistem menyimpan data yang telah diisi pada basis data 7. Sistem menampilkan pesan data baru telah berhasil di entry 8. Sistem menampilkan daftar data
Alur Kejadian Alternatif	1. Sistem menampilkan form data baru kembali

	2. Aktor mengisi kembali form pembuatan scenario 3. Aktor menekan tombol '*Submit*' 4. Sistem memeriksa field kosong pada form data baru 5. Sistem menyimpan data yang telah diisi pada basis data 6. Sistem menampilkan pesan data baru telah berhasil di entry 7. Sistem menampilkan daftar data
Pengecualian	-

Tabel 4.6 Tabel Use Case Memasukan Data Baru

4.3.4 Membuat User Baru

Tabel 4.7 berikut merupakan tabel use case dari Aplikasi Monitoring Kinerja FTEIC membuat user baru.

Nama	Membuat User Baru
Kode	UC004
Deskripsi	Aktor dapat membuat user baru
Tipe	Fungsional
Pemicu	Aktor menekan tombol '*Tambah Pengguna Baru*' di halaman user
Aktor	Admin
Kondisi Awal	Halaman Awal ditampilkan
Kondisi Akhir	Aktor dapat menambah pengguna baru
Alur Kejadian Secara Normal	1. Aktor menekan tombol '*Tambah Pengguna Baru*' di halaman user 2. Sistem akan menampilkan form pembuatan user baru 3. Aktor mengisi form pembuatan user baru 4. Aktor menekan tombol '*Tambah*' 5. Sistem memeriksa field kosong pada form pembuatan user baru

	6. Sistem menyimpan data yang telah diisi pada basis data 7. Sistem menampilkan pesan pembuatan user baru telah berhasil 8. Sistem menampilkan halaman awal
Alur Kejadian Alternatif	1. Menampilkan form pembuatan user baru 2. Aktor mengisi kembali form pembuatan user baru 3. Aktor menekan tombol '*Tambah* 4. Sistem memeriksa field kosong pada form pembuatan user baru 5. Sistem menyimpan data yang telah diisi pada basis data 6. Sistem menampilkan pesan pembuatan user baru telah berhasil 7. Sistem menampilkan halaman awal
Pengecualian	-

Tabel 4.7 Tabel Use Case Membuat User Baru

4.3.5 Memvalidasi Data Baru

Tabel 4.8 berikut merupakan tabel use case dari Aplikasi Monitoring Kinerja FTEIC memvalidasi data baru.

Nama	Memvalidasi Data Baru
Kode	UC005
Deskripsi	Aktor dapat memvalidasi data baru
Tipe	Fungsional
Pemicu	Aktor menekan tombol '*Validasi*' di salah satu halaman data
Aktor	Admin
Kondisi Awal	Halaman data ditampilkan
Kondisi Akhir	Aktor memasuki halaman data
Alur Kejadian Secara	1. Sistem akan menampilkan data yang akan divalidasi

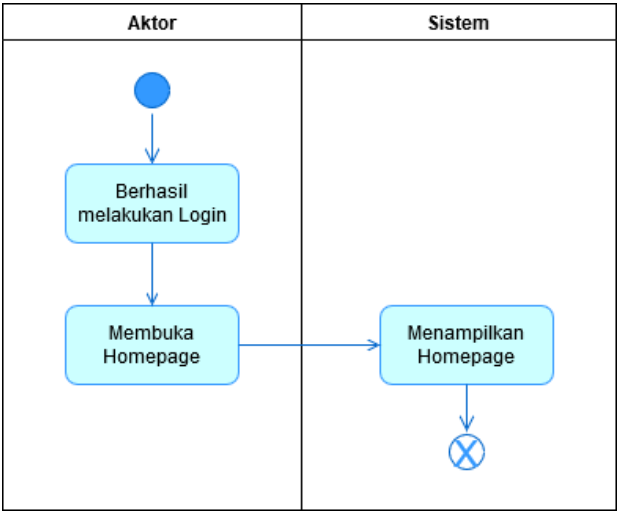
Normal	2. Aktor menekan tombol '*Validasi*' di halaman data 3. Aktor menekan tombol '*Ya'* 4. Sistem menampilkan pesan data sudah tervalidasi 5. Sistem menampilkan halaman data
Alur Kejadian Alternatif	1. Aktor menutup halaman data 2. Sistem menampilkan halaman data
Pengecualian	-

Tabel 4.8 Tabel Use Case Memvalidasi Data Baru

4.4 Diagram Aktivitas

4.4.1 Membuka Halaman Awal

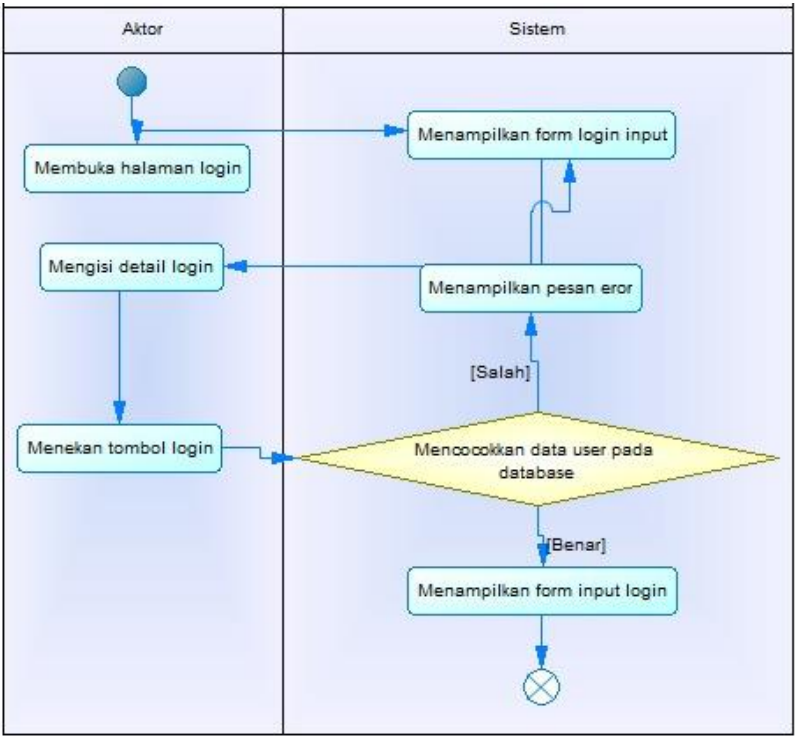
Gambar 4.2 berikut merupakan Diagram Aktivitas dari Aplikasi Monitoring Kinerja FTEIC untuk membuka halaman awal.



Gambar 4.2 Diagram Aktivitas Membuka Halaman Awal

4.4.2 Melakukan Login

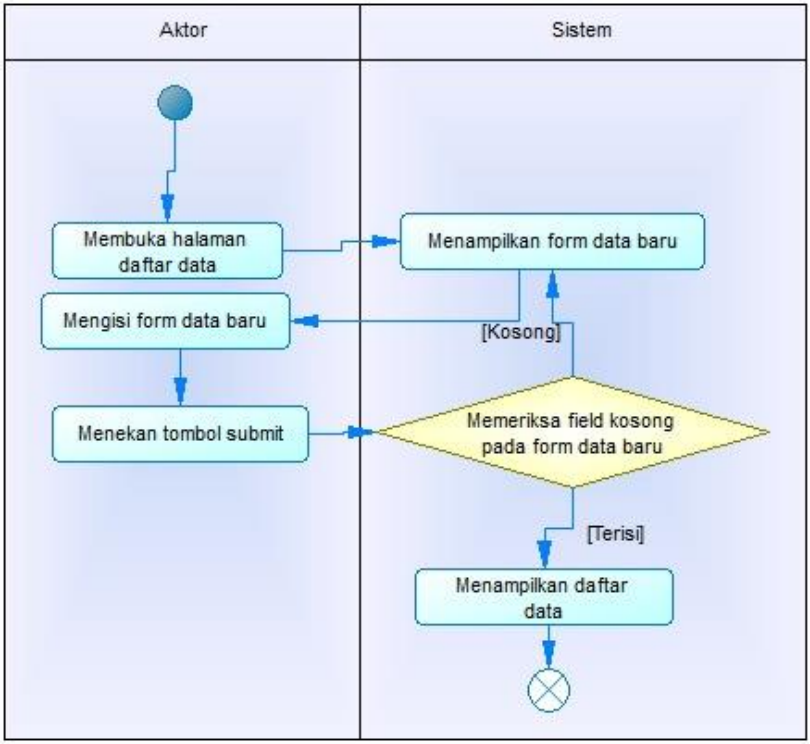
Gambar 4.3 berikut merupakan Diagram Aktivitas dari Aplikasi Monitoring Kinerja FTEIC untuk melakukan login.



Gambar 4.3 Diagram Aktivitas Melakukan Login

4.4.3 Memasukan Data Baru

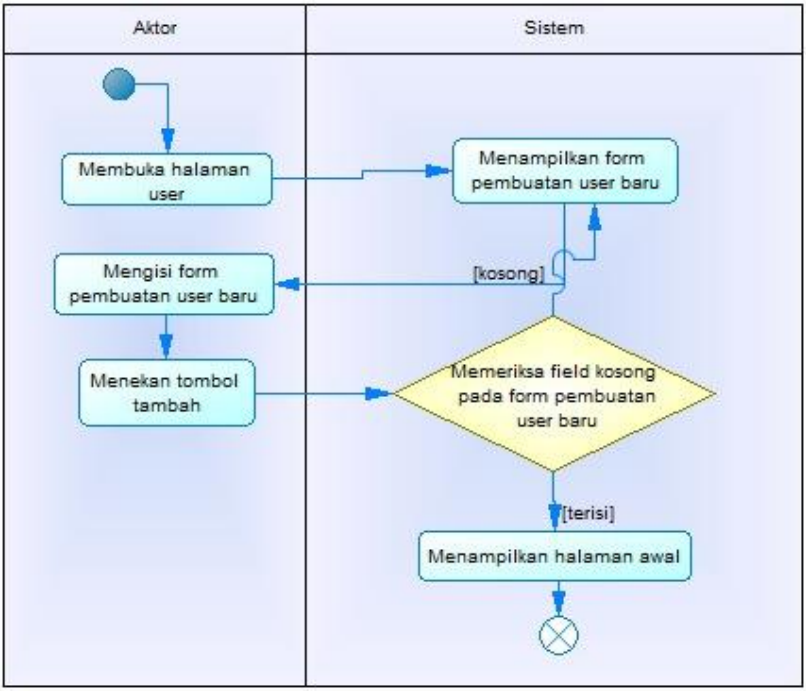
Gambar 4.4 berikut merupakan Diagram Aktivitas dari Aplikasi Monitoring Kinerja FTEIC untuk memasukan data baru.



Gambar 4.4 Diagram Aktivitas Memasukan Data Baru

4.4.4 Membuat User Baru

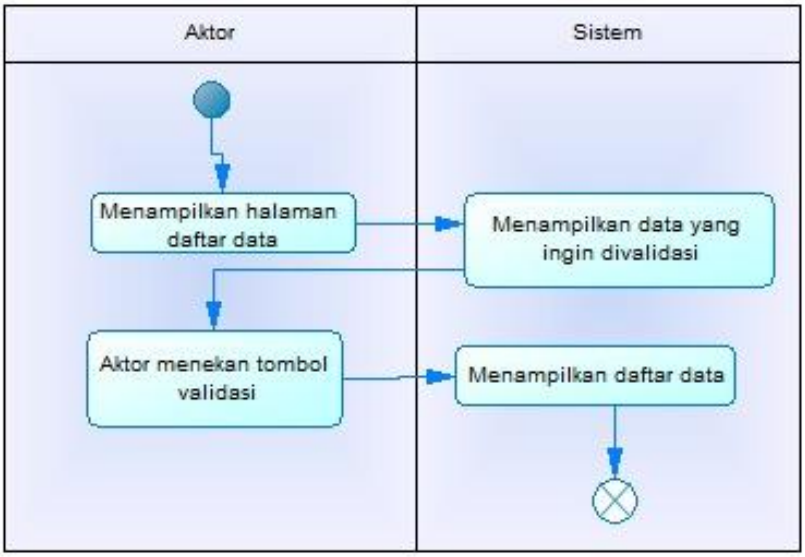
Gambar 4.5 berikut merupakan Diagram Aktivitas dari Aplikasi Monitoring Kinerja FTEIC untuk membuat user baru.



Gambar 4.5 Diagram Aktivitas Membuat User Baru

4.4.5 Memvalidasi Data Baru

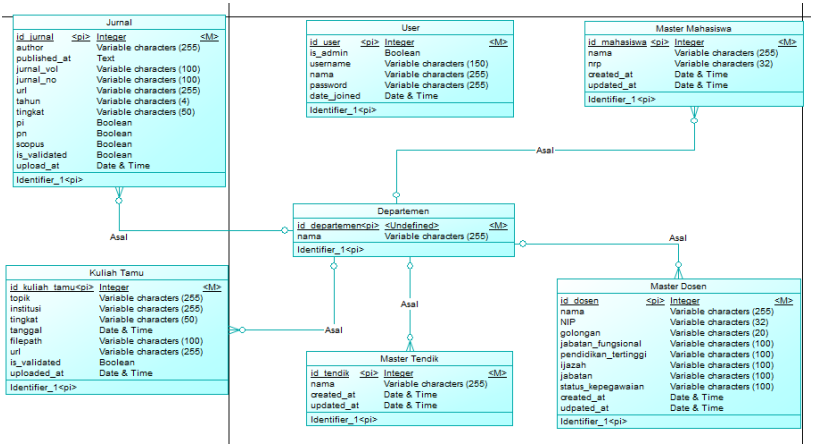
Gambar 4.6 berikut merupakan Diagram Aktivitas dari Aplikasi Monitoring Kinerja FTEIC untuk memvalidasi data baru.



Gambar 4.6 Diagram Aktivitas Memvalidasi Data Baru

4.5 *Conceptual Data Model*

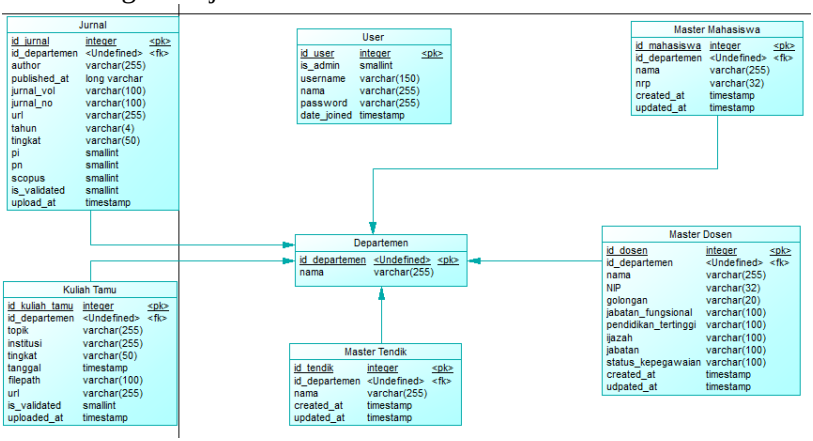
Gambar 4.7 berikut adalah *Conceptual Data Model* dari Aplikasi Monitoring Kinerja FTEIC.



Gambar 4.7 Conceptual Data Model Aplikasi Monitoring Kinerja FTEIC

4.6 *Physical Data Model*

Gambar 4.8 berikut adalah *Physical Data Model* dari Aplikasi Monitoring Kinerja FTEIC



Gambar 4.8 Physical Data Model Aplikasi Monitoring Kinerja FTEIC

4.5. Struktur Tabel

Berikut adalah struktur tabel pada Aplikasi Monitoring Kinerja FTEIC meliputi nama tabel, nama atribut, tipe data dan keterangan atribut.

Tabel 4.9 berikut adalah struktur tabel users

Users			
No	Nama Atribut	Tipe Data	Keterangan
1.	id_user	Integer	Primary key dan auto increment
2.	is_admin	Boolean	Status admin user
3.	username	Variable Character (150)	Username user
4.	nama	Variable Character (255)	Nama user
5.	password	Variable Character (255)	Password user
6.	date_joined	Date Time	Tanggal pembuatan akun

Tabel 4.9 Struktur Tabel Users

Tabel 4.10 berikut adalah struktur tabel departemen.

Departemen			
No	Nama Atribut	Tipe Data	Keterangan
1.	id_departemen	Integer	Primary key dan auto increment
2.	Nama	Variable Character (255)	Nama departemen

Tabel 4.10 Struktur Tabel Departemen

Tabel 4.11 berikut adalah struktur tabel master dosen.

Master Dosen			
No	Nama Atribut	Tipe Data	Keterangan
1.	id_dosen	Integer	Primary key dan auto increment

2.	nama	Variable Character (255)	Nama dosen
3.	NIP	Variable Character (32)	NIP dosen
4.	golongan	Variable Character (20)	Golongan dosen
5.	jabatan_fungsional	Variable Character (100)	Jabatan fungsional dosen
6.	pendidikan_tertinggi	Variable Character (100)	Pendidikan tertinggi dosen
7.	ijazah	Variable Character (100)	Ijazah terakhir dosen
8.	departemen	Integer	ID Departemen
9.	jabatan	Variable Character (100)	Jabatan dosen
10.	status_kepegawaian	Variable Character (100)	Status kepegawaian dosen
11.	created_at	Date Time	Tanggal dibuat
12.	updated_at	Date Time	Tanggal diubah

Tabel 4.11 Struktur Tabel Master Dosen

Tabel 4.12 berikut adalah struktur tabel master mahasiswa.

Master Mahasiswa			
No	Nama Atribut	Tipe Data	Keterangan
1.	id_mahasiswa	Integer	Primary key dan auto increment
2.	nama	Variable Character (255)	Nama mahasiswa
3.	nrp	Variable Character (14)	NRP mahasiswa
4.	Departemen	Integer	ID departemen
5.	created_at	Date Time	Tanggal dibuat
6.	updated_at	Date Time	Tanggal diubah

Tabel 4.12 Struktur Tabel Users

Tabel 4.13 berikut adalah struktur tabel master tendik.

Master Tendik			
No	Nama Atribut	Tipe Data	Keterangan
1.	id_tendik	Integer	Primary key dan auto increment
2.	nama	Variable Character (255)	Nama tendik
3.	Departemen	Integer	ID departemen
4.	created_at	Date Time	Tanggal dibuat
5.	updated_at	Date Time	Tanggal diubah

Tabel 4.13 Struktur Tabel Users

Tabel 4.14 berikut adalah struktur tabel jurnal.

Jurnal			
No	Nama Atribut	Tipe Data	Keterangan
1.	id_jurnal	Integer	Primary key dan auto increment
2.	author	Variable Character (255)	Penulis jurnal
3.	published_at	Text	Tempat publikasi jurnal
4.	jurnal_vol	Variable Character (100)	Volume jurnal
5.	jurnal_no	Variable Character (100)	Nomor jurnal
6.	url	Variable Character (255)	Alamat URL jurnal
7.	tahun	Variable Character (4)	Tahun terbit jurnal
8.	tingkat	Variable Character (50)	Tingkat jurnal
9.	pi	Boolean	Pi dari jurnal
10.	pn	Boolean	Pn dari jurnal
11.	scopus	Boolean	Scopus dari jurnal
12.	is_validated	Boolean	Status tervalidasi
13.	upload_at	Date Time	Tanggal diupload
14.	departemen	Integer	ID departemen

Tabel 4.14 Struktur Tabel Jurnal

Tabel 4.15 berikut adalah struktur tabel kuliah tamu.

Session			
No	Nama Atribut	Tipe Data	Keterangan
1.	id_kuliah_tamu	Integer	Primary key dan auto increment
2.	topik	Variable Character (255)	Nama topik kuliah tamu
3.	institusi	Variable Character (255)	Tempat diadakan kuliah tamu
4.	tingkat	Variable Character (100)	Tingkat kuliah tamu
5.	tanggal	Date	Tanggal kuliah tamu
6.	filepath	Variable Character (100)	Letak file
7.	departemen	Integer	ID Departemen
8.	url	Variable Character (255)	Alamat URL Kuliah tamu
9.	Is_validated	Boolean	Status tervalidasi
10.	uploaded_at	Date Time	Tanggal diupload

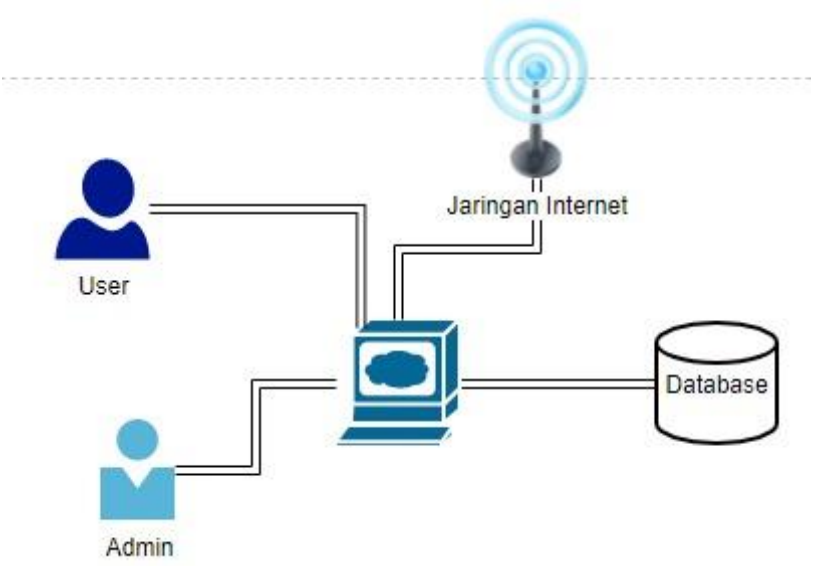
Tabel 4.15 Struktur Tabel Kuliah Tamu

BAB V
IMPLEMENTASI SISTEM

Bab ini membahas tentang implementasi dari perancangan sistem dan pengaplikasian sistem dalam bentuk situs web. Implementasi ini terbagi menjadi 4, yaitu implementasi arsitektur sistem, implementasi lapisan control dan implementasi antarmuka pengguna.

5.1 Implementasi Arsitektur Sistem

Pada bagian ini akan digambarkan arsitektur sistem. Adapun diagram arsitektur sistem yang diterapkan pada Aplikasi Monitoring Kinerja FTEIC terlihat pada gambar 5.1.



Gambar 5.1 Diagram Arsitektur Sistem

5.2 Implementasi Lapisan Kontrol dan Lapisan Model

Implementasi lapisan kontrol berisi logika yang digunakan aplikasi seperti kontrol untuk memasukkan data ke database.

5.2.1 Lapisan Kontrol Aplikasi Monitoring Kinerja FTEIC

5.2.1.1 User Controller

Lapisan kontrol ini berguna untuk mengatur data pada user dan autentikasi user untuk melakukan login. Berikut adalah potongan kode user controller.

```
class ObtainAuthToken(APIView):
    throttle_classes = ()
    permission_classes = ()
    parser_classes = (parsers.FormParser,
parsers.MultiPartParser, parsers.JSONParser,)
    renderer_classes = (renderers.JSONRenderer,)
    serializer_class = AuthTokenSerializer
    if coreapi is not None and coreschema is not None:
        schema = ManualSchema(
            fields=[
                coreapi.Field(
                    name="username",
                    required=True,
                    location='form',
                    schema=coreschema.String(
                        title="Username",
                        description="Valid username for
authentication",
                    ),
                ),
                coreapi.Field(
                    name="password",
                    required=True,
                    location='form',
                    schema=coreschema.String(
                        title="Password",
                        description="Valid password for
authentication",
                    ),
                ),
            ],
            encoding="application/json",
        )

    def post(self, request, *args, **kwargs):
        serializer =
self.serializer_class(data=request.data,

context={'request': request})
        serializer.is_valid(raise_exception=True)
        user = serializer.validated_data['user']
        token, created =
Token.objects.get_or_create(user=user)
        return Response({'token': token.key,
'is admin': user.is_admin, 'nama': user.get_nama()})
```

```
class UserInfoView(APIView):
    permission_classes = (IsAuthenticated,)

    def get(self, request):
        user =
User.objects.get(username=request.user.username)
        serializer = UserSerializer(user)
        return Response(serializer.data)

class UserChangePasswordView(APIView):
    permission_classes = (IsAuthenticated, )

    def post(self, request):
        serializer =
ChangePasswordSerializer(data=request.data,
context={'request': request})
        if serializer.is_valid():

request.user.set_password(serializer.validated_data['ne
w_password'])
            request.user.save()
            return Response(status=status.HTTP_200_OK)

        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)
```

5.2.1.2 Departemen Controller

Lapisan kontrol ini berguna untuk mengatur data pada departemen. Berikut adalah potongan kode departemen controller.

```
class DepartemenAPIView(APIView):
    def get(self, request):
        datas = Departemen.objects.all()
        serializer = DepartemenSerializer(datas,
many=True)
        return Response(serializer.data)
```

5.2.1.3 Master Dosen Controller

Lapisan kontrol ini berguna untuk mengatur data pada dosen. Berikut adalah potongan kode master dosen controller.

```
class MasterDosenAPIView(APIView):
    permission_classes = (IsAuthenticated, )

    def get(self, request):
        datas = MasterDosen.objects.all()
        serializer = MasterDosenSerializer(datas,
many=True)
        return Response(serializer.data)

    def post(self, request):
        serializer =
MasterDosenSerializer(data=request.data)

        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data,
status=status.HTTP_201_CREATED)

        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)

class MasterDosenDetailsAPIView(APIView):
    permission_classes = (IsAuthenticated, )

    def get_object(self, pk):
        try:
            return MasterDosen.objects.get(pk=pk)
        except MasterDosen.DoesNotExist:
            raise NotFound

    def get(self, request, pk):
        data = self.get_object(pk)
        serializer = MasterDosenSerializer(data)
        return Response(serializer.data)

    def put(self, request, pk):
        data = self.get_object(pk)
        serializer = MasterDosenSerializer(data,
data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data)

        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)

    def delete(self, request, pk):
        data = self.get_object(pk)
        data.delete()
        return
Response(status=status.HTTP_204_NO_CONTENT)
```

5.2.1.4 Master Mahasiswa Controller

Lapisan kontrol ini berguna untuk mengatur data pada mahasiswa. Berikut adalah potongan kode master mahasiswa controller.

```
class MasterMahasiswaAPIView(APIView):
    permission_classes = (IsAuthenticated, )

    def get(self, request):
        datas = MasterMahasiswa.objects.all()
        serializer = MasterMahasiswaSerializer(datas,
many=True)
        return Response(serializer.data)
    def post(self, request):
        serializer =
MasterMahasiswaSerializer(data=request.data)

        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data,
status=status.HTTP_201_CREATED)

        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)


class MasterMahasiswaDetailsAPIView(APIView):
    permission_classes = (IsAuthenticated, )

    def get_object(self, pk):
        try:
            return MasterMahasiswa.objects.get(pk=pk)
        except MasterMahasiswa.DoesNotExist:
            raise NotFound

    def get(self, request, pk):
        data = self.get_object(pk)
        serializer = MasterMahasiswaSerializer(data)
        return Response(serializer.data)

    def put(self, request, pk):
        data = self.get_object(pk)
        serializer = MasterMahasiswaSerializer(data,
data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data)

        return Response(serializer.errors,
status=status.HTTP 400 BAD REQUEST)
```

```
    def delete(self, request, pk):
        data = self.get_object(pk)
        data.delete()
        return
Response(status=status.HTTP_204_NO_CONTENT)
```

5.2.1.5 Master Tendik Controller

Lapisan kontrol ini berguna untuk mengatur data pada tendik. Berikut adalah potongan kode master tendik controller.

```
class MasterTendikAPIView(APIView):
    permission_classes = (IsAuthenticated, )

    def get(self, request):
        datas = MasterTendik.objects.all()
        serializer = MasterTendikSerializer(datas,
many=True)
        return Response(serializer.data)

    def post(self, request):
        serializer =
MasterTendikSerializer(data=request.data)

        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data,
status=status.HTTP_201_CREATED)

        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)

class MasterTendikDetailsAPIView(APIView):
    permission_classes = (IsAuthenticated, )

    def get_object(self, pk):
        try:
            return MasterTendik.objects.get(pk=pk)
        except MasterTendik.DoesNotExist:
            raise NotFound

    def get(self, request, pk):
        data = self.get_object(pk)
        serializer = MasterTendikSerializer(data)
        return Response(serializer.data)
```

```
    def put(self, request, pk):
        data = self.get_object(pk)
        serializer = MasterTendikSerializer(data,
data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data)

        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)

    def delete(self, request, pk):
        data = self.get_object(pk)
        data.delete()
        return
Response(status=status.HTTP_204_NO_CONTENT)
```

5.2.1.6 Jurnal Controller

Lapisan kontrol ini berguna untuk mengatur data pada jurnal. Berikut adalah potongan kode jurnal controller.

```
class JurnalViewSet(viewsets.ViewSet):
    parser_classes = (MultiPartParser, FormParser)

    def get_all_objects(self, request):
        if request.user.is_admin:
            queryset = Jurnal.objects.all()
        else:
            queryset =
Jurnal.objects.filter(is_validated=True)
        return queryset

    def get_permissions(self):
        if self.action == 'destroy':
            permission_classes = [isAdminPermission]
        else:
            permission_classes = [IsAuthenticated]
        return [permission() for permission in
permission_classes]

    def list(self, request):
        jurnals = self.get_all_objects(request)
        serializer = JurnalSerializer(jurnals,
many=True)
        return Response(serializer.data)
```

```
    def create(self, request):
        serializer =
JurnalSerializer(data=request.data)

        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data,
status=status.HTTP_201_CREATED)

        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)

    def retrieve(self, request, pk=None):
        queryset = self.get_all_objects(request)
        jurnal = get_object_or_404(queryset, pk=pk)
        serializer = JurnalSerializer(jurnal)
        return Response(serializer.data)

    def destroy(self, request, pk=None):
        queryset = Jurnal.objects.all()
        jurnal = get_object_or_404(queryset, pk=pk)
        jurnal.delete()
        return
Response(status=status.HTTP_204_NO_CONTENT)

# Validated API
@api_view(['POST'])
@permission_classes([isAdminPermission])
def set_validate(request, pk):
    if request.method == 'POST':
        queryset = Jurnal.objects.all()
        jurnal = get_object_or_404(queryset, pk=pk)
        if not jurnal.is_validated:
            jurnal.is_validated = True
            jurnal.save()
        return Response(status=status.HTTP_200_OK)

Filtering API
class JurnalList(ListAPIView):
    permission_classes = (IsAuthenticated, )
    serializer_class = JurnalSerializer
    queryset = Jurnal.objects.all()
    filter_backends = [DjangoFilterBackend]
    filter_fields = [
        'judul', 'author', 'published_at',
        'jurnal_vol', 'jurnal_no', 'url',
        'tahun', 'tingkat', 'pi', 'pn',
        'scopus', 'departemen'
    ]
```

```
# Export Excel API
@api_view(['GET'])
@permission_classes([IsAuthenticated])
def export_data(request):
    response =
HttpResponse(content_type='application/vnd.openxmlform
ats-officedocument.spreadsheetml.sheet')
    response['Content-Disposition'] = 'attachment;
filename={date}-
jurnal.xlsx'.format(date=datetime.now().strftime('%d-
%m-%Y'))

    if request.user.is_admin:
        queryset = Jurnal.objects.all()
    else:
        queryset =
Jurnal.objects.filter(is_validated=True)

    param = request.GET
    if 'tahun' in param:
        queryset =
queryset.filter(tahun=param['tahun'])
    if 'departemen' in param:
        queryset =
queryset.filter(departemen=param['departemen'])

    workbook = Workbook()

    # Get active worksheet/tab
    worksheet = workbook.active
    worksheet.title = 'Jurnal'

    # Define the titles for columns
    columns = [
        'Judul',
        'Author',
        'Dipublish di',
        'Volume',
        'Nomor',
        'Url',
        'Tahun',
        'Tingkat',
        'pi',
        'pn',
        'Scopus',
        'Departemen'
    ]
    row_num = 1
```

```
    # Assign the titles for each cell of the header
    for col_num, column_title in enumerate(columns,
1):
        cell = worksheet.cell(row=row_num,
column=col_num)
        cell.value = column_title

    # Iterate through all kuliah tamu data
    for data in queryset:
        row_num += 1

        # Define the data for each cell in the row
        row = [
            data.judul,
            data.author,
            data.published_at,
            data.jurnal_vol,
            data.jurnal_no,
            data.url,
            data.tahun,
            data.tingkat,
            int(data.pi),
            int(data.pn),
            int(data.scopus),
            data.departemen.nama
        ]

        # Assign the data for each cell of the row
        for col_num, cell_value in enumerate(row, 1):
            cell = worksheet.cell(row=row_num,
column=col_num)
            cell.value = cell_value

    workbook.save(response)

    return response
```

5.2.1.7 Kuliah Tamu Controller

Lapisan kontrol ini berguna untuk mengatur data pada kuliah tamu. Berikut adalah potongan kode kuliah tamu controller.

```
class KuliahTamuViewSet(viewsets.ViewSet):
    parser_classes = (MultiPartParser, FormParser)

    def get_all_objects(self, request):
        if request.user.is_admin:
            queryset = KuliahTamu.objects.all()
        else:
            queryset =
KuliahTamu.objects.filter(is_validated=True)
        return queryset

    def get_permissions(self):
        if self.action == 'destroy':
            permission_classes = [isAdminPermission]
        else:
            permission_classes = [IsAuthenticated]
        return [permission() for permission in
permission_classes]

    def list(self, request):
        queryset = self.get_all_objects(request)
        serializer = KuliahTamuSerializer(queryset,
many=True)
        return Response(serializer.data)

    def create(self, request):
        serializer =
KuliahTamuSerializer(data=request.data)

        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data,
status=status.HTTP_201_CREATED)

        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)

    def retrieve(self, request, pk=None):
        queryset = self.get_all_objects(request)
        kultam = get_object_or_404(queryset, pk=pk)
        serializer = KuliahTamuSerializer(kultam)
        return Response(serializer.data)

    def destroy(self, request, pk=None):
        queryset = KuliahTamu.objects.all()
        kultam = get_object_or_404(queryset, pk=pk)
        kultam.delete()
        return
Response(status=status.HTTP_204_NO_CONTENT)
```

```
# Validated API
@api_view(['POST'])
@permission_classes([isAdminPermission])
def set_validate(request, pk):
    if request.method == 'POST':
        queryset = KuliahTamu.objects.all()
        kultam = get_object_or_404(queryset, pk=pk)
        if not kultam.is_validated:
            kultam.is_validated = True
            kultam.save()
        return Response(status=status.HTTP_200_OK)

# Filtering API
class KuliahTamuList(ListAPIView):
    permission_classes = (IsAuthenticated, )
    serializer_class = KuliahTamuSerializer
    queryset = KuliahTamu.objects.all()
    filter_backends = [DjangoFilterBackend]
    filter_fields = ['topik', 'pemateri', 'institusi',
'tingkat', 'tanggal', 'departemen', 'url']

# Export Excel API
@api_view(['GET'])
@permission_classes([IsAuthenticated])
def export_data(request):
    response =
HttpResponse(content_type='application/vnd.openxmlform
ats-officedocument.spreadsheetml.sheet')
    response['Content-Disposition'] = 'attachment;
filename={date}-kuliah-
tamu.xlsx'.format(date=datetime.now().strftime('%d-%m-
%Y'))

    if request.user.is_admin:
        queryset = KuliahTamu.objects.all()
    else:
        queryset =
KuliahTamu.objects.filter(is_validated=True)

    param = request.GET
    if 'tahun' in param:
        queryset =
queryset.filter(tanggal__year=param['tahun'])
    if 'departemen' in param:
        queryset =
queryset.filter(departemen=param['departemen'])

    workbook = Workbook()
```

```
    # Get active worksheet/tab
    worksheet = workbook.active
    worksheet.title = 'Kuliah Tamu'

    # Define the titles for columns
    columns = [
        'Topik',
        'Pemateri',
        'Institusi',
        'Tingkat',
        'Tanggal',
        'Departemen',
        'URL'
    ]
    row_num = 1

    # Assign the titles for each cell of the header
    for col_num, column_title in enumerate(columns,
1):
        cell = worksheet.cell(row=row_num,
column=col_num)
        cell.value = column_title

    # Iterate through all kuliah tamu data
    for data in queryset:
        row_num += 1

        # Define the data for each cell in the row
        row = [
            data.topik,
            data.pemateri,
            data.institusi,
            data.tingkat,
            data.tanggal,
            data.departemen.nama,
            data.url
        ]

        # Assign the data for each cell of the row
        for col_num, cell_value in enumerate(row, 1):
            cell = worksheet.cell(row=row_num,
column=col_num)
            cell.value = cell_value

    workbook.save(response)

    return response
```

5.2.2 Lapisan Model Aplikasi Monitoring Kinerja FTEIC

5.2.2.1 User Model

Lapisan ini berguna untuk mengelola user. Berikut adalah potongan code dari User Model.

```
from django.db import models
from django.contrib.auth.models import
AbstractBaseUser
from django.contrib.auth.models import
PermissionsMixin
from django.utils.translation import gettext_lazy as _
from django.utils import timezone

from .managers import UserManager
from apps.masters.models import (MasterMahasiswa,
                                 MasterDosen,
                                 MasterTendik)

class User(AbstractBaseUser, PermissionsMixin):
    username = models.CharField(max_length=150,
unique=True)
    is_admin = models.BooleanField(default=False)
    is_active = models.BooleanField(default=True)

    mahasiswa = models.OneToOneField(MasterMahasiswa,
on_delete=models.CASCADE, null=True, blank=True)
    dosen = models.OneToOneField(MasterDosen,
on_delete=models.CASCADE, null=True, blank=True)
    tendik = models.OneToOneField(MasterTendik,
on_delete=models.CASCADE, null=True, blank=True)
    date_joined =
models.DateTimeField(auto_now_add=True)

    objects = UserManager()

    USERNAME_FIELD = 'username'
    REQUIRED_FIELDS = []

    def __str__(self):
        return self.username
```

5.2.2.2 Master Model

Lapisan ini berguna untuk mengelola data departemen, master dosen, mahasiswa, dan tendik. Berikut adalah potongan code dari Master Model.

```
from django.db import models

class Departemen(models.Model):
    nama = models.CharField(max_length=100)

    def __str__(self):
        return self.nama

class MasterDosen(models.Model):
    nama = models.CharField(max_length=255)
    NIP = models.CharField(max_length=32, unique=True)
    golongan = models.CharField(max_length=20,
blank=True)
    jabatan_fungsional =
models.CharField(max_length=100, blank=True)
    pendidikan_tertinggi =
models.CharField(max_length=100, blank=True)
    ijazah = models.CharField(max_length=100,
blank=True)
    departemen = models.ForeignKey(Departemen,
on_delete=models.CASCADE, null=True)
    jabatan = models.CharField(max_length=100,
null=True, blank=True)
    status_kepegawaian =
models.CharField(max_length=100, blank=True)
    created_at =
models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return self.nama

class MasterTendik(models.Model):
    nama = models.CharField(max_length=255)
    departemen = models.ForeignKey(Departemen,
on_delete=models.CASCADE)
    created_at =
models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)
```

```
class MasterMahasiswa(models.Model):
    nama = models.CharField(max_length=255)
    NRP = models.CharField(max_length=14, unique=True)
    departemen = models.ForeignKey(Departemen,
on_delete=models.CASCADE)
    created_at =
models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return self.nama
```

5.2.2.3 Jurnal Model

Lapisan ini berguna untuk mengelola data jurnal. Berikut adalah potongan code dari Jurnal Model.

```
from django.db import models
from ..masters.models import Departemen

# Create your models here.
class Jurnal(models.Model):
    judul = models.CharField(max_length=255)
    author = models.CharField(max_length=255)
    published_at = models.TextField()
    jurnal_vol = models.CharField(max_length=100)
    jurnal_no = models.CharField(max_length=100)
    url = models.CharField(max_length=255)
    tahun = models.CharField(max_length=4)
    tingkat = models.CharField(max_length=50)
    pi = models.BooleanField(default=False)
    pn = models.BooleanField(default=False)
    scopus = models.BooleanField(default=False)
    is_validated = models.BooleanField(default=False)
    uploaded_at = models.DateTimeField(auto_now=True)
    departemen = models.ForeignKey(to=Departemen,
on_delete=models.DO_NOTHING, null=True)

    def __str__(self):
        return self.judul
```

5.2.2.4 Kuliah Tamu Model

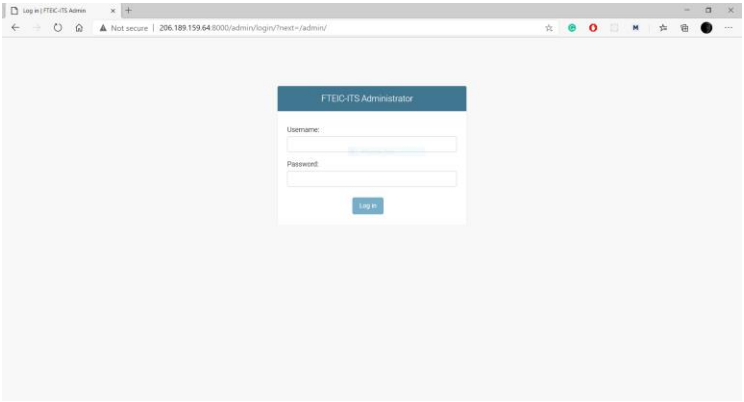
Lapisan ini berguna untuk mengelola data departemen, master dosen, mahasiswa, dan tendik. Berikut adalah potongan code dari Master Model.

```
from django.db import models
from ..masters.models import Departemen

class KuliahTamu(models.Model):
    topik = models.CharField(max_length=255)
    pemateri = models.TextField()
    institusi = models.CharField(max_length=255)
    tingkat = models.CharField(max_length=100)
    tanggal = models.DateField()
    filepath =
models.FileField(upload_to='kuliahtamu/',null=True)
    departemen = models.ForeignKey(to=Departemen,
on_delete=models.DO_NOTHING)
    url = models.CharField(max_length=255, null=True)
    is_validated = models.BooleanField(default=False)
    uploaded_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return self.topik
```

5.3 Implementasi Antarmuka Pengguna

5.3.1 Halaman Login

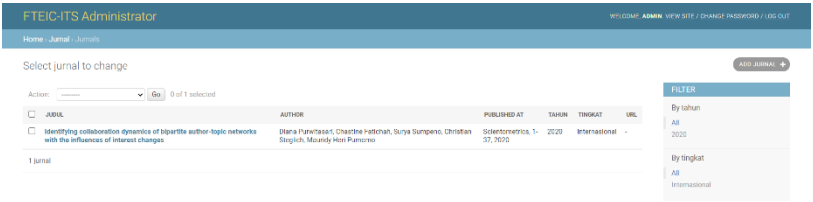
Gambar 5.2 berisi tampilan antarmuka login, dengan form berisi username dan password, serta tombol ‘Sign In’ agar pengguna dapat melakukan login ke sistem menggunakan hak akses yang sesuai.



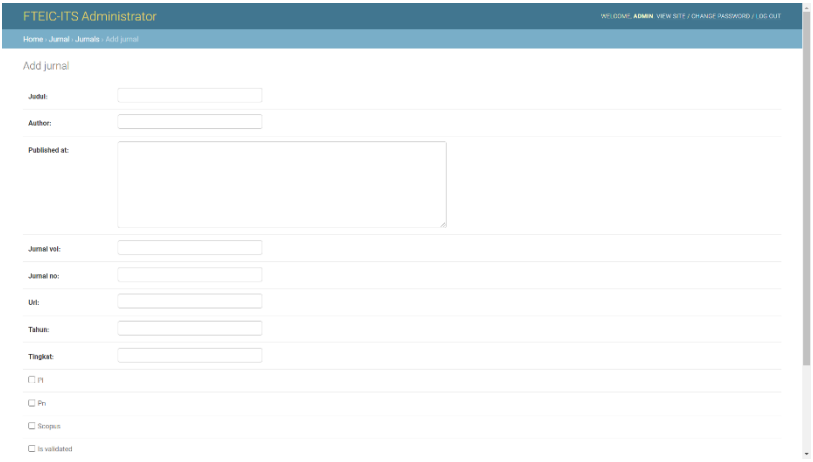
Gambar 5.2 Halaman Login

5.3.2 Halaman Jurnal

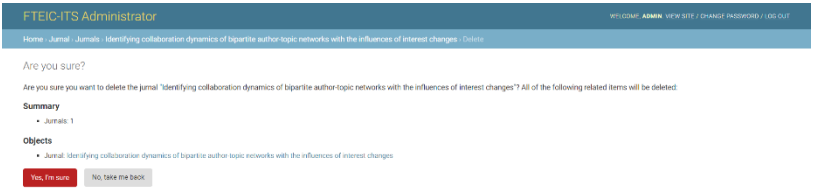
Berikut tampilan antarmuka jurnal, dengan menampilkan opsi menambahkan, melihat data jurnal, merubah, dan menghapus.



Gambar 5.3 Halaman Awal Jurnal



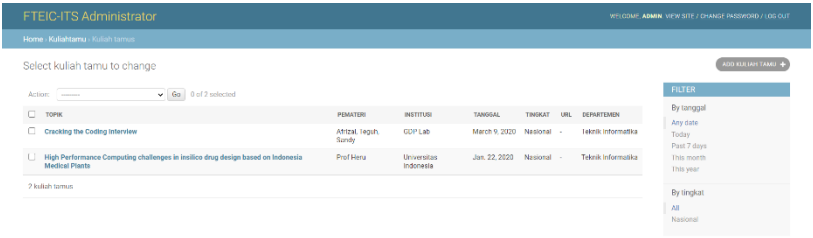
Gambar 5.4 Form Menambahkan Jurnal



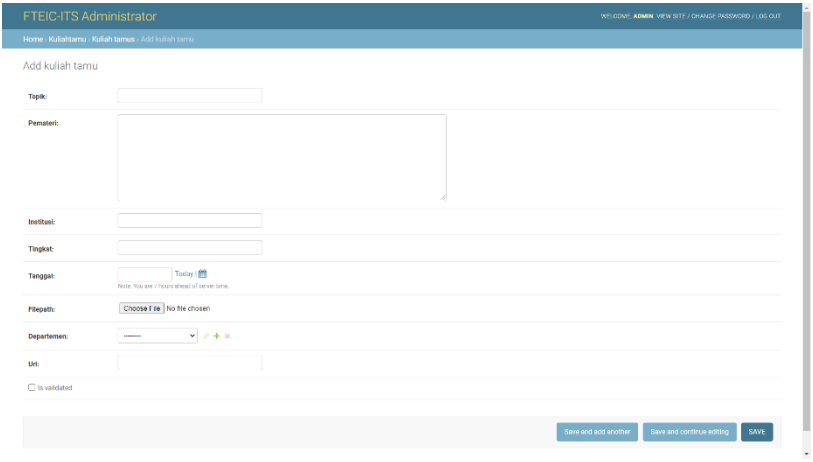
Gambar 5.5 Halaman Hapus Jurnal

5.3.3 Halaman Kuliah Tamu

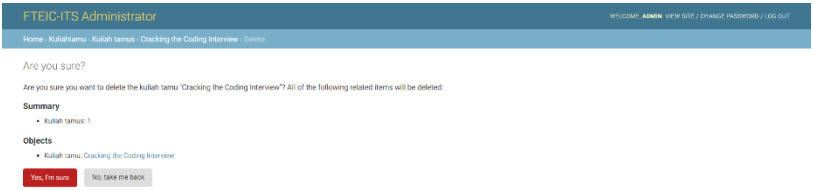
Berikut tampilan antarmuka kuliah tamu, dengan menampilkan opsi menambahkan, melihat data kuliah tamu, merubah, dan menghapus.



Gambar 5.6 Halaman Awal Kuliah Tamu



Gambar 5.7 Form Menambahkan Kuliah Tamu



Gambar 5.8 Halaman Hapus Kuliah Tamu

5.3.4 Halaman Login

Berikut tampilan antarmuka logout, dengan menampilkan bukti bahwa user telah melakukan logout

Gambar 5.9 Halaman Logout

[Halaman ini sengaja dikosongkan]

BAB VI
PENGUJIAN DAN EVALUASI

Bab ini menjelaskan tahap uji coba dilakukan terhadap Aplikasi Monitoring Kinerja FTEIC. Pengujian dilakukan untuk memastikan kualitas perangkat lunak yang dibangun dan kesesuaian hasil eksekusi perangkat lunak dengan analisis dan perancangan perangkat lunak.

6.1. Tujuan Pengujian

Pengujian dilakukan terhadap Aplikasi Monitoring Kinerja FTEIC guna menguji kesesuaian dan ketepatan fungsionalitas dari seluruh sistem aplikasi.

6.2. Skenario Pengujian

6.2.1 Login Pengguna

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka aplikasi pada browser.
- Memasukkan *username* dan *password* lalu klik tombol LOGIN
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu melihat halaman dashboard yang berisi tombol kuliah tamu, konferensi / jurnal, prestasi dosen, training, dan sertifikasi.

6.2.2 Logout Pengguna

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka halaman dashboard pada browser.
- Menekan tombol *logout* di *dropdown*.
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu melihat halaman login.

6.2.3 Membuka Halaman Jurnal

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka /dashboard pada browser.
- Menekan tombol Jurnal.
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu melihat halaman Jurnal

yang berisi daftar Jurnal, *input* untuk pencarian, *filter* pencarian, dan tombol tambah.

6.2.4 Membuka Halaman Detail Jurnal

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka /dashboard pada browser.
- Menekan tombol Jurnal.
- Menekan salah satu daftar Jurnal.
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu melihat halaman detail Jurnal yang berisi judul kegiatan, departemen, URL, konf hal, *check box* scopus, tanggal selesai, author, dipublish di, tingkat, tempat, tanggal mulai, tombol hapus, dan tombol validasi jika Jurnal belum divalidasi.

6.2.5 Menghapus Jurnal

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka /dashboard pada browser.
- Menekan tombol Jurnal.
- Menekan salah satu daftar Jurnal.
- Menekan tombol hapus.
- Menekan tombol YA pada *modal* yang muncul.
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu muncul pemberitahuan bahwa data sudah terhapus, dan data sudah tidak ada di halaman Jurnal.

6.2.6 Memvalidasi Jurnal

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka /dashboard pada browser.
- Menekan tombol Kuliah Tamu.
- Menekan salah satu daftar Kuliah Tamu.
- Menekan tombol validasi.
- Menekan tombol YA pada *modal* yang muncul.

- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu muncul pemberitahuan bahwa data sudah tervalidasi, dan tombol validasi hilang dari halaman detail Jurnal.

6.2.7 Menambah Daftar Jurnal

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka /dashboard pada browser.
- Menekan salah satu tombol Jurnal.
- Menekan tombol TAMBAH.
- Mengisi *form*.
- Menekan tombol SUBMIT.
- Menekan tombol YA pada *modal* yang muncul.
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu melihat pemberitahuan data berhasil di *entry*, dan data Jurnal terdapat pada halaman Jurnal.

6.2.8 Membuka Halaman Kuliah Tamu

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka /dashboard pada browser.
- Menekan tombol Kuliah Tamu.
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu melihat halaman Kuliah Tamu yang berisi daftar Kuliah Tamu, *input* untuk pencarian, *filter* pencarian, dan tombol tambah.

6.2.9 Membuka Halaman Detail Kuliah Tamu

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka /dashboard pada browser.
- Menekan tombol Kuliah Tamu.
- Menekan salah satu daftar Kuliah Tamu.
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu melihat halaman detail Kuliah Tamu yang berisi judul kegiatan, departemen, URL,

konf hal, *check box* scopus, tanggal selesai, author, dipublish di, tingkat, tempat, tanggal mulai, tombol hapus, dan tombol validasi jika Kuliah Tamu belum divalidasi.

6.2.10 Menghapus Kuliah Tamu

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka /dashboard pada browser.
- Menekan tombol Kuliah Tamu.
- Menekan salah satu daftar Kuliah Tamu.
- Menekan tombol hapus.
- Menekan tombol YA pada *modal* yang muncul.
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu muncul pemberitahuan bahwa data sudah terhapus, dan data sudah tidak ada di halaman Kuliah Tamu.

6.2.11 Memvalidasi Kuliah Tamu

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka /dashboard pada browser.
- Menekan tombol Kuliah Tamu.
- Menekan salah satu daftar Kuliah Tamu.
- Menekan tombol validasi.
- Menekan tombol YA pada *modal* yang muncul.
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu muncul pemberitahuan bahwa data sudah tervalidasi, dan tombol validasi hilang dari halaman detail Kuliah Tamu.

6.2.12 Menambah Daftar Kuliah Tamu

Skenario pengujian aplikasi adalah sebagai berikut:

- Membuka /dashboard pada browser.
- Menekan salah satu tombol Kuliah Tamu.
- Menekan tombol TAMBAH.
- Mengisi *form*.
- Menekan tombol SUBMIT.

- Menekan tombol YA pada *modal* yang muncul.
- Memastikan kebutuhan-kebutuhan berikut sudah berfungsi sesuai prosedur yang disepakati yaitu melihat pemberitahuan data berhasil di *entry*, dan data Kuliah Tamu terdapat pada halaman Kuliah Tamu.

6.3. Evaluasi Pengujian

Hasil pengujian dilakukan terhadap pengamatan mengenai perilaku sistem Aplikasi Monitoring Kinerja FTEIC. Hasil evaluasi pengujian dapat dilihat pada Tabel 6.1.

No.	Kebutuhan	Uji Coba	Status
1.	Login Pengguna	Pengguna memasukkan username dan password pada halaman login. Pengguna dapat melihat halaman dashboard	Berhasil
2.	Logout Penggun	Pengguna menekan tombol logout. Pengguna melihat halaman login dan tidak bisa mengakses dashboard	Berhasil
3.	Membuka Halaman Jurnal	Pengguna membuka halaman dashboard. Pengguna menekan tombol Konferensi / Jurnal. Pengguna melihat halaman jurnal.	Berhasil
4.	Membuka Halaman Detail Jurnal	Pengguna membuka halaman dashboard. Pengguna menekan tombol Konferensi / Jurnal Pengguna menekan salah satu data jurnal. Pengguna melihat halaman detail jurnal.	Berhasil
5.	Menghapus Jurnal	Pengguna membuka halaman dashboard. Pengguna menekan tombol Konferensi /	Berhasil

		Jurnal. Pengguna menekan salah satu data jurnal. Pengguna menekan tombol hapus. Pengguna menekan tombol ya pada modal. Data jurnal terhapus.	
6.	Memvalidasi Jurnal	Pengguna membuka halaman dashboard. Pengguna menekan tombol Konferensi / Jurnal. Pengguna menekan salah satu data jurnal. Pengguna menekan tombol validasi. Pengguna menekan tombol ya pada modal. Data jurnal tervalidasi, tombol validasi hilang.	Berhasil
7.	Menambah Daftar Jurnal	Pengguna membuka halaman dashboard. Pengguna menekan tombol Konferensi / Jurnal. Pengguna menekan tombol tambah. Pengguna mengisi data jurnal pada form. Pengguna menekan tombol submit. Pengguna menekan tombol ya pada modal. Data jurnal berhasil ditambah.	Berhasil
8.	Membuka Halaman Kuliah Tamu	Pengguna membuka halaman dashboard. Pengguna menekan tombol Kuliah Tamu. Pengguna melihat halaman kuliah tamu.	Berhasil
9.	Membuka Halaman Detail Kuliah Tamu	Pengguna membuka halaman dashboard. Pengguna menekan tombol Kuliah Tamu. Pengguna menekan salah satu data kuliah tamu. Pengguna melihat halaman	Berhasil

		detail kuliah tamu.	
10.	Menghapus Kuliah Tamu	Pengguna membuka halaman dashboard. Pengguna menekan tombol Kuliah Tamu. Pengguna menekan salah satu data kuliah tamu. Pengguna menekan tombol hapus. Pengguna menekan tombol ya pada modal. Data kuliah tamu terhapus.	Berhasil
11.	Memvalidasi Kuliah Tamu	Pengguna membuka halaman dashboard. Pengguna menekan tombol Kuliah Tamu. Pengguna menekan salah satu data kuliah tamu. Pengguna menekan tombol validasi. Pengguna menekan tombol ya pada modal. Data kuliah tamu tervalidasi, tombol validasi hilang.	Berhasil
12.	Menambah Daftar Kuliah Tamu	Pengguna membuka halaman dashboard. Pengguna menekan tombol Kuliah Tamu. Pengguna menekan tombol tambah. Pengguna mengisi data kuliah tamu pada form. Pengguna menekan tombol submit. Pengguna menekan tombol ya pada modal. Data kuliah tamu berhasil ditambah.	Berhasil

Tabel 6.1 Hasil Evaluasi Pengujian Aplikasi Monitoring Kinerja FTEIC

Dengan hasil pengujian yang telah ditunjukkan, dapat disimpulkan bahwa secara keseluruhan Aplikasi Monitoring Kinerja FTEIC telah memenuhi kriteria-kriteria yang sudah disebutkan pada bagian-bagian sebelumnya sehingga mampu melewati tahap pengujian yang telah dilakukan.

[Halaman ini sengaja dikosongkan]

BAB VII
KESIMPULAN

7.1 Kesimpulan

Kesimpulan yang didapat setelah melakukan pengembangan Aplikasi Monitoring Kinerja FTEIC adalah sebagai berikut:

- Aplikasi yang dibangun telah sesuai dengan permintaan dan dapat dengan mudah dioperasikan oleh pengguna.
- Dengan adanya Aplikasi Monitoring Kinerja FTEIC, memudahkan pengguna untuk melihat data yang ada di Fakultas FTEIC ITS.

7.2 Saran

Berikut ini adalah beberapa saran yang penulis berikan untuk arah perkembangan selanjutnya:

- Membuat dan atau merubah dokumen standar simulasi aplikasi sesuai dengan penambahan fitur-fitur baru.

[Halaman ini sengaja dikosongkan]

DAFTAR PUSTAKA

[1] Mark Lutz, 2009. *Learning Python Fourth Edition*. [online] Available at: https://cfm.ehu.es/ricardo/docs/python/Learning_Python.pdf [Accessed at 7 January 2021]

[2] Alfin F, 2020. *Berkenalan Dengan Django Rest Framework.* [online] Available at: https://halovina.com/berkenalan-dengan-django-rest-framework/ [Accessed at 7 January 2021]

[3] Mozilla Developer Network (2021). What is JavaScript? | MDN [online] Availabel at: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript [Accessed 7 Jan. 2021]

[4] Mozilla Developer Network (2021). What is a web server? | MDN [online] Available at: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_web_server [Accessed 7 Jan. 2021]

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Fandy Kuncoro Adianto, lahir pada tanggal 19 Juni 1999 di Lumajang. Penulis merupakan mahasiswa yang sedang menempuh studi di Departemen Informatika, Institut Teknologi Sepuluh Nopember (ITS). Penulis aktif berorganisasi di dalam kampus dengan menjadi ketua National Logic Competition pada rangkaian acara schematics 2019.