# **Proposal**

**IMPLEMENTASI *MICROSERVICES ARCHITECTURE* UNTUK *SCHEDULE* *HANDLE* PADA APLIKASI *MULTIPLE SEND MESSAGE* MENGGUNAKAN ALGORITMA *PRIORITY SCHEDULING* (PS)**

**(Studi Kasus: Fakultas Teknik Universitas Halu Oleo)**

**Diajukan Untuk Memenuhi Satu Syarat Memproleh Gelar Sarjana Teknik**

****

**MARLINA**

**E1E1 17 016**

**JURUSAN TEKNIK INFORMATIKA**

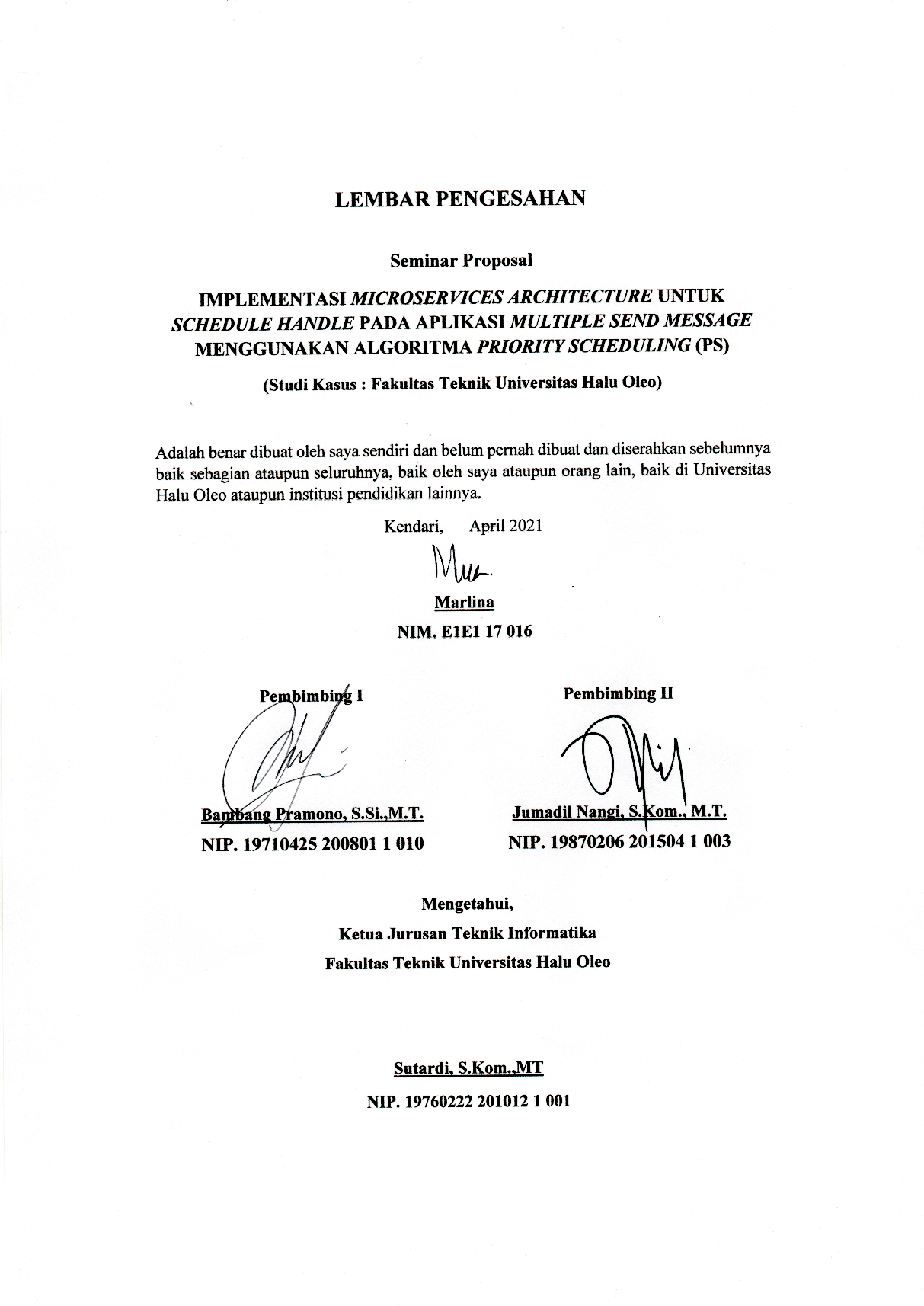
**FAKULTAS TEKNIK**

**UNIVERSITAS HALU OLEO**

**KENDARI**

**2021**

# **LEMBAR PENGESAHAN**



# 

# **DAFTAR ISI**

[HALAMAN PROPOSAL i](#_Toc68341756)

[LEMBAR PENGESAHAN ii](#_Toc68341757)

[DAFTAR ISI iii](#_Toc68341758)

[DAFTAR TABEL v](#_Toc68341759)

[DAFTAR GAMBAR vi](#_Toc68341760)

[BAB I PENDAHULUAN 1](#_Toc68341761)

[1.1 Latar Belakang 1](#_Toc68341762)

[1.2 Rumusan Masalah 2](#_Toc68341763)

[1.3 Batasan Masalah 3](#_Toc68341764)

[1.4 Tujuan Penelitian 3](#_Toc68341765)

[1.5 Manfaat Penelitian 3](#_Toc68341766)

[1.6 Sistematika Penulisan 4](#_Toc68341767)

[1.7 Tinjauan Pustaka 4](#_Toc68341768)

[BAB II LANDASAN TEORI 7](#_Toc68341769)

[2.1 Penjadwalan 7](#_Toc68341770)

[2.2 Arsitektur *Microservice* 7](#_Toc68341771)

[2.3 *Priority Scheduling* 8](#_Toc68341772)

[2.4 PHP (*Hypertext Preprocessor*) 11](#_Toc68341773)

[2.5 MySql 12](#_Toc68341774)

[2.6 Xampp 12](#_Toc68341775)

[2.7 *CodeIgniter* 13](#_Toc68341776)

[2.8 Json 13](#_Toc68341777)

[2.9 *Application programming interface* (API) 14](#_Toc68341778)

[2.10 *Database* 14](#_Toc68341779)

[2.11 *Flowchart* 15](#_Toc68341780)

[2.12 UML (*Unifield Modeling Language*) 16](#_Toc68341781)

[2.13 Diagram UML 16](#_Toc68341782)

[2.14 RUP 21](#_Toc68341783)

[BAB III METODE PENELITIAN 23](#_Toc68341784)

[3.1 Metode Pengumpulan Data 23](#_Toc68341785)

[3.2 Metode Pengembangan Sistem 23](#_Toc68341786)

[3.3 Waktu dan Tempat Penelitian 24](#_Toc68341787)

[3.3.1 Waktu 24](#_Toc68341788)

[3.3.2 Tempat 24](#_Toc68341789)

[3.4 Analisis Sistem 25](#_Toc68341790)

[3.4.1 Analisis Kebutuhan Sistem 25](#_Toc68341791)

[3.4.2 Kebutuhan Fungsional 25](#_Toc68341792)

[3.4.3 Kebutuhan Nonfungsional 26](#_Toc68341793)

[3.5 Analisis Perancangan Sistem 27](#_Toc68341794)

[3.5.1 Analisis Perancangan *Flowchart* 27](#_Toc68341795)

[3.5.2 Analisis Perancangan UML 28](#_Toc68341796)

[3.6 Perancangan Antarmuka (*Interface*) 45](#_Toc68341797)

[3.6.1 Menu Pilih *Login* 46](#_Toc68341798)

[3.6.2 Menu Registrasi Dosen 46](#_Toc68341799)

[3.6.3 Menu Registrasi Mahasiswa 47](#_Toc68341800)

[3.6.4 Menu Registrasi Staf 48](#_Toc68341801)

[3.6.5 Menu *Login* 48](#_Toc68341802)

[3.6.6 Menu Beranda Superadmin 49](#_Toc68341803)

[3.6.7 Menu Beranda Admin 50](#_Toc68341804)

[3.6.8 Menu Data Admin 50](#_Toc68341805)

[3.6.9 Menu Tambah Data Admin 51](#_Toc68341806)

[3.6.10 Menu Laporan 51](#_Toc68341807)

[3.6.11 Menu Data Pendaftar 52](#_Toc68341808)

[3.6.12 Menu Informasi Penjadwalan 52](#_Toc68341809)

[3.6.13 Menu *List* Penjadwalan 54](#_Toc68341810)

[DAFTAR PUSTAKA 54](#_Toc68341811)

# **DAFTAR TABEL**

[Tabel 2. 1 Contoh](#_Toc67756324) *[Priority Scheduling](#_Toc67756324)* [8](#_Toc67756324)

[Tabel 2. 2 *Flowchart* 15](#_Toc67756325)

[Tabel 2. 3 Simbol *Use Case Diagram* 17](#_Toc67756326)

[Tabel 2. 4 Simbol *Activity Diagram* 18](#_Toc67756327)

[Tabel 2. 5 Simbol *Class Diagram* 19](#_Toc67756328)

[Tabel 2. 6 Simbol *Multiplicity* 20](#_Toc67756329)

[Tabel 2. 7 Simbol *Sequence Diagram* 21](#_Toc67756330)

[Tabel 3. 1 Gannt Chart Waktu Penelitian 24](#_Toc68126896)

[Tabel 3. 2 Spesifikasi Perangkat Keras 26](#_Toc68126897)

[Tabel 3. 3 Spesifikasi Perangkat Lunak 26](#_Toc68126898)

[Tabel 3. 4 Deskripsi *Case*  *Login* (Superadmin dan Admin) 29](#_Toc68126899)

[Tabel 3. 5 Deskripsi *Case*  Mengelolah Data Admin 30](#_Toc68126900)

[Tabel 3. 6 Deskripsi *Case*  Melihat Laporan 30](#_Toc68126901)

[Tabel 3. 7 Deskripsi *Case*  Mengelolah Admin 31](#_Toc68126902)

[Tabel 3. 8 Deskripsi *Case*  Mengelolah Data Informasi Penjadwalan 32](#_Toc68126903)

[Tabel 3. 9 Deskripsi *Case*  *List* Penjadwalan 32](#_Toc68126904)

[Tabel 3. 10 Deskripsi *Case*  Registrasi 33](#_Toc68126905)

[Tabel 3. 11 Deskripi *Case*  Menerima Informasi 33](#_Toc68126906)

[Tabel 3. 12 Deskripsi *Case*  data pendaftar 34](#_Toc68126907)

[Tabel 3. 13 Deskripsi *Case*  *Priority Scheduling* 34](#_Toc68126908)

[Tabel 3. 14 Deskripsi *Case*  *Logout* 35](#_Toc68126909)

# **DAFTAR GAMBAR**

[Gambar 3. 1 *Flowchart* 28](#_Toc67756424)

[Gambar 3. 2 *Use* *Case*  29](#_Toc67756425)

[Gambar 3. 3 *Activity Diagram Login* 36](#_Toc67756426)

[Gambar 3. 4 *Activity Diagram* Mengelolah Data Admin 36](#_Toc67756427)

[Gambar 3. 5 *Activity Diagram User* 37](#_Toc67756428)

[Gambar 3. 6 Data Pendaftar Dosen 37](#_Toc67756429)

[Gambar 3. 7 Data Pendaftar Staf Jurusan 38](#_Toc67756430)

[Gambar 3. 8 Data Pendaftar Mahasiswa 38](#_Toc67756431)

[Gambar 3. 9 *Activity Diagram* Informasi Penjadwalan 39](#_Toc67756432)

[Gambar 3. 10 *Activity Diagram List* Penjadwalan 39](#_Toc67756433)

[Gambar 3. 11 *Activity Diagram* Menerima Informasi 40](#_Toc67756434)

[Gambar 3. 12 *Activity Diagram* Laporan 41](#_Toc67756435)

[Gambar 3. 13 *Sequence Diagram Login* 42](#_Toc67756436)

[Gambar 3. 14 *Sequence Diagram* Mengelolah Data Admin 42](#_Toc67756437)

[Gambar 3. 15 *Sequence Diagram User* 43](#_Toc67756438)

[Gambar 3. 16 *Sequence Diagram* Data Pendaftar 43](#_Toc67756439)

[Gambar 3. 17 *Sequence Diagram* Informasi Penjadwalan 44](#_Toc67756440)

[Gambar 3. 18 *Sequence Diagram List* Penjadwalan 44](#_Toc67756441)

[Gambar 3. 19 *Sequence Diagram* Laporan 45](#_Toc67756442)

[Gambar 3. 20 *Class Diagram* 46](#_Toc67756443)

[Gambar 3. 21 Menu Pilih *Login* 47](#_Toc67756444)

[Gambar 3. 22 Menu Registrasi Dosen 47](#_Toc67756445)

[Gambar 3. 23 Menu Registrasi Mahasiswa 48](#_Toc67756446)

[Gambar 3. 24 Menu Registrasi Staf 48](#_Toc67756447)

[Gambar 3. 25 Menu *Login* 49](#_Toc67756448)

[Gambar 3. 26 Menu Beranda Superadmin 49](#_Toc67756449)

[Gambar 3. 27 Menu Beranda Admin 50](#_Toc67756450)

[Gambar 3. 28 Menu Data Admin 50](#_Toc67756451)

[Gambar 3. 29 Menu Tambah Data Admin 51](#_Toc67756452)

[Gambar 3. 30 Menu Laporan 51](#_Toc67756453)

[Gambar 3. 31 Menu Data Pendaftar 52](#_Toc67756454)

[Gambar 3. 32 Menu Informasi Penjadwalan (Dosen) 52](#_Toc67756455)

[Gambar 3. 33 Menu Informasi Penjadwalan (Mahasiswa) 53](#_Toc67756456)

[Gambar 3. 34 Menu Informasi Penjadwalan (Staf) 53](#_Toc67756457)

[Gambar 3. 35 Menu *List* Penjadwalan 54](#_Toc67756458)

# **BAB I PENDAHULUAN**

1. **Latar Belakang**

Pada umumnya setiap informasi mengenai jadwal kegiatan fakultas mempunyai rencana dan jadwal pelaksanaan tertentu, kapan pelaksanaan agenda tersebut harus dimulai, kapan harus diselesaikan, bagaimana agenda tersebut akan dikerjakan, dan bagaimana penyediaan sumber dayanya. Saat ini penyampaian informasi dihadapkan pada pengembangan yang signifikan, dimana setiap informasi salah satunya informasi fakultas seketika perlu ada perubahan-perubahan dalam memberikan sebuah informasi penjadwalan kegiatan fakultas. Salah satu penyebab keterlambatan informasi fakultas tersebut dikarenakan tidak semua informasi sampai ke pihak yang bersangkutan dilihat dari cara penyebaran dan pengendalian informasi.

Oleh karena itu dengan kemajuan teknologi informasi dan komunikasi telah membuka tahap untuk penyebaran informasi fakultas untuk memperoleh informasi secara otonom seperti memproses mendapatkan, meyusun, menyimpan data dalam berbagai cara untuk menghasilkan informasi yang berkualitas dengan perkembangan teknologi yang semakin pesat dalam segala bidang mendorong staf fakultas untuk mengatasi hal dalam menyebarkan sebuah informasi yang ada di fakultas. Sehingga informasi dengan mudah tersampaikan ke pihak yang bersangkutan.

Wujud nyata dari teknologi dapat dilihat adanya penerapan sistem komputerasasi bahwa sistem tersebut diharapkan mampu menyelesaikan suatu pekerja secara efektif dan efisien yang digunakan untuk keperluan fakultas. Pemanfaat teknologi tentunya disesuaikan dengan kebutuhan desain sistem tersebut. Teknologi tidak juga dipandang hanya sebagai pelengkap, melainkan sebagai salah satu penentu atas terlaksananya sasaran dan strategi. Dari berbagai macam algoritma penjadwalan yang ada, seperti *First come first server* (FCFS) penjadwalan dengan ketentuan-ketentuan sederhana, yaitu proses- proses diberi

jatah waktu memproses diurutkan berdasarkan waktu kedatangan proses ke sistem dan dijalankan sampai selesai (Fratiwi et al., 2020) dengan algoritma ini penjadwalan informasi lebih lama untuk menunggu satu proses yang dijalankan. Adapun algoritma *Round robin* semua sumber antrian dianggap sama sehingga diberi waktu yang disebut quantum, jika quantum habis maka proses berlanjut ke antrian berikutnya (Wijaya & Gunawan 2018). Namun dari kedua algoritma tersebut tidak dapat melakukan prioritas informasi berdasarkan tingkat kepentingan. Oleh karena itu dibutuhkan sistem yang mendukung informasi agar dapat tersampaikan ke pihak yang bersangkutan dengan menggunakan algoritma *priority scheduling* merupakan algoritma penjadwalan yang mendahulukan proses yang memiliki prioritas tertinggi, tiap-tiap proses dilengkapi dengan nomor prioritas (nilai integer terkecil biasanya merupakan prioristas terbesar). Pengerjaan mendahulukan proses yang memiliki prioritas terbesar. *Priority scheduling* juga dapat dijalankan secara *preemptive* maupun non-preemptive. Pada *preemptive*, jika ada suatu proses yang baru datang memiliki prioritas yang lebih tinggi daripada proses yang sedang dijalankan, maka proses yang sedang berjalan tersebut dihentikan, lalu dialihkan untuk proses yang baru datang tersebut. Sementara itu, pada *non-preemptive*, proses yang baru datang tidak dapat mengganggu proses yang sedang berjalan, tetapi hanya diletakkan di depan *queue* (Astri Dwi & Juniarahmatunisa 2019)

Berdasarkan latar belakang tersebut penelitian yang akan dilakukan ini berjudul “Implementasi *Microservices Architecture* Untuk­ *Schedule* *Handle* Pada Aplikasi *Multiple Send Message* Menggunakan Algoritma *Priority Scheduling* (Ps)”

1. **Rumusan Masalah**

Berdasarkan latar belakang masalah yang telah diuraikan diatas, maka penulis dapat merumuskan masalah penelitian yaitu:

1. Bagaimana membangun sistem informasi *multiple send message* yang dapat membantu staf dalam penyebaran sebuah informasi jadwal kegiatan di Fakultas Teknik?
2. Bagaimana menerapkan algoritma *Priority Scheduling* pada sistem informasi penjadwalan untuk staf Fakultas Teknik?
3. **Batasan Masalah**

Adapun batasan masalah yang akan dibahas dalam sistem ini sebagai berikut:

1. Penelitian ini hanya dilakukan dalam lingkup Fakultas Teknik Universitas Halu Oleo
2. Tidak dapat mengirim pesan secara bersamaan.
3. Sistem yang akan dibangun merupakan aplikasi berbasis *website*
4. Sistem ini menggunakan algoritma *Priority Scheduling* untuk menentukan penjadwalan *multiple send message* sebuah informasi jadwal kegiatan Fakultas yang akan disampaikan.
5. **Tujuan Penelitian**

Adapun tujuan dari penelitian ini sebagai berikut:

1. Dapat membangun sistem informasi *multiple send message* yang dapat membantu staf dalam penyebaran sebuah informasi di Fakultas Teknik.
2. Menganalisis penerapan metode *Priority Scheduling* untuk staf dalam menyampaikan sebuah informasi *multiple send message* lebih efektif dan efesien.
3. **Manfaat Penelitian**

Adapun manfaat dari penelitian ini sebagai berikut:

1. Diharapkan sistem informasi *multiple send message* tersebut dapat mempermudah staf Fakultas Teknik dalam menyampaikan sebuah informasi jadwal kegiatan di Fakultas Teknik.
2. Diharapkan algoritma *Priority Scheduling* dapat diterapkan untuk penjadwalan staf untuk menyampaikan sebuah informasi *multiple send message.*
3. **Sistematika Penulisan**

Adapun sistematika penulisan pada proposal penelitian ini sebagai berikut:

**BAB I PENDAHULUAN**

Bab ini berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, sistematika penulisan dan tinjauan pustaka.

**BAB II LANDASAN TEORI**

Bagian ini berisi tentang teori-teori penunjang yang berhubungan dengan penelitian berupa teori dari metode yang akan digunakan dalam penelitian implementasi *microservices architecture* untuk *schedule* *handle* pada aplikasi *multiple send message* menggunakan algoritma *priority scheduling* (ps)**.**

**BAB III METODE PENELITIAN**

Bab ini berisi tahapan tentang metode pengumpulan data, metode pengembangan sistem, analisis kebutuhan fungsional maupun non fungsional sistem dan analisis perancangan sistem serta rancangan antarmuka.

1. **Tinjauan Pustaka**

Penelitian ini dibuat berdasarkan penelitian yang pernah dilakukan oleh Simarmata & Harahap pada tahun 2019 dengan judul penelitian “Sistem penjadwalan iklan menggunakan metode *Priority Schedulling* pada PT. kidung indah selaras suara (Radio kiss FM) untuk efektivitas dan efisiensi produksi siaran” mengenai penggunaan metode *prority schedulling* dalam sistem penjadwalan iklan pada radio kiss FM untuk produksi siaran adalah: a) sistem penjadwalan iklan terintegrasi merupakan sistem yang membantu pengelolaan siaran iklan pada radio sehingga dalam pencatatan order iklan penjadwalan iklan tersebut. b) dalam sistem ini, penyiar dapat menayangkan iklan yang sudah dijadikan tanpa harus mencari *file* iklan didalam komputer, sehingga iklan dapat ditayangkan sesuai jadwalnya. c) sistem dapat mencetak laporan bukti siar yang telah ditayangkan, sehingga akan meningkatkan efektivas dan efesiensi dalam membuat laporan bukti siar.

Penelitian yang dilakukan oleh Fitri dkk. pada tahun 2016 dengan judul penenelitian “penerapan *Constraint Satisfaction Problem* pada metode *Priority Scheduling* untuk penjadwalan khutbah jum’at para mubaligh di IKMI pekanbaru”. Berdasarkan hasil percobaan, penyusunan jadwal dengan teknik yang ditawarkan sudah berhasil mendapatkan akurasi yang baik (93.7%) dengan waktu rata-rata penyusunan adalah 3,538 detik. Meskipun begitu, percobaan dilakukan pada data yang relatif kecil, yaitu 50 orang mubaligh dan 50 buah masjid. Kesalahan penyusunan jadwal yang sering terjadi adalah mubaligh tua ditempatkan pada masjid yang jauh dari tempat tinggalnya. Oleh sebab itu, untuk penelitian selanjutnya perlu diuji bagaimana performa teknik yang ditawarkan pada data yang lebih besar, Selain itu, perlu juga dilakukan modifikasi algoritma agar tidak terjadi kesalahan pada penyusunan mubaligh tua. Misalnya dengan mengganti LCV dengan *most constraint variable* (MCV), sehingga penyusunan jadwal mubaligh tua yang memiliki banyak constraint dapat didahulukan.

Penelitian yang dilakukan oleh Jefri1 & Kristina, pada tahun 2015 dengan judul penelitian “Perancangan Aplikasi pemesanan makanan berbasis android menggunakan metode *priority scheduling* untuk rumah makan” Aplikasi ini membantu bagian dapur dalam memperbarui stok bahan makanan yang tersedia sehingga mempermudah memantau jumlah stok masih tersedia, dan juga membantu pelayan dalam melihat menu makanan yang tersedia, sehingga apabila menu yang dipesan tidak ada maka pelayan bisa langsung memberitahukan kepada pelanggan, bagian dapur juga terbantu karena bagian dapur dapat melihat daftar pesanan yang harus diproses, pembuatan antrian pesanan tersebut akan mendapat nilai prioritas yang semakin tinggi. Semakin tinggi nilai prioritas maka pesanan akan semakin cepat mendapat giliran untuk diproses.

Penelitian ini dilakukan oleh Lokapitasari Belluano dkk. pada tahun 2020 dengan judul penenelitian “Sistem informasi program kreativitas mahasiswa berbasis *web service* dan *microservice*” mengenai Pengembangan sistem informasi menggunakan *web service* dengan model arsitektur *microservice* adalah suatu bentuk aplikasi sistem informasi internal Universitas berbasis *microservice* untuk membantu institusi dalam mengelola pelaporan kegiatan program kreativitas mahasiswa internal berupa aplikasi xSIA PKM yang mampu memelihara infrastruktur sistem aplikasi terhadap perubahan alur proses bisnis yang seketika dapat terjadi dalam lingkup institusi sesuai kebutuhan, kebijakan dan kepentingan internal, tanpa mengganggu jalannya sistem akademik lainnya. Keandalan sistem *microservice* dapat bekerja maksimal dalam interval waktu 0.00/detik dengan penggunaan sumber daya sistem yang kecil karena komunikasi data diakomodir sesuai kebutuhan unit-unit. Aplikasi berjalan menggunakan *Advanced* *Message Queuing protocol* yang mengatur komunikasi data secara *asyncronous* (waktu tunggu) antar *microservice*. Sedangkan transaksi data antar unit *microservice* menggunakan *Rest Protocol* yang bersifat *synchronous* dimana data diminta sesuai kebutuhan unit.

**BAB II   
LANDASAN TEORI**

1. **Penjadwalan**

Penjadwalan (*scheduling*), didefinisikan sebagai proses pengalokasian sumber untuk memilih sekumpulan tugas dalam jangka waktu tertentu. Definisi ini dapat dijabarkan dalam arti yang berbeda, yaitu:

1. Penjadwalan merupakan sebuah fungsi pengambilan keputusan dalam menentukan jadwal yang paling tepat.
2. Penjadwalan merupakan teori yang berisi sekumpulan prinsip, model, teknik, dan konklusi logis dalam proses pengambilan keputusan.

Sistem penjadwalan adalah informasi berupa data dalam proses mengatur rencana kegiatan dengan pembagian waktu pelaksanaan (Simarmata & Harahap, 2019).

1. **Arsitektur *Microservice***

Arsitektur *microservice* merupakan alternatif arsitektur yang lebih terukur dan lebih *fleksible*. Pada arsitektur *microservice*, sistem informasi dirancang untuk terdistribusi dan menyediakan layanan secara lebih fokus dan spesifik. Permasalahan besar akan dipecah menjadi beberapa solusi kecil yang disusun dalam satu *service*, dimana setiap *service* memiliki tanggung jawabnya sendiri. Dengan pendekatan ini, suatu sistem informasi akan terdiri dari beberapa *service* yang dapat dikelola dan didistribusikan secara independent, hal ini akan lebih memudahkan sistem untuk beradaptasi terhadap perubahan kebutuhan.(Munawar & Hodijah, 2018)

Terdapat beberapa kelebihan dari arsitektur *microservice* (Mufrizal & Indarti, 2019)yaitu:

1. Arsitektur *microservice* membuat kode aplikasi lebih sedikit dan bersifat independen sehingga dapat dilakukan pengujian aplikasi secara independen.
2. Memudahkan pemeiliharaan perangkat lunak.
3. Dapat melakukan proses distribusi perangkat lunak secara independen
4. Mudahnya dalam melakukan *scalabilit*
5. Developer dapat bebas dalam mengembangkan aplikasi dengan berbagai bahasa pemrograman dan *framework.*
6. ***Priority Scheduling***

*Priority Scheduling* merupakan algoritma penjadwalan yang mendahulukan proses yang memiliki prioritas tertinggi. Setiap proses memiliki prioritasnya masing-masing. Prioritas tersebut dapat ditentukan melalui beberapa karakteristik antara lain:

1. Time Limit
2. Memory Requirement
3. Akses File
4. Perbandingan antara I/O Burst dengan CPU Burst
5. Tingkat Kepentingan Proses

*Priority scheduling* juga dapat dijalankan secara *preemptive* maupun *nonpreemptive*. Pada *preemptive* jika ada suatu proses yang baru datang memiliki prioritas yang lebih tinggi daripada proses yang sedang dijalankan, maka proses yang sedang berjalan tersebut dihentikan, lalu CPU dialihkan untuk proses yang baru datang tersebut. Sementara itu, pada *non-preemptive*, proses yang baru datang tidak dapat menganggu proses yang sedang berjalan, tetapi hanya diletakkan di dalam *queue* (Simarmata & Harahap, 2019). Berikut contoh analisis Algoritma *Priority Scheduling*:

1. Diketahui 4 proses dengan urutan sebagai berikut

**Tabel 2. 1 Contoh Priority Scheduling**

|  |  |  |  |
| --- | --- | --- | --- |
| Proses | *Burst Time* | *Priority* | *Arrival Time* |
| P1 | 8 | 2 | 0 |
| P2 | 5 | 1 | 3 |
| P3 | 2 | 2 | 5 |
| P4 | 6 | 1 | 6 |

1. Kemudian membuat *Gantt Chart* seperti gambar berikut

0

p1(2)

3

p2(1)

p1(2)

5

P1

P2

p2(1)

p1(2)

p3(2)

6

p4(1)

p2(1)

p1(2)

p3(2)

8

p4(1)

p1(2)

p3(2)

14

P3(2)

P1(2)

P4

16

P3

21

P1

P1(2)

Mengitung Average dari AWT dengan rumus sebagai berikut:

Rumus:

CT (*Compitition Time*) = Proses detik selesai

TAT (*Turn Around Time*) = *Compitition time* - *Arrival time* atau CT – AT

WT (*Waiting Time*) = *Turn around time* - *Burst time* atau TAT – BT

*Average Turn Around Time* =

AWT (*Average Waiting Time*) =

1. Penyelesaian TAT (*Turn Around Time*)

P1 = (CT - AT)

= (21 - 0)

= 21

P2 = (CT - AT)

= (8 -3)

= 5

P3 = (CT - AT)

= (16 -5)

= 11

P4 = (CT - AT)

= (14 -6)

= 8

1. Menghitung rumus *Average Turn Around*

Dengan menghitung rumus *Average Turn Around Time* dari:

*Average Turn Around Time* =

=

= 11,25 ms

1. Menghitung *Waiting Time*

Berikut penyelesaian WT (*Waiting Time*)

P1 = (TAT - BT)

= (21 - 8)

= 13

P2 = (TAT - BT)

= (5 - 5 )

= 0

P3 = (TAT - BT)

= (11 - 2)

= 9

P4 = (TAT - BT)

= (8 - 6)

= 2

1. Rumus *Average Waiting Time*

Dengan menghitung rumus AWT (*Average Waiting Time*) dari:

AWT (*Average Waiting Time*) =

AWT (*Average Waiting Time*) =

= 6 ms

1. **PHP (*Hypertext Preprocessor*)**

*Hypertext* *Preprocessor* atau disingkat PHP adalah suatu bahasa pemrograman yang digunakan untuk membuat web dinamis, walau bisa juga digunakan untuk membuat program lain. Tentunya bahasa pemrograman PHP berbeda dengan HTML, pada PHP *script*/kode yang dibuat tidak dapat ditampilkan pada halaman muka *website* begitu saja, tapi harus diproses terlebih dahulu oleh *web* *server* lalu ditampilkan dalam bentuk halaman *website* di *browser*, *script* PHP juga dapat disisipkan pada HTML dan *script* PHP selalu diawasi selalu diawali dengan <php dan diakhiri dengan ?>.

Manajemen *database* yang biasanya digunakan untuk pemrograman PHP misalnya seperti *MySQL*, tapi ada juga yang menggunakan *Oracle*, *Microsoft* *Acces* dan lain-lain. PHP disebut juga sebagai bahasa pemrograman *script* *server* side karena PHP diproses pada komputer *server*. Berikut beberapa kelebihan bahasa pemrograman PHP:

1. PHP adalah merupakan bahasa yang bersifat *open source* dan gratis**.**
2. Proses [belajar pemrograman PHP](https://www.jagoanhosting.com/blog/belajar-pemrograman-web/) yang lebih singkat dibandingkan dengan bahasa lain seperti JSP, ASP dll.
3. Karena bahasa pemrograman terpopuler, maka komunitas banyak dokumentasi lebih besar.
4. Sebagian besar *server web hosting* mendukung PHP secara *default* tidak seperti bahasa lain seperti ASP yang membutuhkan IIS. Ini menjadikan PHP pilihan yang hemat biaya.
5. PHP diperbarui secara teratur untuk mengikuti tren teknologi terbaru.
6. PHP adalah bahasa *scripting* sisi server yang berarti hanya perlu menginstalnya di server dan komputer klien yang meminta sumber daya dari server tidak perlu menginstal PHP hanya peramban web saja sudah cukup.
7. PHP bekerja sama denganMySQL hal ini tidak berarti dapat menggunakan PHP dengan sistem manajemen basis data lainnya. Anda masih dapat menggunakan PHP dengan *Postgres*, Peramal, *MS SQL Server* ODBC dll.
8. PHP adalah bahasa lintas *platform* yang berarti dapat pada aplikasi di sejumlah sistem operasi yang berbeda seperti *windows*, *Linux*, *Mac* OS dll.
9. **MySql**

Menurut Josi (2017) MySQL merupakan sebuah *Relational Database* *Management* *System* (RDBMS) yang bersifat open *source*. Perangkat lunak database pada umumnya disandingkan dengan bahasa pemrograman server web seperti PHP atau JSP. MySQL (*My Structured Query Language*) adalah sebuah program pembuat dan pengelola *database* atau yang sering disebut dengan DBMS (*Database Management System*), sifat DBMS ini ialah *open source*. Selain itu MySQL juga merupakan program pengakses *database* yang bersifat jaringan, sehingga bisa digunakan untuk aplikasi Multi *User*.

1. **Xampp**

XAMPP adalah sebuah software yang berfungsi untuk menjalankan website berbasis PHP dan menggunakan pengolah data MYSQL di komputer lokal”. XAMPP berperan sebagai server web pada komputer lokal. XAMPP juga dapat disebut sebuah C panel server virtual, yang dapat membantu melakukan *preview* sehingga dapat dimodifikasi website tanpa harus *online* atau terakses dengan internet(appkey, 2020). Sebagai informasi kata XAMPP merupakan singkatan dari:

1. X: berarti program ini dapat dijalankan diberbagai *platform*, misalnya *Windows*, *Linux*, *mac OS*, dan *Solaris*.
2. A: *Apache*, merupakan aplikasi *web server*, dan bertugas untuk menghasilkan halaman web yang benar kepada *user* berdasarkan kode PHP yang dituliskan oleh pembuat halaman web. Jika diperlukan juga berdasarkan kode PHP yang dituliskan, maka dapat saja suatu *database* diakses terlebih dahulu (misalnya dalam MySQL) untuk mendukung halaman web yang dihasilkan.
3. M: MySQL, merupakan aplikasi database server. Pengembangnya disebut *Structured Query Language* (SQL). SQL merupakan bahasa terstruktur yang digunakan untuk mengolah *database* beserta isinya. Pengguna dapat memanfaatkan MySQL untuk menambahkan, mengubah dan menghapus data yang berada dalam *database*.
4. P: PHP, bahasa pemrograman lainnya yang serupa, dan lain sebagainya
5. P: *Pearl*, bahasa pemrograman untuk segala keperluan dikembangkan pertama oleh Larry Wall di mesin Unix.
6. ***CodeIgniter***

*Codeigniter* adalah sebuah *framework* untuk web yang dibuat dalam format PHP. Format yang dibuat ini selanjutnya dapat digunakan untuk membuat sistem aplikasi web yang kompleks. *Codeigniter* dapat mempercepat proses pembuatan web, karena semua class dan modul yang dibutuhkan sudah ada dan programmer hanya tinggal menggunakannya kembali pada aplikasi web yang akan dibuat. (Prabowo, 2015). Terdapat 3 jenis komponen yang membangun suatu MVC *pattern* dalam suatu aplikasi yaitu (Istiono et al., 2016) :

1. *View,* merupakan bagian yang menangani *presentation logic.* Pada suatu aplikasi web bagian ini biasanya berupa file *template* HTML, yang diatur oleh *controller. View* berfungsi untuk menerima dan mempresentasikan data kepada *user*. Bagian ini tidak memiliki akses langsung terhadap bagian model.
2. Model, biasanya berhubungan langsung dengan *database* untuk memanipulasi data (*insert, update, delete, search*), menangani validasi dari bagian *controller*, namun tidak dapat berhubungan langsung dengan bagian *view*.
3. *Controller,* merupakan bagian mengatur hubungan antara bagian model dan bagian *view*, *controller* berfungsi untuk menerima *request* dan data dari *user* kemudian menentukan apa yang akan diproses oleh aplikasi.
4. **Json**

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman *JavaScript*, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran data. JSON terbuat dari dua struktur:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur *(struct*), kamus (*dictionary*), tabel *hash* (*hash table*), daftar berkunci (*keyed list*), atau *associative array*.
2. Daftar nilai terurutkan *(an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (array), vektor (*vector*), daftar (list), atau urutan (*sequence*).(Herdiana, 2014)
3. ***Application programming interface* (API)**

*Application programming interface* (API) merupakan suatu dokumentasi yang terdiri dari *interface,* fungsi, kelas, struktur dan sebagainya untuk membangun sebuah perangkat lunak. Dengan adanya API ini, maka memudahkan programmer untuk “membongkar” suatu *software*, kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. API dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya yang memungkinkan programmer menggunakan sistem *function*. Proses ini dikelola melalui sistem operasi. Keunggulan dari API ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya dapat saling berhubungan dan berinteraksi.(Ramadhani, 2015)

1. ***Database***

*Database* adalah kumpulan file-file yang saling berelasi bisa ditunjukkan dengan kunci tiap file yang ada, satu *database* menunjukkan suatu kumpulan data yang dipakai dalam satu lingkup. Merancang *database* merupakan suatu hal yang sangat penting. Kesulitan utama merancang *database* adalah bagaimana merancang *database* sehingga *database* sesuai dengan keperluan masa kini dan masa yang akan datang. Perancangan model konseptual perlu dilakukan disamping relasinya berdasarkan proses yang diinginkan. Sedangkan pengertian sistem basis data adalah sebagai koleksi dari data-data yang terorganisasi sedemikian rupa sehingga data mudah disimpan dan dimanipulasi (diperbaruhi, dicari, diolah dengan perhitungan-perhitungan teretentu, serta dihapus). (Fadlil et al., 2008).

1. ***Flowchart***

Fungsi *flowchart* adalah digunakan untuk menganalisa, mendesain, mendokumentasikan dan memanajemen sebuah proses atau program di berbagai bidang. Secara khusus, *flowchart* berfungsi untuk membantu menggambarkan situasi apa yang sedang terjadi dan yang akan terjadi dari sebuah simbol dan tanda penghubungnya. Selain itu, *flowchart* ini mampu memperjelas sebuah alur dari suatu sistem baik itu kekurangan atau kelebihan dari berbagai proses di dalam tahapan suatu sistem (Irmayani syafitri, 2021)

**Tabel 2. 2 Flowchart**

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | Garis Hidup (*lifeline*) | Menyatakan kehidupan suatu objek. |
|  | Objek | Menyatakan objek yang berinteraksi. |
|  | Waktu Aktif | Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya. |
|  | Pesan Tipe *Call* | Menyatakan suatu objek memanggil operasi atau metode yang ada pada objek lain atau dirinya sendiri. |
|  | Pesan Tipe *Send* | Menyatakan bahwa suatu objek mengirimkan data/ masukan/ informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim. |

Sumber (Irmayani syafitri, 2021)

1. **UML (*Unifield Modeling Language*)**

Pengertian UML (*Unified Modelling Language*) menurut Nugroho 2014 adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek. Abstraksi konsep dasar UML terdiri dari *structural classification, dynamic behavior*, dan model *management* dapat kita pahami main *concepts* sebagai term yang akan muncul pada saat membuat diagram dan *view* adalah kategori dari diagram tersebut. UML mendefinisikan diagram-diagram sebagai *Use Case diagram*, *Class diagram*, *Statechart diagram*, *Activity diagram*, *Sequence diagram*, *Collaboration diagram*, *Component diagram*, dan *deployment diagram*.

1. **Diagram UML**

Seperti yang telah dikemukakan sebelumnya, UML merupakan *tool* yang tepat untuk memodelkan sebuah rancangan perangkat lunak. Seperti hanya *tool* untuk memodelkan perangkat lunak terstruktur, UML pun memiliki diagram-diagram tertentu dan terstandar. Notasi yang digunakam dalam pemodelan UML yaitu:

1. *Use Case Diagram*

*Use Case diagram* menampilkan sekumpulan *use case* dan *actor* (pelaku) dan hubungan diantara *use case* dan *actor* tersebut. *Use case diagram* digunakan untuk penggambaran *use case* statik dari suatu *system*. *Use case* menjelaskan apa yang dilakukan sistem (atau subsistem) tetapi tidak menspesifikasikan cara kerjanya. *Flow of event* digunakan untuk menspesifikasikan kelakuan dari *use Case* . *Flow of event* menjelaskan *use case* dalam bentuk tulisan dengan sejelas-jelasnya, diantaranya bagaimana, kapan *use case* dimulai dan berakhir, ketika *use case* berinteraksi dengan aktor, obyek apa yang digunakan, alur dasar dan alur *alternative*. (Kiswanto, 2016)

**Tabel 2. 3 Simbol Use Case Diagram**

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | *Use Case* | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal *frase* nama *use Case* . |
|  | Aktor | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat. |
|  | Asosiasi | Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor. |
|  | Ekstensi (*extend*) | Relasi *use case* tambahan ke sebuah *use case*  dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use Case* tambahan itu. |
|  | Generalisasi | Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari yang lainnya. |
|  | Menggunakan (*include*) | Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini. |

Sumber: (Kiswanto, 2016)

1. *Activity Diagram*

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses pada perangkat lunak. Yang perlu di perhatikan adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan yang di lakukan aktor (Kiswanto, 2016)

**Tabel 2. 4 Simbol Activity Diagram**

|  |  |  |
| --- | --- | --- |
| Simbol | **Nama** | Deskripsi |
|  | Status Awal | Status awal aktivitas sistem. |
|  | Aktivitas | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja. |
|  | Percabangan (*decision*) | Asosiasi percabangan digunakan untuk mendefinisikan jika ada pilihan aktivitas lebih dari satu. |
|  | Penggabungan (*join*) | Asosiasi penggabungan digunakan untuk mendefinisikan beberapa aktivitas yang digabungkan menjadi satu. |
|  | Status Akhir | Status akhir yang dilakukan sistem. |
|  | *Swimlane* | Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi. |

Sumber: (Kiswanto, 2016)

1. *Class Diagram*

*Class Diagram*  adalah diagram yang digunakan untuk menampilkan beberapa kelas yang ada dalam sistem perangkat lunak yang akan dikembangkan. *Class Diagram*  menunjukkan antar kelas dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan.

Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut :

1. *Main Class*

*Class* yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.

1. *Class* yang menangani tampilan sistem (*view*)

*Class* yang mendefinisikan dan mengatur tampilan ke pemakai.

1. *Class* yang diambil dari pendefinisian *use case* (*controller*)

*Class* yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case* , kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.

1. *Class* yang diambil dari pendefinisian data (model)

*Class* yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data. Adapun simbol – simbol yang digunakan dalam *class* *diagram* adalah sebagai berikut.

**Tabel 2. 5 Simbol Class Diagram**

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | *Class* | Kelas pada stuktur sistem. |
|  | Antarmuka (*interface*) | Sama dengan konsep antarmuka dalam pemrograman berorientasi objek. |
|  | Asosiasi | Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multipicity*. |
|  | Asosiasi Berarah (*directed association*) | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*. |
|  | Generalisasi | Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus). |
|  | Kebergantungan (*dependency*) | Relasi antar kelas dengan makna kebergantungan antar kelas. |
|  | Agregasi (*aggreggation*) | Relasi antar kelas dengan makna semua bagian (*whole part*). |

Sumber: (Kiswanto, 2016)

Dalam class diagram, terdapat *symbol multiplicity*. *Multiplicity* adalah jumlah banyaknya objek sebuah kelas yang berelasi dengan sebuah objek lain dari kelas lain yang berasosiasi dengan kelas tersebut (Kiswanto, 2016). Macam-macam *multiplicity* disajikan dalam Tabel berikut :

**Tabel 2. 6 Simbol Multiplicity**

|  |  |
| --- | --- |
| **Multiplicity** | **Arti** |
| \* | Banyak |
| 0 | Tepat 0 |
| 1 | Tepat 1 |
| 0…\* | Nol atau lebih |
| 1…\* | Satu atau lebih |

Sumber: (Kiswanto, 2016)

1. *Sequence diagram*

Suatu *sequence diagram* adalah suatu diagram interaksi yang menekankan pada pengaturan waktu dari pesan-pesan. *Sequence diagram* berasosiasi dengan *use* *case*  selama proses pengembangan. Dalam *Unified Model Language* (UML), objek dalam *sequence* *diagram* digambar dengan segiempat yang berisi nama objek yang diberi garis bawah. Objek dapat diberi nama dengan tiga cara : (nama objek), (nama objek dan *class*) atau hanya nama *class* (*anonymous* *object*) (Kiswanto, 2016). Berikut notasi *sequence diagram* seperti terlihat pada gambar dibawah ini:

**Tabel 2. 7 Simbol Sequence Diagram**

|  |  |  |
| --- | --- | --- |
| Simbol | **Nama** | Deskripsi |
|  | Garis Hidup (*lifeline*) | Menyatakan kehidupan suatu objek. |
|  | Objek | Menyatakan objek yang berinteraksi. |
|  | Waktu Aktif | Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya. |
|  | Pesan Tipe *Call* | Menyatakan suatu objek memanggil operasi atau metode yang ada pada objek lain atau dirinya sendiri. |
|  | Pesan Tipe *Send* | Menyatakan bahwa suatu objek mengirimkan data/ masukan/ informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim. |

Sumber: (Kiswanto, 2016)

1. **RUP**

RUP *(Rational Unified Process)* adalah salah satu metode khusus pemrograman berorientasi objek yang digunakan untuk pengembangan sisitem. RUP adalah metode yang menggunakan *use case -driven* dan menyediakan struktur untuk pengembangan perangkat lunak.

Process (RUP) merupakan proses pengembangan perangkat lunak yang dilakukan secara iteratif (berulang) dan inkremental (bertahap dengan progres menaik). Iteratif bisa dilakukan dalam setiap tahap atau iteratif tahap pada proses pengembangan perangkat lunak untuk menghasilkan perbaikan fungsi yang inkremental (bertambah naik) dimana setiap iterasi akan memperbaiki iterasi berikutnya. Salah satu *Unifield Process* yang dikenal adalah RUP *(Rational Unifield Process* (Taryana Suryana, 2007)

Dalam metode RUP terdapat empat *fase* untuk melakukan pengembangan sistem yaitu:

1. *Inception* (permulaan)

Pada tahap *inception* dilakukan pendefinisian ruang lingkup dan perkiraan jadwal serta melakukan analisis kebutuhan *user* dan melakukan perancangan awal perangkat lunak (perancangan arsitektur dan *use case* ).

1. *Elaboration* (perencanaan)

Pada tahap *elaboration* dilakukan perencanaan arsitektur sistem. Pada perencanaan arsitektur sistem dimulai dari melakukan spesifikasi fitur perangkat lunak yang akan digunakan hingga analisis dan desain sistem serta implementasi sistem yang fokus pada arketipe sistem (*prototype*).

1. *Construction* (konstruksi)

Pada tahap *construction* dilakukan pengimplementasian rancangan sistem yang telah dibuat. Pada tahap ini akan fokus pada implementasi perangkat lunak pada kode program serta melakukan pengujian sistem.

1. *Transition* (transisi)

Pada tahap *transition* dilakukan *deployment* perangkat lunak. Pada tahap ini dilakukan evaluasi sistem yang telah dirancang, dianalisis dan dibangun. Tahapan ini bertujuan untuk melakukan pelatihan pada *user* yang akan menggunakan sistem serta melakukan pengujian sistem apakah sudah memenuhi harapan *user.*

# **BAB III METODE PENELITIAN**

1. **Metode Pengumpulan Data**

Metode pengumpulan data yang digunakan yaitu kajian pustaka yang digunakan untuk mencari literatur atau sumber pustaka yang berkaitan dengan perangkat lunak yang dibuat dan membantu memperjelas teori-teori yang ada. Sumber literatur berupa buku, *e-book*, *paper*, jurnal, karya ilmiah, data dari Fakultas Teknik Universitas Halu Oleo dan beberapa situs penunjang.

1. **Metode Pengembangan Sistem**

Metode pengembangan sistem yang digunakan dalam sistem ini adalah metode *Rational Unified Process* (RUP). Tahap ini memiliki empat *fase* dalam pengembangan sistem yaitu:

1. *Inception* (Permulaan)

Pada *fase* ini dilakukan proses pengidentifikasian sistem yang dilakukan dengan menganalisis kebutuhan aplikasi, melakukan kajian terhadap penelitian yang terkait dengan informasi jadwal kegiatan dengan menggunakan algoritma *Priority Scheduling*.

1. *Elaboration* (Perluasan/perencanaan)

Setelah menentukan ruang lingkup penelitian, tahap selanjutkan dilakukan perancangan dan analisis sistem menggunakan UML (*Unified Modeling Language*) yang meliputi *use case , activity, class* dan *sequence diagram*. Selain itu melakukan analisis sistem pada tahap tersebut dilakukan perancangan tampilan *interface* dari sistem yang akan dibangun.

1. *Construction* (Konstruksi)

Proses yang dilakukan pada tahan ini adalah mengimplementasikan kode sesuai perancangan yang telah dilakukan sebelumnya sehingga menjadi aplikasi yang dapat digunakan. Pada tahap ini akan dibangun semua tampilan yang ada pada

sistem informasi jadwal kegiatan Fakultas menggunakan metode *Priority Scheduling*, mulai dari registrasi *user* yang mencakup mahasiswa, dosen dan staf jurusan, tampilan *login*, beranda, *form* pengisian informasi jadwal kegiatan Fakultas.

1. *Transition* (Transisi)

Pada tahap ini akan dilakukan proses pengujian pada aplikasi yang telah dibangun.

1. **Waktu dan Tempat Penelitian**
2. **Waktu**

Waktu pelaksanaan penelitian tugas akhir dilaksanakan mulai dari bulan April 2021 sampai dengan juli 2021. Rincian kegiatan dapat dilihat pada Tabel 3.1 berikut:

**Tabel 3. 1 Gannt Chart Waktu Penelitian**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No | Uraian | Waktu (2021) | | | | | | | | | | | | | | | |
| Maret | | | | April | | | | Mei | | | | Juni | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 1 | *Inception* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | *Elaboration* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | *Construction* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | *Transition* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. **Tempat**

Penelitian tugas akhir judul **“Implementasi *Microservices Archticture* Untuk *Schedule* Handle Pada Aplikasi *Multiple Send Message* Menggunakan Algoritma *Priority Scheduling* (Ps)”** dilakukan pada Fakultas Teknik Universitas Halu Oleo.

1. **Analisis Sistem**

Analisis adalah tahapan yang mempunyai tujuan untuk mengetahui dan mengamati semua yang terletak pada suatu *system*. Pada analisis sistem yang memiliki beberapa pembahasan diantara lain.

1. **Analisis Kebutuhan Sistem**

Analisis kebutuhan sistem tersebut bertujuan untuk mengidentifikasi permasalahan yang terdapat pada suatu sistem, dimana aplikasi digabung yang meliputi perangkat lunak dan hasil analisis terhadap sistem dan elemen-elemen sistem.

1. **Kebutuhan Fungsional**

Analisis kebutuhan fungsional adalah data yang dibutuhkan agar sistem dapat berjalan sesuai dengan prosedur. Setelah melalui tahap analisis kemudian akan ditetapkan kebutuhan-kebutuhan untuk membangun sistem. Pada analisis kebutuhan fungsional terdapat beberapa kebutuhan-kebutuhan dalam membangun sistem yaitu analisis kebutuhan *input*, analisis kebutuhan proses dan analisis kebutuhan *output*.

* 1. **Analisis Kebutuhan I*nput***

Adapun *input* pada aplikasi yang dibangun yaitu:

* + - 1. Data user yang meliputi mahasiswa, dosen dan staf jurusan yang melakukan registrasi.
      2. Data informasi jadwal kegiatan yang ada di Fakultas teknik.

1. **Analisis Kebutuhan Proses**

Analisis proses pada aplikasi ini bertujuan untuk mengidentifikasi permasalahan yang ada pada sistem dimana aplikasi dibangun meliputi perangkat lunak dan analisis terhadap sistem.

1. **Analisis Kebutuhan *Output***

*Output* yang diperoleh pada sistem ini yaitu mahasiswa, dosen dan staf jurusan bisa mendapatkan informasi jadwal kegiatan fakultas.

1. **Kebutuhan Nonfungsional**

Analisis kebutuhan nonfungsional adalah langkah sebuah pembangun sistem menganalisis sumberdaya yang dibutuhkan untuk membangun sebuah sistem yang akan dibangun. Pada analisis kebutuhan nonfungsional terbagi menjadi dua tahap yaitu:

* 1. **Kebutuhan Perangkat Keras**

Untuk menerapkan rancangan yang telah dijelaskan sebelumnya, dibutuhkan beberapa perangkat keras sebagai sarana untuk mengimplementasikan sistem yang dibangun. Berikut spesifikasi minimum laptop atau komputer yang dibutuhkan.

**Tabel 3. 2 Spesifikasi Perangkat Keras**

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Perangkat** | **Spesifikasi** |
| 1 | *Laptop* | Acer Aspire E 14 |
| 2 | *Processor* | Intel ® Core (TM) i5 – 6200U CPU @ 2.30 GHz 2.40 GHz |
| 3 | *Monitor* | 14 Inch |
| 4 | *Memori* | Ram 12 GB, DDR4 L Memory |
| 5 | *Hardisk* | 500 GB, 128 GB SSD Adata |

* 1. **Kebutuhan Perangkat Lunak**

Adapun perangkat lunak atau *software* yang dibutuhkan adalah sebagai berikut:

**Tabel 3. 3 Spesifikasi Perangkat Lunak**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama** | **Fungsi** | **Spesifikasi** |
| 1 | *Windows* | Sistem operasi | Windows 10 Pro |
| 2 | *Apache* | Web server | Apache 2.4.39 |
| 3 | *MariaDB* | Basisdata | MariaDB 10.3.16 |
| 4 | MySQL | Bahasa *Query* |  |
| 5 | PHPMyAdmin | Alat management basis data | PHPMyAdmin 4.9.0.1 |
| 6 | XAMPP | Lingkungan pengembangan | XAMPP 3.2.4 |
| 7 | PHP : *HyperText Processor* | Bahasa pemograman web | PHP 7.2.20 |
| 8 | *Framework Codeigniter* | *Framework PHP* | Codeigniter 3.1.11 |
| 9 | *Mozilla Firefox* | *Web browser* | Firefox 84.0.1 (64bit) |
| 10 | *Visual Studio Code* | *Text editor* | VSC 1.52.1 |

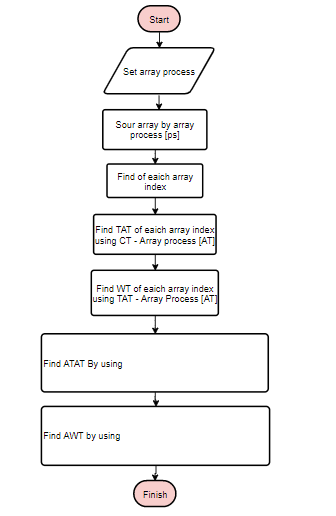
1. **Analisis Perancangan Sistem**

Perancangan sistem yang akan dibangun terdiri atas perancangan *flowchart* dan perancangan UML serta perancangan *User interface.*

1. **Analisis Perancangan *Flowchart***

Setelah menganalisis sistem, makan didapatkan *flowchart* diagram penerapan metode *Priority scheduling* pada Implementasi *Microservices archticture* Untuk *Schedule* Handle Pada Aplikasi *multiple send message* menggunakan algoritma *Priority scheduling* (Ps). Adapun alur kerja *flowchart* diagram sebagai berikut:

1. Admin menginputkan informasi penjadwalan
2. Mencari nilai *turn around time* dan nilai *waiting time*
3. Mencari nilai rata dari *turn around time* dan nilai *waiting time*



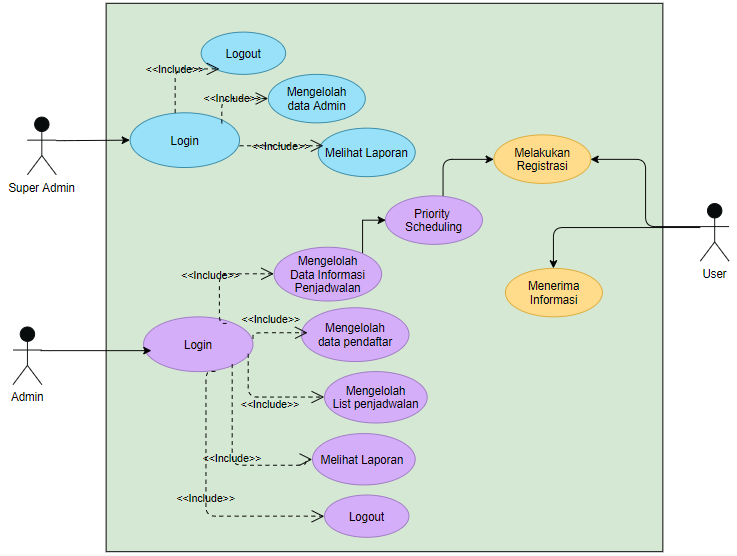
**Gambar 3. 1 Flowchart**

1. **Analisis Perancangan UML**

Aplikasi dibangun dengan menggunakan *Unified Modeling Language* (UML). Dimana UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram yang terdiri dari *Use Case Diagram*, *Activity diagram, Squence diagram* dan *Class diagram*.

1. ***Use Case Diagram***

*Use Case diagram* digunakan dalam memodelkan fungsionalitas-fungsionalitas sistem dilihat dari pengguna yang ada diluar sistem (aktor). Berikut adalah *Use Case diagram* untuk sistem yang akan dibangun:



**Gambar 3. 2 Use Case**

**Tabel 3. 4 Deskripsi Case Login (Superadmin dan Admin)**

|  |  |
| --- | --- |
| **Nama** | **Login** |
| Deskripsi singkat: | Superadmin mengelolah data admin pada sistem |
| Persyaratan | Superadmin harus *input* *username* dan *password* |
| Kondisi akhir | Masuk untuk *manage* sistem |
| Situasi kesalahan | i. Ketika yang masuk bukan superadmin  ii. Salah *input* *username* dan *password* |
| Status sistem dalam saat terjadi kesalahan | * + - 1. Kembali ke halaman sistem ketika salah menginputkan *username* dan *password.*       2. Tidak dapat masuk kedalam sistem. |
| Aktor | Superadmin |
| *Trigger* | Keamanan *login* untuk admin |
| Proses Standar | 1. Superadmin masuk ke sistem 2. Superadmin *input username* 3. Superadmin input *password* |
| Proses alternatif | Mengecek data *username* dan *password* |

**Tabel 3. 5 Deskripsi Case Mengelolah Data Admin**

|  |  |
| --- | --- |
| **Nama** | **Mengelolah Data Admin** |
| Deskripsi singkat: | Superadmin mengelolah data Admin pada sistem |
| Persyaratan | Superadmin harus *input* *username* dan *password* |
| Kondisi akhir | Data admin ditampilkan |
| Situasi kesalahan | - |
| Status sistem dalam saat terjadi kesalahan | - |
| Aktor | Superadmin |
| *Trigger* | - |
| Proses Standar | * + - 1. Superadmin masuk ke sistem       2. Superadmin melakukan penambahan data admin       3. Superadmin menyimpan data admin       4. Data admin ditampilkan |
| Proses alternatif | - |

**Tabel 3. 6 Deskripsi Case Melihat Laporan**

|  |  |
| --- | --- |
| **Nama** | **Melihat Laporan** |
| Deskripsi singkat: | Superadmin dan admin dapat melihat laporan jadwal informasi kegiatan fakultas. |
| Persyaratan | Superadmin dan admin harus masuk ke sistem terlebih dahulu. |
| Kondisi akhir | Data jadwal informasi kegiatan ditampilkan |
| Situasi kesalahan | - |
| Status sistem dalam saat terjadi kesalahan | *-* |
| Aktor | Superadmin dan admin |
| *Trigger* | - |
| Proses Standar | * + - 1. Superadmin dan admin masuk ke sistem  1. Superadmin dan admin melihat laporan informasi. 2. Superadmin dan admin dapat mencetak laporan. |
| Proses alternatif | - |

**Tabel 3. 7 Deskripsi Case Mengelolah Admin**

|  |  |
| --- | --- |
| **Nama** | **Mengelolah Admin** |
| Deskripsi singkat: | Admin mengelolah data informasi pada sistem |
| Persyaratan | Admin harus *input* *username* dan *password* |
| Kondisi akhir | Masuk untuk *manage* sistem |
| Situasi kesalahan | i. Ketika yang masuk bukan admin  ii. Salah *input* *username* dan *password* |
| Status sistem dalam saat terjadi kesalahan | * + - 1. Kembali ke halaman sistem ketika salah menginputkan *username* dan *password.*       2. Tidak dapat masuk kedalam sistem. |
| Aktor | Admin |
| *Trigger* | Keamanan *login* untuk admin |
| Proses Standar | 1. Admin masuk ke sistem 2. Admin *input username* 3. Admin input *password* |
| Proses alternatif | Mengecek data *username* dan *password* |

**Tabel 3. 8 Deskripsi Case Mengelolah Data Informasi Penjadwalan**

|  |  |
| --- | --- |
| **Nama** | **Informasi Penjadwalan** |
| Deskripsi singkat: | Admin mengelolah data informasi penjadwalan pada sistem |
| Persyaratan | Admin harus masuk ke sistem |
| Kondisi akhir | Menampilkan halaman *form* untuk mengatur jadwal sebaran informasi kegiatan. |
| Situasi kesalahan | - |
| Status sistem dalam saat terjadi kesalahan | *-* |
| Aktor | Admin |
| *Trigger* | - |
| Proses Standar | 1. Admin masuk ke sistem 2. Admin mengisi *form* informasi 3. Admin memilih tujuan kepada siapa informasi akan di kirim. 4. Data informasi ditampilkan |
| Proses alternatif | *-* |

**Tabel 3. 9 Deskripsi Case List Penjadwalan**

|  |  |
| --- | --- |
| **Nama** | ***List* Penjadwalan** |
| Deskripsi singkat: | Admin mengelolah data informasi pada sistem |
| Persyaratan | Admin harus masuk ke sistem terlebih dahulu |
| Kondisi akhir | Data penjadwalan informasi ditampilkan |
| Situasi kesalahan | - |
| Status sistem dalam saat terjadi kesalahan | *-* |
| Aktor | Admin |
| *Trigger* | - |
| Proses Standar | 1. Admin masuk ke sistem 2. Admin melihat data *list* informasi penjadwalan. |
| Proses alternatif |  |

**Tabel 3. 10 Deskripsi Case Registrasi**

|  |  |
| --- | --- |
| **Nama** | **Registrasi** |
| Deskripsi singkat: | *User* mengisi *form* registrasi |
| Persyaratan | *User* harus Dosen, Staf jurusan dan mahasiswa |
| Kondisi akhir | Mendapatkan informasi |
| Situasi kesalahan | - |
| Status sistem dalam saat terjadi kesalahan | *-* |
| Aktor | *User* |
| *Trigger* | - |
| Proses Standar | 1. *User* melakukan registrasi 2. *User* dapat menerima informasi kegiatan |
| Proses alternatif | - |

**Tabel 3. 11 Deskripi Case Menerima Informasi**

|  |  |
| --- | --- |
| **Nama** | **Menerima Informasi** |
| Deskripsi singkat: | *User* menerima informasi kegiatan |
| Persyaratan | *User* harus melakukan registrasi |
| Kondisi akhir | Mendapatkan sebuah informasi |
| Situasi kesalahan | - |
| Status sistem dalam saat terjadi kesalahan | *-* |
| Aktor | *User* |
| *Trigger* | - |
| Proses Standar | 1. *User* melakukan registrasi 2. *User* dapat menerima informasi kegiatan |
| Proses alternatif | - |

**Tabel 3. 12 Deskripsi Case data pendaftar**

|  |  |
| --- | --- |
| **Nama** | **Data pendaftar** |
| Deskripsi singkat: | Admin mengelolah data pendaftar |
| Persyaratan | Admin harus masuk ke sistem |
| Kondisi akhir | Menampilkan data pendaftar |
| Situasi kesalahan | - |
| Status sistem dalam saat terjadi kesalahan | *-* |
| Aktor | Admin |
| *Trigger* | - |
| Proses Standar | 1. Admin masuk ke sistem 2. Admin mengelolah data pendaftar 3. Admin menampilkan data pendaftar |
| Proses alternatif | - |

**Tabel 3. 13 Deskripsi Case Priority Scheduling**

|  |  |
| --- | --- |
| **Nama** | ***Priority scheduling*** |
| Deskripsi singkat: | *Priority scheduling* mengelolah penjadwalan kegiatan. |
| Persyaratan | Mengelolah penjadwalan informasi dikirim |
| Kondisi akhir | Mengatur jalannya penjadwalan |
| Situasi kesalahan | - |
| Status sistem dalam saat terjadi kesalahan | *-* |
| Aktor | - |
| *Trigger* | - |
| Proses Standar | 1. *Priority scheduling* mengelolah penjadwalan informasi dikirim   2. Informasi tersampaikan |
| Proses alternatif | - |

**Tabel 3. 14 Deskripsi Case Logout**

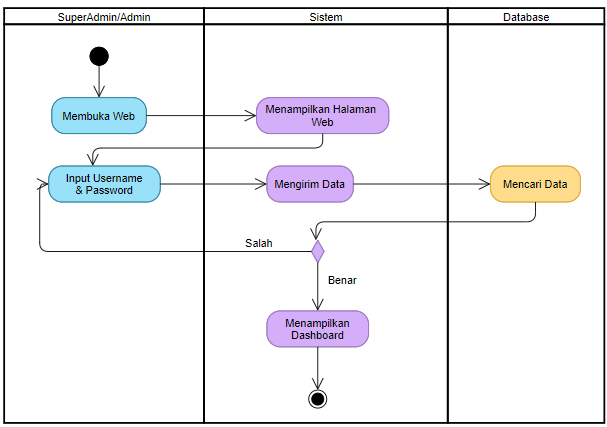
|  |  |
| --- | --- |
| **Nama** | ***Logout*** |
| Deskripsi singkat: | Superadmin dan admin dapat *logout* dari sistem |
| Persyaratan | Superadmin dan admin melakukan *logout* |
| Kondisi akhir | Menu *logout* di tampilkan |
| Situasi kesalahan | - |
| Status sistem dalam saat terjadi kesalahan | *-* |
| Aktor | Superadmin dan admin |
| *Trigger* | - |
| Proses Standar | Superadmin dan admin keluar dari sistem |
| Proses alternatif | - |

1. ***Activity Diagram***

*Activity diagram* menggambarkan berbagai alir aktivitas dalam sistem yang dirancang, bagiamana masing-masing alir berawal, *decision* yang mungkin terjadi dan bagaimana aktivitas tersebut berakhir. Berikut *activity diagram* yang akan menggambarkan alir aktivitas sistem.

1. ***Activity Diagram Login***

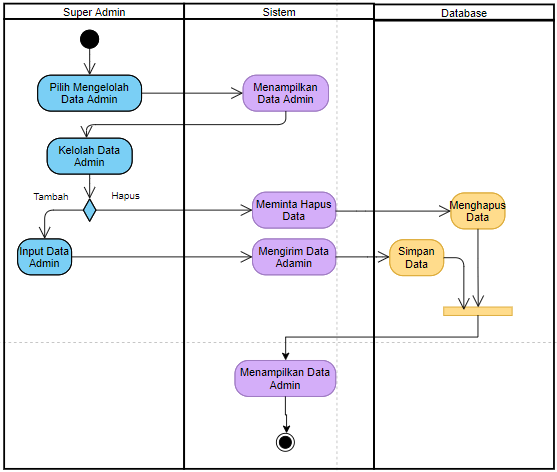
Pada gambar 3.3 merupakan diagram aktivitas yang menunjukkan aktivitas superadmin dan admin melakukan *input* *username* dan *password*, kemudian sistem akan melakukan validasi kebenaran datanya.

****

**Gambar 3. 3 Activity Diagram Login**

1. ***Activity Diagram* Mengelolah Data Admin**

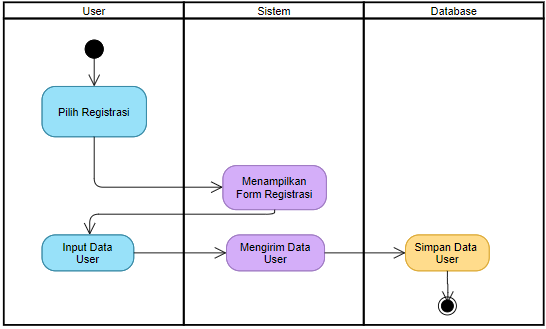
Gambar 3.4 merupakan *activity diagram* yang menunjukkan aktivitas superadmin ketika melakukan penambahan data admin, yang berisikan *form* untuk *input* data admin.

******

**Gambar 3. 4 Activity Diagram Mengelolah Data Admin**

1. ***Activity Diagram User***

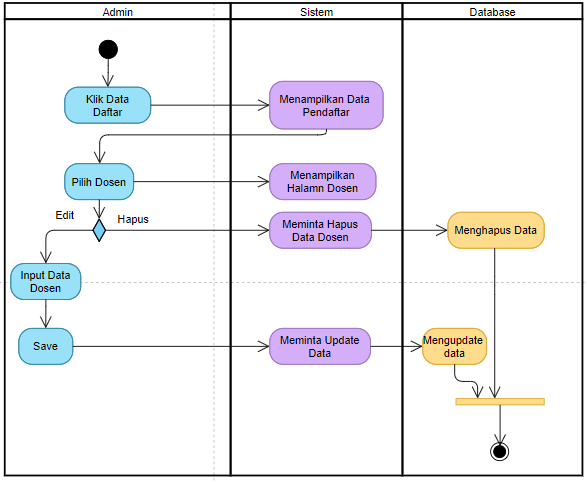
Pada gambar 3.5 merupakan diagram aktivitas yang menunjukkan aktivitas *user* memilih registrasi, dimana *user* akan melakukan registrasi untuk mendapatkan data pendaftar.



**Gambar 3. 5 Activity Diagram User**

1. ***Activity Diagram* Data Pendaftar**
2. Data Pendaftar Dosen

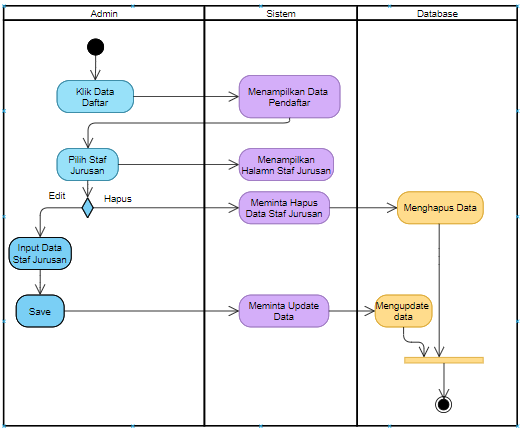
Pada gambar 3.6 merupakan diagram aktivitas yang menunjukkan aktivitas admin dalam mengelolah data pendaftar dosen.

******

**Gambar 3. 6 Data Pendaftar Dosen**

1. Data Pendaftar Staf Jurusan

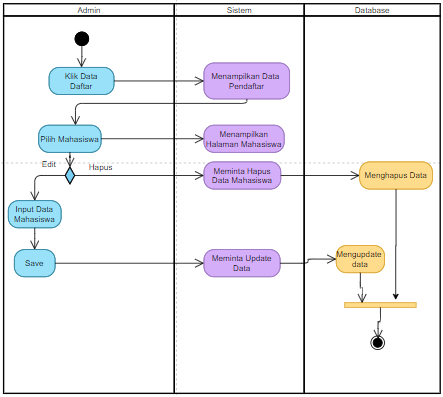
Pada gambar 3.7 merupakan diagram aktivitas yang menunjukkan aktivitas admin dalam mengelolah data pendaftar staf jurusan

.

**Gambar 3. 7 Data Pendaftar Staf Jurusan**

1. Data Pendaftar Mahasiswa

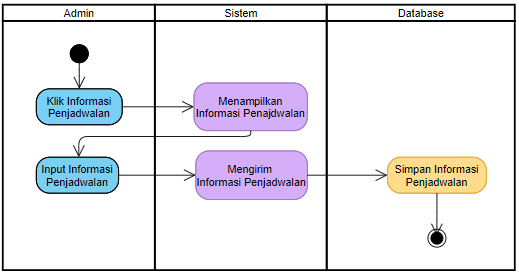
Pada gambar 3.8 merupakan diagram aktivitas yang menunjukkan aktivitas admin dalam mengelolah data pendaftar mahasiswa.



**Gambar 3. 8 Data Pendaftar Mahasiswa**

1. ***Activity Diagram* Informasi Penjadwalan**

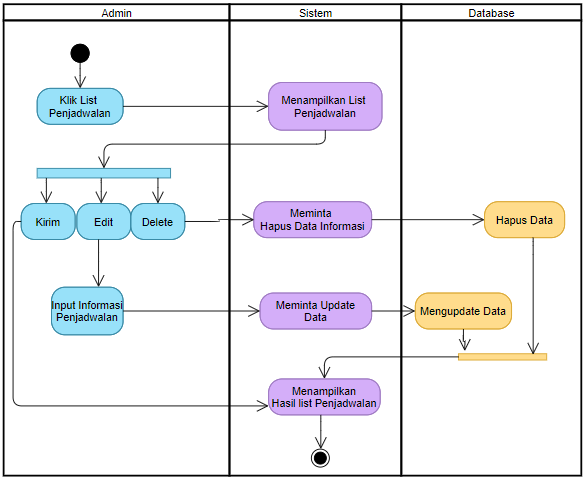
Pada gambar 3.9 merupakan diagram aktivitas yang menunjukkan aktivitas admin memilih menu informasi penjadwalan.



**Gambar 3. 9 Activity Diagram Informasi Penjadwalan**

1. ***Activity Diagram List* Penjadwalan**

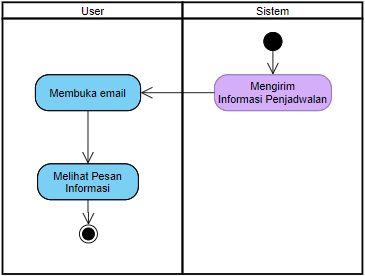
Pada gambar 3.10 merupakan diagram aktivitas yang menunjukkan aktivitas admin memilih menu *list* penjadwalan dan memilih untuk mengedit dan hapus data.



**Gambar 3. 10 Activity Diagram List Penjadwalan**

1. ***Activity Diagram* Menerima Informasi**

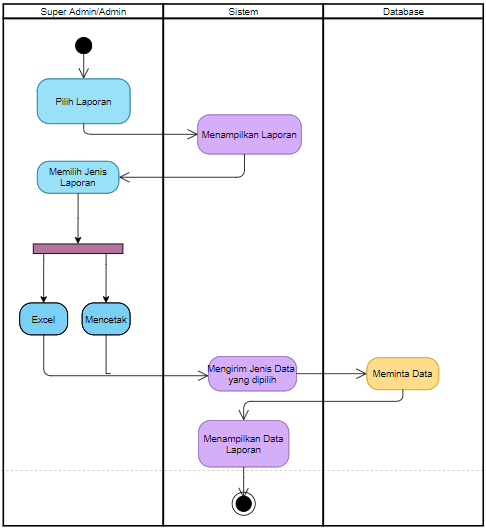
Pada gambar 3.11 merupakan diagram aktivitas yang menunjukkan aktivitas *user* dalam menerima sebuah informasi kegiatan fakultas.



**Gambar 3. 11 Activity Diagram Menerima Informasi**

1. ***Activity Diagram* Laporan**

Pada gambar 3.12 *Activity diagram* menunjukkan aktivitas superadmin dan admin dapat melihat laporan kegiatan fakultas.

******

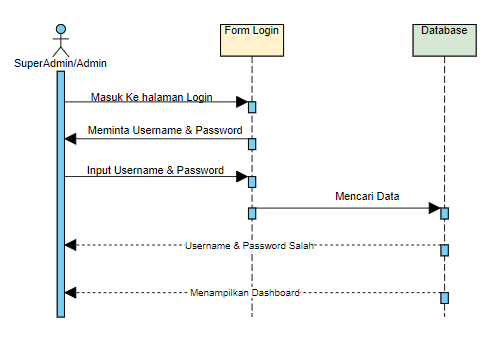
**Gambar 3. 12 Activity Diagram Laporan**

1. ***Sequence Diagram***

*Squence diagram* menggambarkan interaksi antar objek didalam dan disekitar sistem yang digambarkan terhadap waktu. Berikut ini merupakan *sequence diagram* yang akan menggambarkan antar objek dan sistem.

1. ***Sequence Diagram Login***

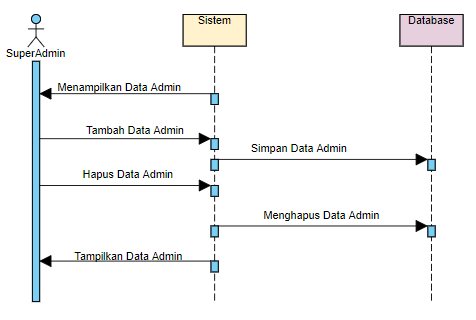
Gambar 3.13 *squence diagram* yang menunjukkan proses *login* bagi admin dan superadmin dimana aktor memasukkan *username* dan *password* lalu di cek ke *database*.

****

**Gambar 3. 13 Sequence Diagram Login**

1. ***Sequence Diagram* Mengelolah Data Admin**

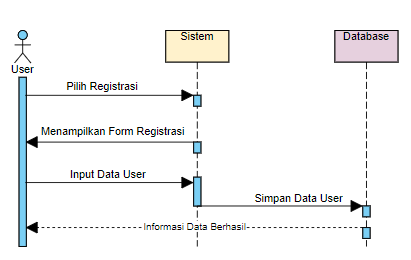
Gambar 3.14 *sequence diagram* merupakan aktivitas yang menunjukkan ketika melakukan penambahan data admin, yang berisikan *form* untuk *input* data admin.



**Gambar 3. 14 Sequence Diagram Mengelolah Data Admin**

1. ***Sequence Diagram* *User***

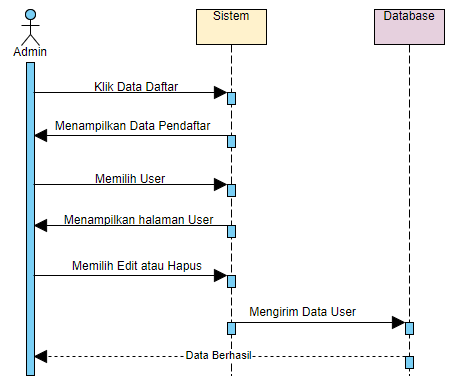
Gambar 3.15 *sequence diagram* aktivitas yang menunjukkan aktivitas *user* memilih registrasi, dimana *user* akan melakukan registrasi untuk mendapatkan data pendaftar.



**Gambar 3. 15 Sequence Diagram User**

1. ***Sequence* Data Pendaftar**

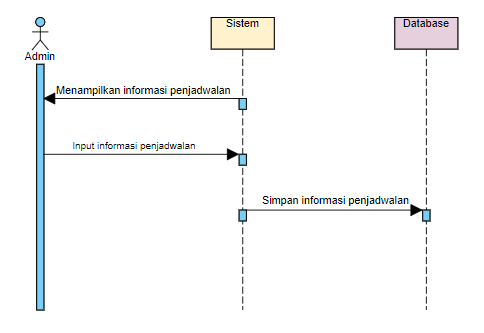
Gambar 3.16 *sequence* data pendaftar merupakan aktivitas yang menunjukkan aktivitas adminmemilih data pendaftar, dimana sistem menampilkan data *user*.



**Gambar 3. 16 Sequence Diagram Data Pendaftar**

1. ***Sequence* Informasi Penjadwalan**

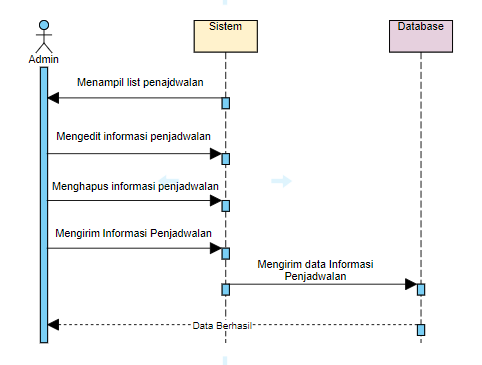
Gambar 3.17 *sequence* informasi penjadwalan merupakan aktivitas yang menunjukkan aktivitas adminmengirim sebuah informasi kegiatan fakultas.



**Gambar 3. 17 Sequence Diagram Informasi Penjadwalan**

1. ***Sequence List* Penjadwalan**

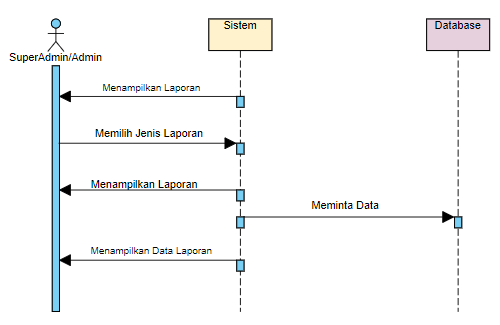
Gambar 3.18 *sequence diagram* informasi penjadwalan merupakan aktivitas yang menunjukkan aktivitas admin melalukan pengeditan dan menghapus data informasi kegiatan fakultas.



**Gambar 3. 18 Sequence Diagram List Penjadwalan**

1. ***Sequence Diagram* Laporan**

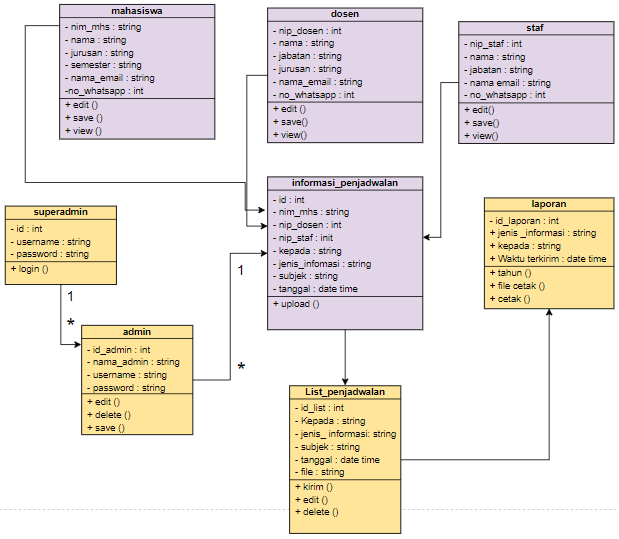
Gambar 3.19 *Sequence* *diagram* menunjukkan aktivitas superadmin dan admin dapat melihat laporan serta mencetaknya berdasarkan tahunnya, kemudian bisa mendapatkan file excelnya.



**Gambar 3. 19 Sequence Diagram Laporan**

1. ***Class Diagram***

*Class diagram* merupakan diagram yang selalu ada dalam dipemodelan sistem yang berorientasi objek. *Class diagram* menunjukkan hubungan antar *class* dalam sistem yang sedang dibangun dan bagaimana saling berkaloborasi untuk mencapai suatu tujuan tersebut. Berikut *class diagram* sistem yang akan dibangun sebagai berikut:

­­­­

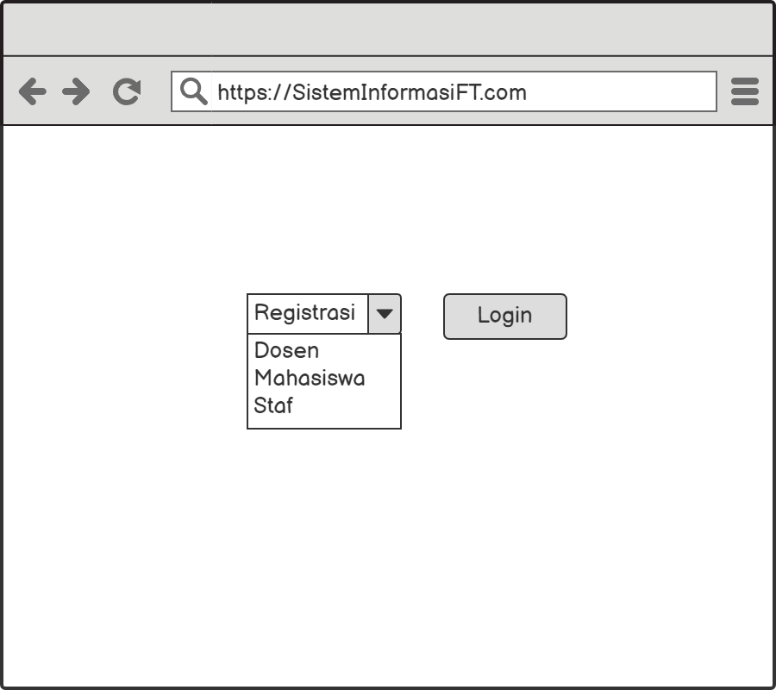
**Gambar 3. 20 Class Diagram**

1. **Perancangan Antarmuka (*Interface*)**

Rancangan antar muka merupakan desain awal dari sisem yang akan berinteraksi langsung dengan *user.* Rancangan ini diperlukan agar sistem yang dibangun bisa lebih terarah dan jelas dari sisi tampilan.

1. **Menu Pilih *Login***

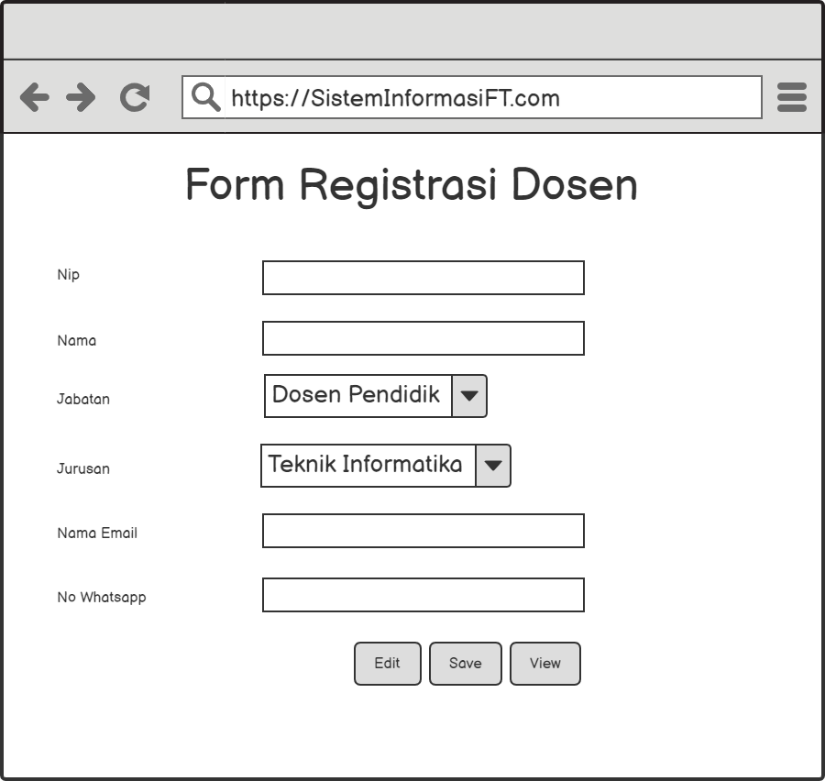
Menu pilih *login* menampilkan registrasi dan *login* ketika memilih registrasi maka *user* menginputkan datanya untuk dapat terdaftar dalam menerima informasi kegiatan fakultas, dan menu *login* untuk superadmin dan admin dapat dilihat pada gambar berikut.



**Gambar 3. 21 Menu Pilih Login**

1. **Menu Registrasi Dosen**

Menu informasi penjadwalan menampilkan *form* pengisian informasi kegiatan yang akan dikirim dapat dilihat pada gambar berikut.



**Gambar 3. 22 Menu Registrasi Dosen**

1. **Menu Registrasi Mahasiswa**

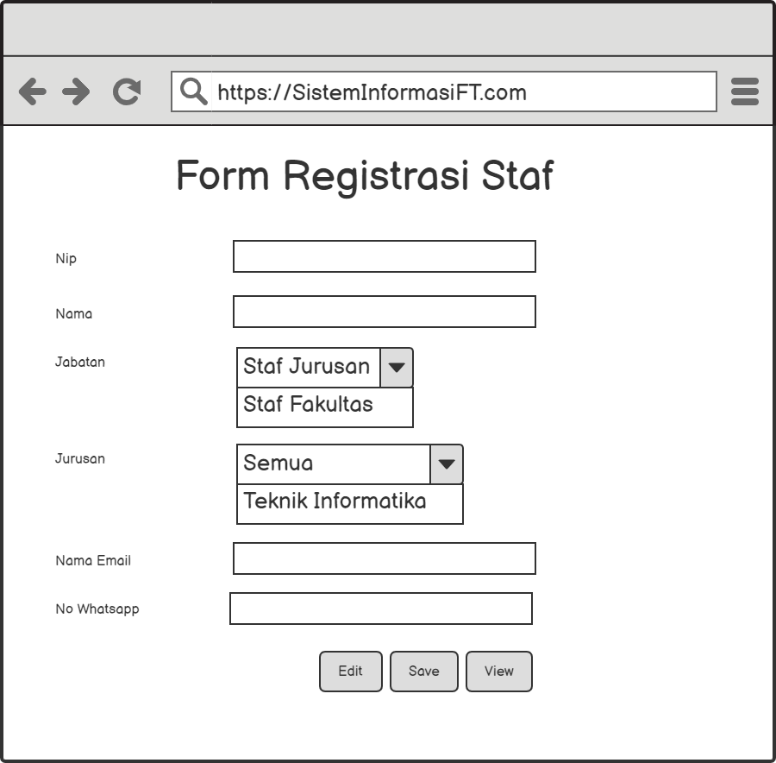
Menu registrasi mahasiwa menampilkan *form* pengisian data mahasiswa untuk dapat nantinya menerima sebuah informasi kegiatan fakultas dapat dilihat pada gambar berikut.



**Gambar 3. 23 Menu Registrasi Mahasiswa**

1. **Menu Registrasi Staf**

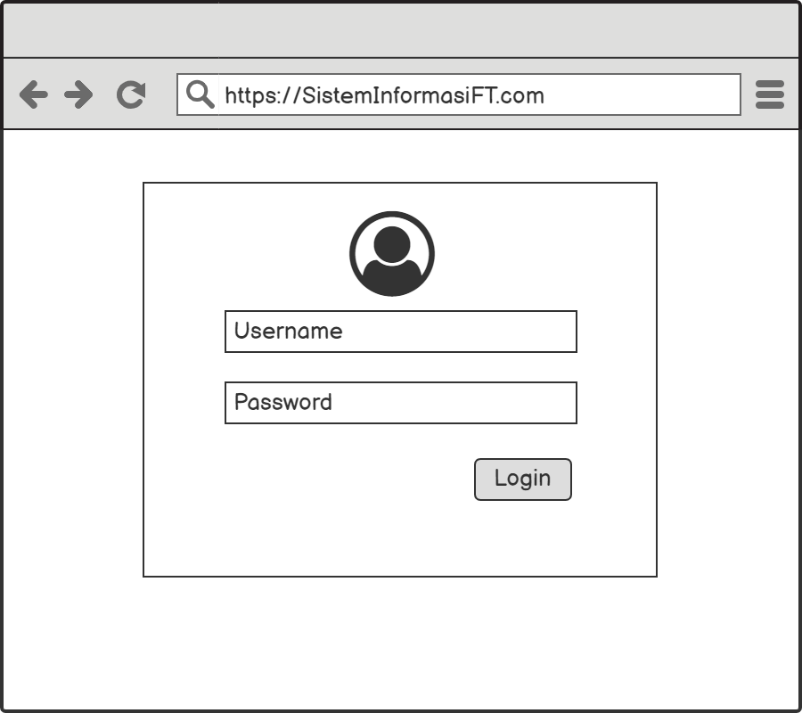
Menu registrasi staf menampilkan *form* pengisian data staf untuk dapat nantinya menerima sebuah informasi kegiatan fakultas dapat dilihat pada gambar berikut.



**Gambar 3. 24 Menu Registrasi Staf**

1. **Menu *Login***

Halaman *login* pada *website* menampilkan *form login* dimana tedapat *username* dan *password*. Setiap *field* harus diisi untuk *login*. Selain itu *form* ini juga telah ditambahkan fungsi validasi agar dapat mengecek data yang benar. Berikut dari tampilan *login* dapat dilihat pada gambar berikut.



**Gambar 3. 25 Menu Login**

1. **Menu Beranda Superadmin**

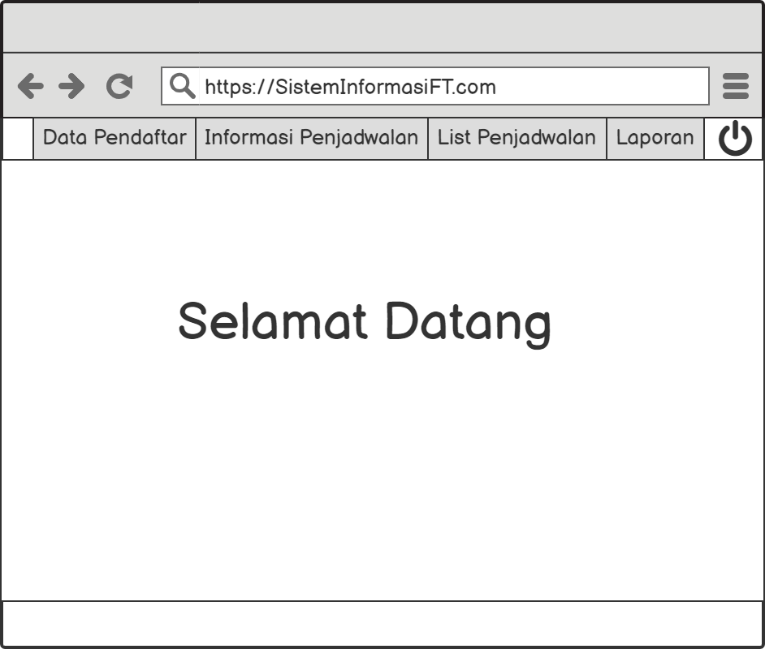
Tampilan beranda superadmin merupakan tampilan pertama yang akan digunakan pada sistem ini dapat dilihat pada gambar berikut.



**Gambar 3. 26 Menu Beranda Superadmin**

1. **Menu Beranda Admin**

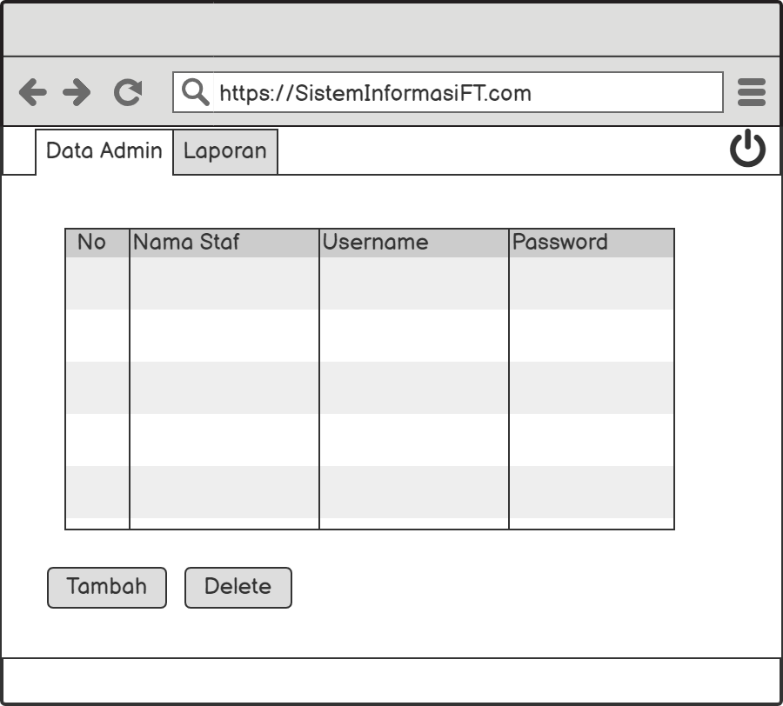
Tampilan beranda admin merupakan tampilan pertama yang akan digunakan pada sistem ini dapat dilihat pada gambar berikut.



**Gambar 3. 27 Menu Beranda Admin**

1. **Menu Data Admin**

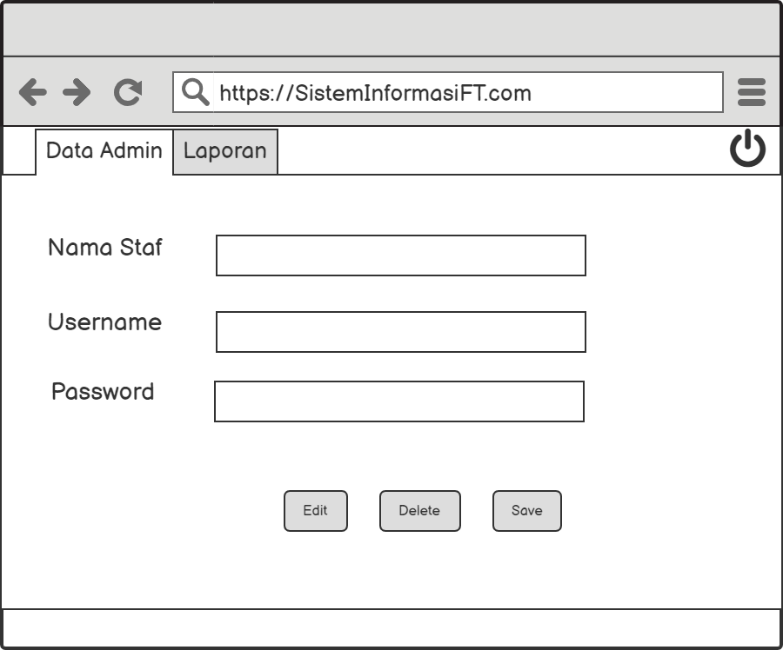
Menu data admin menampilkan data admin dimana superadmin dapat melakukan penghapusan data admin dan juga tambah data admin dapat dilihat gambar berikut.



**Gambar 3. 28 Menu Data Admin**

1. **Menu Tambah Data Admin**

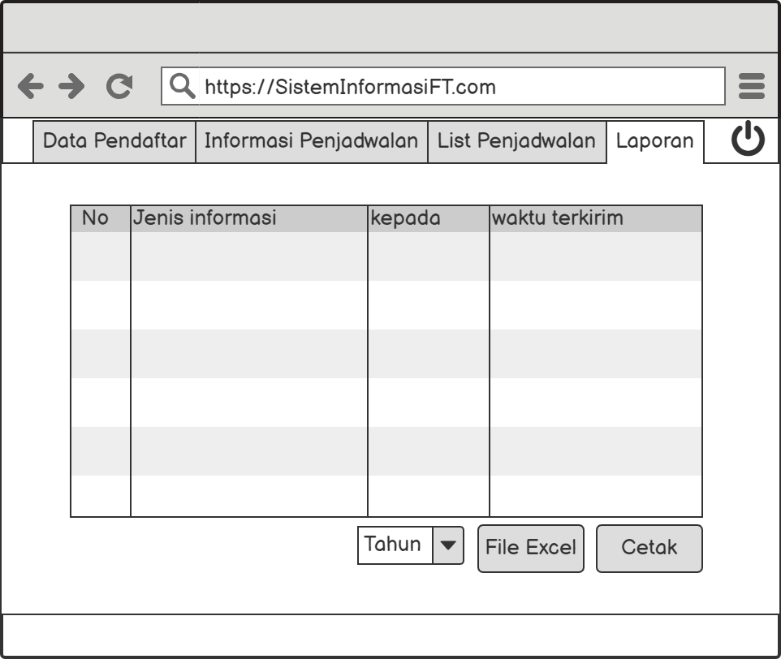
Menu tambah data admin menampilkan *form* pengisian data admin yang akan di inputkan ke dalam data admin dapat dilihat pada gambar berikut.



**Gambar 3. 29 Menu Tambah Data Admin**

1. **Menu Laporan**

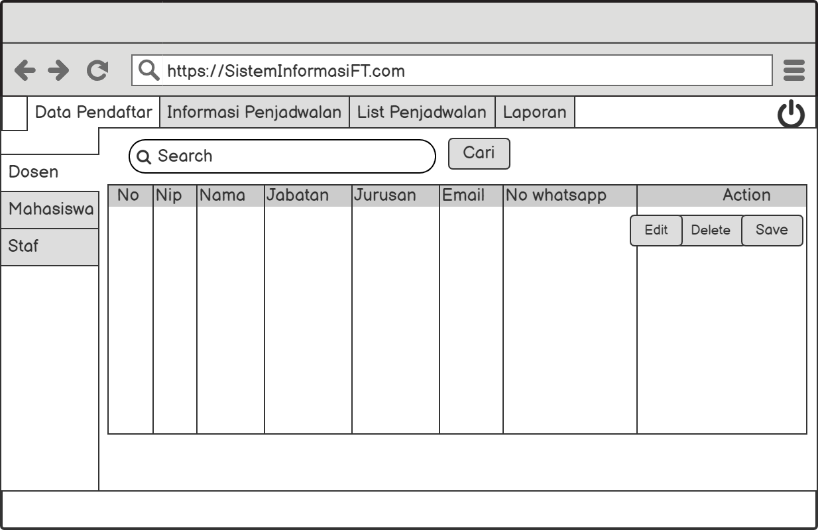
Menu laporan menampilkan sebuah informasi penjadwalan kegiatan yang sudah terkirim dapat dilihat pada gambar berikut.



**Gambar 3. 30 Menu Laporan**

1. **Menu Data Pendaftar**

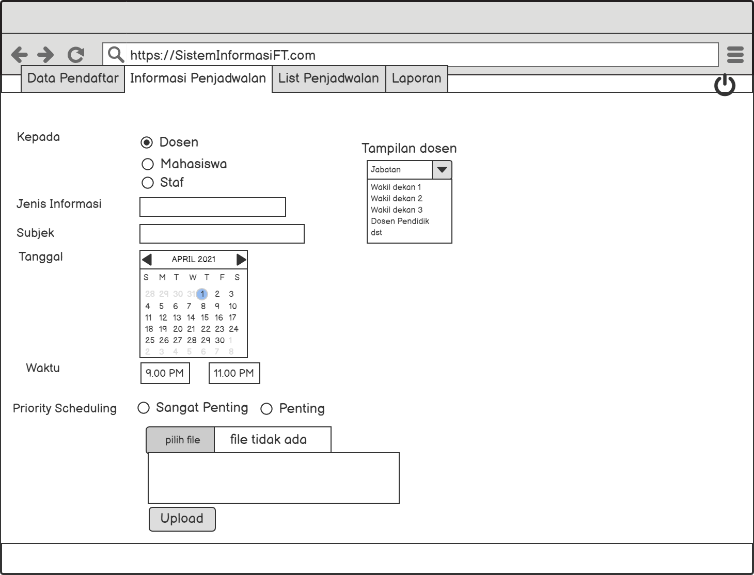
Menu data pendaftar menampilkan menu dosen, mahasiswa dan staf jurusan, dimana setiap menu menampilkan data dari dosen, mahasiswa dan staf jurusan dapat dilihat pada gambar berikut.



**Gambar 3. 31 Menu Data Pendaftar**

1. **Menu Informasi Penjadwalan**
   1. Pemilihan Dosen

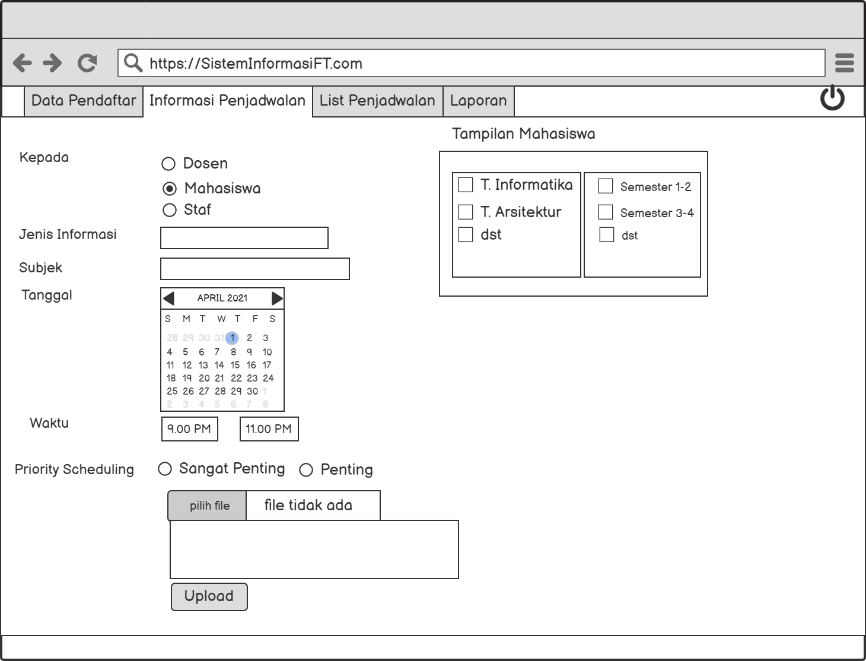
Menu informasi penjadwalan menampilkan pengisian informasi dimana terdapat pemilihan dosen sebagai penerima informasi dapat dilihat pada gambar berikut.



**Gambar 3. 32 Menu Informasi Penjadwalan (Dosen)**

* 1. Pemilihan Mahasiswa

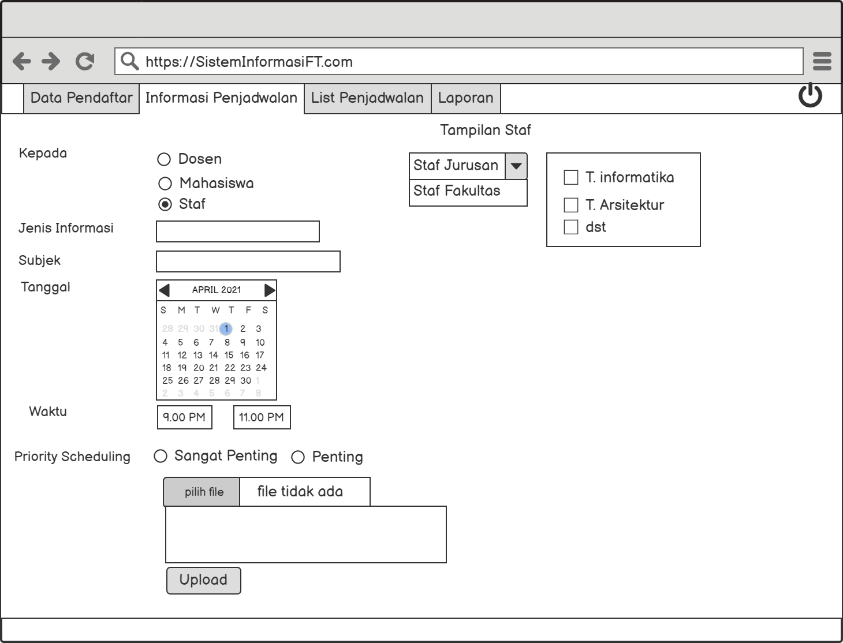
Menu informasi penjadwalan menampilkan pengisian informasi dimana terdapat pemilihan mahasiswa sebagai penerima informasi dapat dilihat pada gambar berikut.



**Gambar 3. 33 Menu Informasi Penjadwalan (Mahasiswa)**

1. Pemilihan Staf

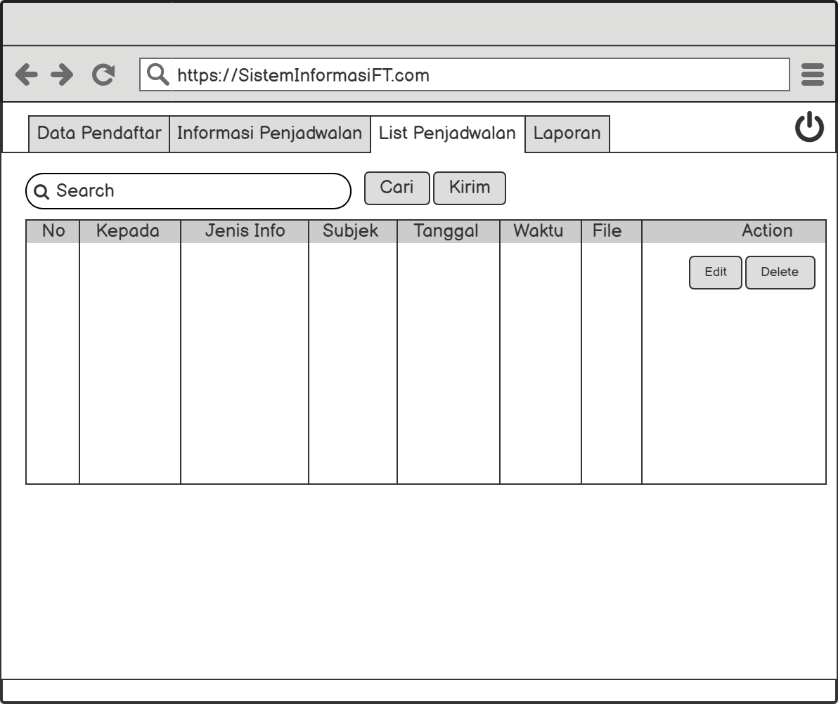
Menu informasi penjadwalan menampilkan pengisian informasi dimana terdapat pemilihan staf sebagai penerima informasi dapat dilihat pada gambar berikut.



**Gambar 3. 34 Menu Informasi Penjadwalan (Staf)**

1. **Menu *List* Penjadwalan**

Menampilkan semua informasi yang sudah terikirim dapat dilihat pada gambar berikut.



**Gambar 3. 35 Menu List Penjadwalan**

# **DAFTAR PUSTAKA**

appkey. (2020). *XAMPP* adalah: Pengertian dan Fungsinya. Juni 4 2020. https://appkey.id/pembuatan-website/teknologi-web/xampp-adalah/

Astri Dwi Juniarahmatunisa. (n.d.). Sistem Penjadwalan Penayangan Iklan Dengan Menggunakan Aalgoritma *Priority Scheduling.*

Fadlil, A., Firdausy, K., & Hermawan, F. (2008). Pengembangan Sistem Basis Data Presensi Perkuliahan Dengan Kartu Mahasiswa Ber-Barcode. Telkomnika (Telecommunication Computing Electronics And Control), 6(1), 65. Https://Doi.Org/10.12928/Telkomnika.V6i1.552

Fitri, A., Permana, I., & Marsal, A. (2016). Penerapan Constraint Satisfaction Problem Pada Metode Priority Schedulinguntuk Penjadwalan Khutbah Jum’at Para Mubaligh Di Ikmi Pekanbaru. Sains, 13(2), 190–194. Http://Ejournal.Uin-Suska.Ac.Id/Index.Php/Sitekin

Fratiwi, D., Mariana, N., Studi, P., Informasi, S., Informasi, F. T., Stikubank, U., Antrian, K. K., Pelayanan, S., Come, F., Serve, F., Semarang, B., Waktu, T. A., Ta, R., & Job, T. A. (2020). Metode Fcfs Dalam Menunjang Sistem Layanan Antrian Pembagian Dana Pensiun Studi Kasus Kantor Pos Bongsari. 978–979.

Herdiana, Y. (2014). Aplikasi Rumus Matematika Sma Berbasis Mobile. Jurnal Ilmiah Komputer Dan Informatika ( Komputa ).

Irmayani Syafitri. (2021). Pengertian Flowchart Beserta Fungsi Dan Simbol-Simbol Flowchart Yang Paling Umum Digunakan. 15 Februari 2021. Https://Www.Nesabamedia.Com/Pengertian-Flowchart/

Istiono, W., Hijrah, & Sutarya. (2016). Pengembangan Sistem Aplikasi Penilaian Dengan Pendekatan Mvc Dan Menggunakan Bahasa Php Dengan Framework Codeigniter Dan Database Mysql Pada Pahoa College Indonesia. Jurnal Ticom, 5(1), 53–59. Https://Media.Neliti.Com/Media/Publications/93757-Id-Pengembangan-Sistem-Aplikasi-Penilaian-D.Pdf

Jefri1, Kristina2, T. W. (2015). Berbasis Android Menggunakan Metode Priority Scheduling. 1–11.

Josi, A. (2017). Penerapan Metode Prototyping Dalam Membangun Website Desa (Studi Kasus Desa Sugihan Kecamatan Rambang). Jti, 9(1), 50–57.

Kiswanto. (2016). Sistem Informasi Akademik Sub-Sistem : Utility Dan Epsbed. Journal Of Chemical Information And Modeling, 53(9), 1689–1699.

Lokapitasari Belluano, P. L., Purnawansyah, P., Panggabean, B. L. E., & Herman, H. (2020). Sistem Informasi Program Kreativitas Mahasiswa Berbasis Web Service Dan Microservice. Ilkom Jurnal Ilmiah, 12(1), 8–16. Https://Doi.Org/10.33096/Ilkom.V12i1.492.8-16

Mufrizal, R., & Indarti, D. (2019). Refactoring Arsitektur Microservice Pada Aplikasi Absensi Pt. Graha Usaha Teknik. Jurnal Nasional Teknologi Dan Sistem Informasi, 5(1), 57–68. Https://Doi.Org/10.25077/Teknosi.V5i1.2019.57-68

Munawar, G., & Hodijah, A. (2018). Analisis Model Arsitektur Microservice Pada Sistem Informasi Dplk. Sinkron: Jurnal Dan Penelitian Teknik Informatika, 3(1), 232–239.

Nugroho:2014. (N.D.). Book\_Adi Nugroho\_Rekayasa Perangkat Lunak\_Pengantar.Pdf.

Prabowo, D. (2015). Website E-Commerce Menggunakan Model View Controller ( Mvc ) Dengan Framework Codeigniter Studi Kasus : Toko Miniatur. Data Manajemen Dan Teknologi Informasi (Dasi), 16(1), 23.

Ramadhani, M. F. (2015). Pembangunan Aplikasi Informasi, Pengaduan, Kritik, Dan Saran Seputar Kota Cimahi Pada Platform Android. Jurnal Ilmiah Komputer Dan Informatika (Komputa), 9.

Simarmata, A. M., & Harahap, M. (2019). Sistem Penjadwalan Iklan Menggunakan Metode Priority Schedulling Pada Pt. Kidung Indah Selaras Suara (Radio Kiss Fm) Untuk Efektivitas Dan Efisiensi Produksi Siaran. Jurnal Teknologi Dan Ilmu Komputer Prima (Jutikomp), 2(1), 66–75. Https://Doi.Org/10.34012/Jutikomp.V2i1.564

Taryana Suryana. (2007). Rational Unified Process (Rup). Rational Unified Process (Rup), 3(September), 1–6.

Wijaya, A., & Gunawan. (2018). Implementasi Algoritma Round Robin Pada Sistem Penjadwalan Mata Kuliah ( Studi Kasus : Universitas Muhammadiyah Bengkulu ). Jurnal Informatika Upgris (Jiu), 4(1), 64–71. Http://Journal.Gris.Ac.Id/Index.Php/Jiu/Article/View/2336/1885up