

**MEMBANGUN APLIKASI SURAT KETERANGAN MAHASISWA
BERBASIS WEB
(STUDI KASUS : FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG)**

SKRIPSI

Karya tulis sebagai syarat memperoleh
Gelar Sarjana Komputer dari Fakultas Teknologi Informasi
Universitas Bale Bandung

Disusun oleh:

UMAM LUQMAN

NPM. C1A140026



**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG
BANDUNG
2019**

LEMBAR PERSETUJUAN PENGUJI

**MEMBANGUN APLIKASI SURAT KETERANGAN MAHASISWA
BERBASIS WEB
(STUDI KASUS : FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG)**

Disusun oleh:
UMAM LUQMAN
NPM. C1A140026

Skripsi ini diterima dan disetujui untuk memenuhi persyaratan mencapai gelar
SARJANA KOMPUTER

Pada
**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG**

Baleendah, Agustus 2019

Disetujui oleh

Penguji 1

Penguji 2

Yudi Herdiana, S.T., M.T.

NIDN.0428027501

Zen Munawar, S.T., M.Kom

NIDN.0422037002

LEMBAR PERSETUJUAN PEMBIMBING

**MEMBANGUN APLIKASI SURAT KETERANGAN MAHASISWA
BERBASIS WEB
(STUDI KASUS : FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG)**

Disusun oleh:

UMAM LUQMAN

NPM. C1A140026

Skripsi ini diterima dan disetujui untuk memenuhi persyaratan mencapai gelar

SARJANA KOMPUTER

pada

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG**

Baleendah, Agustus 2019

Disetujui oleh

Pembimbing 1

Pembimbing 2

Rustiyana, S.T., M.T., M.PD

NIDN.0416017704

Iim Abdurrohim, S.T., MT

NIDN.0413107002

LEMBAR PERSETUJUAN PROGRAM STUDI

**MEMBANGUN APLIKASI SURAT KETERANGAN MAHASISWA
BERBASIS WEB
(STUDI KASUS : FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG)**

Disusun oleh:

UMAM LUQMAN

NPM. C1A140026

Skripsi ini diterima dan disetujui untuk memenuhi persyaratan mencapai gelar

SARJANA KOMPUTER

Pada

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG**

Baleendah, Agustus 2019

Mengesahkan

Ketua Prodi Teknik Informatika

Yaya Suharya, S.Kom., M.T.

NIDN. 0407047706

LEMBAR PERSETUJUAN FAKULTAS

**MEMBANGUN APLIKASI SURAT KETERANGAN MAHASISWA
BERBASIS WEB
(STUDI KASUS : FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG)**

Disusun oleh:

UMAM LUQMAN

NPM. C1A140026

Skripsi ini diterima dan disetujui untuk memenuhi persyaratan mencapai gelar

SARJANA KOMPUTER

Pada

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG**

Baleendah, Agustus 2019

Mengetahui

Dekan Fakultas Teknik Informatika

Yudi Herdiana, S.T., M.T.

NIDN.0428027501

ABSTRAK

Surat keterangan bagi mahasiswa diperlukan disaat mahasiswa ingin melengkapi suatu persyaratan yang menandakan bahwa mahasiswa tersebut berstatus pelajar / mahasiswa, maka di perlukan surat keterangan kuliah, disaat mahasiswa mempunyai urusan pribadi yang tidak dapat ditinggalkan dalam kurun waktu yang lumayan lama sehingga dapat mengganggu perkuliahan, dengan adanya surat cuti mahasiswa tersebut dapat berhenti sementara. Saat ini, sistem yang digunakan dalam pengelolaan arsip surat keterangan bagi mahasiswa di Universitas Bale Bandung, dikelola secara manual dalam sebuah kertas/buku kemudian diketik dan dibuat dengan komputer, kurang efisien dalam pencarian data karena harus mencari arsip catatan siapa saja mahasiswa yang membuat surat keterangan dari setiap halaman buku catatan atau halaman di dalam komputer, sehingga memakan banyak waktu dan kurang efisien. Dalam masalah ini, hal yang paling diharapkan oleh mahasiswa adalah kemudahan dan kecepatan dalam pengurusan surat keterangan untuk mahasiswa. maka dari itu penulis akan membangun aplikasi berbasis web untuk mengeloa serta mengurus surat keterangan mahasiswa untuk mepermudah pihak fakultas dan mahasiswa. aplikasi yang dibangun berbasis *web* sehingga aplikasi yang bisa diakses kapan saja dan di mana saja, upaya membangun aplikasi surat keterangan bagi mahasiswa berbasis web diharapkan akan dapat memudahkan penggunaanya dalam pengurusan surat keterangan bagi mahasiswa di Fakultas Teknologi Informasi Universitas Bale Bandung.

Kata Kunci: Aplikasi, Surat keterangan, Berbasis *Web*

ABSTRACT

A certificate for students is needed when students want to complete a requirement that indicates that the student is a student / student status, so a lecture certificate is needed, when students have personal matters that cannot be left behind in a fairly long period of time so that they can interfere with lectures, with the student leave letter can be paused temporarily, and if the student wishes to return to the lecture program, he can then re-arrange the active letter. At present, the system used in managing certificate records for students at Bale Bandung University, managed manually in a paper / book then typed and made with a computer, is less efficient in data searching because it has to search the records of any student who made a certificate from each page of the notebook or pages on the computer, so it is time-consuming and less efficient. In this problem, the thing most expected by students is the ease and speed in handling certificates for students. therefore the author will build a web-based application to manage and take care of student certificates to facilitate the faculty and students. web-based applications are built so that applications can be accessed anytime and anywhere, efforts to build a certificate application for web-based students are expected to be able to facilitate its users in managing certificates for students at the Faculty of Information Technology, Bale Bandung University.

Keywords: Application, Certificate, Web Based

KATA PENGANTAR

Dengan mengucapkan puji syukur yang sebesar – besarnya ke hadirat Allah SWT, berkat rahmat-Nya penulis dapat menyelesaikan skripsi sebagai usulan penelitian untuk syarat mengikuti skripsi di Fakultas Teknologi Informasi Universitas Bale Bandung.

Selama proses penulisan skripsi ini, penulis menemui beberapa hambatan, tetapi dengan terus semangat dalam mengerjakannya akhirnya skripsi ini dapat selesai. Pada kesempatan yang berharga ini penulis ingin mengucapkan terima kasih kepada:

1. Allah SWT yang telah memberikan karunia dan nikmat sehat selama proses pengerjaan skripsi ini.
2. Bapak Yudi Herdiana, ST. MT. Selaku Dekan Fakultas Teknologi Informasi Universitas Bale Bandung sekaligus merupakan dosen wali bagi penulis.
3. Bapak Rustiyana, S.T.,M.T.,M.Pd. Dosen Pembimbing 1 yang senantiasa membimbing dan mengarahkan penulis dalam menyelesaikan skripsi ini.
4. Bapak Iim Abdurrahim,S.T.,MT sebagai Dosen pembimbing 2 membimbing dan mengarahkan penulis dalam menyelesaikan skripsi ini.
5. Kedua orang tua yang telah memberikan dukungan baik moral maupun materi.
6. Semua pihak yang tidak dapat disebutkan satu persatu namanya, yang telah membantu dalam pelaksanaan penyusunan skripsi.

Akhir kata, penulis mengharapkan kritik dan saran yang membangun. Semoga penyusunan skripsi ini dapat bermanfaat bagi semua pihak.

Baleendah, Agustus 2019

Umam Luqman

DAFTAR ISI

LEMBAR PERSETUJUAN PENGUJI	
LEMBAR PERSETUJUAN PEMBIMBING	
LEMBAR PERSETUJUAN PROGRAM STUDI.....	
LEMBAR PERSETUJUAN FAKULTAS.....	
ABSTRAK.....	i
ABSTRACT.....	ii
KATA PENGANTAR	iii
DAFTAR ISI.....	iv
DAFTAR GAMBAR	viii
DAFTAR TABEL.....	x
BAB I.....	1
PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah.....	3
1.4 Tujuan.....	3
1.5 Metodologi	3
1. <i>Preliminary Investigation</i> (Investigasi Awal).....	4
2. <i>Problem Analysis</i> (Analisis Masalah).....	4
3. <i>Requirement Analysis</i> (Analisis Kebutuhan)	4
Metode Pengembangan Sistem	4
4. <i>Documentation</i> (Dokumentasi / Laporan Akhir)	5
1.6 Sistematika	5

BAB I PENDAHULUAN	5
BAB II TINJAUAN PUSTAKA.....	5
BAB III : METODOLOGI.....	5
BAB IV ANALISIS DAN PERANCANGAN	5
BAB V : KESIMPULAN DAN SARAN	5
BAB II.....	6
TINJAUAN PUSTAKA	6
2.1 Landasan Teori.....	6
2.2 Definisi Teori	7
2.2.1 Definisi Aplikasi	7
2.2.2 Internet	8
2.2.3 Web	8
2.2.4 Smartphone	9
2.2.5 Aplikasi Web	9
2.2.6 Model Driven Development(MDD)	10
2.2.7 Unified Modeling Language(UML)	13
2.2.8 Basis Data	20
2.2.9 HTML (Hyper Text Markup Language).....	21
2.2.10 CSS (Cascading Style Sheet).....	22
2.2.11 Bahasa Pemograman.....	23
2.2.12 Python	24
2.2.13 Java Script.....	26
2.2.14 Text Editor	26
2.2.15 Text Editor Atom.....	27
2.2.16 Framework Django (python)	27
2.2.17 Web browser	28

2.2.18	React Js	29
2.2.19	Virtualenv	29
2.2.20	Yarn	29
2.2.21	Pip	30
2.2.22	Mockup	30
2.2.23	Balsamiq Mockup	30
2.2.24	Astah Community	31
BAB III		32
METODOLOGI		32
3.1	Metodologi Penelitian	32
3.1.1	Preliminary Investigation (Investigasi awal)	33
3.1.2	<i>Problem Analysis</i> (Analisis masalah)	34
3.1.3	Requirement Analysis (Analisis kebutuhan).....	34
3.2	Metodologi Pengembangan Sistem	35
3.2.1	<i>Design</i> (Desain).....	35
3.2.2	<i>Construction</i> (Kontruksi/pengkodean).....	35
3.2.3	Implementation (Implementasi)	36
3.2.4	Dokumentation (Dokumentasi /Laporan Akhir)	36
BAB IV		37
ANALISIS DAN PERANCANGAN.....		37
4.1	ANALISIS	37
4.1.1	Analisis Masalah	37
4.1.2	Analisis Software	38
4.1.3	Analisis Pengguna	38
4.1.4	User Interface	39
4.1.5	Fitur – Fitur	39

4.1.6 Analisis Data	40
4.2 Perancangan.....	40
4.2.1 Unified Modeling Language (UML).....	41
4.2.1.1 Use Case Diagram.....	41
4.2.1.2 Class Diagram	47
4.3 Perancangan Basis Data	48
4.3.1 Struktur Tabel.....	48
4.3.2 Activity Diagram.....	49
4.3.3 Desaign.....	56
4.3.4 Hasil	68
4.3.4.1 Menjalankan Sistem	68
BAB V.....	80
KESIMPULAN DAN SARAN.....	80
5.1 Kesimpulan.....	80
5.2 Saran	81
DAFTAR PUSTAKA	82
LAMPIRAN.....	83

DAFTAR GAMBAR

Gambar 2.1 Model Driven Development.....	11
Gambar 3.1 metodologi penelitian.....	32
Gambar 4.1 Use case diagram.....	41
Gambar 4.2 skema class diagram.....	47
Gambar 4.3 Activity diagram login/gagal login	49
Gambar 4.4 Activity Diagram Data Masuk	50
Gambar 4.5 Activity diagram Tambah data.....	51
Gambar 4.6 Activity diagram Edit / rubah data	52
Gambar 4.7 Activity diagram Hapus data.....	53
Gambar 4.8 Activity diagram memilih jenis surat	54
Gambar 4.9 Activity diagram cetak surat	55
Gambar 4.10 Rancangan halaman awal Login	57
Gambar 4.11 Rancangan halaman sesudah Login Admin	58
Gambar 4.12 Rancangan halaman export user.....	59
Gambar 4.13 Halaman antar muka Admin	60
Gambar 4.14 Halaman edit member	61
Gambar 4.15 Halaman Agenda	63
Gambar 4.16 Halaman Export Agenda	64
Gambar 4.17 Halaman Setting	65
Gambar 4.18 Halaman awal member (mahasiswa).....	66
Gambar 4.19 Halaman setting member (mahasiswa)	67
Gambar 4.20 Halaman login	69
Gambar 4.21 halaman awal admin.....	70
Gambar 4.22 halaman Export member	70
Gambar 4.23 halaman hasil Export member menggunakan ms.excel	71
Gambar 4.24 halaman hasil Export member menggunakan notpad	71
Gambar 4.25 tambah member	72
Gambar 4.26 halaman Edit member.....	73
Gambar 4.27 halaman agenda.....	73

Gambar 4.28 halaman export data Agenda	74
Gambar 4.29 halaman hasil Export agenda menggunakan ms.excel	74
Gambar 4.30 halaman hasil Export agenda menggunakan notpad	75
Gambar 4.31 halaman setting pada admin	75
Gambar 4.32 halaman awal member.....	76
Gambar 4.33 tampilan hasil surat keterangan mahasiswa	77
Gambar 4.34 tampilan hasil surat cuti mahasiswa	78
Gambar 4.35 halaman setting pada member.	79

DAFTAR TABEL

Tabel 2.1 Simbol Use case diagram.....	17
Tabel 2.2 Simbol activity diagram	19
Tabel 4.1 Analisis Data	40
Tabel 4.2 Deskripsi Admin dan Member (Mahasiswa).	42
Tabel 4.3 Use case Admin	42
Tabel 4.4 Use case Member (Mahasiswa)	42
Tabel 4.5 Use case Login	43
Tabel 4.6 Usecase Mengelola Data Masuk	43
Tabel 4.7 Usecase Tambah data.....	44
Tabel 4.8 Usecase Edit/rubah data	45
Tabel 4.9 Usecase Hapus data.....	45
Tabel 4.10 Usecase Memilih jenis surat	46
Tabel 4.11 Usecase Mencetak surat	46
Tabel 4.12 Usecase Logout	47
Tabel 4.13 struktur tabel user	48
Tabel 4.14 struktur tabel agenda	48

BAB I

PENDAHULUAN

1.1 Latar Belakang

Di dalam ruang lingkup mahasiswa saat sedang mengikuti perkuliahan terkadang disibukan dengan berbagai hal pribadi yang membutuhkan surat untuk mengakui bahwa mahasiswa tersebut masih mengikuti program perkuliahan diperguruan tinggi yang bersangkutan. Surat keterangan untuk mahasiswa ini mempunyai dua jenis surat yaitu :

1. Surat keterangan kuliah yang menandakan bahwa mahasiswa tersebut masih mengikuti perkuliahan di universitas.
2. Surat keterangan cuti kuliah yang menandakan bahwa mahasiswa tersebut mempunyai kepentingan sehingga tidak dapat mengikuti perkuliahan dalam sementara waktu.
3. Dan jika mahasiswa ingin Aktif kembali yang menandakan bahwa mahasiswa tersebut akan kembali mengikuti program perkuliahan yang sebelumnya pernah berhenti sementara waktu. Dengan membawa bukti surat cuti.

Universitas Bale Bandung atau disingkat UNIBBA merupakan instansi yang bergerak dalam bidang pendidikan, Fakultas Teknologi Informasi(FTI) Pada tahun 2008 berdasarkan SK DIKTI Nomor 80/D/O/2008 tanggal 22 Mei 2008 Fakultas Teknologi Informasi(FTI) bergabung dengan Universitas Bale Bandung (UNIBBA) dibawah binaan Yayasan Pendidikan Bale Bandung (YPBB) yang didirikan oleh Bapak R.H. Lily Sumantri (beliau pernah menjadi Bupati Kabupaten Bandung pada Tahun 1980). FTI UNIBBA saat ini memiliki 2 (dua) program studi yaitu: Teknik Informatika dan Sistem Informasi.

Saat ini, sistem yang digunakan dalam pengelolaan arsip surat keterangan bagi mahasiswa Fakultas Teknologi Informasi di Universitas Bale Bandung, dikelola secara manual kurang efisien dalam pencarian data karena harus mencari arsip catatan siapa saja mahasiswa yang membuat surat keterangan.

Setelah menganalisis permasalahan yang dialami di Fakultas Teknologi Informasi di Universitas Bale Bandung, solusi yang dapat membantu dalam mengatasi permasalahan tersebut adalah menggunakan sebuah aplikasi yang dapat memberikan kemudahan dalam mengelola surat keterangan mahasiswa baik dalam proses input mahasiswa saat pembuatan surat otomatis masuk kedalam database, sehingga waktu yang dibutuhkan dalam mengelola surat keterangan mahasiswa menjadi lebih efektif dan efisien selain itu dapat mengarsipkan surat berdasarkan bulan dan tahun.

Berdasarkan permasalahan di atas, maka penulis mengusulkan untuk membuat sebuah aplikasi pengelolaan surat keterangan mahasiswa, dengan harapan aplikasi ini bisa bermanfaat bagi pihak Fakultas Teknologi Informasi di Universitas Bale Bandung dan bagi mahasiswa yang sedang melakukan pengurusan surat keterangan mahasiswa. Dalam pelaksanaannya, penulis melakukan pembangunan sebuah sistem informasi berbasis *web* yang berjudul “MEMBANGUN APLIKASI SURAT KETERANGAN MAHASISWA BERBASIS WEB”.

1.2 Rumusan Masalah

Berdasarkan latar belakang permasalahan di atas, adapun rumusan masalah adalah:

1. Bagaimana melakukan penyimpanan data yang tepat, sehingga mudah ditemukan kembali.
2. Bagaimana mempercepat pembuatan surat keterangan aktif dari mahasiswa yang valid.
3. Bagaimana mempercepat membuat surat keterangan cuti dengan validasi cuti hanya boleh maksimal 1(satu) semester dengan batas cuti 2 (dua) semester.

1.3 Batasan Masalah

Adapun beberapa batasan masalah diatas meliputi :

1. Aplikasi ini hanya berfungsi untuk membuat surat keterangan mahasiswa, surat cuti, dan surat aktif kembali yang dibutuhkan mahasiswa sebagai penunjang sarana perkuliahan.
2. Aplikasi ini dapat diakses oleh pengguna yang dibuat mempunyai dua level pengguna, yaitu Admin, dan member.
3. Admin mempunyai hak akses penuh khususnya untuk menambah data mahasiswa, dan mengelola data bahkan dapat mencetak surat secara langsung.
4. Sedangkan member mempunyai batasan hanya berfokus pada pengurusan surat dan cetak surat

1.4 Tujuan

Tujuan yang ingin dicapai dalam penelitian skripsi ini adalah terbangunnya aplikasi Pengelolaan Surat Keterangan Mahasiswa Fakultas Teknologi Informasi di Universitas Bale Bandung, dan dapat memberikan kemudahan Admin dalam mengelola data, mencari data mahasiswa yang membuat surat keterangan mahasiswa, dengan terhubung langsung kedatabse maka pencarian data didalam database akan lebih mudah dilakukan, dan mencetak surat tanpa harus membuat ulang sehingga dapat menghemat waktu, untuk surat aktif kembali dengan syarat mengisi krs dan aplikasi ini berbasis web sehingga memudahkan mahasiswa untuk pengurusan surat karena dapat diakses dimana saja melalui komputer/ laptop dan telepon pintar (smartphone) tanpa harus datang ke fakultas.

1.5 Metodologi

Dalam pengumpulan data dan penyelesaian masalah dalam penelitian, penulis menggunakan metodologi Model Driven Development (MDD) yang di tulis oleh The McGraw-Hill Companies. Teknik pengembangan berbasis model (MDD) menekankan gambar model untuk membantu memvisualisasikan dan menganalisis masalah, mendefinisikan kebutuhan bisnis, dan merancang sistem informasi.

Berikut ini gambaran singkat metodologi yang digunakan penulis:

1. *Preminary Investigation* (Investigasi Awal)

Dalam tahap awal ini penulis melakukan pengumpulan data dan melakukan wawancara terhadap pihak-pihak yang bersangkutan untuk kebutuhan analisis masalah.

2. *Problem Analisis* (Analisis Masalah)

Penulis melakukan analisis masalah yang terjadi di Fakultas Teknologi Informasi di Universitas Bale Bandung dalam mengurus surat keterangan mahasiswa.

3. *Requirement Analysis* (Analisis Kebutuhan)

Penulis melakukan analisis kebutuhan apa saja yang diperlukan untuk membuat aplikasi yang akan dibangun dan alat/tools yang dibutuhkan dalam pengembangan sistem.

Metode Pengembangan Sistem

Dalam metodologi pengembangan sistem, ada beberapa tahapan yaitu:

1. Design (Desain/Rancangan)

Dalam tahap desain/rancangan, penulis membuat rancangan sistem menggunakan Astah Community untuk membuat use case diagram, activity diagram dan class diagram. Kemudian membuat rancangan user interface penulis menggunakan Baslamiq Mockup.

2. Construction (Konstruksi / penulisan kode)

Dalam tahap penulisan kode, penulis menggunakan Atom untuk teks editor, dan bahasa pemrograman menggunakan PYTHON dan untuk design web agar lebih dinamis menggunakan framework Nextjs, Django rest framework, React Js.

3. Implementation (Implementasi)

Dalam tahap implementasi, penulis mengimplementasikan aplikasi berbasis web yang di buat dengan cara upload ke hosting web server.

4. *Documentation* (Dokumentasi / Laporan Akhir)

Dalam tahap ini, penulis melakukan penyelesaian laporan akhir yaitu berupa skripsi yang terdapat dokumentasi.

1.6 Sistematika

Sistematika penulisan terdiri dari 6 buah bab, berikut adalah uraian dari keseluruhan bab.

BAB I PENDAHULUAN

Bab ini menjelaskan latar belakang, rumusan masalah, batasan, rumusan penelitian, penjelasan metodologi yang digunakan dalam pengumpulan data dan penyelesaian penelitian secara singkat dan sistematika yang digunakan dalam penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini menjelaskan landasan teori yang bersumber dari jurnal-jurnal yang sesuai dengan objek penelitian dan dasar teori yang membangun skripsi ini.

BAB III : METODOLOGI

Bab ini menjelaskan metodologi yang digunakan dalam penelitian dan kerangka penelitian atau tahap – tahap penulis dalam melakukan penelitian di Fakultas Teknologi Informasi Universitas Bale Bandung, yang selanjutnya dapat diperoleh suatu jalan keluar untuk mengatasi masalah yang sedang dihadapi oleh Fakultas Teknologi Informasi Universitas Bale Bandung.

BAB IV ANALISIS DAN PERANCANGAN

Bab ini menjelaskan tentang analisis sistem yang sedang berjalan di Fakultas Teknologi Informasi Universitas Bale Bandung, serta menjelaskan segala bentuk perancangan sistem yang akan dibuat. Berisikan hasil analisis dan perancangan dari aplikasi yang akan dibuat.

BAB V : KESIMPULAN DAN SARAN

Berisikan kesimpulan dari penelitian dan saran yang dapat mengembangkan aplikasi di masa yang akan datang.

BAB II

TINJAUAN PUSTAKA

2.1 Landasan Teori

1. “Perancangan sistem pembuatan surat keterangan mahasiswa berbasis web pada fakultas sains dan teknologi di Universitas Islam Negeri Syarif Hidayatullah Jakarta” oleh Geilandri Rathella Dharmawan pada jurnal ini hasil penelitian perancangan sistem pembuatan surat menjadi lebih mudah dan dapat menghemat waktu, dikarenakan dapat diakses melalui komputer/laptop dan telepon pintar (smartphone). Dalam pembuatan surat keterangan mahasiswa menggunakan komputer sehingga mahasiswa diharuskan datang ke universitas hal ini dapat menghabiskan tenaga dan waktu, kemudian dibuatlah aplikasi pemograman guna mengatasi masalah yang dihadapi fakultas sains dan teknologi di Universitas Islam Negeri Syarif Hidayatullah Jakarta dalam hal sistem pembuatan surat keterangan mahasiswa berbasis web. (Geilandri Rathella,2014).
2. “Pembangunan perangkat lunak sistem informasi kebutuhan surat akademik mahasiswa di Fakultas Teknologi Informasi dan Sains Universitas Khatolik Parahyangan”. pada jurnal ini hasil penelitian proses pembuatan surat akademik masih dilakukan secara manual. Prosedur yang dilalui mahasiswa apabila hendak membuat surat akademik adalah meminta formulir surat ke pegawai TU lalu mengisi formulir yang disediakan , apabila permintaan surat akademik sedang kosong atau sedikit maka surat dapat langsung diproses dan mahasiswa tidak perlu menunggu lama untuk mendapatkannya. Namun yang menjadi kendala adalah apabila permintaan surat sedang banyak, biasanya mahasiswa diminta untuk kembali keesokan harinya untuk mengambil surat. Sehingga mahasiswa pemohon tidak bisa mendapatkan surat yang dibutuhkan dengan cepat. Untuk menyelesaikan masalah tersebut, diburuhkan suatu perangkat lunak sehingga mahasiswa dapat membuat sendiri surat akademik yang diperlukannya dengan cepat mulai dari pengisian formulir sampai proses

pencetakan surat tersebut. Perangkat lunak yang akan dibangun mengaplikasikan konsep mailmerge, yaitu proses membuat template surat untuk pengedaran secara massal. (Dony Erlangga,2017).

3. “Pengembangan sistem informasi pelayanan surat keterangan mahasiswa di Universitas Muhammadiyah Malang berbasis web menggunakan Php dan MySQL”. pada jurnal ini hasil penelitian proses pembuatan surat keterangan aktif mahasiswa, Dalam pembuatan surat keterangan aktif mahasiswa menggunakan komputer sehingga mahasiswa diharuskan datang ke universitas hal ini dapat menghabiskan tenaga dan waktu, kemudian dibuatlah aplikasi pemograman guna mengatasi masalah yang dihadapi Universitas Muhammadiyah Malang dalam hal pembuatan surat keterangan aktif mahasiswa berbasi web. (Khamdy,2015).

2.2 Definisi Teori

2.2.1 Definisi Aplikasi

Menurut Supriyanto (2005:2), Aplikasi adalah program yang memiliki aktivitas pemrosesan perintah yang diperlukan untuk melaksanakan permintaan pengguna dengan tujuan tertentu.

Menurut Jogiyanto (1999:12) adalah penggunaan dalam suatu komputer, instruksi (instruction) atau pernyataan (statement) yang disusun sedemikian rupa sehingga komputer dapat memproses input menjadi output.

Menurut Kamus Kamus Besar Bahasa Indonesia (1998 : 52) adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu. Aplikasi adalah suatu program komputer yang dibuat untuk mengerjakan dan melaksanakan tugas khusus dari pengguna.

Aplikasi adalah program komputer yang dibuat oleh suatu perusahaan komputer untuk membantu manusia dalam mengerjakan tugas – tugas tertentu, misalnya Ms. Word, Ms. Excel. Aplikasi berbeda dengan sistem operasi (yang menjalankan komputer), *utility* (yang melaksanakan perawatan atau tugas – tugas umum) dan bahasa pemrograman (yang digunakan untuk membuat program – program tertentu).

2.2.2 Internet

Internet adalah suatu jaringan komunikasi yang menghubungkan satu media elektronik dengan media yang lainnya. Standar teknologi pendukung yang dipakai secara global adalah Transmission Control Protocol atau Internet Protocol Suite (disingkat sebagai istilah TCP/IP). TCP/IP ini merupakan protokol pertukaran paket (dalam istilah asingnya Switching Communication Protocol) yang bisa digunakan untuk miliaran lebih pengguna yang ada di dunia. Sementara itu, istilah “internetworking” berarti cara/prosesnya dalam menghubungkan rangkaian internet beserta penerapan aturannya yang telah disebutkan sebelumnya.

Menurut Supriyanto (2006) Internet adalah suatu hubungan antara berbagai jenis komputer dan juga dengan jaringan di dunia yang punya sistem operasi dan juga aplikasi yang berbeda-beda, dimana hubungan tersebut memanfaatkan kemajuan perangkat komunikasi semacam telepon dan satelit yang memakai protokol standar dalam melakukan hubungan komunikasi, yaitu protokol TCP/IP (Transmission Control/Internet Protocol).

Menurut Heywood (1996) internet adalah istilah teknologi yang muncul mulanya pada akhir tahun 60-an yaitu pada saat United States Department of Defense (DoD) memerlukan suatu standar baru dalam melakukan komunikasi Internetworking. Standar baru ini haruslah merupakan standar yang sanggup menghubungkan berbagai jenis komputer di DoD dengan komputer milik kontraktor militer, organisasi penelitian atau juga yang ilmiah seperti di universitas. Jaringan ini harus kuat, aman dan tahan kerusakan sehingga mampu juga dioperasikan pada kondisi minimum akibat bencana maupun perang.

2.2.3 Web

Web adalah salah satu aplikasi internet yang terdiri dari perangkat lunak, kumpulan protokol dan seperangkat aturan yang memungkinkan kita untuk mengakses informasi di internet. Web menggunakan hypertext (teks yang terhubung ke teks lainnya) dan mendukung file multimedia sehingga dapat digunakan oleh pengguna internet di seluruh dunia. Dengan aplikasi web kita bias mengkomunikasikan berbagai informasi sekaligus mencari informasi baru di

internet. World Wide Web ditemukan oleh Tim Berners-Lee, seorang ilmuwan yang bekerja di pusat penelitian fisika CERN. (Sujatmiko, 2012).

2.2.4 Smartphone

Smartphone adalah sebuah telepon genggam yang memiliki fitur atau kemampuan tingkat tinggi, sering kali dalam penggunaannya menyerupai komputer, sehingga banyak orang mengartikan smartphone sebagai komputer genggam yang memiliki fasilitas telepon. Fitur – fitur yang dapat ditemukan pada smartphone antara lain telepon, sms, internet, ebook viewer, editing dokumen dan masih banyak lagi yang lainnya. Kita juga dapat menambahkan aplikasi lain kedalam smartphone layaknya kita menginstall aplikasi pada komputer.

Sebelum smartphone dikenal oleh kita seperti sekarang ini, kita mengenal adanya telepon seluler dan PDA (Personal Digital Assistant, ponsel berfungsi untuk telepon dan sms sementara PDA memiliki fungsi asisten digital pribadi dari sini munculah ide untuk menggabungkan fungsi keduanya.

2.2.5 Aplikasi Web

Aplikasi web merupakan sebuah aplikasi yang menggunakan teknologi browser untuk menjalankan aplikasi dan diakses melalui jaringan komputer (Remick, 2011). Sedangkan menurut (Rouse, 2011) aplikasi web adalah sebuah program yang disimpan di Server dan dikirim melalui internet dan diakses melalui antarmuka browser. Dari pengertian diatas dapat disimpulkan aplikasi web merupakan aplikasi yang diakses menggunakan web browser melalui jaringan internet atau intranet. Aplikasi web juga merupakan suatu perangkat lunak komputer yang dikodekan dalam bahasa pemrograman yang mendukung perangkat lunak berbasis web seperti HTML, JavaScript, CSS, Ruby, Python, Php, Java dan bahasa pemrograman lainnya.

Aplikasi web adalah sebuah aplikasi yang dapat diakses menggunakan web browser atau penjelajah web melalui jaringan internet atau intranet. Meskipun hingga saat ini ternyata lebih banyak, lebih luas, dan lebih komersil dalam pemakaiannya. Banyak dari perusahaan-perusahaan berkembang yang

menggunakan aplikasi berbasis web dalam merencanakan sumber daya mereka dan untuk mengelola perusahaan mereka. Beberapa yang lain mendefinisikan bahwa pengertian aplikasi web adalah program yang tersimpan pada server kemudian dikirim melalui internet dan diakses melalui antar muka atau interface berupa web browser.

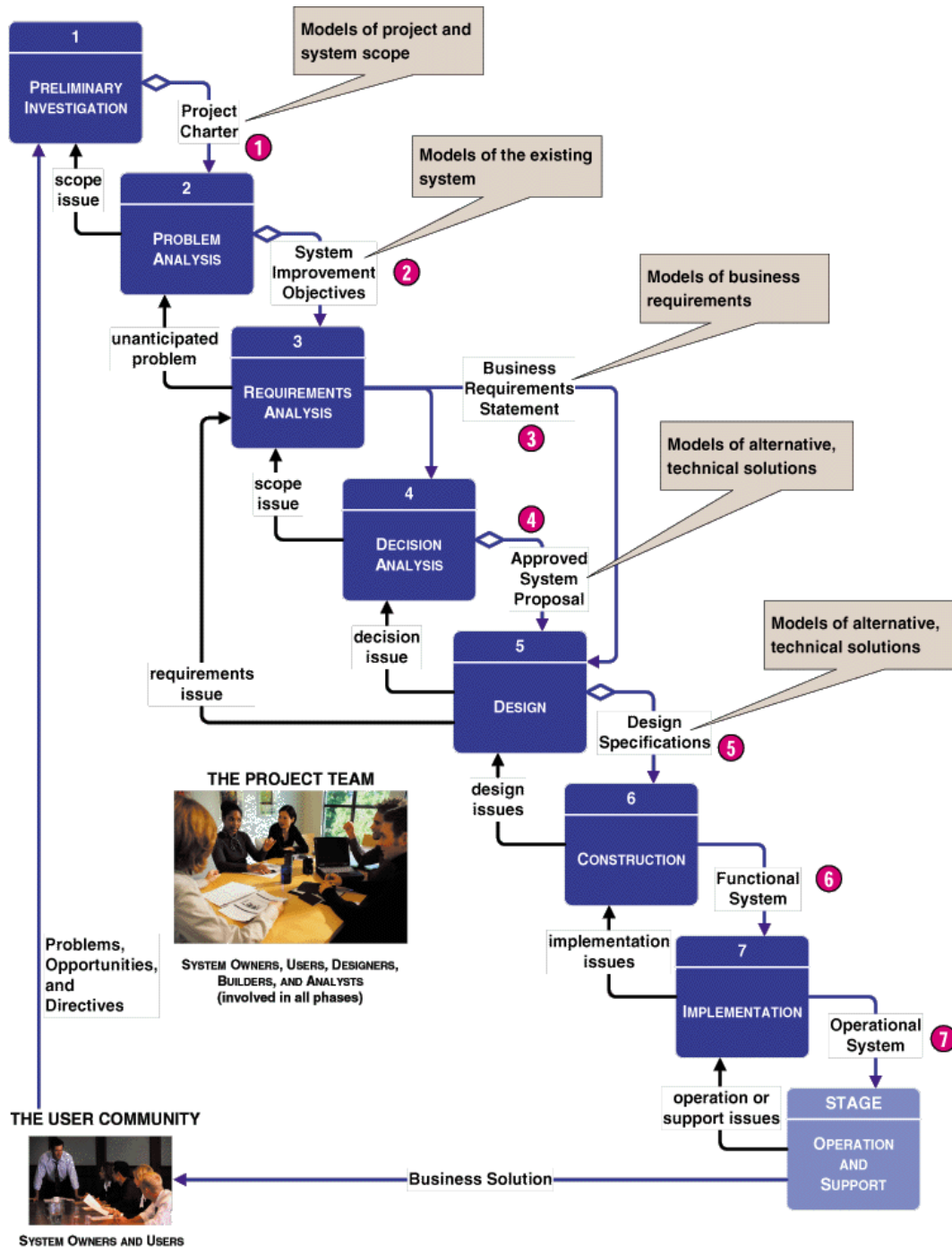
2.2.6 Model Driven Development(MDD)

Model-driven development (MDD) adalah format untuk menulis dan mengimplementasikan perangkat lunak dengan cepat, efektif dan dengan biaya minimum. Metodologi ini juga dikenal sebagai *pengembangan perangkat lunak model-driven* (MDSD), *model-driven engineering* (MDE) dan *model-driven architecture* (MDA).

Pendekatan MDD berfokus pada pembangunan model perangkat lunak. Model adalah diagram yang menentukan cara kerja sistem perangkat lunak sebelum kode dihasilkan. Setelah perangkat lunak dibuat, dapat diuji menggunakan model-based testing (MBT) dan kemudian digunakan.

Pemodelan adalah tindakan menggambar satu atau lebih representasi grafis (atau gambar) dari suatu sistem. Pemodelan adalah teknik komunikasi berdasarkan pepatah lama, "sebuah gambar bernilai seribu kata."

Teknik pengembangan berbasis model (MDD) menekankan gambar model untuk membantu memvisualisasikan dan menganalisis masalah, mendefinisikan kebutuhan bisnis, dan merancang sistem informasi. berikut ini gambar *Model Driven Development* yang digunakan :



Gambar 2.1 Model Driven Development

- Analisis dan desain sistem terstruktur - berpusat pada proses
- Teknik informasi (IE) - berpusat pada data
- Analisis dan desain berorientasi obyek (OOAD) - terpusat pada objek (integrasi data dan masalah proses)

Langkah langkah

1. *Preliminary investigation* (investigasi awal)

Tahap ini merupakan tahap awal dari pengembangan sistem. Fase ini berisikan investigasi awal ketika ingin merancang sebuah sistem, seperti wawancara, tinjauan langsung dan mempelajari dokumen perusahaan.

2. *Problem analysis* (Analisis masalah)

Problem Analysis ialah menganalisa masalah-masalah yang terdapat di lapangan. Tahap ini merupakan pengembangan dari tahap pertama. Pada tahap ini dilakukan analisis terhadap sistem yang telah ada saat itu. Tahap ini memberikan pemahaman yang lebih dalam bagi tim proyek mengenai permasalahan yang dihadapi. Analisis ini dilakukan untuk menjawab pertanyaan apakah keuntungan yang diperoleh setelah pemecahan masalah lebih besar daripada biaya yang dikeluarkan.

3. *Requirements analysis* (Analisis Kebutuhan)

Requirement Analysis ialah melakukan analisa terhadap kebutuhan perusahaan.. Pekerjaan pada tahap ini adalah mendefinisikan apa saja yang perlu dilakukan oleh system dan apa yang dibutuhkan dan diinginkan oleh pengguna dari sistem baru.

Tahap ini memerlukan perhatian yang besar karena jika terjadi kesalahan dalam menerjemahkan kebutuhan dan keinginan pengguna sistem maka dapat mengakibatkan adanya rasa tidak puas pada sistem final dan perlu diadakan modifikasi yang tentunya akan kembali mengeluarkan biaya.

4. *Decision analysis* (Analisis Keputusan)

Decision Analysis ialah melakukan analisa terhadap keputusan yang akan diambil berdasarkan solusi-solusi yang ditawarkan. Dalam analisis keputusan, umumnya terdapat berbagai alternatif untuk mendesain sistem informasi yang baru.

5. *Design* (Desain)

Setelah diperoleh proposal sistem yang disetujui, maka dapat mulai dilakukan proses desain dari sistem target. Tujuan dari tahap ini adalah untuk mentransformasikan *business requirement statement* menjadi spesifikasi desain untuk proses *konstruksi*. Dengan kata lain, tahap desain menyatakan bagaimana teknologi akan digunakan dalam sistem yang baru. Tahap ini memerlukan ide dan opini dari pengguna, *vendor*, dan spesialis IT.

6. *Construction* (Kontruksi)

Construction ialah tahapan melaksanakan pengujian pada komponen sistem secara individu dan sistem secara keseluruhan.

7. *Implementation* (Implementasi)

Mengimplementasikan penghubung antara sitem baru dan sistem lama, termasuk instalasi dari *software* yang dibeli atau disewa. Pada tahap ini dilakukan konstruksi basis data, program aplikasi, dan penghubung antara sistem dan pengguna. Beberapa dari komponen ini telah ada sebelumnya. Setelah dilakukan pengujian, maka sistem dapat mulai diimplementasikan.

2.2.7 Unified Modeling Language(UML)

Unified Modeling Language merupakan salah satu alat bantu yang dapat digunakan dalam bahasa pemograman yang berorientasi objek, saat ini UML akan mulai menjadi standar masa depan bagi industri pengembangan sistem/perangkat lunak yang berorientasi objek sebab pada dasarnya UML digunakan oleh banyak perusahaan raksasa seperti IBM, Microsoft, dan sebagainya. Unified Modeling Language merupakan salah satu alat bantu yang dapat digunakan dalam bahasa pemograman yang berorientasi objek, saat ini UML akan mulai menjadi standar masa depan bagi industri pengembangan sistem/perangkat lunak yang berorientasi objek sebab pada dasarnya UML digunakan oleh banyak perusahaan raksasa seperti IBM, Microsoft, dan sebagainya.

Definisi UML menurut para ahli adalah sebagai berikut:

Menurut Booch (2005:7) UML adalah Bahasa standar untuk membuat rancangan *software*. UML biasanya digunakan untuk menggambarkan dan membangun, dokumen artifak dari software –intensive system.

Menurut Nugroho (2010:6), UML (*Unified Modeling Language*) adalah ‘bahasa’ pemodelan untuk sistem atau perangkat lunak yang berparadigma ‘berorientasi objek’. Pemodelan (modeling) sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami.

1. Menurut Nugroho (2009:4), UML (*Unified Modeling Language*) adalah Metodologi kolaborasi antara metoda-metoda Booch, OMT (*Object Modeling Technique*), serta OOSE (*Object Oriented Software Engineering*) dan beberapa metoda lainnya, merupakan metodologi yang paling sering digunakan saat ini untuk analisa dan perancangan sistem dengan metodologi berorientasi objek mengadaptasi maraknya penggunaan bahasa “pemrograman berorientasi objek” (OOP).
2. Menurut Herlawati (2011:10), bahwa beberapa literature menyebutkan bahwa UML menyediakan sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa diagram yang digabung, misanya diagram komunikasi, diagram urutan dan diagram pewaktuan digabung menjadi diagram interaksi.

Adapun beberapa fungsi UML Antara lain:

1. Untuk menggambarkan batasan sistem dan fungsi-fungsi sistem secara umum, dibuat dengan use case dan actor
2. menggambarkan kegiatan atau proses bisnis yang dilaksanakan secara umum, dibuat dengan interaction diagrams
3. menggambarkan representasi struktur statik sebuah sistem dalam bentuk class diagrams
4. membuat model behavior ”yang menggambarkan kebiasaan atau sifat sebuah sistem” dengan state transition diagrams

5. menyatakan arsitektur implementasi fisik menggunakan component and development diagram, untuk menyampaikan atau memperluas functionality dengan stereotypes.

Beberapa literature menyebutkan bahwa UML menyediakan sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa diagram yang digabung, misanya diagram komunikasi, diagram urutan dan diagram pewaktuan digabung menjadi diagram interaksi. Namun demikian model-model itu dapat dikelompokkan berdasarkan sifatnya yaitu statis atau dinamis. Jenis diagram yang akan digunakan yaitu antara lain:

1. Class Diagram

Bersifat statis, Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram ini umum dijumpai pada pemodelan sistem berorientasi objek. Meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas aktif.

2. Use-Case Diagram

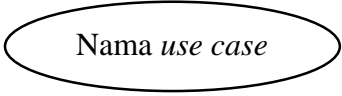



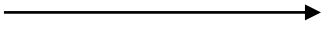
Usecase diagram adalah diagram *usecase* yang digunakan untuk menggambarkan secara ringkas siapa yang menggunakan sistem dan apa saja yang bisa dilakukannya. Diagram usecase tidak menjelaskan secara detail tentang penggunaan usecase, namun hanya memberi gambaran singkat hubungan antara usecase, aktor, dan sistem. Melalui diagram usecase dapat diketahui fungsi-fungsi apa saja yang ada pada sistem (Rosa-Salahudin, 2011: 130). Nama suatu *usecase* harus didefinisikan sesimple mungkin dan dapat dipahami.



Komponen-komponen yang ada pada usecase adalah :

1. Aktor. Merupakan orang, proses atau sistem lain yang berinteraksi dengan sistem yang akan dibuat. Jadi walaupun simbol aktor dalam diagram usecase berbentuk orang, namun aktor belum tentu orang.
2. Usecase. Merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang slaing berinteraksi atau bertukar pesan antar unit maupun aktor.

3. Relasi. Merupakan hubungan yang terjadi pada sistem baik antar aktor maupun usecase maupun anantara usecase dan aktor. Relasi yang digunakan dalam diagram usecase antara lain :
 - a. Association. Merupakan relasi yang digunakan untuk menggambarkan interaksi antara usecase dan aktor. Asosiasi juga menggambarkan berapa banyak objek lain yang bisa berinteraksi dengan suatu objek atau disebut multiplicity (Multiplicity dapat dilihat pada postingan Class Diagram).
 - b. Generalization. Merupakan relasi yang menggambarkan inheritance baik aktor maupun usecase.
 - c. Dependency Merupakan relasi yang menggambarkan ketergantungan antara usecase yang satu dengan usecase yang lain. Ada dua macam dependency yaitu include dan extends. Include menggambarkan bahwa jalannya suatu usecase memicu jalannya usecase lain. Misalnya usecase login diinclude oleh usecase memilih menu, artinya usecase memilih menu akan memicu dijalankannya usecase login. Sebelum aktor menjalankan usecase memilih menu, aktor harus menjalankan usecase login dulu. Dalam penggambaran diagram usecase, panah mengarah kepada usecase yang diinclude. Sedangkan extends menggambarkan bahwa suatu usecase dijalankan karena ada persyaratan tertentu dari usecase lain. Misal, dalam sebuah sistem user tidak bisa menjalankan login sebelum dia mendaftar akun. Dalam diagram usecase, usecase daftar akun mengextends usecase login. Artinya aktor harus menjalankan usecase daftar akun dulu sebelum menjalankan usecase login karena usecase login memiliki syarat aktor yang melakukan login harus sudah melakukan pendaftaran akun. Arah panah dependency mengarah pada usecase yang memiliki syarat. (MSDN, n.d).

Tabel 2.1 Simbol Use case diagram

Simbol	Deskripsi
<p><i>Use Case</i></p>  <p>Nama use case</p>	<p>Fungsional yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit / aktor; biasanya dinyatakan dengan menggunakan kata kerja diawal frase nama <i>use case</i></p>
<p>Aktor / Actor</p>  <p>Nama actor</p>	<p>Orang proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat tersebut, jadi walaupun simbol aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya menggunakan kata benda di awal frase nama actor</p>
<p>Asosiasi/ association</p> 	<p>Komunikasi antara aktor dan use case yang berpartisipasi pada use case atau use case memiliki interaksi dengan aktor</p>
<p>Ekstensi / extend</p> <p><< extend >></p> 	<p>Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan dapat berdiri sendiri walaupun tanpa use case tambahan itu; biasanya use case tambahan memiliki nama depan yang sama dengan use case yang ditambahkan.</p>
<p>Generalisasi / generalization</p> 	<p>Hubungan generalisasi dan spesialis (umum – khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang umum dari lainnya.</p>

<p>Menggunakan / include / uses</p> <p><<include>></p>  <p><<uses>></p> 	<p>Relasi use case tambahan ke sebuah use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini.</p>
--	--




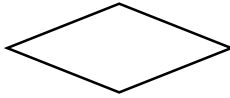


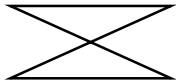

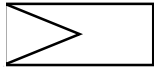

3. Activity Diagram

Activity Diagram adalah diagram yang menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Komponen yang ada pada activity diagram antara lain :

1. Activity atau state : Menunjukkan aktivitas yang dilakukan.
2. Initial activity atau initial state : Menunjukkan awal aktivitas dimulai.
3. Final Activity atau final state : Menunjukkan bagian akhir dari aktivitas.
4. Decission : Digunakan untuk menggambarkan test kondisi untuk memastikan bahwa control flow atau object flow mengalir lebih ke satu jalur. Jumlah jalur sesuai yang diinginkan.
5. Merge : Berfungsi menggabungkan flow yang dipecah oleh decission.
6. Synchronization: Diabgi menjadi 2 yaitu fork dan join. Fork digunakan untuk memecah behaviour menjadi activity atau action yang paralel, sedangkan join untuk menggabungkan kembali activity atau action yang paralel.
7. Swimlanes : Memecah activity diagram menjadi baris dan kolom untuk membagi tanggung jawab obyek-obyek yang melakukan aktivitas.
8. Transition : Menunjukkan aktivitas selanjutnya setelah aktivitas sebelumnya.

Tabel 1.2 Simbol activity diagram

Simbol	Keterangan
	Titik Awal
	Titik Akhir
	Activity
	Pilihan untuk mengambil keputusan
	Fork; Digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan paralel menjadi satu
	Rake; Menunjukkan adanya dekomposisi
	Tanda waktu
	Tanda pengiriman
	Tanda penerimaan
	Aliran akhir

2.2.8 Basis Data

Basis data (*database*) adalah kumpulan informasi yang disimpan didalam komputer secara sistematis sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut (Sujatmiko, 2012).

Basis data (*database*) adalah suatu kumpulan data yang disusun dalam bentuk tabel-tabel yang saling berkaitan maupun berdiri sendiri dan disimpan secara bersama-sama pada suatu media. Basis data dapat digunakan oleh satu atau lebih program aplikasi secara optimal, data disimpan tanpa mengalami ketergantungan pada program yang akan menggunakannya (Kadir, 1999).

Adapun komponen – komponen yang ada pada basis data adalah sebagai berikut:

1. *Table*

Tabel adalah kumpulan dari suatu *field* dan *record*. Dalam hal ini biasanya *field* ditunjukkan dalam bentuk kolom dan *record* ditunjukkan dalam bentuk baris.

2. *Field*

Field adalah sebutan untuk mewakili suatu *record*. Misalnya seorang pegawai dapat dilihat datanya melalui *field* yang diberikan padanya seperti nip, nama, alamat, dan lain-lain.

3. *Record*

Record adalah kumpulan elemen-elemen yang saling berkaitan menginformasikan tentang suatu isi data secara lengkap. Satu *record* mewakili satu data atau informasi tentang seseorang misalnya, nomor daftar, nama pendaftar, alamat, tanggal masuk.

4. *Primary Key*

Primary key adalah suatu kolom (*field*) yang menjadi titik acuan pada sebuah tabel, bersifat unik dalam artian tidak ada satu nilai pun yang sama atau

kembar dalam tabel tersebut, dan dalam satu tabel hanya boleh ada satu *primary key*.

5. *Foreign Key*

Foreign key atau disebut juga kunci relasi adalah suatu kolom dalam tabel yang digunakan sebagai “kaitan” untuk melengkapi satu hubungan yang didapati dari tabel induk, dan biasanya hubungan yang terjalin antar tabel adalah satu ke banyak (*one to many*).

6. *Index* adalah struktur basis data secara fisik, yang digunakan untuk optimalisasi pemrosesan data dan mempercepat proses pencarian data.

2.2.9 HTML (Hyper Text Markup Language)

HTML adalah singkatan dari Hrypertext Markup Language. HTML merupakan file teks yang ditulis menggunakan aturan-aturan kode tertentu untuk kemudian disajikan ke user melalui satu aplikasi web browser. Setiap informasi yang tampil di web selalu dibuat menggunakan kode HTML. Oleh karena itu, dokumen HTML sering disebut juga sebagai web Page (halaman web). Untuk membuat dokumen HTML, kita tidak tergantung pada aplikasi tertentu; karena dokumen HTML dapat dibuat menggunakan aplikasi Text Editor apa pun, bisa Notepad (untuk lingkungan Linux), dan sebagainya. (Raharjo,2012)

Disebut hypertext karena di dalam HTML sebuah text biasa dapat berfungsi lain, kita dapat menjadi Link yang dapat berpindah dari satu halaman ke halaman lainnya hanya dengan meng-klik text tersebut. Kemampuan ext inilah yang dinamakan hypertext, walaupun pada implementasinya nanti tidak hanya ext yang dapat dijadikan link.

HTML bukanlah bahasa pemrograman (programming language), tetapi bahasa markup (markup language), hal ini terdengar sedikit aneh, tapi di dalam HTML tidak akan ditemukan struktur yang biasa ditemukan dalam bahasa pemrograman seperti IF, LOOP, maupun variabel.

Hypertext 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 terintegrasi. Dengan kata lain, berkas yang dibuat dalam perangkat lunak pengolah kata dan disimpan dalam format ASCII normal sehingga menjadi halaman web dengan perintah-perintah HTML. Bermula dari sebuah bahasa yang sebelumnya banyak digunakan di dunia penerbitan dan percetakan yang disebut dengan SGML (Standard Generalized Markup Language), HTML adalah sebuah standar yang digunakan secara luas untuk menampilkan halaman web. HTML saat ini merupakan standar Internet yang didefinisikan dan dikendalikan penggunaannya oleh World Wide Web Consortium (W3C). HTML dibuat oleh kolaborasi Caillau TIM dengan Berners-lee Robert ketika mereka bekerja di CERN pada tahun 1989 (CERN adalah lembaga penelitian fisika energi tinggi di Jenewa).

2.2.10 CSS (Cascading Style Sheet)

CSS adalah kepanjangan dari "Cascading Style Sheets". adalah kumpulan kode program yang digunakan untuk memberi sentuhan gaya atau corak tampilan (style) pada sebuah element atau struktur halaman web yang dibuat oleh HTML. Dengan CSS dapat mengubah desain text, warna, gambar dan latar belakang dari (hampir) semua kode tag HTML. CSS biasanya selalu dikaitkan dengan HTML, karena keduanya memang saling melengkapi. HTML ditujukan untuk membuat struktur, atau konten dari halaman web tersebut, sedangkan CSS digunakan untuk tampilan dari halaman web tersebut. Istilahnya, "HTML for content, CSS for Persentation".

CSS dikembangkan oleh komunitas internasional yang disebut *World Wide Web Consortium (W3C)*, yang merupakan komunitas seluruh dunia yang bergabung untuk berkontribusi memberi masukan, menulis dan mengembangkan spesifikasi dari CSS untuk mendefinisikan standar bahasa CSS yang akan digunakan oleh semua orang, juga agar vendor yang mengembangkan aplikasi web browser dapat mengimplementasikannya. Spesifikasi CSS pertama kali diperkenalkan pada tahun 1996.

Untuk mempelajari CSS, tentunya harus mengetahui HTML dasar yang membentuk struktur halaman sebuah website yang dibuat oleh tag-tag atau elemen-elemen HTML. Dengan demikian, dapat mengaplikasikan *style* untuk masing-masing tag atau elemen tersebut.

CSS mendeskripsikan presentasi atau tampilan sebuah halaman web. Sehingga, kode CSS digunakan untuk mempercantik dan memperbaiki tampilan antar muka sebuah website. Seperti, mewarnai tulisan, memberi background (latar) baik dengan warna ataupun gambar, menentukan font, memberi animasi, mengatur tampilan sesuai ukuran layar web browser yang digunakan, dan hal-hal lainnya yang berkaitan dengan desain sebuah website.

2.2.11 Bahasa Pemrograman

Bahasa pemrograman, atau sering diistilahkan juga dengan bahasa komputer atau bahasa pemrograman komputer, adalah instruksi standar untuk memerintah komputer. Bahasa pemrograman ini merupakan suatu himpunan dari aturan sintaks dan semantik yang dipakai untuk mendefinisikan program komputer. Bahasa ini memungkinkan seorang programmer dapat menentukan secara persis data mana yang akan diolah oleh komputer, bagaimana data ini akan disimpan/diteruskan, dan jenis langkah apa yang akan diambil dalam berbagai situasi secara persis.

Menurut tingkat kedekatannya dengan mesin komputer, bahasa pemrograman terdiri dari:

1. Bahasa Mesin, yaitu memberikan perintah kepada komputer dengan memakai kode bahasa biner, contohnya 01100101100110
2. Bahasa Tingkat Rendah, atau dikenal dengan istilah bahasa rakitan (bah.Ingggris Assembly), yaitu memberikan perintah kepada komputer dengan memakai kode-kode singkat (kode mnemonic), contohnya kode_mesin|MOV, SUB, CMP, JMP, JGE, JL, LOOP, dsb.
3. Bahasa Tingkat Menengah, yaitu bahasa komputer yang memakai campuran instruksi dalam kata-kata bahasa manusia (lihat contoh Bahasa

Tingkat Tinggi di bawah) dan instruksi yang bersifat simbolik, contohnya {, }, ?, <<, >>, &&, ||, dsb.

4. Bahasa Tingkat Tinggi, yaitu bahasa komputer yang memakai instruksi berasal dari unsur kata-kata bahasa manusia, contohnya begin, end, if, for, while, and, or, dsb. Komputer dapat mengerti bahasa manusia itu diperlukan program compiler atau interpreter.

Sebagian besar bahasa pemrograman digolongkan sebagai Bahasa Tingkat Tinggi, hanya bahasa C yang digolongkan sebagai Bahasa Tingkat Menengah dan Assembly yang merupakan Bahasa Tingkat Rendah.

Penulis dalam membangun aplikasi surat keterangan mahasiswa berbasis web ini menggunakan bahasa pemrograman tingkat tinggi yaitu bahasa pemrograman Python dan javascript.

2.2.12 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.

Python termasuk dalam bahasa pemrograman tinggi yang dapat melakukan eksekusi sejumlah instruksi multi guna secara langsung (interpretatif) dengan

metode orientasi objek (Object Oriented Programming) serta menggunakan semantik dinamis untuk memberikan tingkat keterbacaan syntax. Sebagai bahasa pemrograman tinggi, python dapat dipelajari dengan mudah karena sudah dilengkapi dengan manajemen memori otomatis (pointer).

Python merupakan bahasa pemrograman tingkat tinggi yang diracik oleh Guido van Rossum. Python banyak digunakan untuk membuat berbagai macam program, seperti: program CLI, Program GUI (desktop), Aplikasi Mobile, Web, IoT, Game, Program untuk Hacking, dsb. Python juga dikenal dengan bahasa pemrograman yang mudah dipelajari, karena struktur sintaknya rapi dan mudah dipahami.

Python dikembangkan oleh Guido van Rossum pada tahun 1990 di Stichting Mathematisch Centrum (CWI), Amsterdam sebagai kelanjutan dari bahasa pemrograman ABC. Versi terakhir yang dikeluarkan CWI adalah 1.2.

Tahun 1995, Guido pindah ke CNRI di Virginia Amerika sambil terus melanjutkan pengembangan Python. Versi terakhir yang dikeluarkan adalah 1.6. Tahun 2000, Guido dan para pengembang inti Python pindah ke BeOpen.com yang merupakan sebuah perusahaan komersial dan membentuk BeOpen PythonLabs. Python 2.0 dikeluarkan oleh BeOpen. Setelah mengeluarkan Python 2.0, Guido dan beberapa anggota tim PythonLabs pindah ke DigitalCreations.

Saat ini pengembangan Python terus dilakukan oleh sekumpulan pemrogram yang dikoordinir Guido dan Python Software Foundation. Python Software Foundation adalah sebuah organisasi non-profit yang dibentuk sebagai pemegang hak cipta intelektual Python sejak versi 2.1 dan dengan demikian mencegah Python *dimiliki* oleh perusahaan komersial. Saat ini distribusi Python sudah mencapai versi 2.7.14 dan versi 3.6.3

Nama Python dipilih oleh Guido sebagai nama bahasa ciptaannya karena kecintaan Guido pada acara televisi Monty Python's Flying Circus. Oleh karena itu seringkali ungkapan-ungkapan khas dari acara tersebut seringkali muncul dalam korespondensi antar pengguna Python.

2.2.13 Java Script

Javascript adalah Bahasa script dan Bahasa pemrograman tingkat tinggi. Javascript sangat populer di internet dan dapat bekerja disebagian web browser, javascript dapat disisipkan dalam halaman web menggunakan tag script.

Javascript pertama kali dikembangkan oleh Brendan Eich dari Netscape di bawah nama mocha, yang nantinya namanya diganti menjadi LiveScript, dan akhirnya menjadi JavaScript. Pada pertengahan tahun 1997, JavaScript 1.1 diajukan ke badan standarisasi Eropa: *European Computer Manufacturers Association (ECMA)* untuk membuat sebuah standar bahasa pemograman script web browser. Atas dasar ini, dibentuklah sebuah komite dengan anggota yang terdiri dari programmer dari berbagai perusahaan internet pada saat itu, seperti *Netscape, Sun, Microsoft, Borland, NOMBAS* serta beberapa perusahaan lain yang tertarik dengan perkembangan JavaScript.

2.2.14 Text Editor

Text Editor adalah suatu software aplikasi atau suatu program komputer yang memungkinkan sebagai penggunaanya untuk membuat, mengubah atau mengedit file teks yang ada berupa text biasa. Text editor ini sebenarnya bisa digunakan untuk membuat program komputer dan mengedit source code dari bahasa pemograman. Selain itu, text editor juga bisa dimanfaatkan untuk membuat halaman web atau template web design dan juga membuat aplikasi tertentu. Software aplikasi satu ini memang secara umum ditujukan untuk mempermudah aktivitas pemrograman.

Text Editor memiliki fitur-fitur sangat kecil dan sederhana. Namun ada juga beberapa text editor kini sudah menawarkan fungsi luas dan kompleks. Unix dan Linux adalah contohnya. Dalam sistem operasinya sudah tersedia Editor VI (atau varian), tapi banyak juga yang mencakup editor Emacs. Sementara sistem operasi dari Windows itu sendiri menyediakan Notepad standar. Walaupun sudah tersedia secara bawaan, banyak programmer lebih menyukai text editor lainnya yang memang fiturnya lebih banyak atau lengkap. Penulis menggunakan text editor Atom.

2.2.15 Text Editor Atom

Atom adalah teks editor dan editor kode gratis dan terbuka untuk macOS, Linux, dan Microsoft Windows dengan dukungan untuk plug-in yang ditulis dalam Node.js, dan tertanam Git Control, yang dikembangkan oleh GitHub. Atom adalah aplikasi desktop yang dibangun menggunakan teknologi web.

2.2.16 Framework Django (python)

Django adalah high-level web framework Python yang memiliki pengembangan cepat dalam keamanan dan perawatan website. Dibangun oleh pengembang berpengalaman, Django menjangkau banyak sekali kerumitan dalam pengembangan web, jadi kamu bisa fokus dalam menulis aplikasi tanpa harus menulis ulang jalurnya. Django dapat digunakan secara gratis dan open source, memiliki komunitas yang terus tumbuh dan aktif, dokumentasi yang jelas, dan banyak pilihan untuk dukungan yang gratis maupun berbayar.

Django adalah *free* dan *open-source framework full-stack* berkonsep MVC untuk membuat aplikasi web dengan bahasa pemrograman Python. *Full-stack* berarti Django melingkupi sisi depan (*front-end*) dan juga sisi belakang (*back-end*). *Front-end* adalah bagian aplikasi web yang terlihat oleh pengguna, sedangkan *back-end* adalah bagian yang berhubungan dengan *database* dan logika aplikasi web.

Django awalnya dikembangkan pada tahun 2003 dan 2005 oleh beberapa *web developers* yang bertugas merawat *web portal* dari *newspaper website*. Setelah membuat beberapa website, para web developer tersebut mulai membuat ulang kode-kode yang mereka pernah tulis dengan menerapkan beberapa *design pattern*, lalu disebarkan sebagai proyek *open-source* dengan nama “Django” pada bulan Juli 2005.

Filosofi dari Django sendiri, yaitu “*The web framework for perfectionists with deadlines*” memungkinkan *web developer* membangun website secepat mungkin dengan kriteria tertentu.

Django adalah kerangka kerja web gratis dan open-source berbasis-Python, yang mengikuti pola arsitektur model-template-view. Ini dikelola oleh Django Software Foundation, sebuah organisasi independen yang didirikan sebagai 501 nirlaba.

Tujuan utama Django adalah untuk memudahkan pembuatan situs web yang kompleks dan digerakkan oleh basis data. Kerangka kerja menekankan reusability dan "pluggability" komponen, kode lebih sedikit, pengembangan cepat, dan prinsip jangan ulangi sendiri .^[7] Python digunakan di seluruh, bahkan untuk pengaturan file dan model data. Django juga menyediakan antarmuka administrasi membuat, membaca, memperbarui dan menghapus yang dihasilkan secara dinamis melalui introspeksi dan dikonfigurasi melalui model admin.

Beberapa situs terkenal yang menggunakan Django termasuk Public Broadcasting Service , Instagram , Mozilla , *The Washington Times* , Disqus , Bitbucket , dan Nextdoor. Itu digunakan di Pinterest , tetapi kemudian situs pindah ke kerangka kerja yang dibangun di atas Flask .

2.2.17 Web browser

Web Browser adalah suatu program atau software yang digunakan untuk menjelajahi internet atau untuk mencari informasi dari suatu web yang tersimpan didalam komputer. Awalnya, web browser berorientasi pada teks dan belum dapat menampilkan gambar. Namun, web browser sekarang tidak hanya menampilkan gambar dan teks saja, tetapi juga memutar file multimedia seperti video dan suara. Web browser juga dapat mengirim dan menerima email, mengelola HTML, sebagai input dan menjadikan halaman web sebagai hasil output yang informative.

Dengan menggunakan web browser, para pengguna internet dapat mengakses berbagai informasi yang terdapat di internet dengan mudah. Beberapa contoh web browser diantaranya Internet Explorer, Mozilla, Firefox, Safari, Opera, dll.

2.2.18 React Js

React JS adalah sebuah pustaka/library javascript yang bersifat opensource untuk membangun User Interface yang dibuat oleh Facebook. React JS hanya mengurus semua hal yang berkaitan dengan tampilan dan logika di sekitarnya. React bukanlah sebuah framework MVC. React adalah library yang bersifat composable user interface, yang artinya kita dapat membuat berbagai UI yang bisa kita bagi menjadi beberapa komponen. React JS ini diciptakan dengan tujuan untuk : “Membangun aplikasi skala besar dengan data yang berubah dan terus berubah dari waktu ke waktu.”

ReactJS adalah library javascript yang bisa memudahkan dalam membuat tampilan UI yang interaktif (satu data yang sama bisa saling berkomunikasi), dimana bagian bagiannya nanti akan dibentuk dalam komponen agar mudah dipakai berkali-kali.

2.2.19 Virtualenv

Virtualenv adalah tools untuk membuat lingkungan python virtual yang terisolasi. Lingkungan python disini yang dimaksud meliputi binary (executable), library dan semua package yang di install oleh package manager python seperti pip dan easy_install. Terisolasi artinya tertutup dan tidak bisa diakses dari dunia luar. Program Python yang berjalan di dalam virtualenv memiliki modul-modulnya sendiri dan program dari luar tidak bisa mengaksesnya.

2.2.20 Yarn

Yarn adalah suatu package manager buatan Facebook untuk sistem yang akan dibuat. Package manager adalah suatu paket instalasi, update, configure dan remove software. Jadi jika melakukan instalasi menggunakan package manager, jadi tidak perlu melakukan konfigurasi secara terpisah. cukup hanya perlu memasukkan perintah tertentu seperti add ataupun install.

2.2.21 Pip

Pip adalah sebuah tool yang dapat di gunakan untuk manajemen paket python. Pip adalah singkatan dari Pip Installs Python atau PIP Installs Packages. Yang berfungsi untuk menginstall modules. pip adalah manajer paket standar untuk Python . Ini memungkinkan untuk menginstal dan mengelola paket tambahan yang bukan bagian dari pustaka standar Python .

2.2.22 Mockup

Mockup merupakan sebuah rancangan awal dari desain web. Hal inilah yang menunjukkan mockup termasuk salah satu hal yang perlu dipertimbangkan sejak awal. Mockup bisa dibuat manual menggunakan software pengedit gambar seperti photoshop, balsamiq mockup atau lainnya. Mockup juga bisa diartikan sebagai prototipe suatu halaman website atau gambar model yang dibuat secara menyeluruh dan mendetil.

Untuk menghasilkan website yang bagus, sekiranya perlu memiliki beberapa mockup web. Hal ini mengingat mockup berfungsi memberi gambaran tampilan desain yang ingin dibuat. Dengan demikian bisa mendapat pilihan guna mendapat hasil desain terbaik. Selain itu, prototipe juga diperlukan agar memberi pengalaman fungsional saat akan dicoba. Sehingga bisa mendapat gambaran lebih jelas tentang website yang bakal dibuat, dalam mendesain tampilan user interfacenya penulis menggunakan balsamiq mockup.

2.2.23 Balsamiq Mockup

Balsamiq mockup adalah program aplikasi yang digunakan dalam pembuatan tampilan user interface sebuah aplikasi. Software ini sudah menyediakan tools yang dapat memudahkan dalam membuat desain prototyping aplikasi yang akan di buat. Software ini berfokus pada konten yang ingin digambar dan fungsionalitas yang dibutuhkan oleh pengguna.

Alih-alih menggambar sketsa (wireframe) atau prototype rancangan desain website di atas kertas balsamiq mockups membantu seorang web desainer membuat

tampilan web dalam bentuk gambar di komputer. Tujuannya selain agar membuat tampilan (desain) website menarik juga dapat menyesuaikan dengan kebutuhan customer (pelanggan). Dengan alat pembuat mockup maka seorang web desainer dapat menganalisa tata letak, desain dan fungsi.

Kelebihan Balsamiq Mockups dibanding software pembuat mockup lainnya adalah aplikasi ini berbasis cloud, disertai aplikasi desktop yang memungkinkan kita dengan cepat dan mudah membuat rancangan website. Dengan konten yang terbuat seperti dari gambaran tangan, akan membuat kita fokus pada pemecahan masalah user interface yang lebih besar, daripada pada perincian website

2.2.24 Astah Community

Astah adalah sebuah aplikasi yang bertujuan untuk membuat sebuah model rancangan diagram yang sering digunakan dalam melakukan pengembangan aplikasi. Astah secara umum dibagi menjadi dua, yaitu versi professional dan versi community.

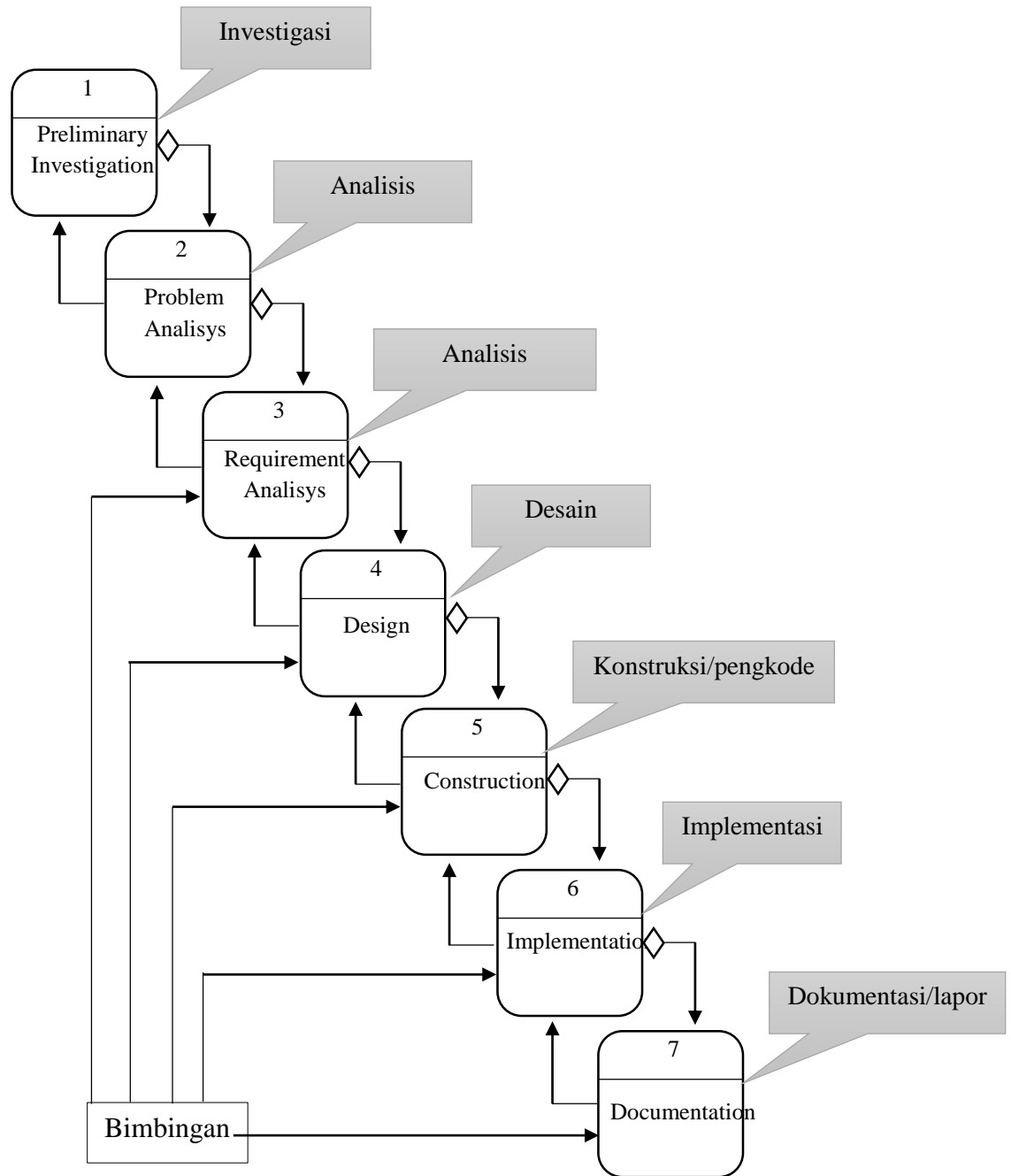
Astah Community merupakan perangkat lunak yang penulis gunakan untuk membuat model UML. Software ini berperan penting dalam perancangan program atau software, dengan adanya software ini kita dapat membuat rancangan software seperti use case, DFD, perancangan UML dan lainnya. Astah di buat oleh perusahaan Jepang bernama Change Vision. Astah pertama kali di kembangkan oleh Kenji Hiranabe CEO dari Change Vision Inc pada tahun 1996. Astah terasa “Ringan” bahkan untuk pemula sekalipun, hal ini di karenakan antar-muka yang user-friendly sehingga memungkinkan pengguna membuat diagram UML dengan mudah dan cepat.

Astah community merupakan tool gratis dengan fitur – fitur dasar, dilengkapi dengan fitur mencetak diagram, mengimpor/mengekspor ke atau dari program java. Astah community yang dulunya bernama Astah Jude telah di gunakan sedikitnya 120.000 orang di seluruh dunia pada tahun 2006 (wikipedia.org).

BAB III

METODOLOGI

3.1 Metodologi Penelitian



Gambar 3.1 metodologi penelitian

3.1 Deskripsi Tahapan penelitian

3.1.1 Preliminary Investigation (Investigasi awal)

Dalam investigasi awal, penulis melakukan observasi dan pengamatan langsung di Universitas Bale Bandung tepatnya di Fakultas Teknologi Informasi yang beralamat di Jl. R.A.A. Wirantakusumah No.07 Baleendah. Kemudian penulis bertemu dengan Dekan yang bernama bapak Yudi.Herdiana dan meminta izin untuk melakukan penelitian di tempatnya.

Setelah melakukan izin, penulis melakukan pengumpulan data dengan wawancara kepada bapak Yudi selaku dekan di Universitas Bale Bandung tepatnya di Fakultas Teknologi Informasi mengenai sistem pengurusan untuk membuat surat keterangan mahasiswa dan surat cuti kuliah.

Dalam pengurusan surat keterangan mahasiswa sistem yang berjalan saat ini di Fakultas Teknologi Informasi Universitas Bale Bandung yakni karena masih manual sehingga dalam proses pembuatan surat mahasiswa adalah mahasiswa di haruskan datang ke fakultas, untuk bertemu dengan bagian yang mengurus surat untuk membuat surat keterangan mahasiswa maupun surat cuti kuliah, apabila sudah membuat surat keterangan mahasiswa atau surat cuti kuliah kemudian di cetak, setelah dicetak surat mahasiswa di haruskan meminta persetujuan dari dekan yang meliputi tanda tangan dekan dan cap universitas.

Dikarenakan sistem pembuatan yang masih manual menjadi penyebab banyak memakan waktu dalam pengurusan surat keterangan mahasiswa dan pihak fakultas pun sulit dalam mengarsipkan surat mahasiswa berdasarkan bulan dan tahun.

3.1.2 *Problem Analysis* (Analisis masalah)

Setelah melakukan pengumpulan data dalam investigasi awal, penulis merumuskan masalah-masalah dalam membuat surat keterangan mahasiswa diantaranya:

1. Dalam membuat surat keterangan mahasiswa, mahasiswa harus bertemu dengan pihak fakultas bagian pengurusan surat mahasiswa dan dekan.
2. Sistem pembuatan surat yang masih manual karena harus dibuat dulu dan diketik dalam komputer kemudian dapat dicetak.
3. Bagian pengurusan surat mahasiswa kesulitan dalam mengarsipkan surat berdasarkan bulan dan tahun.
4. Mahasiswa kesulitan untuk mendapatkan cap dan tanda tangan jika Dekan sedang tidak ada di tempat.

3.1.3 *Requirement Analysis* (Analisis kebutuhan)

Dalam segi kebutuhan alat/*tools* yang digunakan dalam pengembangan sistem, penulis menggunakan Laptop Asus A455LD yang dipergunakan untuk berbagai tahapan dalam penelitian, seperti perancangan sistem, dokumentasi hingga pembuatan laporan baik proposal maupun skripsi.

Sedangkan bahan yang digunakan dalam melaksanakan penelitian ini diantaranya adalah:

1. Atom text editor
2. Astah Community
3. Balsamiq Mockup
4. Google Chrome

Setelah mengetahui kebutuhan apa saja yang akan digunakan, penulis tidak menerapkan analisis keputusan karena sudah mengetahui apa yang akan dibuat, jadi penulis langsung ketahap pembangunan sistem yakni tahap desain.

3.2 Metodologi Pengembangan Sistem

3.2.1 *Design (Desain)*

Desain untuk rancangan aplikasi yang akan dibuat penulis berdasarkan hasil analisis yang sudah dilaksanakan sebelumnya. Alat/tools yang akan digunakan dalam tahap rancangan ini yaitu Astah Community dan Balsamiq Mockup.

Dalam perancangan sistem, penulis menggunakan Astah Community. Tahap pertama yaitu membuat *use case diagram* untuk menggambarkan fungsionalitas utama setiap aktor pada sistem. Selanjutnya membuat *activity diagram* untuk menggambarkan aliran dari suatu aktivitas ke aktivitas lainnya pada sistem. Selanjutnya penulis membuat *class diagram* untuk menggambarkan bagaimana relasi antar model dan fungsi-fungsi dalam model tersebut.

Dalam perancangan *user interface* surat keterangan mahasiswa berbasis web ini, penulis akan membuat *mockup* dengan menggunakan aplikasi Balsamiq Mockup. Penulis akan membuat desain *user interface* dengan memperhatikan desain yang *user friendly* atau mudah di pahami saat mengoperasikannya , agar pengguna dapat dengan mudah menggunakan sistem informasi yang dibangun. Hal ini bertujuan untuk memudahkan mahasiswa dalam mengoprasikannya.

3.2.2 *Construction (Kontruksi/pengkodean)*

Dalam pengkodean penulis melakukan secara bertahap di mulai dengan merancang dan membuat design aplikasi yang telah dibuat dalam bentuk mockup, dan penulis menggunakan Atom sebagai *text* editor dalam membuat source code untuk aplikasi surat keterangan mahasiswa berbasis web di Fakultas Teknologi Informasi Universitas Bale Bandung.

menggunakan framework Django (python) sebagai *server* dan penulis menggunakan bahasa pemrograman Python dan java script dalam pembuatan aplikasinya.

Demi memudahkan dalam megoprasikannya penulis melakukan tahapan mempercantik dan menyempurnakan tampilan dengan menggunakan Framework Next Js,dan Django Rest Framework.

3.2.3 Implementation (Implementasi)

Dalam hal implementasi penulis mengerjakan sesuai yang direncanakan yaitu membuat aplikasi surat keterangan mahasiswa berbasis web yang sesuai dengan kebutuhan dan sesuai data-data yang telah dikumpulkan oleh penulis pada tahap sebelumnya. Agar aplikasi berjalan dengan lancar penulis juga melakukan pengujian terhadap aplikasi yang di buat memastikan aplikasi berjalan dengan baik dan terbebas dari *error* dan *bugs* agar aplikasi berjalan dengan lancar saat di gunakan.

3.2.4 Dokumentation (Dokumentasi /Laporan Akhir)

Dalam tahapan akhir ini penulis melaksanakan penyelesaian laporan akhir yaitu berupa skripsi yang di dalamnya terdapat tahapan yang dilakukan penulis dimulai dalam merancang,membangun aplikasi dan dokumentasi pada saat implementasi dan penggunaan aplikasi surat keterangan berbasis web.

BAB IV

ANALISIS DAN PERANCANGAN

4.1 ANALISIS

4.1.1 Analisis Masalah

Dalam proses pengelolaan kepengurusan surat keterangan mahasiswa, pihak yang terkait dalam proses pembuatan surat keterangan mahasiswa adalah ,pihak fakultas meliputi bagian pengurus surat, dekan, dan mahasiswa selaku yang membutuhkan surat keterangan mahasiswa, pihak fakultas selaku pengurus surat keterangan mahasiswa, dan Dekan selaku yang mempersetujui dalam pembuatan surat keterangan bagi mahasiswa tersebut.

Dalam pengurusan surat keterangan mahasiwa sistem yang berjalan saat ini yakni karena masih manual sehingga dalam proses pembuatan surat mahasiswa adalah mahasiswa di haruskan datang ke fakultas, untuk bertemu dengan bagian yang mengurus surat untuk membuat surat keterangan mahasiswa maupun surat cuti kuliah, apabila sudah membuat surat keterangan mahasiswa atau surat cuti kuliah kemudian di cetak, setelah dicetak surat mahasiswa diharuskan meminta persetujuan dari dekan yang meliputi tanda tangan dekan dan cap universitas.

Setelah mengetahui bagaimana alur dari pengurusan pembuatan surat keterangan mahasiswa masalah yang sering muncul adalalah mahasiswa diharuskan datang langsung ke pihak fakultas, setelah selesai membuat surat keterangan mahasiswa diharuskan ke Dekan untuk meminta persetujuan dan cap universitas, apabila dekan tidak ada difakultas sehingga mahasiswa diharuskan datang ke esokan harinya, dan sistem pembuatan yang masih manual menjadi penyebab banyak memakan waktu dalam pengurusan surat keterangan mahasiswa dan pihak fakultaspun sulit dalam mengarsipkan surat mahasiswa berdasarkan bulan dan tahun.

4.1.2 Analisis Software

Berikut adalah daftar spesifikasi software minimum yang digunakan dalam pembuatan aplikasi ini :

1. Atom text editor, aplikasi yang digunakan dalam code editor. Versi yang digunakan adalah Atom Text editor version 1.41.0. Penulis menggunakan aplikasi ini karena mempunyai kelebihan yaitu ringan dan tidak banyak memakan banyak ruang di hardisk dikarenakan ukurannya yang kecil, dengan berbagai fitur yang dapat memudahkan dalam proses membuat code saat pembuatan aplikasi.
2. Google chrome dan Mozzila firefox, aplikasi browser yang digunakan sebagai pembuka atau hasil selama proses coding. Penulis menggunakan kedua aplikasi tersebut dan untuk memandirikan interface yang digunakan browser masing-masing.
3. Os Windows, Sistem Operasi yang digunakan penulis dalam pembuatan aplikasi ini adalah Windows 10 pro 64 bit.

4.1.3 Analisis Pengguna

Penganalisaan pengguna adalah yang memakai aplikasi ini, penggunanya adalah pihak fakultas selaku Admin dan mahasiswa selaku member dengan terbangunnya aplikasi ini diharapkan dapat memudahkan mahasiswa dalam pengurusan surat mahasiswa seperti surat keterangan mahasiswa, surat cuti mahasiswa, karena aplikasi ini berbasis web sehingga dapat diakses dengan komputer/ laptop dan smartphone tanpa harus datang ke fakultas. Serta memudahkan pihak fakultas dalam pengurusan dan mngarsipkan surat keterangan mahasiswa.

4.1.4 User Interface

User Interface dari aplikasi ini sangat berpengaruh terhadap mahasiswa dan pihak fakultas dalam menggunakannya. Hal ini mencakup dalam piranti yang digunakan sebagai piranti masukan dan pengeluaran dari aplikasi ini. Karena aplikasi ini yang ditujukan untuk mempermudah terhadap pihak fakultas selaku Admin dan mahasiswa selaku member, aplikasi ini didesain dengan sederhana sehingga mudah dioprasikannya dan fitur-fitur yang disediakan sebisa mungkin tidak membuat penggunaanya bingung saat mengoprasikannya.

4.1.5 Fitur – Fitur

Fitur-fitur yang dibuat dalam aplikasi ini dimaksudkan agar pihak fakultas lebih mudah dalam mengelola pengurusan surat keterangan mahasiswa dan memudahkan mahasiswa dalam pengurusan pembuatan surat keterangan mahasiswa. Berikut adalah fitur-fitur yang disediakan oleh aplikasi ini:

1. Dapat memonitor siapa saja mahasiswa yang telah terdaftar dan membuat surat keterangan mahasiswa.
2. Menu Export dapat mengarsipkan data mahasiswa dan agenda surat berdasarkan bulan dan tahun.
3. Pengelolaan data mahasiswa, untuk menampilkan data mahasiswa dan dapat hapus, rubah data mahasiswa dan tambah data mahasiswa.
4. Menu Surat Dapat langsung membuat surat keterangan tanpa harus membuat ulang karena sudah tersedia di aplikasi.
5. Pembuatan surat keterangan mahasiswa dengan memilih jenis surat keterangan mahasiswa yang meliputi: Surat keterangan mahasiswa, Surat cuti mahasiswa.

4.1.6 Analisis Data

Berrikut adalah analisis data berupa data masukan, proses dan keluaran yang menukung aplikasi ini:

Tabel 4.1 Analisis Data

Masukan	Proses	Keluaran
Admin mengisi username dan password	Data di validasi dengan dengan database	Data Admin
Admin menambah data mahasiswa	data mahasiswa disimpan dalam database	Data mahasiswa
Mahasiswa mengisi username dan password	Data di validasi dengan dengan database	Data mahasiswa
Mahasiswa mencetak surat dengan memilih jenis surat	Aplikasi memproses cetak surat	Print out surat

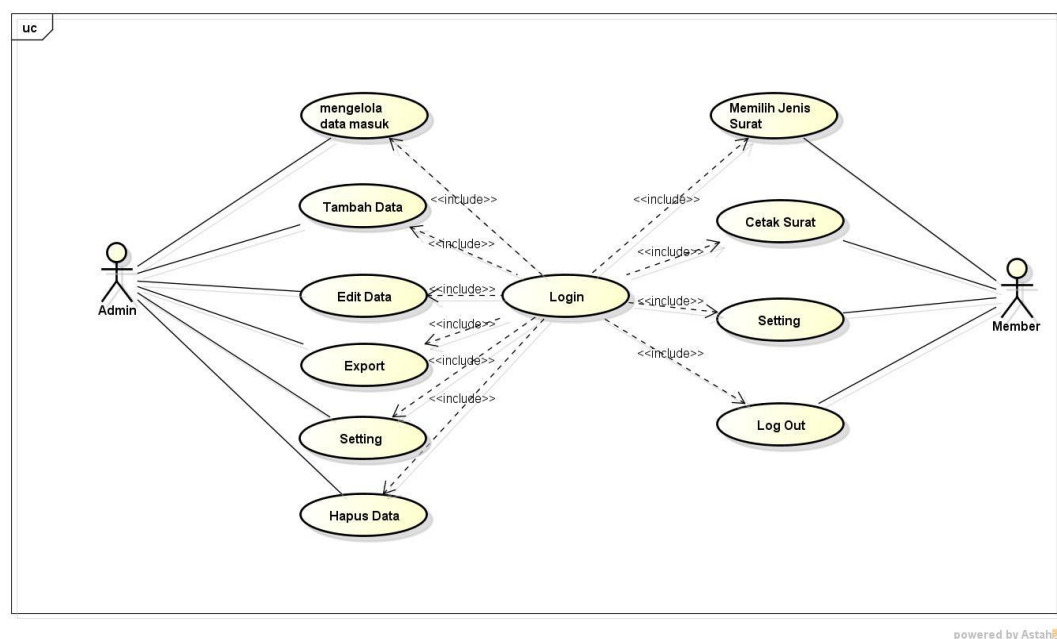
4.2 Perancangan

Pada skripsi ini perancangan model yang digunakan penulis adalah *use case* diagram untuk menggambarkan secara ringkas siapa yang menggunakan sistem dan apa saja yang bisa dilakukannya., *scenario use case* untuk menunjukan scenario utama dari usecase yang telah dibuat pada use case diagram. Pada skenario use case dijelaskan bagaimana urutan fungsionalitas berlangsung dari kondisi awal sampai kondisi akhir, *class* diagram untuk memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. dan *activity* diagram menggambarkan aktivitas sistem.

4.2.1 Unified Modeling Language (UML)

4.2.1.1 Use Case Diagram

Use case diagram ini dibuat untuk menunjukkan fungsionalitas utama dari setiap level user pada aplikasi yang digambarkan dengan actor. Terdapat 2 aktor dalam use case diagram ini yaitu pihak fakultas selaku admin dan mahasiswa selaku member.



Gambar 4.1 Use case diagram

a. Skenario Use Case

skenario use case ini dibuat untuk menunjukkan scenario utama dari usecase yang telah dibuat pada use case diagram. Pada skenario use case dijelaskan bagaimana urutan fungsionalitas berlangsung dari kondisi awal sampai kondisi akhir yang menunjukkan hasil akhir dari setiap use case.

Berikut adalah deskripsi pendefinisian admin dan member pada perancangan aplikasi Surat Keterangan Mahasiswa berbasis web:

Tabel 2.2 Deskripsi Admin dan Member (Mahasiswa).

No	Aktor	Deskripsi
1	Admin	Admin bertugas untuk mengelola data member (mahasiswa) yaitu menambah, merubah, dan menghapus member (mahasiswa).
2	Member	Member adalah mahasiswa atau pengguna aplikasi apabila sudah login dapat memilih jenis surat dan mencetak surat.

Tabel 4.3 Use case Admin

No	Use Case	Deskripsi
1	Login/Logout	Merupakan proses untuk melakukan Login (masuk) dan Logout (keluar)
2	Mengelola member	Merupakan menambah member, merubah, dan menghapus
3	Export data	Merupakan backup data member dan agenda surat berdasarkan bulan/tahun
4	Setting	Merupakan merubah password

Tabel 4.4 Use case Member (Mahasiswa)

No	Use Case	Deskripsi
1	Login/Logout	Merupakan proses untuk melakukan Login (masuk) dan Logout (keluar)
2	Memilih jenis surat	Merupakan memilih jenis surat, yang terdiri dari surat keterangan kuliah atau cuti kuliah
3	Setting	Merupakan perubahan pada password

Nama use case : Login

Tabel 4.5 Use case Login

Aksi Aktor	Reaksi Sistem
Scenario normal	
1. Memasukan username dan password	2. Memeriksa valid tidaknya data masukan dengan memeriksa ke table user
	3. Masuk ke aplikasi Surat
Scenario alternative	
1. Memasukan username dan password	
	3. Memeriksa valid tidaknya data masukan dengan memeriksa ke table user
	3. Menampilkan pesan gagal login
1. Memasukan username dan password yang valid	2. Memeriksa valid tidaknya data masukan dengan memeriksa ke table user
	3. Masuk ke aplikasi Sistem

Nama use case : Mengelola Data Masuk

Tabel 4.6 Usecase Mengelola Data Masuk

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	1. Memeriksa status login
2. Halaman utama Admin	
	3. Menampilkan Data masuk

Nama use case : Menambah Data

Tabel 4.7 Usecase Tambah data

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	1. Memeriksa status login
2. Memilih menu Tambah data	
	3. Menampilkan tampilan menu isi Tambah data
4. Mengisi data sesuai dengan kolom yang ada	
	5. Menyimpan data masukan ke basis data
	6. Menampilkan pesan sukses disimpan
Skenario Alternatif	
	1. Memeriksa status login
2. Mengisi data sesuai dengan kolom yang ada	
	3. Menampilkan pesan data gagal disimpan
4. Memperbaiki data	
	5. Menyimpan data masuk ke basis data
	6. Menampilkan pesan sukses disimpan

Nama use case : Edit/rubah Data Skenario

Tabel 4.8 Usecase Edit/rubah data

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	1. Memeriksa status login
2. Memilih Edit data	
	3. Menampilkan tampilan menu isi Edit data
4. Mengisi data sesuai dengan kolom yang ada	
	5. Menyimpan data masukan ke basis data
	6. Menampilkan pesan sukses disimpan

Nama use case : Menghapus Data Skenario

Tabel 4.9 Usecase Hapus data

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	1. Memeriksa status login
2. Memilih hapus data	
	3. Menampilkan pesan Yakin data akan di hapus
4. Memilih data yakin dihapus	
	5. Menampilkan pesan Data berhasil dihapus

Nama use case : Memilih jenis surat scenario

Tabel 4.10 Usecase Memilih jenis surat

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	1. Memeriksa status login
2. Memilih menu surat	
	3. Menampilkan tampilan menu isi jenis surat
4. Memilih jenis surat	
	5. Menampilkan surat yang dipilih

Nama use case : mencetak surat

Tabel 4.11 Usecase Mencetak surat

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	1. Memeriksa status login
2. Memilih jenis surat	
	3. Mengunduh surat yang dipilih
	4. Menampilkan pesan surat berhasil dibuat
5. Cetak surat	
	6. Menampilkan surat dalam format pdf

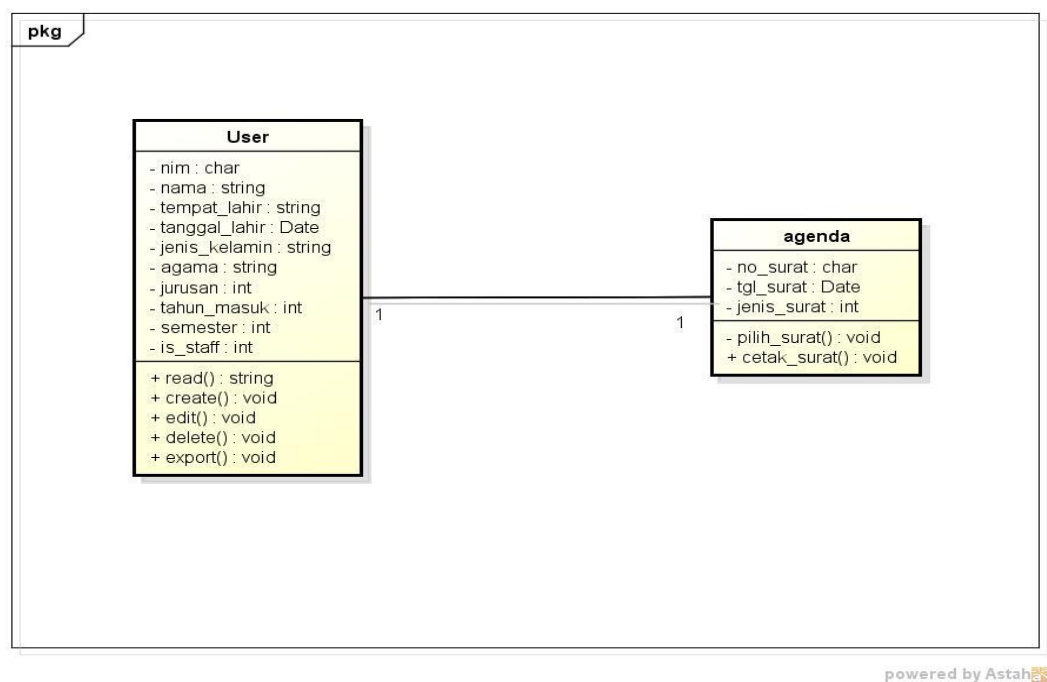
Nama use case : Logout

Tabel 4.12 Usecase Logout

Aksi Aktor	Reaksi Sistem
Scenario normal	
1. Memilih menu Logout	
	2. Melakukan Logout / keluar

4.2.1.2 Class Diagram

Class diagram menunjukkan hubungan yang terjadi pada database aplikasi. Hubungan antara tabel-tabel yang saling berkolaborasi dalam mengelola data di Aplikasi Surat Keterangan Mahasiswa berbasis Web. Berikut sekemanya :



Gambar 1.2 skema class diagram

4.3 Perancangan Basis Data

Basis data dibutuhkan untuk menyiapkan semua data-data pokok yang dibutuhkan untuk dijadikan informasi yang ditampilkan pada aplikasi yang dibuat.

4.3.1 Struktur Tabel

Struktur tabel dilakukan untuk mengetahui struktur basis data yang akan dibuat pada aplikasi sebagai media penyimpanan data agar dapat diakses dengan mudah dan cepat. Adapun struktur rancangan basis data yang akan dibuat pada aplikasi ini adalah sebagai berikut :

Tabel 4.13 struktur tabel user

Field	Type	Key	Default
nim	text(10)	Primary	TRUE
nama	text(50)		
tempat_lahir	text(50)		
tanggal_lahir	date		TRUE
jenis_kelamin	char(1)		
agama	text(10)		
jurusan	text(2)		TRUE
tahun_masuk	text(12)		TRUE
semester	int		TRUE
is_satff	boolean		FALSE

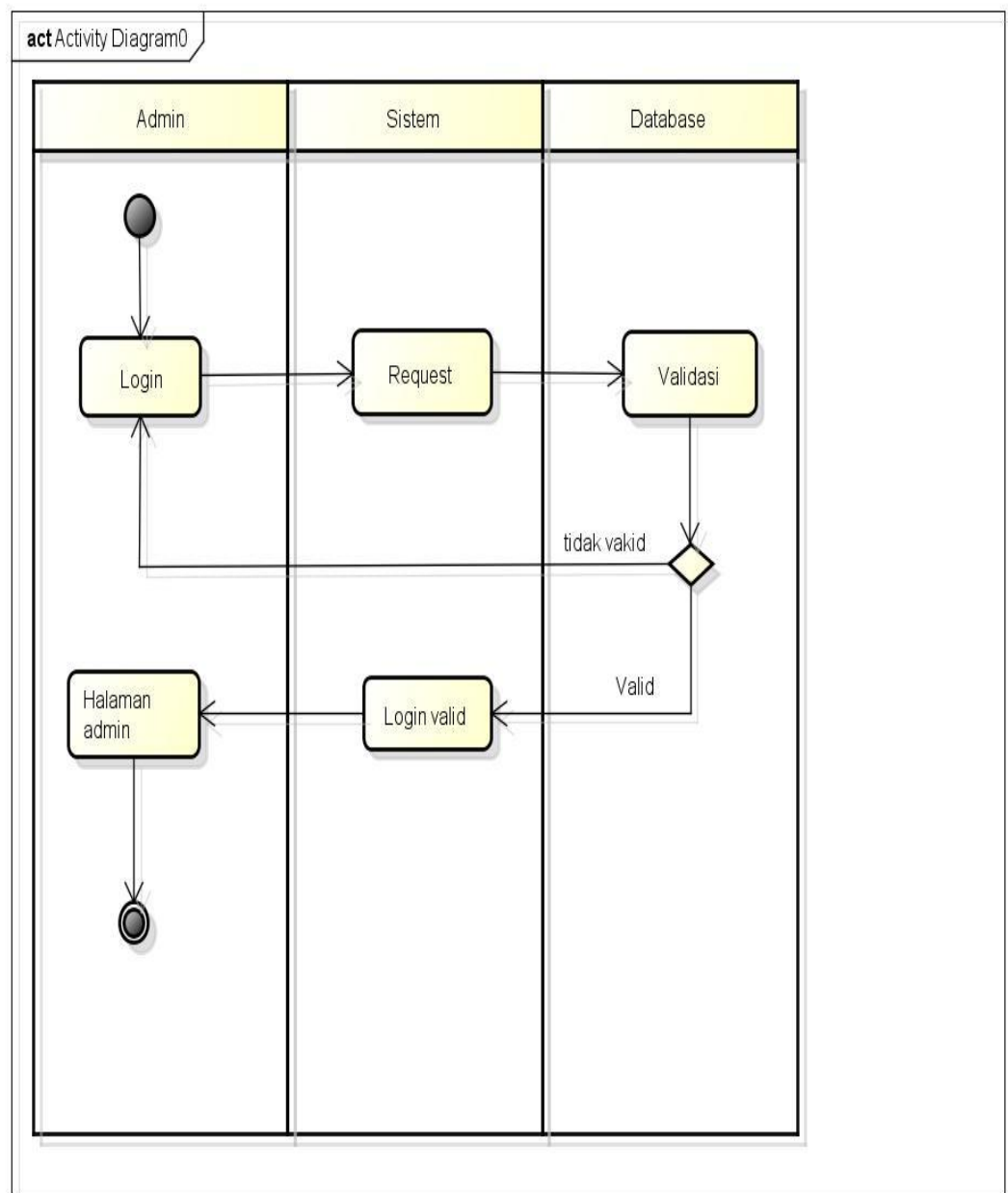
Tabel 4.14 struktur tabel agenda

Field	Type	Key
no_surat	Primary	TRUE
tgl_surat	auto	TRUE
jenis_surat	char(2)	

4.3.2 Activity Diagram

Activity Diagram ini dibuat untuk menunjukan aktivitas yang dilakukan user dan timbal balik yang dilakukan aplikasi terhadap aktivitas user secara sistematis.

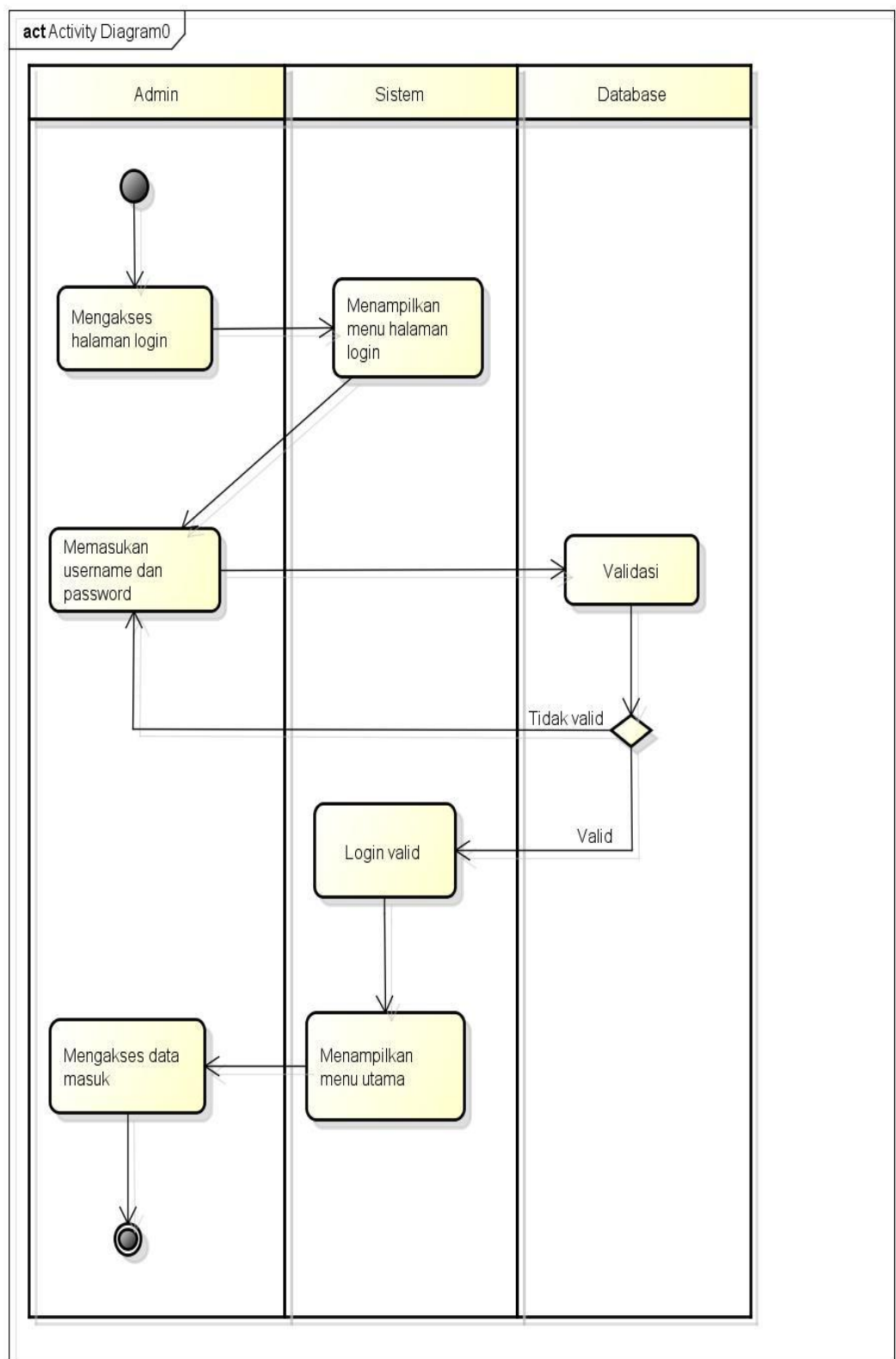
1. Activity diagram login/gagal login



powered by Astah

Gambar 4.3 Activity diagram login/gagal login

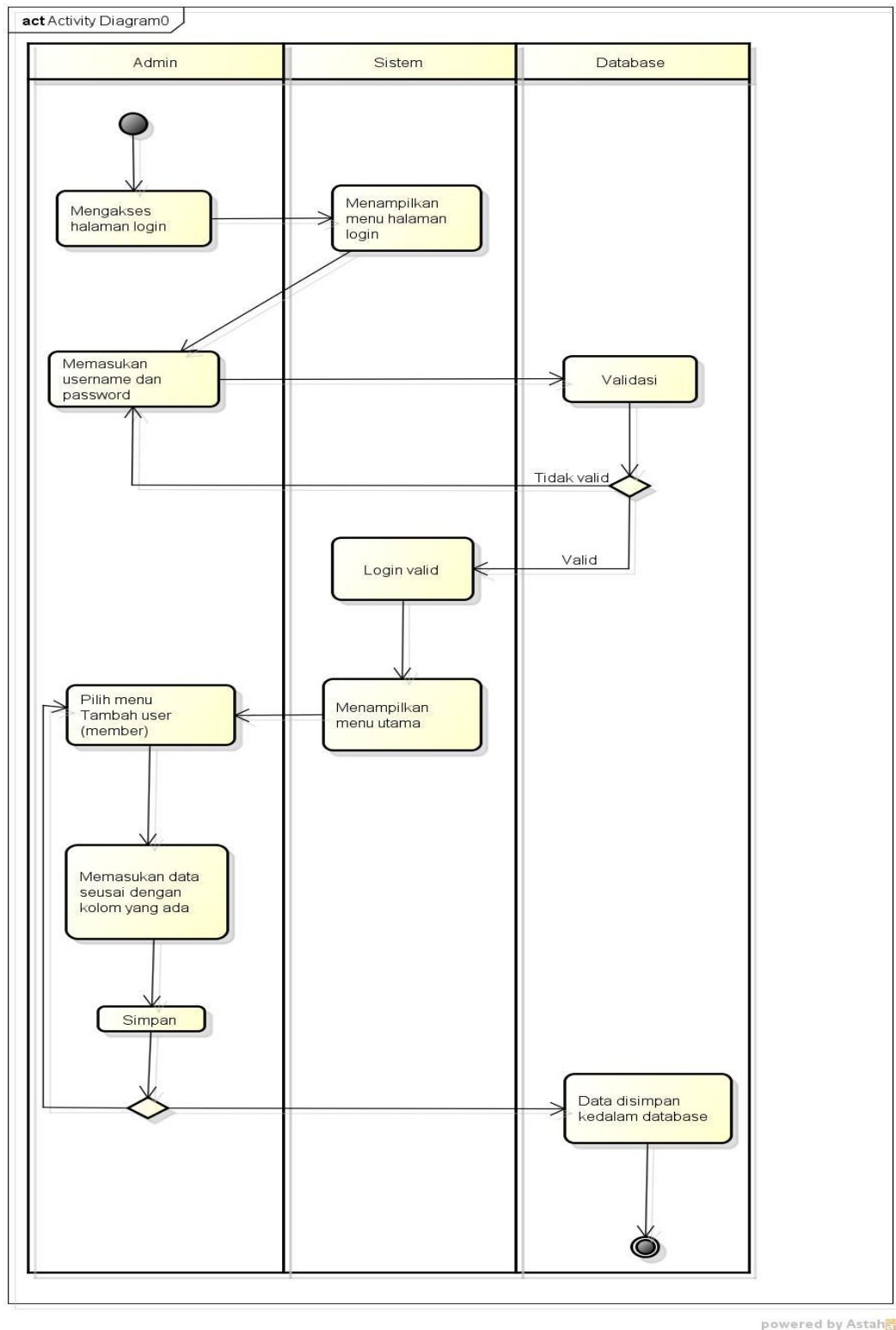
2. Activity Diagram Data Masuk



powered by Astah

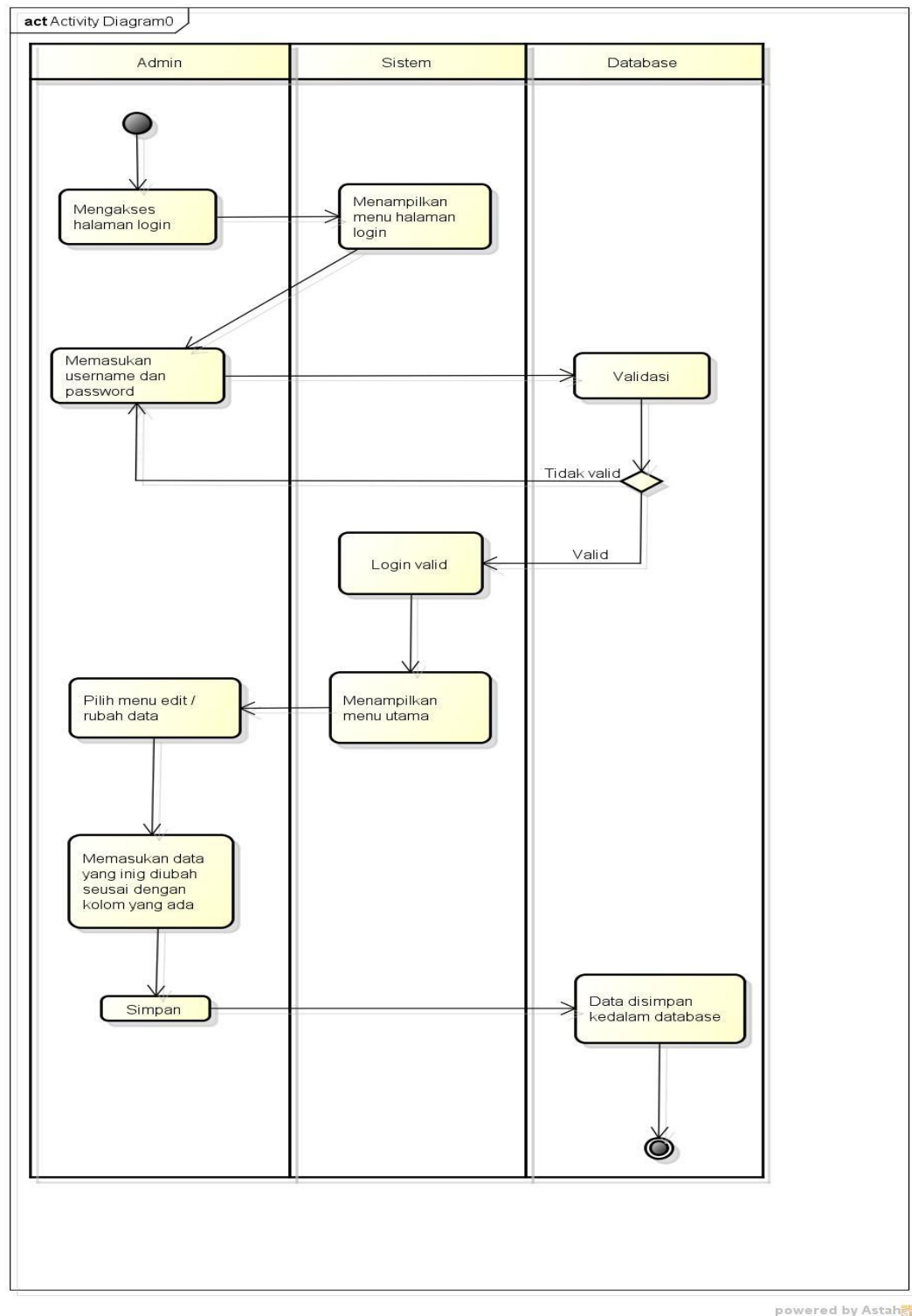
Gambar 4.4 Activity Diagram Data Masuk

3. Activity diagram Tambah data



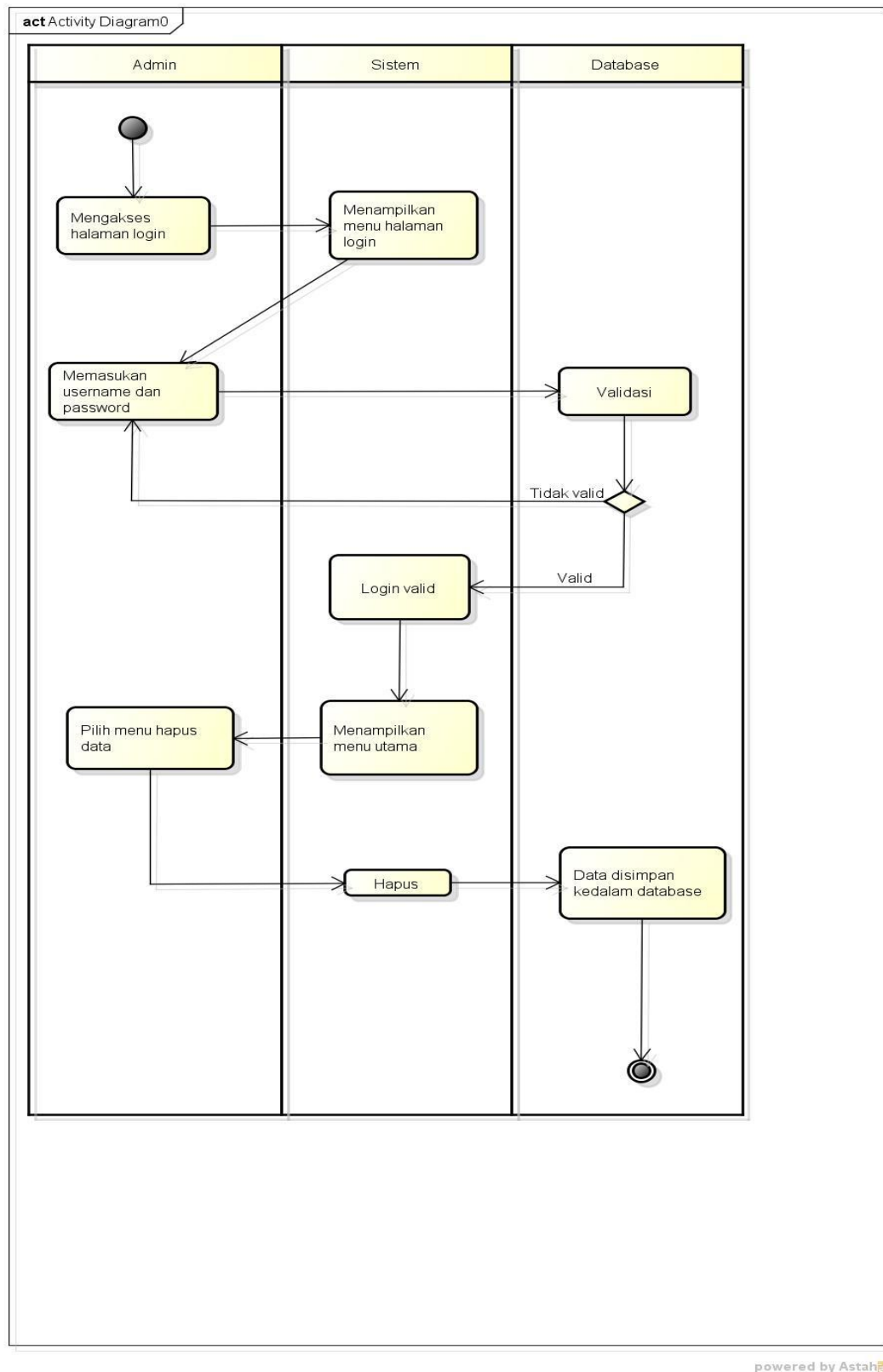
Gambar 4.5 Activity diagram Tambah data

4. Activity diagram Edit / rubah data



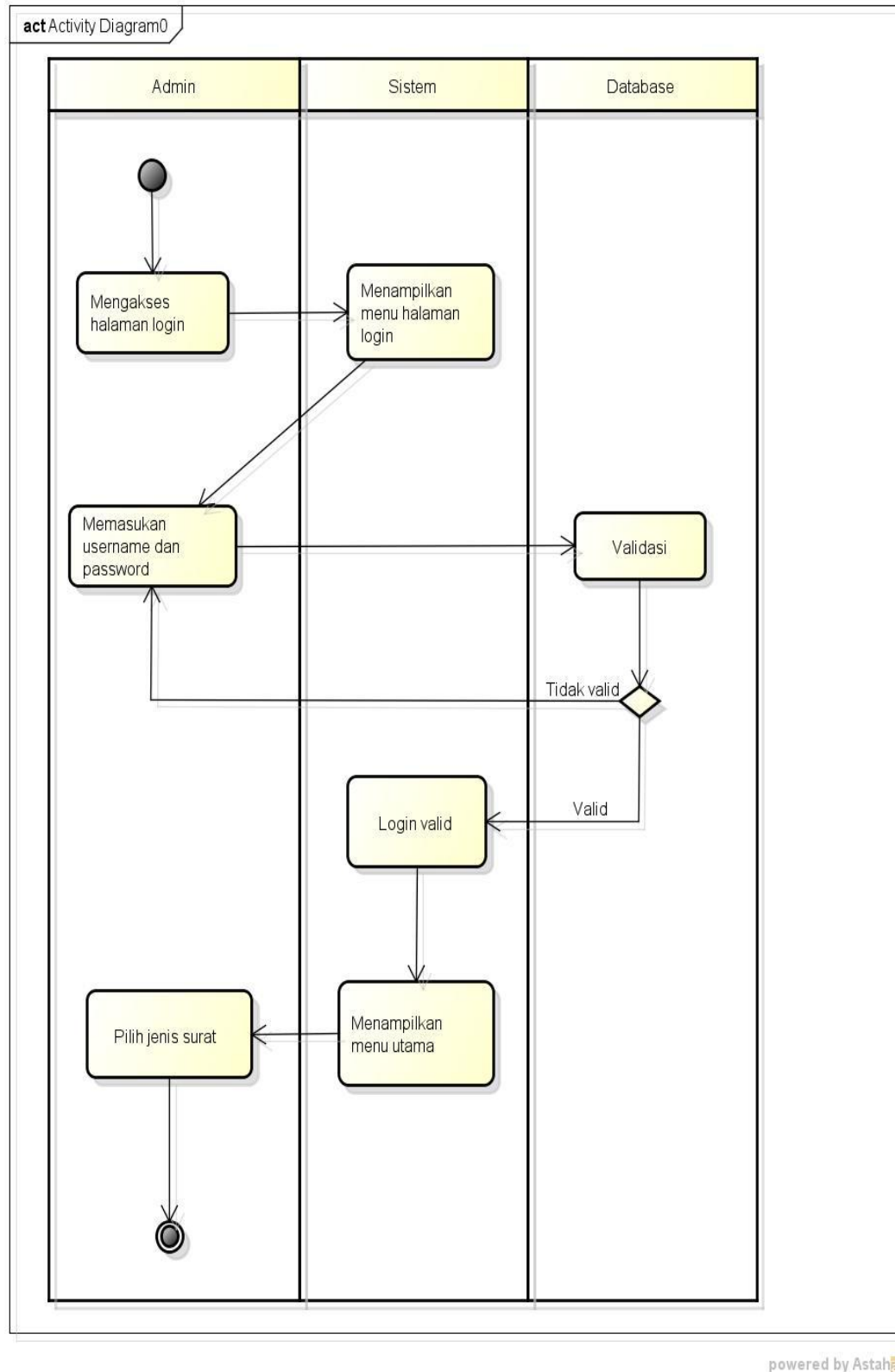
Gambar 4.6 Activity diagram Edit / rubah data

5. Activity diagram Hapus data



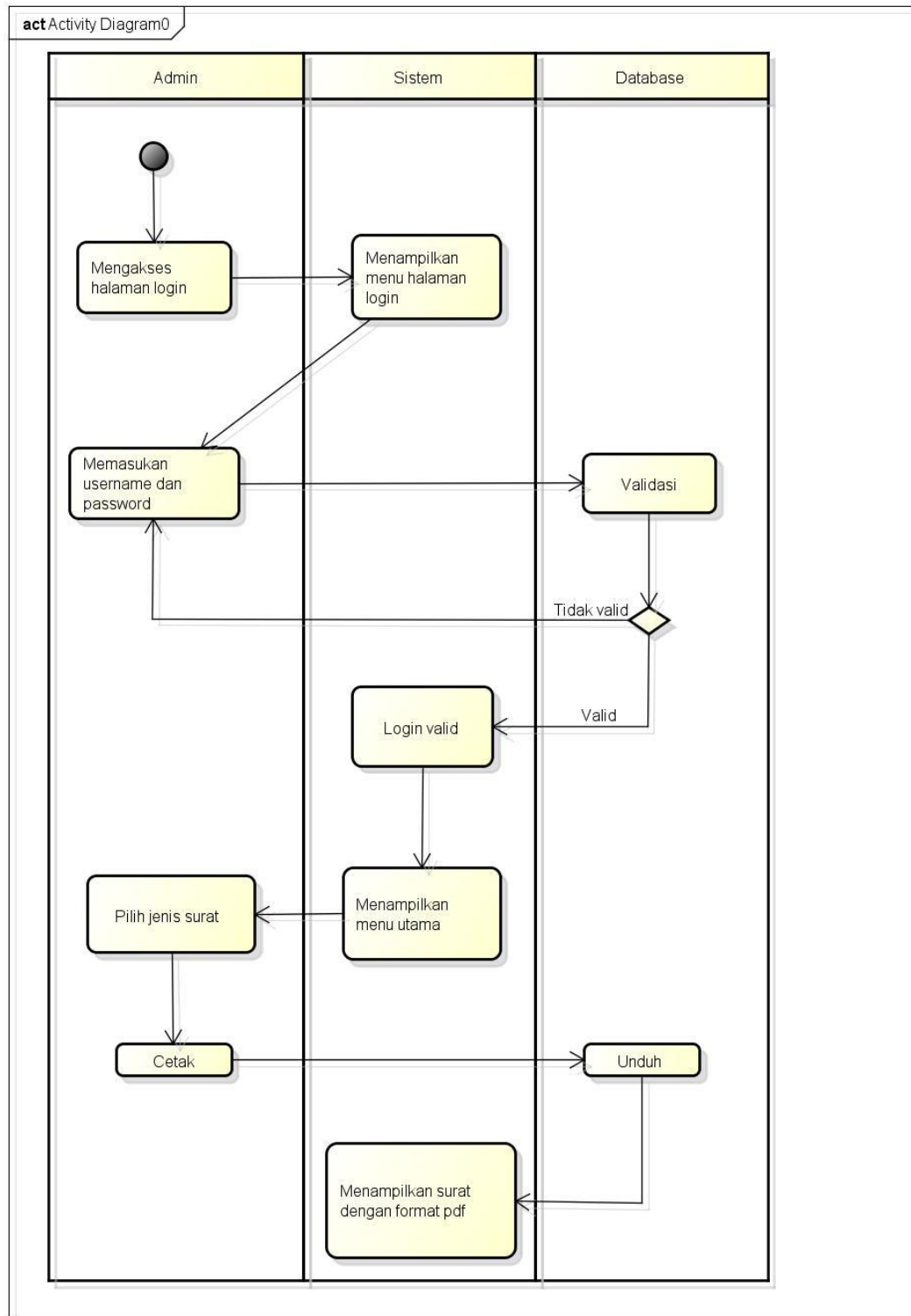
Gambar 4.7 Activity diagram Hapus data

6. Activity diagram memilih jenis surat



Gambar 4.8 Activity diagram memilih jenis surat

7. Activity diagram cetak surat



powered by Astah

Gambar 4.9 Activity diagram cetak surat

4.3.3 Desaign

Desiagn tampilan aplikasi berbasis web merupakan sebuah rancangan awal dari membuat tampilan halaman aplikasinya. Untuk membuat sebuah aplikasi yang menarik dan mudah saat digunakan maka diperlukan perancangan tampilan yang bagus dan menarik serta mudah bagi orang lain saat mengoprasikannya.

Sehingga penulis berusaha membuat tampilan yang bagus dan mendesaign rancangannya dengan mockup halaman aplikasi berbasis web agar mudah saat membuat tampilan websitenya. Dengan membuat mockup tampilan halaman aplikasinya dapat mempermudah saat membangun aplikasinya seperti menentukan fitur apa saja yang akan di seidakan di dalam halaman pada aplikasinya.

Hal inilah yang menunjukkan mockup termasuk salah satu hal yang perlu dipertimbangkan sejak awal sebelum memulai tahapan membangun aplikasi, Mockup juga bisa diartikan sebagai prototipe suatu halaman website atau gambar model yang dibuat secara menyeluruh dan mendetil.

Untuk perancangan desain antar muka dari aplikasi surat keterangan mahasiswa berbasis web ini dibuat dalam bentuk mockup dengan menggunakan aplikasi balsamiq mockup hal ini bertujuan untuk memudahkan dalam pembuatan tampilan user interface di aplikasi yang akan dibuat.

Berikut adalah desain antar muka untuk aplikasi surat keterangan mahasiswa berbasis webnya :

a. Rancangan halaman antar muka awal dan login

Gambar di atas merupakan rancangan tampilan awal halaman aplikasi untuk login bagi Admin maupun member (mahasiswa) sebelum masuk ke Aplikasi berikut tampilan desainnya :

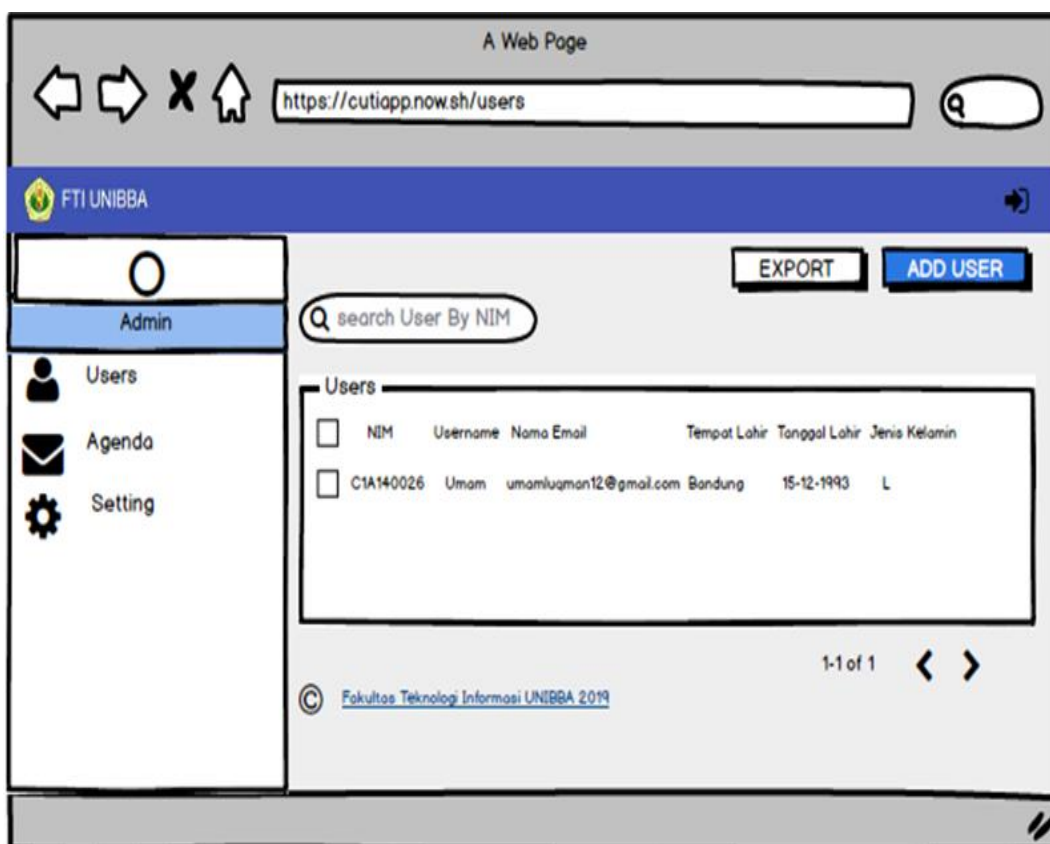
Gambar 4.10 Rancangan halaman awal Login

Dalam tampilan halaman awal aplikasi surat keterangan mahasiswa berbasis web ini tepatnya di halaman awal sebelum login, tampilan bagian sebelah kanan dari aplikasi terdapat visi FTI Unibba, dan di samping kiri terdapat form username untuk menampung nama user sebelum login dan dibawah form username terdapat form password untuk menampung password dan tombol sign now untuk mengeksekusi setelah login sehingga data username dan password akan di tampung dan di validasi didalam database apabila gagal harus melakukan login dengan mengisi username dan password yang sesuai dengan

data yang ditampung di dalam database dan apabila berhasil login maka akan diarahkan ke halaman selanjutnya.

b. Rancangan halaman sesudah login Admin

Halaman awal Admin setelah login, disuguhkan dengan berbagai fitur seperti add user, export user, penulis juga ingin membuat admin dapat melihat member atau mahasiswa yang sudah terdaftar di aplikasi. Admin juga mempunyai akses untuk menambah member (mahasiswa), merubah dan menghapus member. Berikut tampilan desainnya :

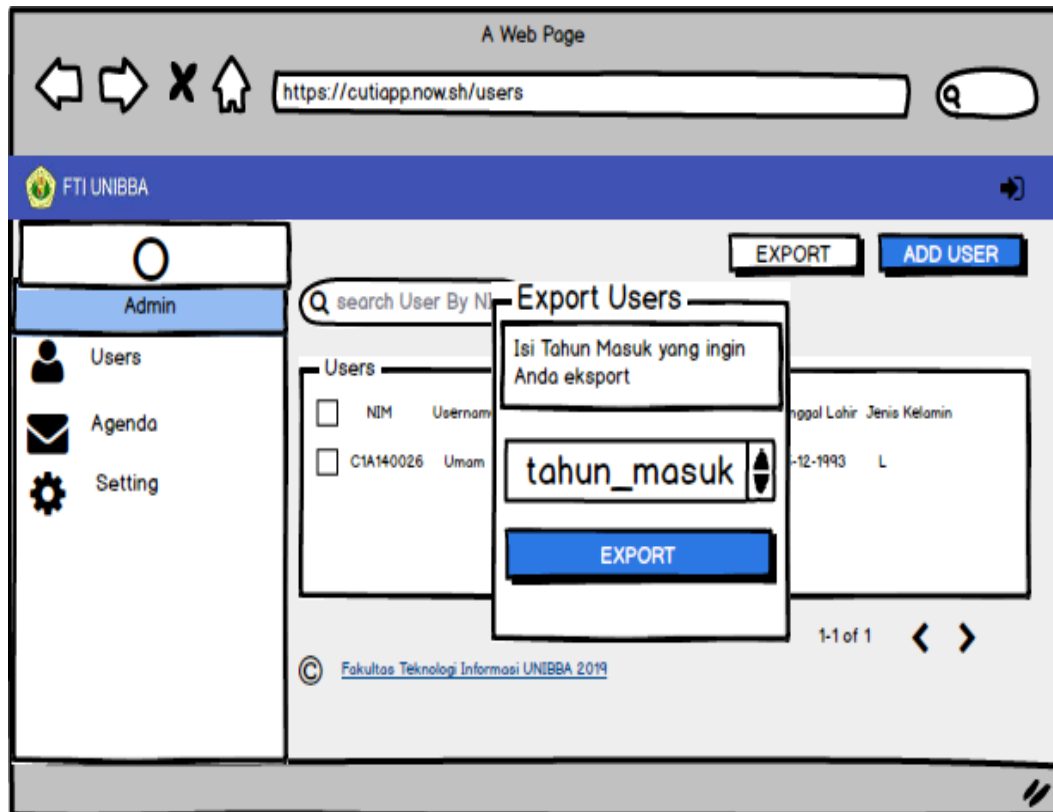


Gambar 4.11 Rancangan halaman sesudah Login Admin

Dalam tampilan halaman admin setelah melakukan login terdapat berbagai menu yaitu export user untuk mengarsipkan member (mahasiswa) berdasarkan tahun masuk, add user untuk menambah member pada aplikasi dan pada bagian sidebar terdapat agenda untuk berpindah ke halaman agenda, dan setting untuk berpindah ke halaman setting pada admin.

c. Rancangan halaman Export member

Halaman Export data ini Admin dapat mengexport atau mengarsipkan data user berdasarkan tahun masuk, hal ini dapat mempermudah admin dalam mengarsipkan member (mahasiswa) berikut tampilan desain mockupnya :



Gambar 4.12 Rancangan halaman export user

Didalam halaman export user terdapat form untuk mengisi tahun masuk hal ini bertujuan untuk mengarsipkan user berdasarkan tahun masuk, tombol export untuk mengeksekusi setelah tahun masuk sudah diisi.

Pada bagian sidebar terdapat menu users untuk kembali ke halaman awal, menu agenda untuk berpindah ke halaman agenda, dan menu setting untuk berpindah ke halaman setting pada admin. Dan disamping atas terdapat tombol logout untuk keluar dari aplikasi.

d. Rancangan halaman tambah user

Halaman tambah user ini dapat dilakukan oleh Admin bagi siapa saja mahasiswa yang memerlukan surat keterangan maupun cuti bagi mahasiswa diwajibkan membuat akun kepada admin hal ini dilakukan agar tidak ada mahasiswa yang bersasal dari universitas lain yang membuat surat keterangan maupun cuti mahasiswa. berikut tampilan desaign mockupnnya :

The mockup shows a web interface for adding users. It features a sidebar with navigation links: Admin (selected), Users, Agenda, and Setting. The main form is titled 'Account' and includes the following fields:

- nim (text input)
- nama (text input)
- username (text input)
- password (text input)
- email (text input)
- tempat lahir (text input)
- tanggal_lahir (text input)
- Agama (dropdown menu)
- tahun masuk (dropdown menu)
- jenis kelamin (dropdown menu)
- jurusan (dropdown menu)
- semester (dropdown menu)

A 'SAVE' button is located below the form fields. The footer of the page reads '© Fakultas Teknologi Informasi UNIBBA 2019'.

Gambar 4.13 Halaman antar muka Admin

Halaman add user ini bertujuan untuk menambahkan user mahasiswa selaku member, di halaman add user ini terdapat berbagai form yang harus di isi dengan data yang sesuai dengan mahasiswa, meliputi :

- Form nim = mengisi nim mahasiswa
- Form nama = mengisi nama mahasiswa
- Form username = untuk mengisi nama username mahasiswa
- Form password = untuk mengisi password mahasiswa
- Form email = untuk mengisi nama email mahasiswa
- Form tempat lahir = untuk mengisi tempat lahir berdasarkan mahasiswa
- Form tanggal lahir = untuk mengisi tanggal lahir mahasiswa
- Form jenis kelamin = untuk mengisi jenis kelamin berdasarkan mahasiswa

- Form agama = untuk memilih agama berdasarkan agama pada mahasiswa
- Form jurusan = untuk mengisi jurusan berdasarkan jurusan mahasiswa
- Form tahun masuk = untuk mengisi tahun masuk berdasarkan tahun masuk mahasiswa
- Form semester = untuk mengisi semester mahasiswa
- Tombol save untuk menyimpan data apabila sudah selesai melakukan tambah data mahasiswa.
- Disamping atas terdapat fitur logout untuk keluar dari aplikasi.

pada bagian sidebar terdapat menu users untuk kembali kehalaman awal, menu agenda untuk berpindah ke halaman agenda, dan setting untuk berpindah ke halaman setting pada admin.

e. Rancangan halaman Edit member

Halaman edit member ini hanya bisa di lakukan oleh admin untuk merubah data kalau ada yang salah. berikut tampilan desainnya :

A Web Page

https://cutiapp.now.sh/edit-user?nim=C1A140026

FTI UNIBBA

Admin

Users

Agenda

Setting

Account

C1A140026

Umam

password

umamluqman@gmail.com

Bandung

15/12/1993

Laki-laki

Islam

Informatika

2014

8

SAVE

© Fakultas Teknologi Informasi UNIBBA 2019

Gambar 4.14 Halaman edit member

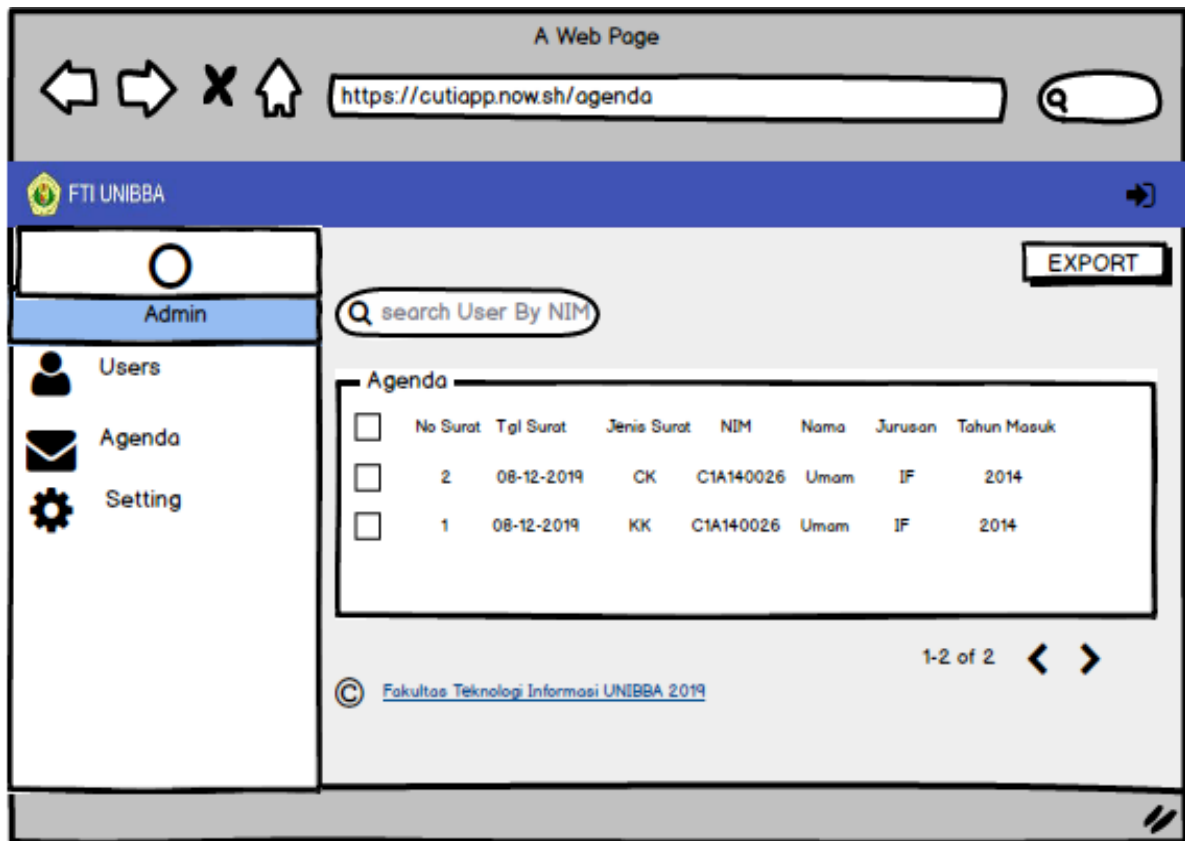
Halaman edit member ini bertujuan untuk merubah data mahasiswa selaku member apabila terdapat data yang salah meliputi :

- Form nim = merubah nim mahasiswa.
- Form nama = merubah nama mahasiswa.
- Form username = untuk merubah nama username mahasiswa.
- Form password = untuk merubah password mahasiswa.
- Form email = untuk merubah nama email mahasiswa.
- Form tempat lahir = untuk merubah tempat lahir berdasarkan mahasiswa.
- Form tanggal lahir = untuk merubah tanggal lahir mahasiswa.
- Form jenis kelamin = untuk merubah jenis kelamin mahasiswa.
- Form agama = untuk merubah agama berdasarkan agama pada mahasiswa.
- Form jurusan = untuk merubah jurusan berdasarkan jurusan mahasiswa.
- Form tahun masuk = untuk merubah tahun masuk mahasiswa.
- Form semester = untuk merubah semester mahasiswa.
- Tombol save untuk menyimpan data apabila sudah selesai melakukan perubahan.

pada bagian sidebar terdapat menu users untuk kembali kehalaman awal, menu agenda untuk berpindah ke halaman agenda, dan menu setting untuk berpindah ke halaman setting pada admin.

f. Rancangan halaman Agenda

Halaman Agenda yang bertujuan agar admin dapat melihat siapa saja member yang telah membuat surat keterangan atau cuti mahasiswa . berikut tampilan desainnya :



Gambar 4.15 Halaman Agenda

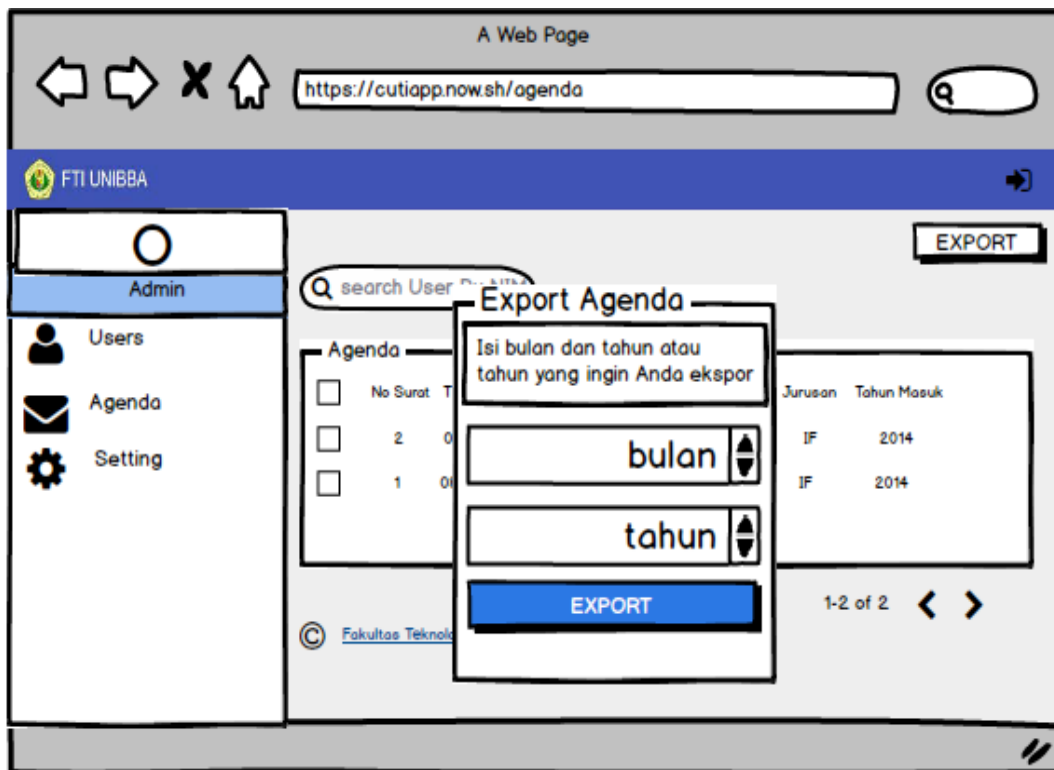
Halaman agenda ini terdapat fitur yang dapat dilakukan oleh admin meliputi:

- Admin dapat melihat data siapa saja mahasiswa yang membuat surat keterangan mahasiswa dan surat cuti kuliah.
- Admin dapat menghapus data mahasiswa yang membuat surat di agenda
- Admin dapat mengarsipkan data agenda berdasarkan tahun.
- Search bertujuan untuk memudahkan dalam pencarian berdasarkan nomor surat.

pada bagian sidebar terdapat menu users untuk kembali kehalaman awal, dan menu setting untuk berpindah ke halaman setting pada admin.

g. Rancangan halaman Export Agenda

Halaman Export Agenda yang dapat dilakukan oleh Admin bertujuan untuk dapat mengexport atau mengarsipkan data mahasiswa yang membuat surat berdasarkan bulan dan tahun. berikut tampilan desainnya :



Gambar 4.16 Halaman Export Agenda

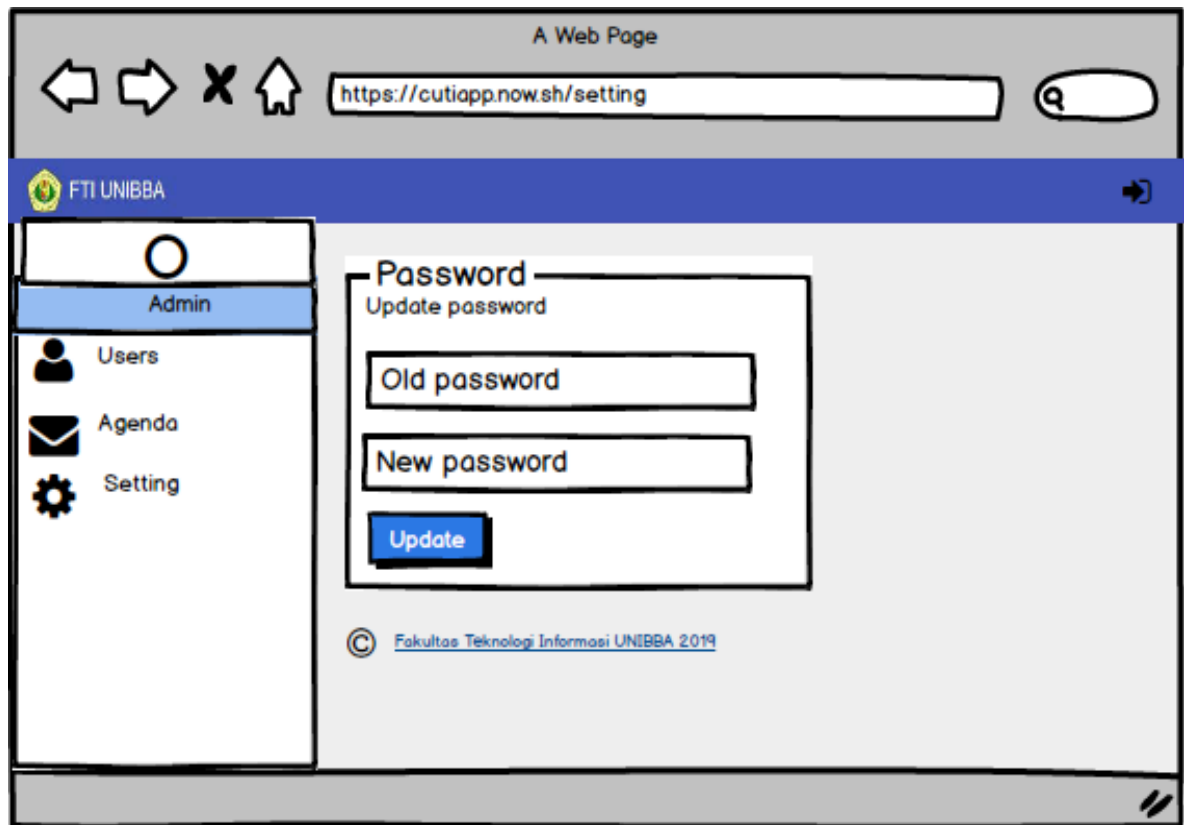
Di dalam desain halaman export agenda ini terdapat beberapa fitur antara lain :

- tombol export yang berguna untuk mengarsipkan data agenda
- form bulan untuk mengisi nama bulan.
- form tahun untuk mengisi nama tahun.
- tombol export untuk mngeksekusi.

pada bagian sidebar terdapat menu users untuk kembali kehalaman awal, dan menu setting untuk berpindah ke halaman setting pada admin.

h. Setting admin

Halaman setting pada admin bertujuan untuk melakukan perubahan atau mengganti password. berikut tampilan desainnya :



Gambar 4.17 Halaman Setting

Desain tampilan setting admin terdapat form yaitu

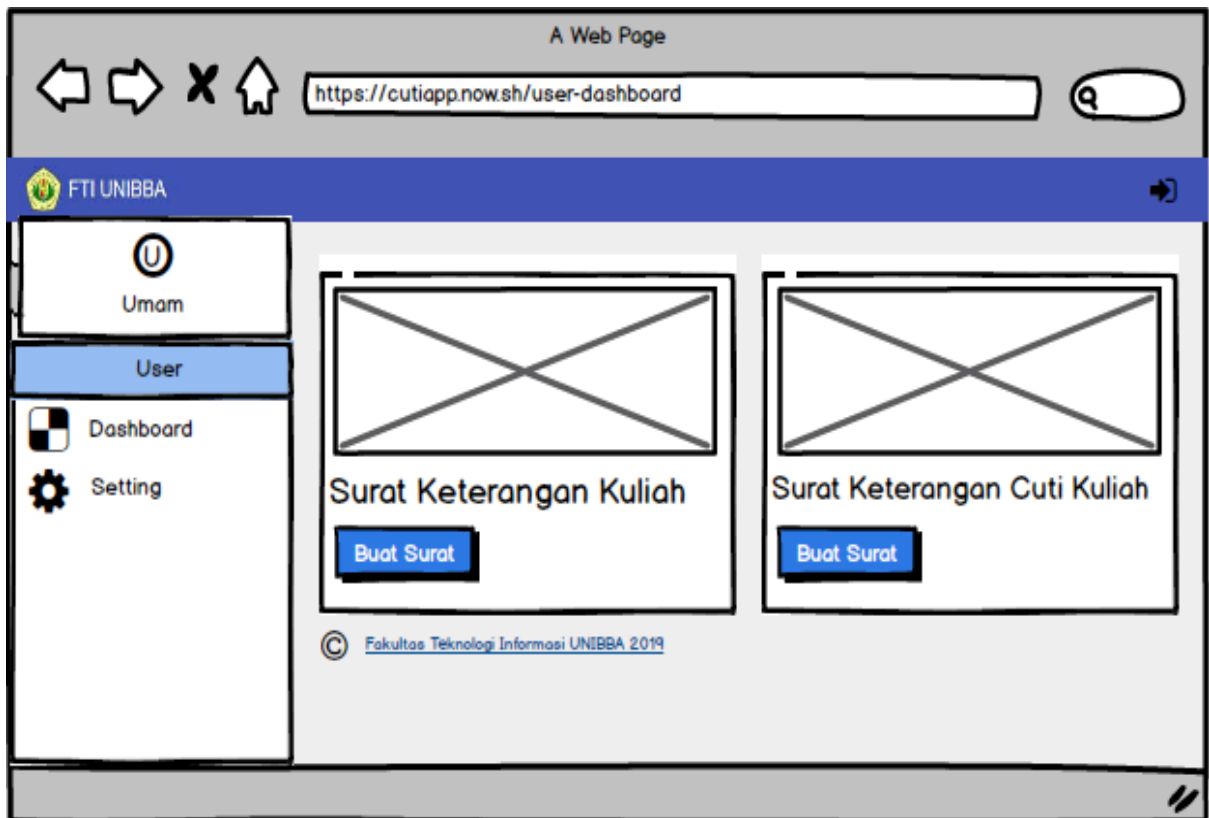
- Form old password = untuk mengisi password lama
- Form new password = untuk mengisi password yang baru
- Tombol update = untuk mengeksekusi apabila telah melakukan perubahan dan otomatis ditampung kedalam database

pada bagian sidebar terdapat menu users untuk kembali kehalaman awal, menu agenda untuk berpindah ke halaman agenda.

i. Rancangan halaman awal member (mahasiswa)

Halaman awal member setelah melakukan login akan disuguhkan dengan beberapa fitur dapat memilih jenis surat sebelum di buat dan di cetak.

berikut tampilan desain mockupnya :



Gambar 4.18 Halaman awal member (mahasiswa)

Penampilan desain halaman awal member terdapat fitur – fitur meliputi :

- Surat keterangan kuliah = untuk membuat surat keterangan kuliah.
- Surat cuti kuliah = untuk membuat surat cuti kuliah.
- Tombol buat surat = untuk mngeksekusi surat yang akan dibuat.

Disamping halaman terdapat sidebar yang terdapat menu halaman meliputi :

- Dashboard = menampilkan halaman awal.
- Setting = menampilkan halaman setting pada member.

Di pojok atas terdapat tombol logout untuk keluar dari aplikasi.

j. Rancangan halaman setting pada member (mahasiswa)

Halaman setting pada member (mahasiswa) dapat melihat rincian data diri dan dapat merubah atau mengganti password. berikut tampilan desainnya :

Gambar 4.19 Halaman setting member (mahasiswa)

Penampilan desain mockup halaman setting pada member user dapat user dapat melihat data diri meliputi :

- nim = merubah nim mahasiswa.
- nama = merubah nama mahasiswa.
- username = untuk merubah nama username mahasiswa.
- email = untuk merubah nama email mahasiswa.
- tempat lahir = untuk merubah tempat lahir berdasarkan mahasiswa.

- tanggal lahir = untuk merubah tanggal lahir mahasiswa.
- jenis kelamin = untuk merubah jenis kelamin mahasiswa.
- agama = untuk merubah agama berdasarkan agama pada mahasiswa.
- jurusan = untuk merubah jurusan berdasarkan jurusan mahasiswa.
- tahun masuk = untuk merubah tahun masuk mahasiswa.
- semester = untuk merubah semester mahasiswa.

Dan disamping data diri member dapat merubah password

- Form old password = untuk mengisi password lama
- Form new password = untuk mengisi password yang baru
- Tombol update = untuk mengeksekusi apabila telah melakukan perubahan dan otomatis ditampung kedalam database
- pada bagian sidebar terdapat menu Dashboard untuk kembali kehalaman awal.

4.3.4 Hasil

4.3.4.1 Menjalankan Sistem

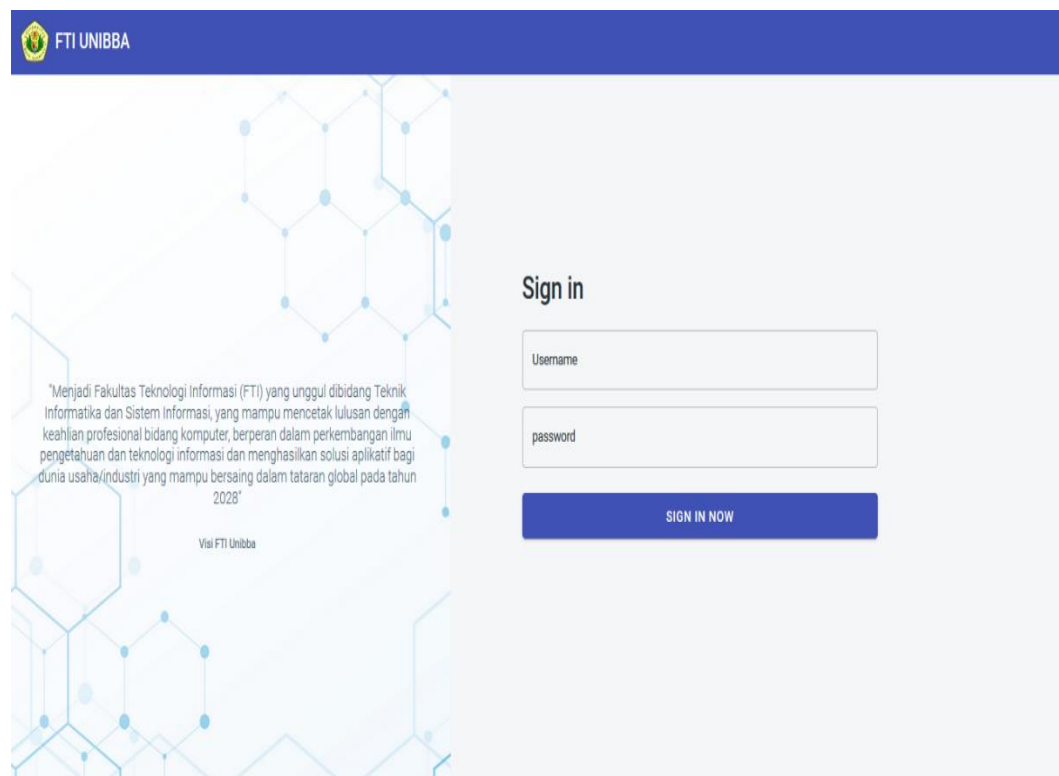
Aplikasi yang dibangun oleh penulis adalah aplikasi membuat surat keterangan mahasiswa berbasis web yang bertujuan untuk memudahkan mahasiswa dalam membuat surat keterangan mahasiswa maupun cuti mahasiswa dan memudahkan pihak fakultas selaku admin dalam mengelola surat keterangan mahasiswa dan dapat dengan mudah mengarsipkannya berdasarkan bulan dan tahun.

Aplikasi surat keterangan mahasiswa berbasis web di Fakultas Teknologi Informasi Universitas Bale Bandung karena bertujuan untuk memudahkan mahasiswa sehingga aplikasi ini dibuat bersifat responsive mengingat bahwa aplikasi ini berbasis web sehingga dapat di akses menggunakan PC / laptop, tablet maupun Smartphone android.

Berikut adalah tampilan pada aplikasi yang sudah berhasil di bangun di Fakultas Teknologi Informasi Universitas Bale Bandung.

Halaman awal login antara Admin dan Member.

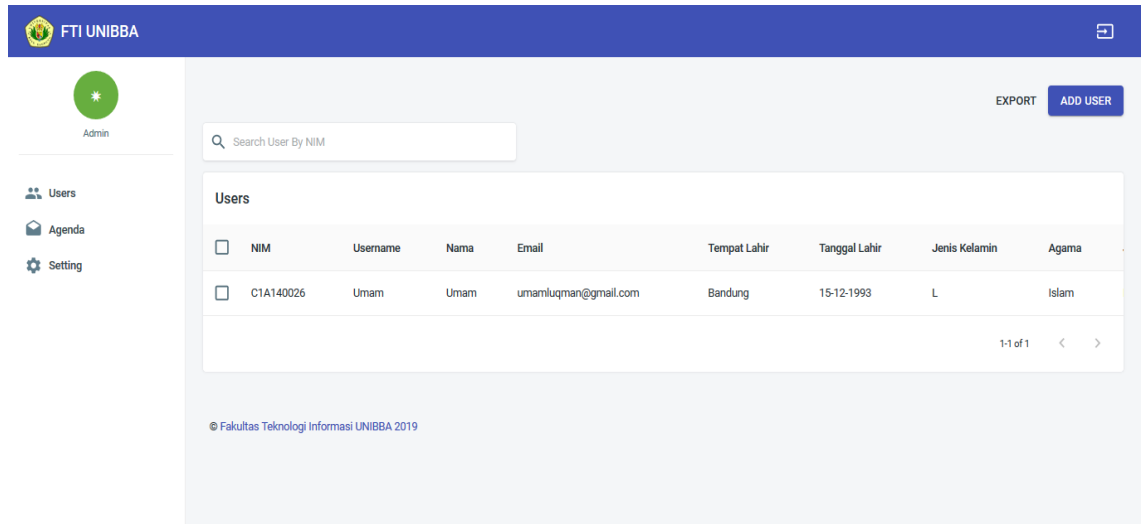
Halaman login merupakan hak akses untuk member (mahasiswa) jika ingin membuat surat keterangan atau cuti mahasiswa dan akases sebagai admin untuk mengelola data dan agenda surat . berikut tampilan halaman pada aplikasinya :



Gambar 4.20 Halaman login

a. Halaman awal admin setelah login

Halaman awal admin setelah melakukan login dapat memonitor mahasiswa yang sudah terdaftar sebelumnya di aplikasi. berikut tampilan halaman pada aplikasinya :

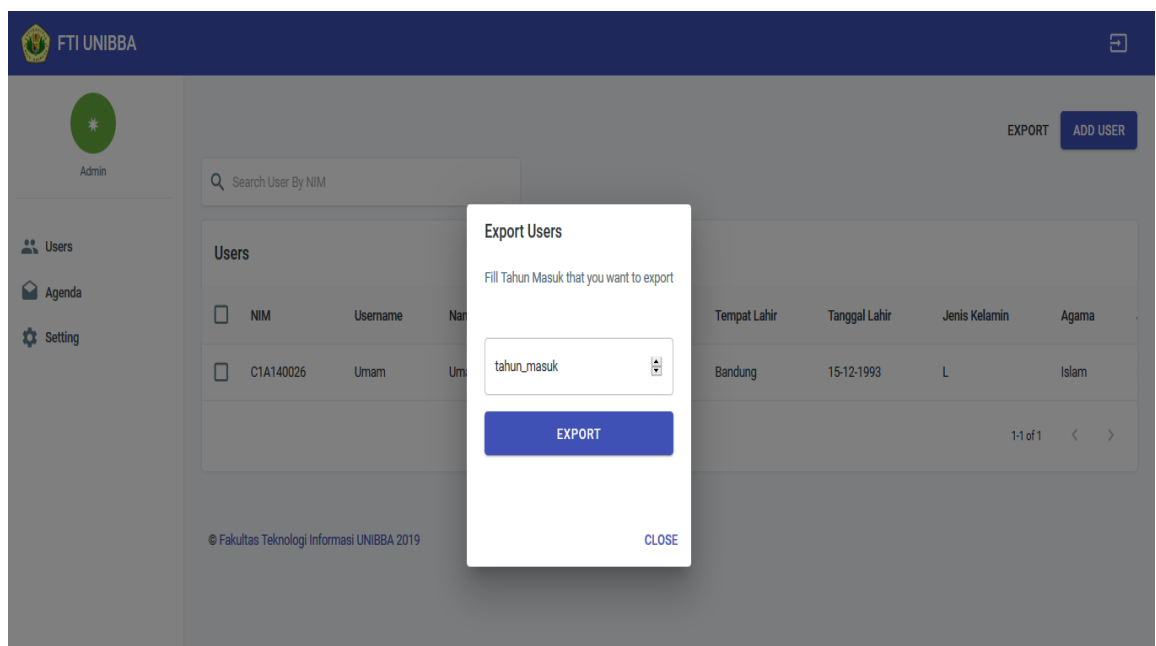


Gambar 4.21 halaman awal admin

b. Export member

Halaman export member yang dapat dilakukan oleh admin untuk mengarsipkan data member (mahasiswa) yang telah terdaftar di aplikasi dapat di arsipkan berdasarkan tahun masuk mahasiswa.

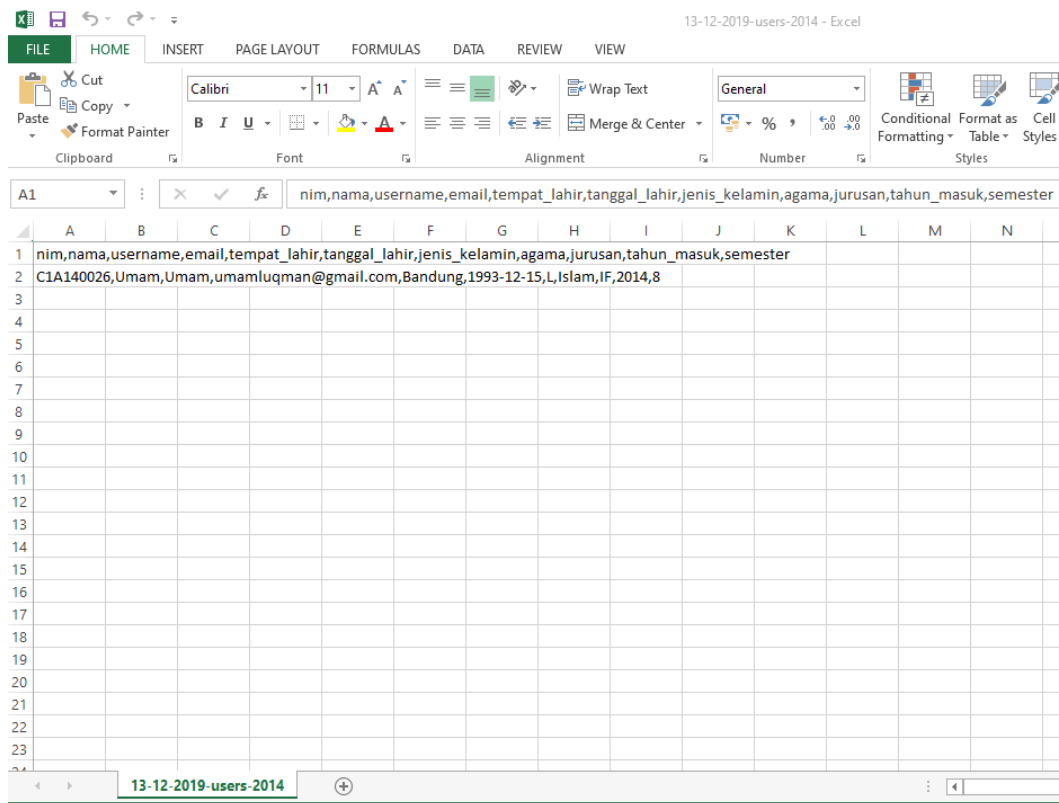
berikut tampilan halaman pada aplikasinya :



Gambar 4.22 halaman Export member

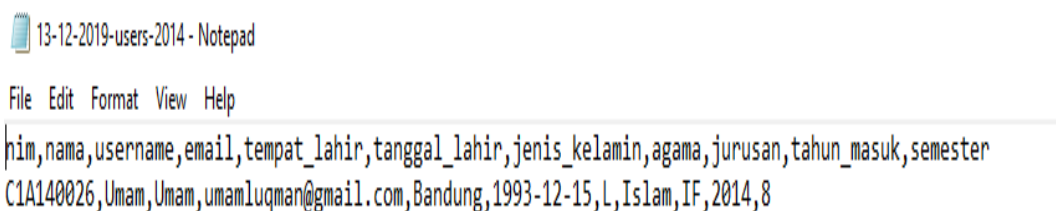
- c. Hasil export member berekstensi .csv dapat dibuka dengan ms.excel atau dengan notepad.

Berikut hasil tampilannya : Menggunakan ms. excel



Gambar 4.23 halaman hasil Export member menggunakan ms.excel

Berikut hasil tampilannya : Menggunakan notepad



Gambar 4.24 halaman hasil Export member menggunakan notpad

d. Tambah member (user)

Halaman tambah data user ini dapat dilakukan oleh admin untuk siapa saja mahasiswa yang ingin membuat surat harap mendaftarkan terlebih dahulu kepada admin, fitur ini bertujuan untuk mencegah agar tidak sembarang mahasiswa dari universitas lain melakukan login dan membuat surat di aplikasi dan aplikasi ini di buat di khususkan hanya untuk mahasiswa Fakultas Teknologi Informasi Universitas Bale Bandung. berikut tampilan halaman pada aplikasinya :

Gambar 4.25 tambah member

e. Edit member

Halaman ini bertujuan untuk merubah data apabila terdapat data member (mahasiswa) yang tidak sesuai dengan data mahasiswa dapat melapor terhadap pihak admin agar data yang tidak sesuai dapat di betulkan. berikut tampilan halaman pada aplikasinya :

FTI UNIBBA

U
Umam
User

Dashboard
Setting

Account

NIM: C1A140026

name: Umam

username: Umam

password:

email: umamluqman@gmail.com

tempat lahir: Bandung

tanggal_lahir: 15/12/1993

jenis_kelamin: Laki-Laki

agama: Islam

jurusan: INFORMATIKA

tahun_masuk: 2014

semester: 8

SAVE

Gambar 4.26 halaman Edit member

f. Agenda

Halaman agenda ini memudahkan admin untuk melihat siapa saja mahasiswa yang membuat surat keterangan dan cuti mahasiswa. berikut tampilan halaman pada aplikasinya :

FTI UNIBBA

Admin

Users
Agenda
Setting

EXPORT

Search Agenda By No Surat

Agenda

<input type="checkbox"/>	No Surat	Tgl Surat	Jenis Surat	NIM	Nama	Jurusan	Tahun Masuk
<input type="checkbox"/>	12	11-12-2019	CK	C1A140026	Umam	IF	2014
<input type="checkbox"/>	11	11-12-2019	KK	C1A140026	Umam	IF	2014
<input type="checkbox"/>	10	11-12-2019	KK	C1A140026	Umam	IF	2014

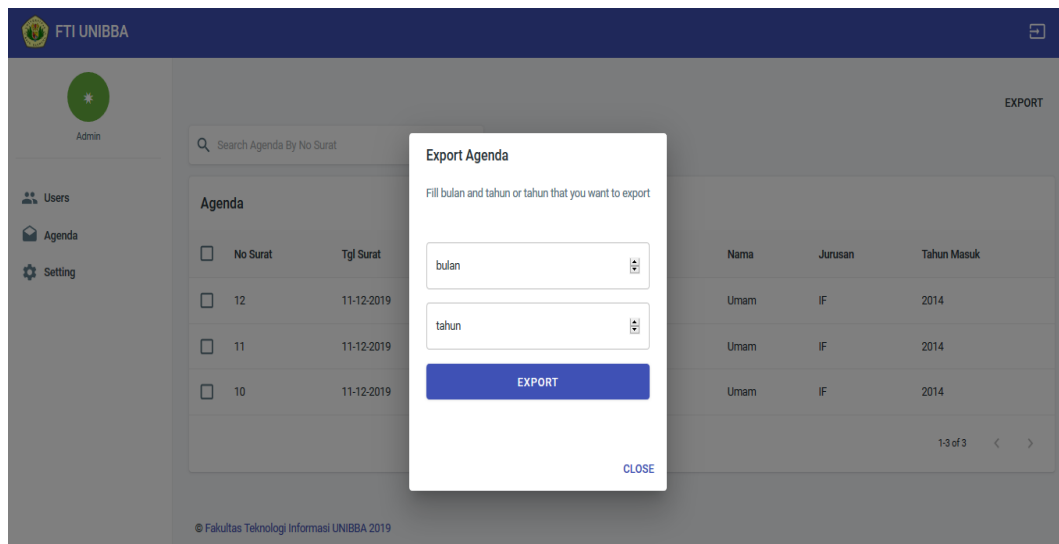
1-3 of 3

© Fakultas Teknologi Informasi UNIBBA 2019

Gambar 4.27 halaman agenda

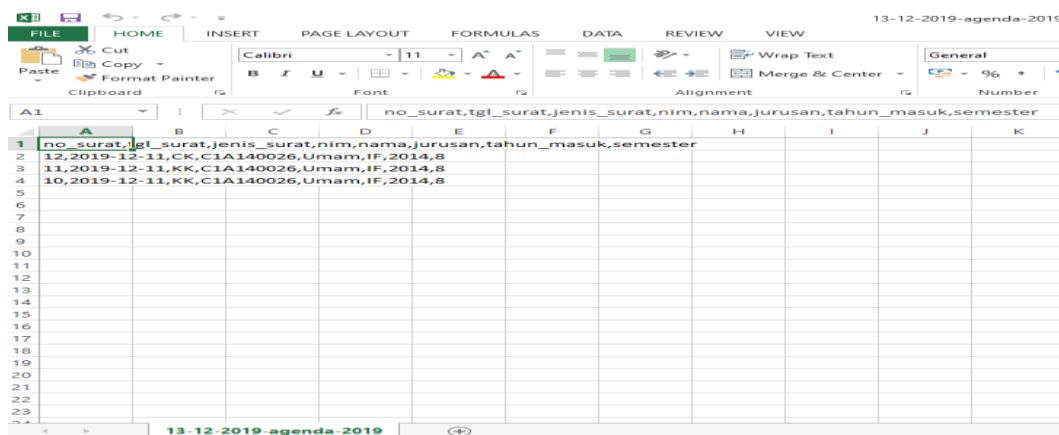
g. Halaman export agenda

Halaman export agenda surat dilakukan oleh admin untuk memudahkan Admin melakukan pengarsipan data siapa saja mahasiswa yang membuat surat keterangan mahasiswa dan cuti mahasiswa berdasarkan bulan dan tahun. berikut tampilan halaman pada aplikasinya :



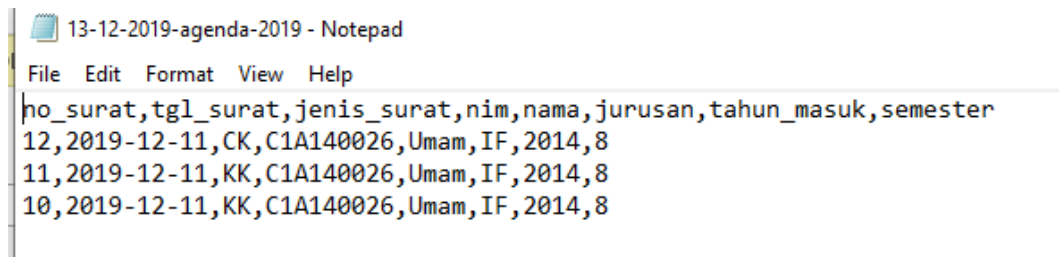
Gambar 4.28 halaman export data Agenda

h. Hasil export agenda berekstensi .csv dapat dibuka dengan ms.excel atau dengan notepad. Berikut hasil tampilannya :



Gambar 4.29 halaman hasil Export agenda menggunakan ms.excel

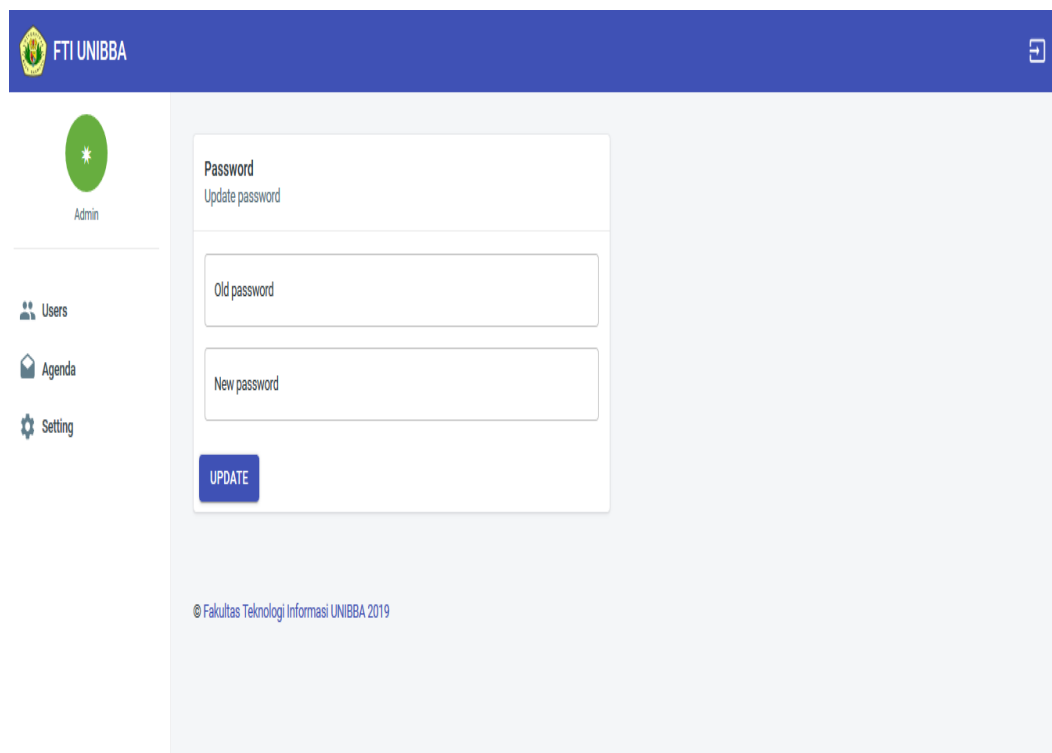
Berikut hasil tampilannya : Menggunakan notepad



Gambar 4.30 halaman hasil Export agenda menggunakan notpad

i. Halaman setting pada admin

Halaman ini Admin dapat merubah atau mengganti password berikut tampilan halaman pada aplikasinya :



Gambar 4.31 halaman setting pada admin

j. Halaman awal member setelah login


Halaman awal member setelah melakukan login langsung di suguhkan dengan pilihan memilih jenis surat manakah yang akan di buat. berikut tampilan halaman pada aplikasinya :



Gambar 4.32 halaman awal member


Setelah member berhasil melakukan login maka halaman pada tampilan pertama setelah login mahasiswa selaku member dapat memilih jenis surat yang tersedia di aplikasi, di dalam aplikasi tersedia surat keterangan mahasiswa dan surat cuti mahasiswa apabila mahasiswa memilih surat keterangan kuliah maka akan otomatis mengunduh file surat keterangan kuliah dalam format pdf, dan apabila mahasiswa memilih menu surat keterangan cuti kuliah maka akan otomatis mendownload surat cuti kuliah dalam format pdf kemudian mahasiswa dapat mencetaknya.

Berikut tampilan hasil surat keterangan mahasiswa setelah berhasil di buat kemudian di unduh dan ditampilkan dalam format pdf.

	<p align="center">UNIVERSITAS BALE BANDUNG FAKULTAS TEKNOLOGI INFORMASI PROGRAM STUDI : TEKNIK INFORMATIKA DAN SISTEM INFORMASI</p>
<hr/> <p align="center">JL. R.A.A WIRANTAKUSUMAH No.7 BALEENDAH KAB. BANDUNG 40258 Tlp.022-5946443,5949921,Fax.022-5940443</p> <hr/>	
<p align="center"><u>SURAT KETERANGAN KULIAH</u> Nomor : 09/KK/FTI-UNIBBA/XII/2019</p>	
<p>Yang bertanda tangan di bawah ini :</p>	
<p>Nama NIK Jabatan</p>	<p>: Yudi Herdiana S.T.,M.T : 04104808008 : Dekan</p>
<p>Dengan ini menyatakan sesungguhnya bahwa :</p>	
<p>Nama NIM Tahun Akademik Program Studi</p>	<p>: Umam.Luqman : C1A140026 : 2019-2020 : Informatika</p>
<p>Tercatat sebagai Mahasiswa pada Fakultas Teknologi Informasi Universitas Bale Bandung (UNIBBA).</p>	
<p>Demikian Surat Keterangan ini dibuat dengan sesungguhnya untuk dipergunakan sebagaimana mestinya, kepada pihak yang terkait mohon maklum adanya dan kami ucapkan terima kasih.</p>	
<p align="right">Baleendah, 10 Desember 2019 Dekan,</p>	
<p align="right">Yudi Herdiana S.T.,M.T NIK. 04104808008</p>	

Gambar 4.33 tampilan hasil surat keterangan mahasiswa

Berikut tampilan hasil surat cuti mahasiswa setelah berhasil di buat kemudian di unduh dan ditampilkan dalam format pdf.

	<p align="center">UNIVERSITAS BALE BANDUNG FAKULTAS TEKNOLOGI INFORMASI PROGRAM STUDI : TEKNIK INFORMATIKA DAN SISTEM INFORMASI</p>
<hr/> <p align="center">Jl. R.A.A WIRANTAKUSUMAH No.7 BALEENDAH KAB. BANDUNG 40258 Tlp.022-5946443,5949921,Fax.022-5940443</p> <hr/>	
<p>Perihal : Permohonan Penghentian Studi Sementara (Cuti Akademik)</p>	
<p>Yth. Rektor Universitas Bale Bandung Baleendah.</p>	
<p>Yang bertanda tangan dibawah ini :</p>	
<p>Nama : Umam NIM : C1A140026 Semester : 8 (Delapan) SKS/IPK : Program Studi : Informatika Sudah/ Belum pernah cuti : Lama cuti yang lalu : Alamat/ No telp/ HP :</p>	
<p>Mengajukan permohonan penghentian studi sementara (cuti akademik)</p>	
<p>Pada Semester/ Tahun Akademik : Alasan Cuti Akademik :</p>	
<p>Atas perhatian saudara kami ucapkan terimakasih</p>	
<p>Mengetahui/ Menyetujui Dekan,</p>	<p align="right">Baleendah, 11 Desember 2019 Hormat Saya,</p>
<p>(Yudi Herdiana S.T.,M.T)</p>	<p align="right">(Umam)</p>
<p>Tembusan : 1. Biro Akademik 2. Dosen Wali</p>	

Gambar 4.34 tampilan hasil surat cuti mahasiswa

k. Halaman setting pada member

Halaman setting pada member dapat melihat data diri mahasiswa tersebut dan mahasiswa tersebut dapat merubah atau mengganti password.

berikut tampilan halaman pada aplikasinya :

The screenshot shows a web application interface for FTI UNIBBA. On the left, there is a sidebar with a user profile icon labeled 'U' and 'Umam User', and two menu items: 'Dashboard' and 'Setting'. The main content area is divided into two panels. The left panel, titled 'User' and 'User Details', contains several input fields with the following values: Nim (C1A140026), Nama (Umam), Username (Umam), Email (umamluqman@gmail.com), Tempat Lahir (Bandung), Tanggal Lahir (15-12-1993), and Jenis Kelamin (L). The right panel, titled 'Password' and 'Update password', contains two input fields labeled 'Old password' and 'New password', and a blue 'UPDATE' button at the bottom.

Gambar 4.35 halaman setting pada member.

Untuk mengganti password mahasiswa tersebut dapat mengisi di bagian password yang terdapat kolom yang disediakan meliputi :

- Form Old password untuk memasukan password yang lama
- Form new password untuk memasukan password yang baru kemudian
- Click tombol update untuk mengeksekusi dan menyimpan password akan otomatis di simpan di dalam database.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil dari analisis dan perancangan serta pembahasan materi pada pembangunan aplikasi surat keterangan mahasiswa berbasis web di Fakultas Teknologi Informasi khususnya di prodi Teknik Informatika, maka dapat diambil beberapa kesimpulan bahwa:

1. Sistem Aplikasi surat keterangan mahasiswa berbasis web ini dapat memberikan kemudahan bagi mahasiswa dalam pengurusan surat keterangan mahasiswa yang meliputi surat keterangan mahasiswa, dan surat cuti mahasiswa
2. karena berbasis web sehingga dapat memudahkan mahasiswa dalam membuat surat dan dapat menghemat waktu kerana tanpa perlu repot datang ke Universitas.
3. Aplikasi yang dibuat telah di desain tampilan yang mudah dimengerti sehingga aplikasi ini dapat memudahkan mahasiswa dalam mengoprasikannya.
4. Tampilan yang responsive sehingga memudahkan mahasiswa dalam mengaksesnya dengan smartphone atau komputer.
5. Hasil dari pembuatan aplikasi ini dalam tahap penguijiannya aplikasi ini berjalan dengan baik.

5.2 Saran

Berdasarkan hasil pengembangan aplikasi ini adalah sebagai berikut:

1. Aplikasi ini masih diperlukan pengembangan karena masih terdapat kekurangan dikarenakan masih dalam tahap pembuatan.
2. Perlunya penambahan fitur edit pada admin untuk mengganti nama dekan pada surat, dikarenakan fitur ini belum ada pada aplikasi.
3. Aplikasi ini diperlukan pengembangan lebih lanjut dikarenakan masih terdapat kekurangan yang harus disesuaikan dengan kebutuhan Fakultas Teknologi Informasi Universitas Bale Bandung.

DAFTAR PUSTAKA

- Nugroho, A. (2009). *Rekayasa Perangkat Lunak Menggunakan UML & Java*. Yogyakarta: Andi Offset.
- Rosa, A., & Shalahuddin, M. (2011). *Modul Pembelajaran Rekayasa Perangkat Lunak*. Bandung:Modula.
- Whittenn, J., & Bentley, L. (2007). *System Analysis dan Design Method*. McGraw Hill.
- Geliandri Rathella Dharmawan, *perancangan system pembuatan surat mahasiswa berbasisweb*. UIN Syarif Hidayatullah.
- Dony Erlangga. *PPL Sistem Informasi kebutuhan surat akademik mahasiswa di FTI dan Sains Khatolik Parahyangan*.
- Khamdy. *Pengembangan SI pelayanan surat keterangan mahasiswa berbasis web di Universitas Muhammadiyah Malang*.
- Raharjo, B., Heryanto, I., & RK, E. (2012). *Modul Pemrograman Web HTML,PHP & MYSQL*. Bandung: Modula.
- Sujatmiko, E. (2012). *Kamus Teknologi Informasi dan Komunikasi*. Surakarta: Aksarra Sinergi Media

LAMPIRAN

I. Database

Database User

Model.py

```
JENIS_KELAMIN_CHOICES = (  
    ('L', 'Laki-Laki'),  
    ('P', 'Perempuan')  
)
```

```
AGAMA_CHOICES = (  
    ('Islam', 'Islam'),  
    ('Kristen', 'Kristen'),  
    ('Budha', 'Budha'),  
    ('Hindu', 'Hindu'),  
    ('Konghucu', 'Konghucu')  
)
```

```
JURUSAN_CHOICE = (  
    ('IF', 'INFORMATIKA'),  
    ('SI', 'SISTEM INFORMASI')  
)
```

```
class CustomUser(AbstractUser):

    nim = models.TextField(max_length=10, primary_key=True, blank=True)

    nama = models.TextField(max_length=50)

    tempat_lahir = models.TextField(max_length=50)

    tanggal_lahir = models.DateField(null=True)

    jenis_kelamin = models.CharField(

        choices=JENIS_KELAMIN_CHOICES,

        max_length=1

    )

    agama = models.TextField(

        choices=AGAMA_CHOICES,

        max_length=10

    )

    jurusan = models.TextField(

        choices=JURUSAN_CHOICE,

        max_length=2,

        null=True

    )

    tahun_masuk = models.TextField(

        max_length=12,

        null=True

    )
```

```
semester = models.IntegerField(null=True)

is_staff = models.BooleanField(default=False)
```

Database Agenda

Models.py

```
JENIS_SURAT_CHOISES = (
    ('KK', 'Keterangan Kuliah'),
    ('CK', 'Cuti Kuliah')
)

class Agenda(models.Model):
    no_surat = models.AutoField(primary_key=True, )
    tgl_surat = models.DateField(auto_now=True)
    jenis_surat = models.CharField(choices=JENIS_SURAT_CHOISES,
max_length=2)
    user = models.ForeignKey(
        settings.AUTH_USER_MODEL,
        to_field='nim',
        related_name='surats',
        on_delete=models.CASCADE
    )
```

II. Page

Halaman awal admin

Index.tsx

```
function SignPage () {

  const [snackControl, setSnackControl] = useState({

    open: false,

    message: "",

    variant: 'success'

  })

  const handleClose = (_event?: SyntheticEvent, reason?: string) => {

    if (reason === 'clickaway') {

      return

    }

    setSnackControl({

      ...snackControl, open: false

    })

  }

  const onSubmit = async (data: { username: string, password: string }) => {

    const url = 'http://localhost:8000/api/auth/login'
```

```
try {

  const response = await fetch(url, {

    method: 'POST',

    headers: {

      'Content-Type': 'application/json'

    },

    body: JSON.stringify({username: data.username, password:
data.password}),

  })

  if (response.status === 200) {

    const { token, user } = await response.json()

    login({ token, user })

  } else {

    const data = await response.json()

    setSnackControl({

      open: true,

      message: data['non_field_errors'][0],

      variant: 'error'

    })

  }

} catch (error) {

  setSnackControl({
```

```
        open: true,

        message: error.message,

        variant: 'error'

    })

}

}

return (

    <Minimal>

        <SignIn onSubmit={onSubmit} />

        <CustomizedSnackbars

            open={snackControl.open}

            message={snackControl.message}

            variant={snackControl.variant as any}

            onClose={handleClose}

        />

    </Minimal>

)

}

export default SignInPage
```

Sign.tsx

```
import React from "react";

import { Grid, TextField, Typography } from "@material-ui/core";

import Button from "@material-ui/core/Button";

import useForm from "react-hook-form";

import { makeStyles, createStyles, Theme } from "@material-ui/core/styles";

const defaultValues = {

  username: "",

  password: ""

};

interface SignInProps {

  className?: string

  onSubmit: (data: typeof defaultValues) => Promise<void>

}

const useStyles = makeStyles((theme: Theme) =>

  createStyles({

    root: {

      backgroundColor: theme.palette.background.default,

      height: "100vh"

    },

  },
```



```
grid: {  
  height: "100%"  
},  
quoteContainer: {  
  [theme.breakpoints.down("md")]: {  
    display: "none"  
  }  
},  
quote: {  
  backgroundColor: theme.palette.background.paper,  
  height: "100%",  
  display: "flex",  
  justifyContent: "center",  
  alignItems: "center",  
  backgroundImage: "url(/bg.jpg)",  
  backgroundSize: "cover",  
  backgroundRepeat: "no-repeat",  
  backgroundPosition: "center"  
},  
quoteInner: {  
  textAlign: "center",  
  flexBasis: "600px",  
  padding: theme.spacing(2)
```

```
},  
  
quoteText: {  
  color: theme.palette.text.secondary,  
  fontWeight: 300  
},  
  
name: {  
  marginTop: theme.spacing(3),  
  color: theme.palette.text.secondary  
},  
  
bio: {  
  color: theme.palette.text.secondary  
},  
  
content: {  
  height: "100%",  
  display: "flex",  
  flexDirection: "column"  
},  
  
contentHeader: {  
  display: "flex",  
  alignItems: "center",  
  paddingTop: theme.spacing(5),  
  paddingBottom: theme.spacing(2),  
  paddingLeft: theme.spacing(2),
```

```
paddingRight: theme.spacing(2)

},

logoImage: {

  marginLeft: theme.spacing(4)

},

contentBody: {

  flexGrow: 1,

  display: "flex",

  alignItems: "center",

  [theme.breakpoints.down("md")]: {

    justifyContent: "center"

  }

},

form: {

  paddingLeft: 100,

  paddingRight: 100,

  paddingBottom: 125,

  flexBasis: 700,

  [theme.breakpoints.down("sm")]: {

    paddingLeft: theme.spacing(2),

    paddingRight: theme.spacing(2)

  }

},
```

```
title: {

  marginTop: theme.spacing(3),

  marginBottom: theme.spacing(3)

},

textField: {

  marginTop: theme.spacing(2)

},

signInButton: {

  margin: theme.spacing(2, 0)

}

})

);

const SignIn: React.SFC<SignInProps> = (props) => {

  const { onSubmit } = props

  const classes = useStyles({ })

  const { register, handleSubmit } = useForm({

    defaultValues: defaultValues

  });

  return (
```

```
<div className={classes.root}>
```

```
  <Grid className={classes.grid} container>
```

```
    <Grid className={classes.quoteContainer} item lg={5}>
```

```
      <div className={classes.quote}>
```

```
        <div className={classes.quoteInner}>
```

```
          <Typography className={classes.quoteText} variant="h5">
```

"Menjadi Fakultas Teknologi Informasi (FTI) yang unggul dibidang Teknik Informatika dan Sistem Informasi, yang mampu mencetak lulusan dengan keahlian profesional bidang komputer, berperan dalam perkembangan ilmu pengetahuan dan teknologi informasi dan menghasilkan solusi aplikatif bagi dunia usaha/industri yang mampu bersaing dalam tataran global pada tahun 2028"

```
        </Typography>
```

```
      </div>
```

```
    <Typography className={classes.name} variant="body2">
```

Visi FTI Unibba

```
    </Typography>
```

```
  </div>
```

```
</div>
```

```
</div>
```

```
</Grid>
```

```
<Grid className={classes.content} item lg={7} xs={12}>
```

```
  <div className={classes.content}>
```

```
    <div className={classes.contentBody}>
```

```
<form className={classes.form}
onSubmit={handleSubmit(onSubmit)}>

  <Typography className={classes.title} variant="h2">

    Sign in

  </Typography>

  <TextField

    label="Username"

    name="username"

    fullWidth

    type="text"

    variant="outlined"

    inputRef={register({

      required: "Required"

    })}

  />

  <TextField

    label="password"

    type="password"

    name="password"

    margin="normal"

    variant="outlined"

    fullWidth

    inputRef={register({
```

```
        required: "Required"

        }}}

    />

    <Button

        className={ classes.signInButton }

        color="primary"

        fullWidth

        size="large"

        type="submit"

        variant="contained"

    >

        Sign in now

    </Button>

</form>

</div>

</div>

</Grid>

</Grid>

</div>

);

};

export default SignIn;
```

Halaman admin

Users.tsx

```
export interface UserData {  
  
  nim: string,  
  
  username: string,  
  
  nama: string,  
  
  email: string,  
  
  tempat_lahir: string,  
  
  tanggal_lahir: string,  
  
  jenis_kelamin: string,  
  
  agama: string,  
  
  jurusan: string,  
  
  tahun_masuk: number,  
  
  semester: number  
  
}  
  
export interface UserPageProps {  
  
  users: Response  
  
  token: string  
  
  data: any  
  
}
```



```
export interface NotifControl {

  open: boolean,

  message: string,

  variant: 'success' | 'warning' | 'error' | 'info'

}

const initialState = {

  notif: {

    open: false,

    message: "",

    variant: 'success' as NotifControl['variant']

  },

  selectedUsers: [],

  userWhoFind: "",

  page: 0

}

export interface ContextJoin {

  gstate: typeof initialState

  dispatch?: React.Dispatch<Action>

}

export type Action =
```

```
| { type: 'setSelectedUsers', users: Array<string> }
```

```
| { type: 'setPage', page: number }
```

```
| { type: 'setNotif', notif: NotifControl }
```

```
| { type: 'setUserWhoFind', value: string }
```

```
export const UsersContext = createContext<ContextJoin |  
undefined>(undefined)
```

```
const usersReducer = (state: typeof initialState, action: Action) => {  
  switch (action.type) {  
    case 'setSelectedUsers':  
      return { ...state, selectedUsers: action.users }  
    case 'setPage':  
      return { ...state, page: action.page }  
    case 'setNotif':  
      return { ...state, notif: action.notif }  
    case 'setUserWhoFind':  
      return { ...state, userWhoFind: action.value }  
    default:  
      return state  
  }  
}
```

```

const UsersPage: NextPage<UserPageProps> = ({ data }) => {

  const [state, dispatch] = useReducer(usersReducer, initialState)

  const { open, message, variant } = state.notif


  const handleClose = (_event?: SyntheticEvent, reason?: string) => {

    if (reason === 'clickaway') {

      return

    }

    dispatch({ type: 'setNotif', notif: { open: false, message: "", variant:
state.notif.variant} })

  }


  return (

    <UsersContext.Provider

      value={{

        gstate: state,

        dispatch: dispatch

      }}>

      <MainLayout

        name={data.nama}

        isStaf={data.is_staff}

      >

```

```
<UserList />

<CustomizedSnackbars

  open={ open}

  message={ message}

  variant={ variant}

  onClose={ handleClose}

/>

</MainLayout>

</UsersContext.Provider>

)

}

UsersPage.getInitialProps = async ctx => {

  const { token } = nextCookie(ctx)

  if (!token) {

    return redirectOnError(ctx)

  }

  const apiUrl = `http://localhost:8000/api/auth/user`

  try {

    const response = await fetch(apiUrl, {
```

```
    credentials: 'include',

    headers: {

        Authorization: 'token ' + token

    }

})

if (response.ok) {

    const data = await response.json()

    if (data.is_staff) {

        return { data: data, token: token }

    } else {

        return redirectOnError(ctx)

    }

} else {

    return redirectOnError(ctx)

}

} catch (error) {

    return redirectOnError(ctx)

}

}
```

```
export default withAuthSync(UsersPage)
```

Main.tsx

```
const useStyles = makeStyles((theme: Theme) => ({

  root: {

    paddingTop: 56,

    height: '100%',

    [theme.breakpoints.up('sm')]: {

      paddingTop: 64

    }

  },

  shiftContent: {

    paddingLeft: 240

  },

  content: {

    height: '100%'

  }

}))

export interface MainProps {

  name: string

  isStaf: boolean

}
```

```
const Main: React.SFC<MainProps> = props => {

  const { children, name, isStaf } = props


  const classes = useStyles({ })

  const theme = useTheme()

  const isDesktop = useMediaQuery(theme.breakpoints.up('lg'), {
    defaultMatches: true
  })


  const [openSidebar, setOpenSidebar] = useState(false)


  const handleSidebarOpen = () => {
    setOpenSidebar(true)
  }


  const handleSidebarClose = () => {
    setOpenSidebar(false)
  }


  const shouldOpenSidebar = isDesktop ? true : openSidebar


  return (
```

```
<div

  className={

    clsx({

      [classes.root]: true,

      [classes.shiftContent]: isDesktop

    })

  }

>

  <Topbar onSidebarOpen={handleSidebarOpen} />

  <Sidebar

    name={name}

    isStaf={isStaf}

    onClose={handleSidebarClose}

    open={shouldOpenSidebar}

    variant={isDesktop ? 'persistent' : 'temporary'}

  />

  <main className={classes.content}>

    {children}

    <Footer />

  </main>

</div>

)

}
```



```
export default Main
```

UserList.tsx

```
const useStyles = makeStyles((theme: Theme) => ({  
  
  root: {  
  
    padding: theme.spacing(3)  
  
  },  
  
  content: {  
  
    marginTop: theme.spacing(2)  
  
  }  
}))
```

```
const UserList: React.SFC = () => {  
  
  const classes = useStyles({ })  
  
  return (  
  
    <div className={classes.root}>  
  
      <UsersToolbar />  
  
      <div className={classes.content}>  
  
        <UsersTable />  
  
      </div>  
  
    </div>  
  )  
}
```

```
)  
  
}  
  
export default UserList
```

Auth.tsx

```
export const login = ({ token, user }) => {  
  
  cookie.set('token', token, { expires: 1 })  
  
  if (user.is_staff) {  
    Router.push('/users')  
  } else {  
    Router.push('/user-dashboard')  
  }  
}  
  
export const auth = ctx => {  
  
  const { token } = nextCookie(ctx)  
  
  if (!token) {  
    if (typeof window === 'undefined') {  
      ctx.res.writeHead(302, { location: '/' })  
      ctx.res.end()  
    }  
  }  
}
```

```
    } else {  
  
      Router.push('/')  
  
    }  
  
  }  
  
  return token  
}  
  
export const logout = async () => {  
  
  const url = 'http://localhost:8000/api/auth/logout'  
  
  const token = cookie.get('token')  
  
  try {  
  
    const response = await fetch(url, {  
  
      method: 'POST',  
  
      headers: {  
  
        Authorization: 'token ' + token  
  
      },  
  
    })  
  
    if (response.ok) {  
  
      cookie.remove('token')  
  
      window.localStorage.setItem('logout', Date.now().toString())  
  
    }  
  
  }  
  
}
```

```
Router.push('/')

} else {

  Router.push('/')

}

} catch (error) {

  Router.push('/')

}

cookie.remove('token')

window.localStorage.setItem('logout', Date.now().toString())

Router.push('/')

}

export const withAuthSync = WrappedComponent => {

  const Wrapper = props => {

    const syncLogout = event => {

      if (event.key === 'logout') {

        console.log('logged out from storage!')

        Router.push('/')

      }

    }

  }

  useEffect(() => {
```

```
window.addEventListener('storage', syncLogout)
```

```
return () => {
```

```
  window.removeEventListener('storage', syncLogout)
```

```
  window.localStorage.removeItem('logout')
```

```
}
```

```
}, [])
```

```
return <WrappedComponent {...props} />
```

```
}
```

```
Wrapper.getInitialProps = async ctx => {
```

```
  const token = auth(ctx)
```

```
  const componentProps = WrappedComponent.getInitialProps &&
```

```
  (await WrappedComponent.getInitialProps(ctx))
```

```
  return { ...componentProps, token }
```

```
}
```

```
return Wrapper
```

```
}
```

Export user

Api.py

```
@api_view(['POST'])
@permission_classes([permissions.IsAdminUser])
def streaming_users_csv_view(request):
    """A view that streams a large CSV file."""
    serialize = ExportCsvSerializer(data=request.data)
    serialize.is_valid(raise_exception=True)
    users = CustomUser.objects.filter(
        Q(is_staff=False) &
        Q(tahun_masuk=serialize.data['tahun_masuk'])
    ).order_by('nim')

    if len(users):
        response = StreamingHttpResponse(
            streaming_content=(iter_items(users, Echo())),
            content_type="text/csv"
        )
        response['Content-Disposition'] = 'attachment;
filename="somefilename.csv"'
        return response
    else:
        return Response(status=status.HTTP_400_BAD_REQUEST)
```

Add user

Add-user.tsx

```
const AddUser: NextPage<{ token: string, data: any }> = (props) => {

  const { token, data } = props

  const [snackControl, setSnackControl] = useState({
    open: false,
    message: "",
    variant: 'success',
    disabled: false
  })

  const onSubmit = async (data: AccountType, setError: any) => {

    setSnackControl({
      ...snackControl, disabled: true
    })

    try {

      const url = 'http://localhost:8000/api/users/'

      const response = await fetch(

        url,

        {
```

```
method: 'POST',

credentials: 'include',

headers: {

  'Content-Type': 'application/json',

  Authorization: 'token ' + token

},

body: JSON.stringify({...data})

}

)

if (response.ok) {

  setSnackControl({

    open: true,

    disabled: false,

    message: `success add user`,

    variant: 'success'

  })

  setTimeout(() => Router.push('/users'), 2000)

} else {

  const data = await response.json()

  setError(extractMessage(data))

  setSnackControl({
```



```
        ...snackControl,

        open: false

    })

}

} catch (error) {

    setSnackControl({

        open: true,

        disabled: false,

        message: error.message,

        variant: 'error'

    })

}

}

const handleClose = (_event?: SyntheticEvent, reason?: string) => {

    if (reason === 'clickaway') {

        return

    }

    setSnackControl({

        ...snackControl, open: false

    })

}
```

```

return (

  <MainLayout

    name={data.nama}

    isStaf={data.is_staff}

  >

    <Account onSubmit={onSubmit} disabled={snackControl.disabled}/>

    <CustomizedSnackbars

      open={snackControl.open}

      message={snackControl.message}

      variant={snackControl.variant as any}

      onClose={handleClose}

    />

  </MainLayout>

)

}

AddUser.getInitialProps = async ctx => {

  const { token } = nextCookie(ctx)

  if (!token) {

    return redirectOnError(ctx)

  }

```

```
const apiUrl = `http://localhost:8000/api/auth/user`
```

```
try {
```

```
  const response = await fetch(apiUrl, {
```

```
    credentials: 'include',
```

```
    headers: {
```

```
      Authorization: 'token ' + token
```

```
    }
```

```
  })
```

```
  if (response.ok) {
```

```
    const data = await response.json()
```

```
    if (data.is_staff) {
```

```
      return { token: token, data: data }
```

```
    } else {
```

```
      return redirectOnError(ctx)
```

```
    }
```

```
  } else {
```

```
    return redirectOnError(ctx)
```

```
  }
```

```
    } catch (error) {  
  
      return redirectOnError(ctx)  
  
    }  
  
  }  
  
}  
  
export default withAuthSync(AddUser)
```

Main.tsx

```
const useStyles = makeStyles((theme: Theme) => ({  
  
  root: {  
  
    paddingTop: 56,  
  
    height: '100%',  
  
    [theme.breakpoints.up('sm')]: {  
  
      paddingTop: 64  
  
    }  
  
  },  
  
  shiftContent: {  
  
    paddingLeft: 240  
  
  },  
  
  content: {  
  
    height: '100%'  
  
  }  
  
}))
```

```
export interface MainProps {  
  name: string  
  isStaf: boolean  
}  
  
const Main: React.SFC<MainProps> = props => {  
  const { children, name, isStaf } = props  
  
  const classes = useStyles({ })  
  
  const theme = useTheme()  
  
  const isDesktop = useMediaQuery(theme.breakpoints.up('lg'), {  
    defaultMatches: true  
  })  
  
  const [openSidebar, setOpenSidebar] = useState(false)  
  
  const handleSidebarOpen = () => {  
    setOpenSidebar(true)  
  }  
  
  const handleSidebarClose = () => {  
    setOpenSidebar(false)  
  }  
}
```

```
}

const shouldOpenSidebar = isDesktop ? true : openSidebar

return (
  <div
    className={
      clsx({
        [classes.root]: true,
        [classes.shiftContent]: isDesktop
      })
    }
  >
    <Topbar onSidebarOpen={handleSidebarOpen} />
    <Sidebar
      name={name}
      isStaf={isStaf}
      onClose={handleSidebarClose}
      open={shouldOpenSidebar}
      variant={isDesktop ? 'persistent' : 'temporary'}
    />
    <main className={classes.content}>
      {children}
    </main>
  </div>
)
```

```
    <Footer />

  </main>

</div>

)

}

export default Main
```

Auth.tsx

```
export const login = ({ token, user }) => {

  cookie.set('token', token, { expires: 1 })

  if (user.is_staff) {

    Router.push('/users')

  } else {

    Router.push('/user-dashboard')

  }

}

export const auth = ctx => {

  const { token } = nextCookie(ctx)

  if (!token) {
```

```
if (typeof window === 'undefined') {

  ctx.res.writeHead(302, { location: '/'})

  ctx.res.end()

} else {

  Router.push('/')

}

}

return token

}

export const logout = async () => {

  const url = 'http://localhost:8000/api/auth/logout'

  const token = cookie.get('token')

  try {

    const response = await fetch(url, {

      method: 'POST',

      headers: {

        Authorization: 'token ' + token

      },

    })

  }
```



```
if (response.ok) {  
  cookie.remove('token')  
  window.localStorage.setItem('logout', Date.now().toString())  
  Router.push('/')  
} else {  
  Router.push('/')  
}  
} catch (error) {  
  Router.push('/')  
}  
  
cookie.remove('token')  
window.localStorage.setItem('logout', Date.now().toString())  
Router.push('/')  
}  
  
export const withAuthSync = WrappedComponent => {  
  const Wrapper = props => {  
    const syncLogout = event => {  
      if (event.key === 'logout') {  
        console.log('logged out from storage!')  
        Router.push('/')  
      }  
    }  
  }  
}
```

```

}

useEffect(() => {

  window.addEventListener('storage', syncLogout)


  return () => {

    window.removeEventListener('storage', syncLogout)

    window.localStorage.removeItem('logout')

  }

}, [])

```

```

return <WrappedComponent {...props} />
}

```

```

Wrapper.getInitialProps = async ctx => {

  const token = auth(ctx)


  const componentProps = WrappedComponent.getInitialProps &&
    (await WrappedComponent.getInitialProps(ctx))


  return { ...componentProps, token }

}

```

```
return Wrapper  
  
}
```

Edit User

Edit-user.tsx

```
const EditUser = ({token, users, nim }) => {  
  
  const [snackControl, setSnackControl] = useState({  
    open: false,  
    message: "",  
    variant: 'success',  
    disabled: false  
  })  
  
  const handleClose = (_event?: SyntheticEvent, reason?: string) => {  
    if (reason === 'clickaway') {  
      return  
    }  
    setSnackControl({  
      ...snackControl, open: false  
    })  
  }  
  
  const onSubmit = async (data: AccountType, setError) => {  
    setSnackControl({  
      ...snackControl, disabled: true  
    })  
    try {  
      const url = `http://localhost:8000/api/users/${nim}`  
      const response = await fetch(  

```

```

url,
{
  method: 'PUT',
  credentials: 'include',
  headers: {
    'Content-Type': 'application/json',
    Authorization: 'token ' + token
  },
  body: JSON.stringify({...data})
}
)

if (response.ok) {
  setSnackControl({
    open: true,
    disabled: false,
    message: `success edit user nim ${nim}`,
    variant: 'success'
  })
  setTimeout(() => Router.push('/users'), 2000)

} else {
  const data = await response.json()
  setError(extractMessage(data))
  setSnackControl({
    ...snackControl,
    open: false
  })
}
} catch (error) {
  setSnackControl({
    open: true,

```

```

        disabled: false,
        message: error.message,
        variant: 'error'
      ))
    }
  }
}

```

```

return (
  <MainLayout
    name={users.nama}
    isStaf={users.is_staff}
  >
    <Account
      onSubmit={onSubmit}
      defaultValues={users}
      disabled={snackControl.disabled}
    />
    <CustomizedSnackbars
      open={snackControl.open}
      message={snackControl.message}
      variant={snackControl.variant as any}
      onClose={handleClose}
    />
  </MainLayout>
)
}

```

```

EditUser.getInitialProps = async ctx => {
  const { token } = nextCookie(ctx)
  const { nim } = ctx.query

  if (!token) {

```

```
    return redirectOnError(ctx)
  }

  const apiUrl = `http://localhost:8000/api/users/${nim}`

  try {
    const response = await fetch(apiUrl, {
      credentials: 'include',
      headers: {
        Authorization: 'token ' + token
      }
    })

    if (response.ok) {
      const data = await response.json()
      return { token: token, users: data.results[0], nim: nim }
    } else {
      return redirectOnError(ctx)
    }
  } catch (error) {
    return redirectOnError(ctx)
  }
}
```

```
export default withAuthSync(EditUser)
```

Agenda

Agenda.tsx

```
export interface AgendaData {
  'no_surat': number,
  'tgl_surat': string,
  'jenis_surat': string,
  'user': {
    'nim': string,
    'nama': string,
    'jurusan': string,
    'tahun_masuk': number,
    'semester': number
  }
}

export interface AgendaPageProps {
  agenda: Response
  token: string
  data: any
}

export interface NotifControl {
  open: boolean,
```

```
message: string,

variant: 'success' | 'warning' | 'error' | 'info'

}

const initialState = {

  notif: {

    open: false,

    message: "",

    variant: 'success' as NotifControl['variant']

  },

  findNoSurat: "",

  selectedAgendas: [],

  page: 1

}

export type Action =

  | { type: 'setSelectedAgendas', agendas: Array<string> }

  | { type: 'setfindNoSurat', value: string }

  | { type: 'setPage', page: number }

  | { type: 'setNotif', notif: NotifControl }

export interface ContextJoin {

  gstate: typeof initialState

}
```



```

dispatch?: React.Dispatch<Action>

}

export const AgendaContext = createContext<ContextJoin | undefined>(undefined)

const agendaReducer = (state: typeof initialState, action: Action) => {
  switch (action.type) {
    case 'setSelectedAgendas':
      return { ...state, selectedAgendas: action.agendas }

    case 'setfindNoSurat':
      return { ...state, findNoSurat: action.value }

    case 'setNotif':
      return { ...state, notif: action.notif }

    default:
      return state
  }
}

const AgendaPage: NextPage<AgendaPageProps> = ({ data }) => {
  const [state, dispatch] = useReducer(agendaReducer, initialState)

  const { open, message, variant } = state.notif

  const handleClose = (_event?: SyntheticEvent, reason?: string) => {

```

```
if (reason === 'clickaway') {  
  return  
}  
  
dispatch({ type: 'setNotif', notif: { open: false, message: "", variant:  
state.notif.variant}})  
  
}  
  
return (  
  <AgendaContext.Provider  
    value={{  
      gstate: state,  
      dispatch: dispatch  
    }}>  
    <MainLayout  
      name={data.nama}  
      isStaf={data.is_staff}  
    >  
      <AgendaList />  
      <CustomizedSnackbars  
        open={open}  
        message={message}  
        variant={variant}  
        onClose={handleClose}
```

```
    />

    </MainLayout>

    </AgendaContext.Provider>

  )
}

AgendaPage.getInitialProps = async ctx => {

  const { token } = nextCookie(ctx)

  if (!token) {

    return redirectOnError(ctx)

  }

  const apiUrl = `http://localhost:8000/api/auth/user`

  try {

    const response = await fetch(apiUrl, {

      credentials: 'include',

      headers: {

        Authorization: 'token ' + token

      }

    })

  })
```

```
if (response.ok) {  
  const data = await response.json()  
  
  if (data.is_staff) {  
    return { token: token, data: data }  
  } else {  
    return redirectOnError(ctx)  
  }  
  
} else {  
  return redirectOnError(ctx)  
}  
  
} catch (error) {  
  return redirectOnError(ctx)  
}  
}  
  
export default withAuthSync(AgendaPage)
```

Main.tsx

```
const useStyles = makeStyles((theme: Theme) => ({  
  root: {  
    paddingTop: 56,  
    height: '100%',  
    [theme.breakpoints.up('sm')]: {
```

```
paddingTop: 64
  }
},
shiftContent: {
  paddingLeft: 240
},
content: {
  height: '100%'
}
}))

export interface MainProps {
  name: string
  isStaf: boolean
}

const Main: React.SFC<MainProps> = props => {
  const { children, name, isStaf } = props

  const classes = useStyles({})
  const theme = useTheme()
  const isDesktop = useMediaQuery(theme.breakpoints.up('lg'), {
    defaultMatches: true
  })

  const [openSidebar, setOpenSidebar] = useState(false)

  const handleSidebarOpen = () => {
    setOpenSidebar(true)
  }

  const handleSidebarClose = () => {
    setOpenSidebar(false)
  }
}
```

```

}

const shouldOpenSidebar = isDesktop ? true : openSidebar

return (
  <div
    className={
      clsx({
        [classes.root]: true,
        [classes.shiftContent]: isDesktop
      })
    }
  >
    <Topbar onSidebarOpen={handleSidebarOpen} />
    <Sidebar
      name={name}
      isStaf={isStaf}
      onClose={handleSidebarClose}
      open={shouldOpenSidebar}
      variant={isDesktop ? 'persistent' : 'temporary'}
    />
    <main className={classes.content}>
      {children}
      <Footer />
    </main>
  </div>
)
}

export default Main

```

Export Agenda

Api.py

```
@api_view(['POST'])
@permission_classes([permissions.IsAdminUser])
def streaming_agenda_csv_view(request):
    """A view that streams a large CSV file."""
    serialize = ExportCsvSerializer(data=request.data)
    serialize.is_valid(raise_exception=True)
    data = serialize.data

    if (data['bulan']):
        agenda = Agenda.objects.filter(
            Q(tgl_surat__month=data['bulan']) & Q(
                tgl_surat__year=data['tahun'])
        ).order_by('-no_surat')
    else:
        agenda = Agenda.objects.filter(
            Q(tgl_surat__year=data['tahun'])
        ).order_by('-no_surat')

    if len(agenda):
        response = StreamingHttpResponse(
            streaming_content=(iter_items(agenda, Echo())),
            content_type="text/csv"
        )
        response['Content-Disposition'] = 'attachment;
filename="somefilename.csv"'
        return response
    else:
        return Response(status=status.HTTP_400_BAD_REQUEST)
```

```

@api_view(['POST'])
def create_surat_kk_request(request):
    serializer = AgendaDetailSerializer(
        data=request.data,
        context={'request': request}
    )
    serializer.is_valid(raise_exception=True)
    data = serializer.save()

    filename = "kk"
    response = HttpResponse(content_type='application/pdf')
    response['Content-Disposition'] = 'attachment; filename="' + \
        filename + '.pdf"'
    create_kk(response, data)
    return response

```

```

@api_view(['POST'])
def create_surat_ck_request(request):

    serializer = AgendaDetailSerializer(
        data=request.data,
        context={'request': request}
    )
    serializer.is_valid(raise_exception=True)
    data = serializer.save()
    filename = "ck"
    response = HttpResponse(content_type='application/pdf')
    response['Content-Disposition'] = 'attachment; filename="' + \
        filename + '.pdf"'
    create_ck(response, data)

    return response

```


Setting.tsx

```
export interface SettingProps {  
  
  data: {  
  
    "nim": string,  
  
    "nama": string,  
  
    "username": string,  
  
    "email": string,  
  
    "tempat_lahir": string,  
  
    "tanggal_lahir": string,  
  
    "jenis_kelamin": string,  
  
    "agama": string,  
  
    "jurusan": string,  
  
    "tahun_masuk": string,  
  
    "semester": string,  
  
    "is_staff": boolean  
  
  }  
}  
  
export interface NotifControl {  
  
  open: boolean,  
  
  message: string,  
  
  variant: 'success' | 'warning' | 'error' | 'info'  
  
}
```

```
const initialState = {  
  notif: {  
    open: false,  
    message: "",  
    variant: 'success' as NotifControl['variant']  
  }  
}  
  
export type Action =  
  | { type: 'setNotif', notif: NotifControl}  
  
export interface ContextProps {  
  data: SettingProps['data'],  
  gstate: typeof initialState  
  dispatch?: React.Dispatch<Action>  
}  
  
export const SettingContext = createContext<ContextProps | undefined>(undefined)  
  
const usersReducer = (state: typeof initialState, action: Action) => {  
  switch (action.type) {  
    case 'setNotif':
```

```

    return { ...state, notif: action.notif}

    default:

        return state
    }
}

const Setting: NextPage<SettingProps> = (props) => {

    const [state, dispatch] = useReducer(usersReducer, initialState)

    const [snackControl, setSnackControl] = useState({

        open: false,

        message: "",

        variant: 'success',

        disabled: false

    })

    const handleClose = (_event?: SyntheticEvent, reason?: string) => {

        if (reason === 'clickaway') {

            return

        }

        setSnackControl({

            ...snackControl, open: false

        })

    }
}

```

```
return (  
  <SettingContext.Provider  
    value={{  
      data: props.data,  
      gstate: state,  
      dispatch: dispatch  
    }}  
  >  
    <MainLayout  
      name={props.data.nama}  
      isStaf={props.data.is_staff}  
    >  
      <SettingView />  
      <CustomizedSnackbars  
        open={snackControl.open}  
        message={snackControl.message}  
        variant={snackControl.variant as any}  
        onClose={handleClose}  
      />  
    </MainLayout>  
  </SettingContext.Provider>  
)
```

```
}

Setting.getInitialProps = async ctx => {

  const { token } = nextCookie(ctx)

  if (!token) {

    return redirectOnError(ctx)

  }

  const apiUrl = `http://localhost:8000/api/auth/user`

  try {

    const response = await fetch(apiUrl, {

      credentials: 'include',

      headers: {

        Authorization: 'token ' + token

      }

    })

    if (response.ok) {

      const data = await response.json()

      return { token: token, data: data }

    } else {
```

```
    return redirectOnError(ctx)

  }

  } catch (error) {

    return redirectOnError(ctx)

  }

}

export default withAuthSync(Setting)
```

Main.tsx

```
const useStyles = makeStyles((theme: Theme) => ({

  root: {

    paddingTop: 56,

    height: '100%',

    [theme.breakpoints.up('sm')]: {

      paddingTop: 64

    }

  },

  shiftContent: {

    paddingLeft: 240

  },

  content: {

    height: '100%'

  }

})
```

```
}  
)))  
  
export interface MainProps {  
  name: string  
  isStaf: boolean  
}  
  
const Main: React.SFC<MainProps> = props => {  
  const { children, name, isStaf } = props  
  
  const classes = useStyles({ })  
  const theme = useTheme()  
  const isDesktop = useMediaQuery(theme.breakpoints.up('lg'), {  
    defaultMatches: true  
  })  
  
  const [openSidebar, setOpenSidebar] = useState(false)  
  
  const handleSidebarOpen = () => {  
    setOpenSidebar(true)  
  }  
}
```

```
const handleSidebarClose = () => {  
  setOpenSidebar(false)  
}  
  
const shouldOpenSidebar = isDesktop ? true : openSidebar  
  
return (  
  <div  
    className={  
      clsx({  
        [classes.root]: true,  
        [classes.shiftContent]: isDesktop  
      })  
    }  
  >  
    <Topbar onSidebarOpen={handleSidebarOpen} />  
    <Sidebar  
      name={name}  
      isStaf={isStaf}  
      onClose={handleSidebarClose}  
      open={shouldOpenSidebar}  
      variant={isDesktop ? 'persistent' : 'temporary'}  
    />  
  </div>  
)
```



```
<main className={classes.content}>

  {children}

  <Footer />

</main>

</div>

)

}

export default Main
```

Halaman awal user dan memilih jenis surat

User-dashboard

```
export interface UserDasboarProps {

  data: any

  token: string

}

export interface NotifControl {

  open: boolean,

  message: string,

  variant: 'success' | 'warning' | 'error' | 'info'

}
```

```
const initialState = {  
  
  notif: {  
  
    open: false,  
  
    message: "",  
  
    variant: 'success' as NotifControl['variant']  
  
  }  
}  
  
export type Action =  
  
  | { type: 'setNotif', notif: NotifControl }  
  
export interface ContextProps {  
  
  gstate: typeof initialState  
  
  dispatch?: React.Dispatch<Action>  
  
}  
  
export const UsersDashboardContext = createContext<ContextProps |  
undefined>(undefined)  
  
const usersReducer = (state: typeof initialState, action: Action) => {  
  
  switch (action.type) {  
  
    case 'setNotif':  
  
      return { ...state, notif: action.notif }
```

```

default:

  return state

}

}

const UserDashboard: NextPage<UserDasboarProps> = ({ data }) => {

  const [state, dispatch] = useReducer(usersReducer, initialState)

  const { open, message, variant } = state.notif

  const handleClose = (_event?: SyntheticEvent, reason?: string) => {

    if (reason === 'clickaway') {

      return

    }

    dispatch({ type: 'setNotif', notif: { open: false, message: "", variant:
state.notif.variant} })

  }

  return (

    <UsersDashboardContext.Provider

      value={{

        gstate: state,

        dispatch: dispatch

      }}

```

```

>

<MainLayout

  name={ data.nama }

  isStaf={ data.is_staff }

>

<CreateSurat />

<CustomizedSnackbars

  open={ open }

  message={ message }

  variant={ variant }

  onClose={ handleClose }

/>

</MainLayout>

</UsersDashboardContext.Provider>

)

}

UserDashboard.getInitialProps = async ctx => {

  const { token } = nextCookie(ctx)

  if (!token) {

    return redirectOnError(ctx)

  }

```

```
const apiUrl = `http://localhost:8000/api/auth/user`
```

```
try {
```

```
  const response = await fetch(apiUrl, {
```

```
    credentials: 'include',
```

```
    headers: {
```

```
      Authorization: 'token ' + token
```

```
    }
```

```
  })
```

```
  if (response.ok) {
```

```
    const data = await response.json()
```

```
    return { data: data, token: token }
```

```
  } else {
```

```
    return redirectOnError(ctx)
```

```
  }
```

```
  } catch (error) {
```

```
    return redirectOnError(ctx)
```

```
  }
```

```
}
```

```
export default withAuthSync(UserDashboard)
```

Setting User (Details User)

Setting.tsx

```
export interface SettingProps {
  data: {
    "nim": string,
    "nama": string,
    "username": string,
    "email": string,
    "tempat_lahir": string,
    "tanggal_lahir": string,
    "jenis_kelamin": string,
    "agama": string,
    "jurusan": string,
    "tahun_masuk": string,
    "semester": string,
    "is_staff": boolean
  }
}
```

```
export interface NotifControl {
  open: boolean,
```

```
message: string,

variant: 'success' | 'warning' | 'error' | 'info'

}

const initialState = {

  notif: {

    open: false,

    message: "",

    variant: 'success' as NotifControl['variant']

  }

}

export type Action =

  | { type: 'setNotif', notif: NotifControl}

export interface ContextProps {

  data: SettingProps['data'],

  gstate: typeof initialState

  dispatch?: React.Dispatch<Action>

}

export const SettingContext = createContext<ContextProps | undefined>(undefined)
```

```
const usersReducer = (state: typeof initialState, action: Action) => {  
  
  switch (action.type) {  
  
    case 'setNotif':  
  
      return { ...state, notif: action.notif}  
  
    default:  
  
      return state  
  
  }  
}
```

```
const Setting: NextPage<SettingProps> = (props) => {  
  
  const [state, dispatch] = useReducer(usersReducer, initialState)  
  
  const [snackControl, setSnackControl] = useState({  
  
    open: false,  
  
    message: "",  
  
    variant: 'success',  
  
    disabled: false  
  
  })  
  
  
  const handleClose = (_event?: SyntheticEvent, reason?: string) => {  
  
    if (reason === 'clickaway') {  
  
      return  
  
    }  
  
    setSnackControl({
```



```
        ...snackControl, open: false

    })
}

return (
  <SettingContext.Provider
    value={{
      data: props.data,
      gstate: state,
      dispatch: dispatch
    }}
  >
    <MainLayout
      name={props.data.nama}
      isStaf={props.data.is_staff}
    >
      <SettingView />
      <CustomizedSnackbars
        open={snackControl.open}
        message={snackControl.message}
        variant={snackControl.variant as any}
        onClose={handleClose}
      />
    </MainLayout>
  </SettingContext.Provider>
)
```

```
</MainLayout>

</SettingContext.Provider>

)
}

Setting.getInitialProps = async ctx => {

  const { token } = nextCookie(ctx)

  if (!token) {

    return redirectOnError(ctx)

  }

  const apiUrl = `http://localhost:8000/api/auth/user`

  try {

    const response = await fetch(apiUrl, {

      credentials: 'include',

      headers: {

        Authorization: 'token ' + token

      }

    })

    if (response.ok) {
```

```
    const data = await response.json()

    return { token: token, data: data }

  } else {

    return redirectOnError(ctx)

  }

} catch (error) {

  return redirectOnError(ctx)

}

}

export default withAuthSync(Setting)
```

AccountDetails.tsx

```
const defaultValues = {

  'nim': '',

  'nama': '',

  'username': '',

  'password': '',

  'email': '',

  'tempat_lahir': '',

  'tanggal_lahir': '',

  'jenis_kelamin': '',

  'agama': '',
```

```
'jurusan': "",

'tahun_masuk': "",

'semester': "",

}

export type AccountType = typeof defaultValues

export interface AccountDetailsProps {

  className?: string

  onSubmit: (data: AccountType, setError: any, nim?: string) => Promise<void>

  defaultValues?: AccountType

  value?: string

  disabled: boolean

}

const jurusanOptions = [

  {"value": "IF", "label": "INFORMATIKA"},

  {"value": "SI", "label": "SISTEM INFORMASI"}

]

const jenisKelaminOptions = [

  {"value": "L", "label": "Laki-Laki"},

  {"value": "P", "label": "Perempuan"}

]
```

```
]
```

```
const agamaOptions = [
```

```
  {"value": "Islam", "label": "Islam"},
```

```
  {"value": "Kristen", "label": "Kristen"},
```

```
  {"value": "Budha", "label": "Budha"},
```

```
  {"value": "Hindu", "label": "Hindu"},
```

```
  {"value": "Konghucu", "label": "Konghucu"},
```

```
]
```

```
const useStyles = makeStyles(() => ({
```

```
  root: {}
```

```
}))
```

```
const AccountDetails: React.SFC<AccountDetailsProps> = props => {
```

```
  const { disabled, onSubmit, defaultValues: dp, className, ...rest } = props
```

```
  const classes = useStyles({})
```

```
  const [selectedDate, setSelectedDate] = useState<any>(
```

```
    dp ? moment(dp.tanggal_lahir) : moment()
```

```
  )
```

```
const handleDateChange = (date: any) => {

  console.log('date from change', date)

  setSelectedDate(date)

}

const { register, handleSubmit, errors, setError } = useForm({

  defaultValues: dp ? dp : defaultValues

})

return (

  <Card

    {...rest}

    className={clsx(classes.root, className)}

  >

    <form

      autoComplete="off"

      onSubmit={

        handleSubmit(

          (data: any) => onSubmit(

            {...data, tanggal_lahir: selectedDate.format('YYYY-MM-DD')},

            setError

          )

        )

      }

    >
```

```
}  
  
noValidate  
  
>  
  
<CardHeader  
  
  title="Account"  
  
</>  
  
<Divider />  
  
<CardContent>  
  
  <Grid  
  
    container  
  
    spacing={3}  
  
    >  
  
      <Grid  
  
        item  
  
        md={6}  
  
        xs={12}  
  
        >  
  
          <TextField  
  
            fullWidth  
  
            label="nim"  
  
            name="nim"  
  
            disabled={dp ? true: false}  
  
            error={errors.nim !== undefined}
```

```
        helperText={errors.nim && errors.nim.message}

        inputRef={register({

            required: "Required"

        })}

        variant="outlined"

    />

</Grid>

<Grid

    item

    md={6}

    xs={12}

>

    <TextField

        fullWidth

        label="nama"

        name="nama"

        error={errors.nama !== undefined}

        helperText={errors.nama && errors.nama.message}

        inputRef={register({

            required: "Required"

        })}

        variant="outlined"

    />
```



```
</Grid>
```

```
<Grid
```

```
  item
```

```
    md={6}
```

```
    xs={12}
```

```
>
```

```
  <TextField
```

```
    fullWidth
```

```
    label="username"
```

```
    name="username"
```

```
    error={errors.username !== undefined}
```

```
    helperText={errors.username && errors.username.message}
```

```
    inputRef={register({
```

```
      required: "Required"
```

```
    })}
```

```
    variant="outlined"
```

```
</Grid>
```

```
<Grid
```

```
  item
```

```
    md={6}
```

```
    xs={12}
```

```
>
```

```
<TextField

  fullWidth

  label="password"

  name="password"

  type="password"

  disabled={dp ? true : false}

  error={errors.password !== undefined}

  helperText={errors.password && errors.password.message}

  inputRef={dp ? undefined : register({

    required: "Required"

  })}

  variant="outlined"

/>

</Grid>

<Grid

  item

  md={6}

  xs={12}

>

  <TextField

    fullWidth

    label="email"

    name="email"
```

```
        type="email"

        error={errors.email !== undefined}

        helperText={errors.email && errors.email.message}

        inputRef={register({

            required: "Required"

        })}

        variant="outlined"

    />

</Grid>

<Grid

    item

    md={6}

    xs={12}

>

    <TextField

        fullWidth

        label="tempat lahir"

        name="tempat_lahir"

        error={errors.tempat_lahir !== undefined}

        helperText={errors.tempat_lahir && errors.tempat_lahir.message}

        inputRef={register({

            required: "Required"

        })}
```

```
        variant="outlined"

    />

</Grid>

<Grid

    item

    md={6}

    xs={12}

>

    <MuiPickersUtilsProvider utils={MomentUtils}>

        <KeyboardDatePicker

            autoOk

            fullWidth

            variant="inline"

            name={ "tanggal_lahir" }

            inputVariant="outlined"

            label="tanggal_lahir"

            format="DD/MM/YYYY"

            value={ selectedDate }

            onChange={ handleDateChange }

        />

    </MuiPickersUtilsProvider>

</Grid>

<Grid
```

```
item

md={6}

xs={12}

>

<TextField

  fullWidth

  label="jenis kelamin"

  name="jenis_kelamin"

  error={errors.jenis_kelamin !== undefined}

  helperText={errors.jenis_kelamin && errors.jenis_kelamin.message}

  inputRef={register({

    required: "Required"

  })}

  select

  SelectProps={{ native: true }}

  variant="outlined"

>

{jenisKelaminOptions.map(option => (

  <option

    key={option.value}

    value={option.value}

    defaultValue=""

  >
```

```
        {option.label}

      </option>

    )))

  </TextField>

</Grid>

<Grid

  item

  md={6}

  xs={12}

  >

    <TextField

      fullWidth

      label="agama"

      name="agama"

      error={errors.agama !== undefined}

      helperText={errors.agama && errors.agama.message}

      inputRef={register({

        required: "Required"

      })}

      select

      SelectProps={{ native: true }}

      variant="outlined"

    >
```

```

{agamaOptions.map(option => (
  <option
    key={option.value}
    value={option.value}
    defaultValue=""
  >
    {option.label}
  </option>
))}
</TextField>
</Grid>
<Grid
  item
  md={6}
  xs={12}
>
  <TextField
    fullWidth
    label="jurusan"
    name="jurusan"
    error={errors.jurusan !== undefined}
    helperText={errors.jurusan && errors.jurusan.message}
    inputRef={register({

```

```
        required: "Required"

      }}}

    select

    SelectProps={{ native: true }}

    variant="outlined"

  >

    {jurusanOptions.map(option => (

      <option

        key={option.value}

        value={option.value}

        defaultValue=""

      >

        {option.label}

      </option>

    ))}

  </TextField>

</Grid>

<Grid

  item

  md={6}

  xs={12}

  >

  <TextField
```



```
        fullWidth

        label="tahun masuk"

        name="tahun_masuk"

        type="number"

        error={errors.tahun_masuk !== undefined}

        helperText={errors.tahun_masuk && errors.tahun_masuk.message}

        inputRef={register({

            required: "Required",

        })}

        variant="outlined"

    />

</Grid>

<Grid

    item

    md={6}

    xs={12}

>

    <TextField

        fullWidth

        label="semester"

        name="semester"

        type="number"

        error={errors.semester !== undefined}
```

```
        helperText={errors.semester && errors.semester.message}

        inputRef={register({
            required: "Required"
        })}

        select

        SelectProps={{ native: true }}

        variant="outlined"
    >

    {[1,2,3,4,5,6,7,8].map(option => (

        <option

            key={option}

            value={option}

            defaultValue=""

            >

                {option}

            </option>

        )]}

    </TextField>

</Grid>

</Grid>

</CardContent>

<Divider />

<CardActions>
```

```
<Button  
  color="primary"  
  variant="contained"  
  type="submit"  
  disabled={ disabled}  
  >  
    Save  
  </Button>  
</CardActions>  
</form>  
</Card>  
)  
}  
  
export default AccountDetails
```