PLANNING

1.Identifying Busines Value

Penerapan aplikasi penggajian pada prusahaan/kantor di harapkan dapat menggantikan sistem confensional ke dalam sistem yang lebih modern. Dengan memanfaatkan teknologi masakini di harapkan penggunaan aplikasi ini dapat membantu dan mempermudah proses penggajian.

* 1. Low Cost

Dengan di terapkannya sistem ini dapat mengurangi biaya seperti pembelian ATK, mempersingkat waktu pengerjaan dan menghemat biaya jam lembur karyawan.

* 1. Incrase Cost

-Mempermudah pengerjaan penghitungan penggajian karyawan dan memangkas penggunaan kertas.

-Mempersingkat waktu pengerjaan.

2.Analyze Feasibility

Uji kelayakan pengembangan sistem penggajian karyawan adalah untuk menentukan apakah pengembangan e-learning layak dilakukan dari segi technical, economic dan organizational.

2.1 Technical Feasibility

Pengembangan sistem penggajian merupakan langkah yang tepat, karena kita bergerak di tengah perkembangan teknologi yang pesat dimana saat ini semua kegiatan perusahaan atau perkantoran menggunakan komputerisasi, seperti absensi karyawan perhitungan jam kerja karyawan dan lain lain.

Secara teknis pengembangan sistem penggajian saat ini dapat dilakukan karena teknologi sekarang mampu mengatasi kebutuhan yang diperlukan untuk proses pengembangan, begitu pula dengan SDMnya.

2.2 Economic feasibility

Penerapan sistem penggajian memiliki keuntungan secara ekonomi yang cukup signifikan, karena bisa mengurangi biaya hingga40%, seperti biaya peralatan alat tulis dan pembayaran jam lembur karyawan.

2.3 Organisasi Feasibility

Penggunaan sistem ini dalam proses penggajian PT.Gulung Tikar Maju Bersama,merupakan hal yang tepat karna dalam sebuah perusahaan memiliki karyawan yang banyak dan akan cukup sulit jika menggunakan sistem confensional/manual dan membutuhkan waktu pengerjaan yang sangat cepat.

Analysis

Analisis Prosedur yang sedang Berjalan

Prosedur merupakan langkah-langkah yang di lakukan oleh sistem  
sehingga dapat memberikan hasil berupa laporan. Di bawah ini adalah  
prosedur sistem yang sedang berjalan yang di sajikan dalam bentuk  
FlowMap, Diagram Konteks dan DFD (Data Flow Diagram).

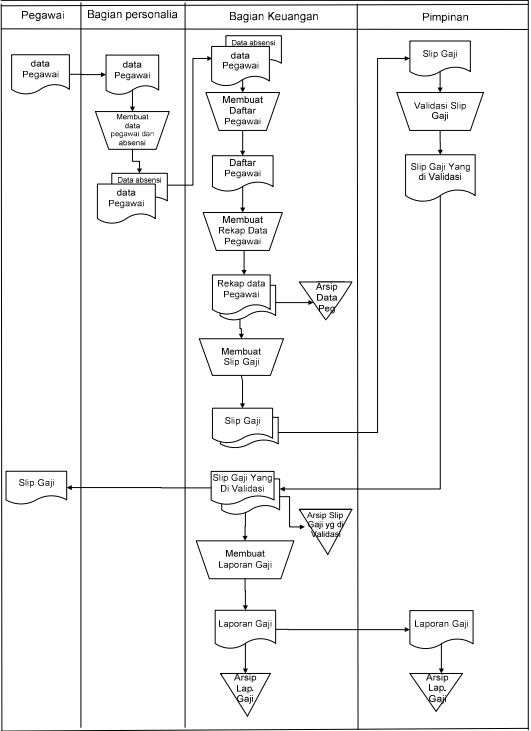
Flow Map

FlowMap merupakan diagram alir yang menggambarkan  
pergerakan proses diantara unit kerja yang berbeda-beda, sekaligus  
menggambarkan arus dari dokumen, aliran data fisik, entitas-entitas  
Sistem Informasi dan kegiatan operasi yang berhubungan dengan  
Sistem Informasi.

Penulis mencoba menggambarkan pergerakan proses Penggajian Pegawai dalam bentuk flowmap.

**Even list**

1. Pegawai memberikan data pegawai kepada bagian keuangan
2. Dibagian keuangan, data pegawai dicatat kedalam buku tentang  
   data pegawai
3. Lalu bagian keuangan membuat rekap data pegawai sebanyak 2  
   rangkap yang data didalamnya berdasarkan dokumen data  
   pegawai yang sudah di catat
4. Satu rangkap data pegawai di arsipkan dibagian keuangan dan  
   satu rangkap lagi dijadikan data untuk proses pembuatan slip  
   gaji
5. Bagian keuangan membuat slip gaji sebanyak 2 rangkap
6. Kemudian 2 rangkap slip gaji tersebut diserahkan kepada  
   pimpinan untuk disahkan
7. Setelah slip gaji disahkan, lalu 2 rangkap slip gaji pegawai  
   diserahkan kembali kebagian keuangan
8. Satu rangkap slip gaji diberikan kepada pegawai sebagai bukti  
   gaji dan satu rangkap lagi dijadikan bahan untuk membuat  
   laporan gaji
9. Laporan gaji pegawai dibuat sebanyak 2 rangkap, satu untuk  
   diarsipkan di bagian keuangan dan satu lagi diberikan kepada  
   pimpinan



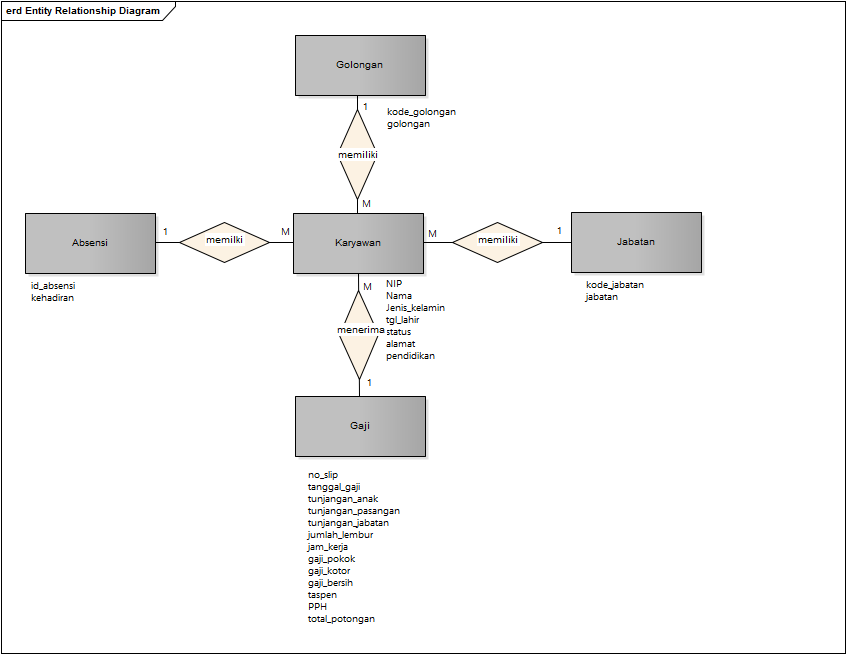
**Flow Map Sistem Yang Sedang Berjalan**

**Design**

Detail Program menggunakan UML untuk memudahkan pendeskripsian program

#### ERD

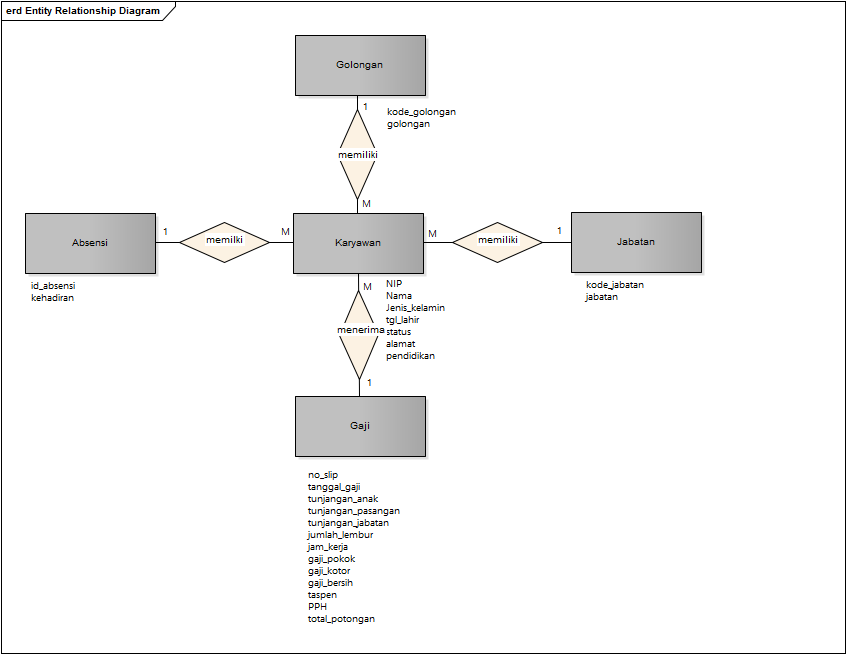
Dalam rekayasa perangkat lunak, sebuah Entity-Relationship Model (ERM) merupakan abstrak dan konseptual representasi data. Entity-Relationship adalah salah satu metode pemodelan basis data yang digunakan untuk menghasilkan skema konseptual untuk jenis/model data semantik sistem. Dimana sistem seringkali memiliki basis data relasional. Diagram untuk menggambarkan model *Entity-Relationship* ini disebut *Entity-Relationship* Diagram*,* ER-Diagram, atau *ERD.*

**

**Gambar 3.2 ERD**

#### 3.2.1.2 ERD ke LRS

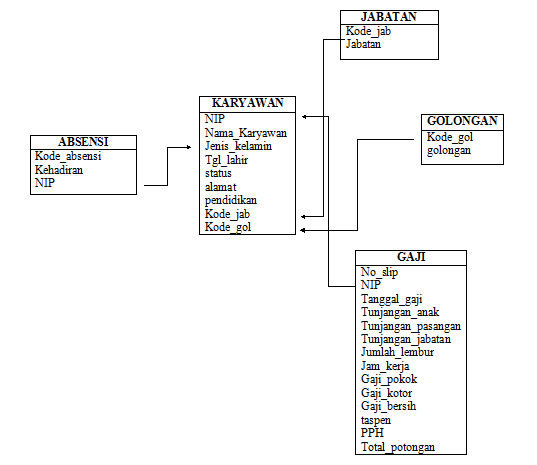
Transformasi merupakan perubahan bentuk dari suatu bentuk ke bentuk yang lain. Dalam pembahasan, kita rubah bentuk menjadi basis data fisik yang kita gunakan untuk membangun suatu data, yakni ERD (*Entity-Relationship* Diagram) ditransformasikan dalam bentuk tabel yang komponen utama pembentukan basis data. Atribut yang terdapat pada masing-masing dinyatakan sebagai *field* atau kolom dari tabel yang sesuai.

****

**Gambar 3.3 ERD KE LRS**

#### 3.2.1.3 LRS

LRS (*Logical Record Structure)* adalah representasi dari struktur *recordrecord* pada tabel-tabel yang terbentuk dari hasil relasi antar himpunan entitas.



**Gambar 3.4 LRS**

#### 3.2.1.4 Spesifikasi Basis Data

Berikut adalah tabel – tabel yang diperlukan untuk membuat aplikasi penerimaan karyawan.

1. Karyawan

Menjelaskan isi dari table karyawan diantaranya :

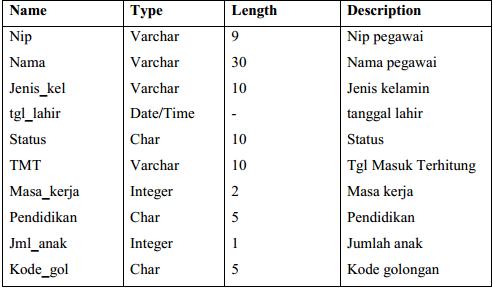
Fungsi : Menampung data karyawan

File Name : Karyawan

Content : Data Karyawan

Primary key : NIP

**Tabel 3.1 Karyawan**

****

1. Gaji

Menjelaskan isi dari table gaji diantaranya :

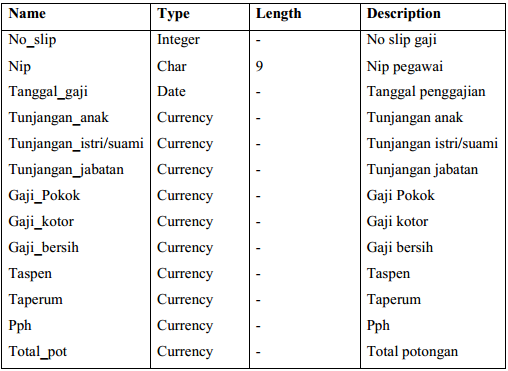
Fungsi : Menampung data gaji

*File Name* : Gaji

*Content* : Data Gaji

*Primary Key* : no\_slip

**Tabel 3.2 Gaji**

****

1. Golongan

Menjelaskan isi dari tabel golongan diantaranya :

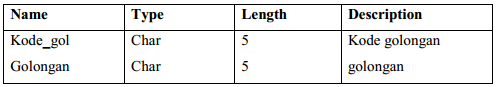
Fungsi : Menampung data golongan

File Name : Golongan

Content : Data Golongan

Primary Key : kode\_gol

**Tabel 3.3 Golongan**

****

1. Jabatan

Menjelaskan isi dari tabel jabatan diantaranya :

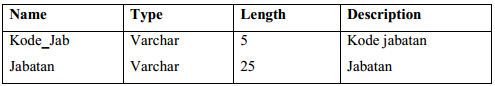
Fungsi : Menampung data jabatan

File Name : Jabatan

Content : Data Jabatan

Primary Key : kode\_jab

**Tabel 3.4 Jabatan**

****

1. Absensi

Menjelaskan isi dari tabel absensi diantaranya :

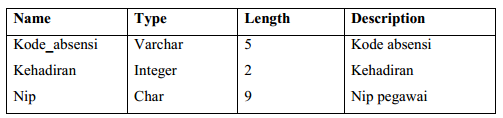
Fungsi : Menampung data absensi

File Name : Absensi

Content : Data Absensi

Primary Key : kode\_absensi

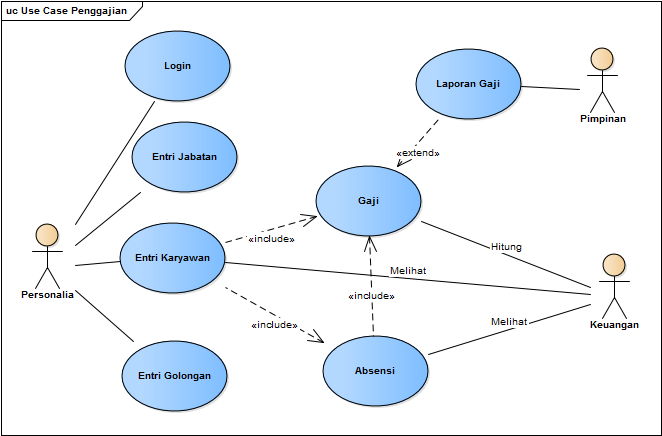
**Tabel 3.5 Absensi**

****

* + 1. Perancangan Aplikasi

#### Use Case Diagram

Dalam pemodelan dengan menggunakan UML, semua perilaku dimodelkan sebagai *use case* yang mungkin dispesifikasikan mandiri dari realisasinya. *Use case* mendeskripsikan kumpulan urutan (*sequence*) dimana setiap urutan menjelaskan interaksi sistem dengan sesuatu diluar sistem (sering dinamakan *actor*). *Use case* menampilkan spesifikasi fungsional yang diharapkan dari sistem/perangkat lunak yang kelak akan kita kembangkan. *Use case* sangat penting dimanfaatkan untuk menangkap seluruh kebutuhan dan harapan pengguna (*user needs and expectation*).

****

**Gambar 3.5 Use Case Penggajian**

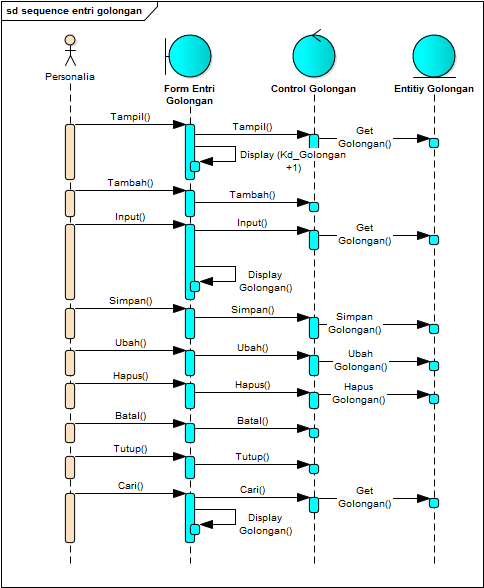
Deskripsi dari *use case* yang digambarkan diatas dapat dilihat pada tabel berikut :

**Tabel 3.6 Deskripsi *Use Case Diagram***

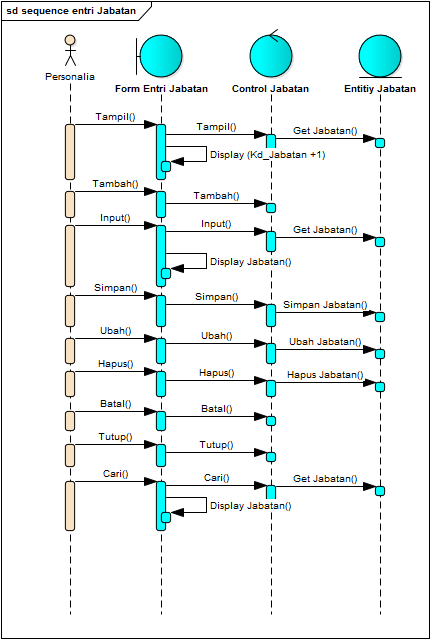
|  |  |  |
| --- | --- | --- |
| Aktor | Nama *Use Case* | Deskripsi *Use Case* |
| Personalia | Login | *Use Case* ini berfungsi untuk verifikasi data login |
| Personalia | Entri Jabatan | *Use Case* ini berfungsi untuk melakukan pengelolaan dalam tambah, edit dan hapus jabatan. |
| Personalia | Entri Karyawan | *Use Case* ini berfungsi untuk melakukan pengelolaan dalam tambah, edit dan hapus karyawan. |
| Personalia | Entri Golongan | *Use Case* ini berfungsi untuk melakukan pengelolaan dalam tambah, edit dan hapus golongan. |
| Keuangan | Entri Gaji | *Use Case* ini berfungsi untuk melakukan pengelolaan dalam tambah, edit dan hapus gaji. |
| Keuangan | Lihat Absensi | *Use Case* ini berfungsi untuk melihat data absensi |
| Pimpinan | Lihat Laporan Gaji | *Use Case* ini berfungsi untuk melihat data Laporan Gaji |

#### 3.2.2.2 Sequence Diagram

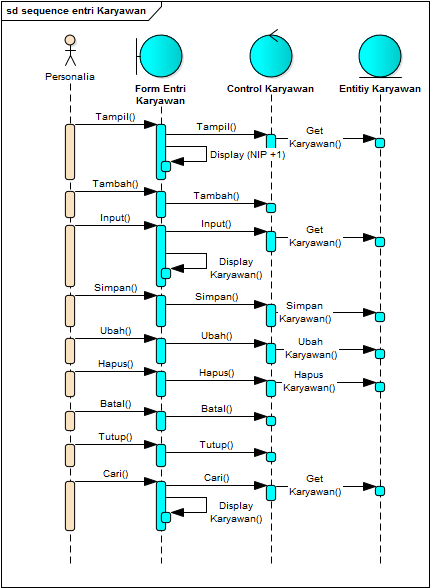
*Sequence* Diagram adalah suatu diagram yang menggambarkan interaksi antar obyek dan mengindikasikasn komunikasi diantara obyek-obyek tersebut. Diagram ini juga menunjukkan serangkaian pesan yang dipertukarkan oleh obyek-obyek yang melakukan suaatu tugas atau aksi tertentu. Bagian paling atas dari diagram menjadi titik awal dan waktu berjalan ke bawah sampai dengan bagian dasar dari diagram garis vertical (*life line*), dilekatkan pada setiap obyek atau *actor*. Kemudian *life line* tersebut digambarkan menjadi kotak ketika obyek ,elakukan suatu operasi, kotak tersebut disebut *activation.* Obyek dikatakan mempunyai *life activation* pada saat tersebut.

****

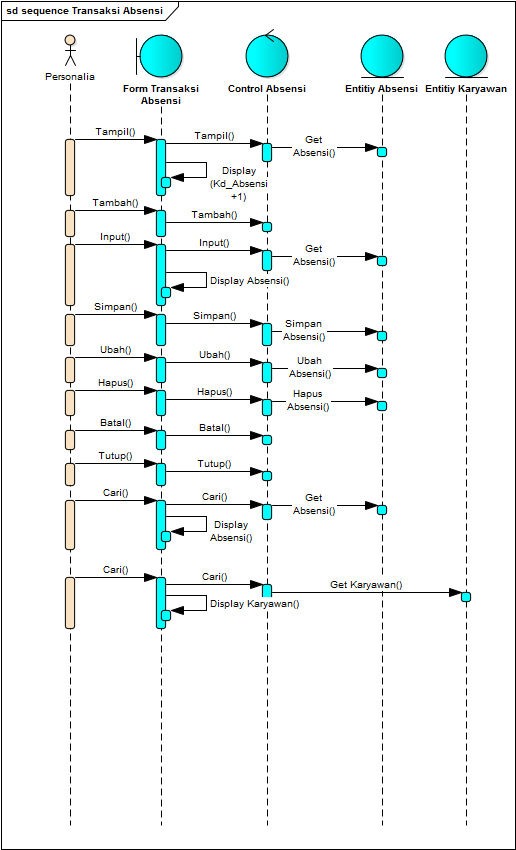
**Gambar 3.6 *Sequence Diagram Golongan***

****

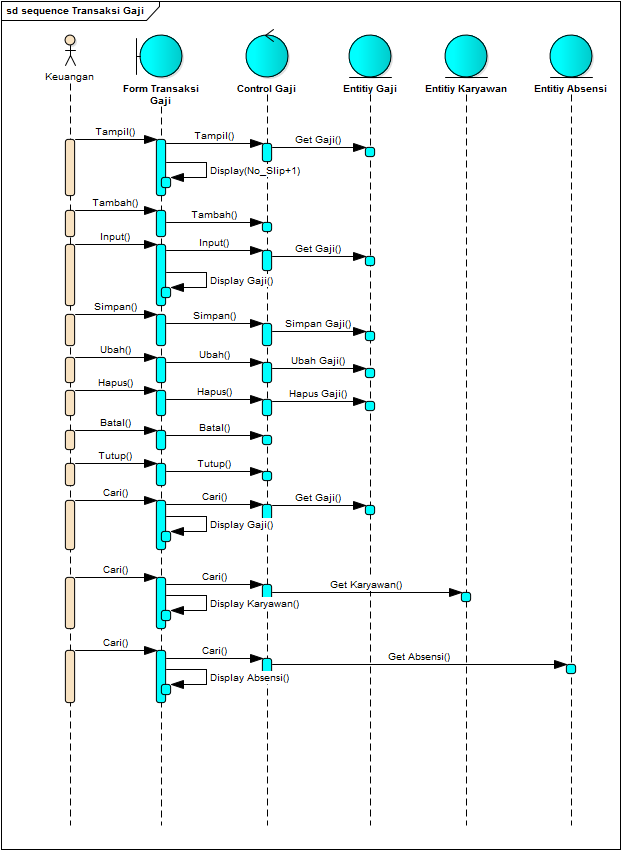
**Gambar 3.7 *Sequence Diagram Jabatan***

****

**Gambar 3.8 *Sequence Diagram Karyawan***

****

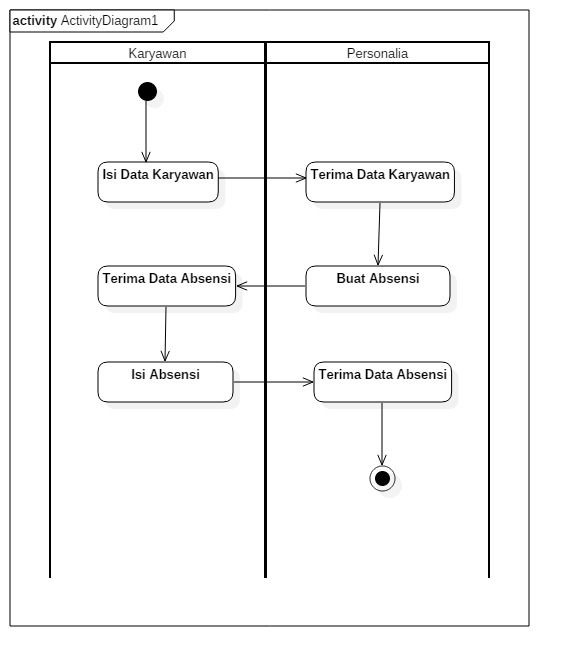
Gambar 3.9 Sequence Diagram Absensi

****

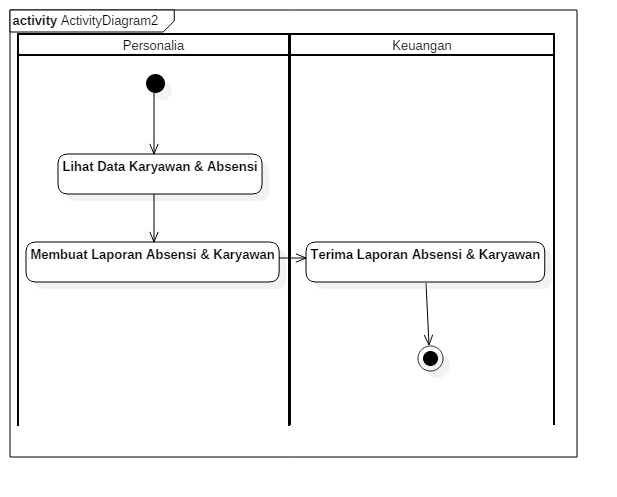
Gambar 3.10 Sequence Diagram Gaji

#### *3.2.2.3 Activity Diagram*

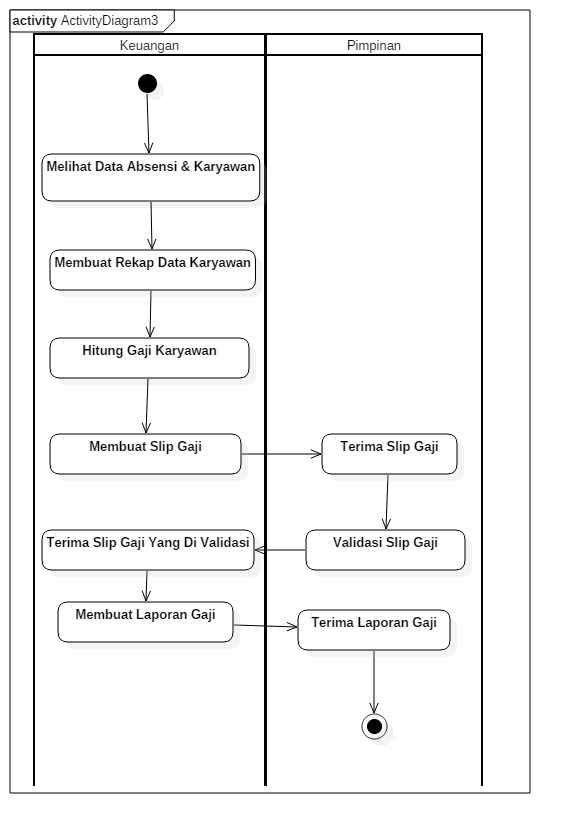
*Activity Diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana berakhir.

****

Gambar 3.11 Activity Diagram Absensi

****

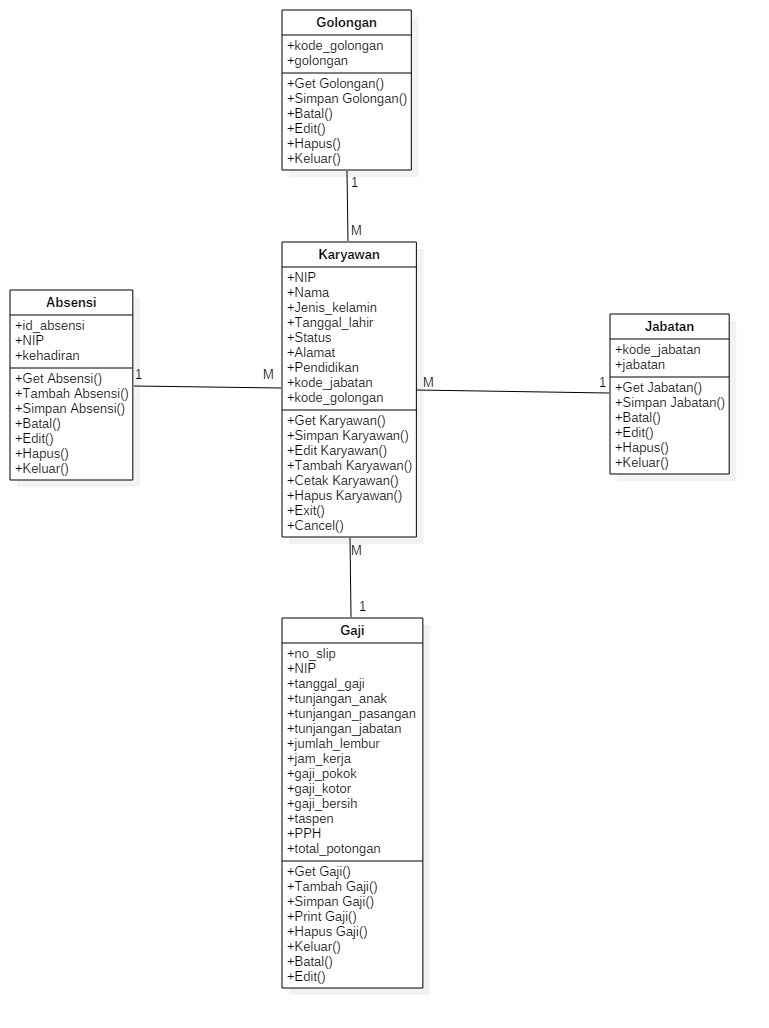
Gambar 3.12 Activity Diagram Tambah Data Karyawan dan Absensi

****

Gambar 3.13 Activity Diagram Gaji

#### 3.2.2.4 *Class Diagram*

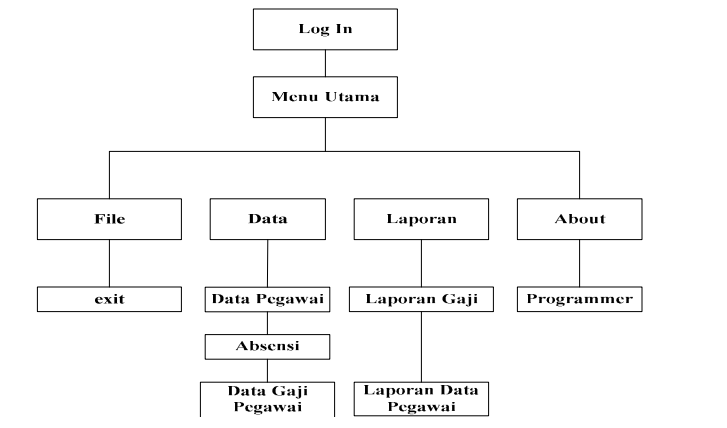
*Class Diagram* digunakan untuk menampilkan kelas-kelas dan paketpaket di dalam sistem. *Class Diagram* memberikan gambaran sistem secara statis san relasi antar mereka. Biasanya, beberapa *class* diagram untuk sistem tunggal. Beberapa diagram akan menampilkan subset dari kelals-kelas dan relasinya. Dapat dibuat beberapa diagram sesuai dengan yang diinginkan untuk mendapat gambaran lengkap terhadap sistem yang dibangun.

****

Gambar 3.14 Class Diagram

#### 3.2.2.4 *User Interface*

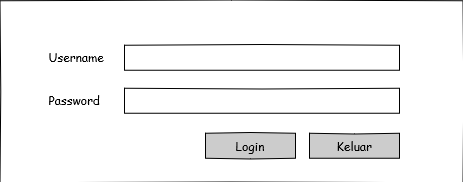
Struktur Tampilan



Gambar 3.15 Struktur Tampilan

1. **Halaman Login**

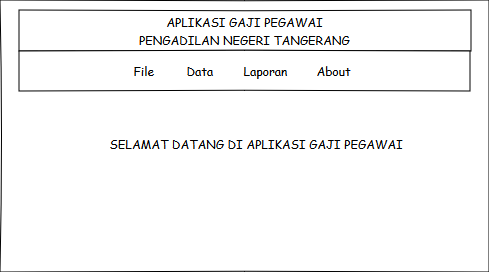
Perancangan login merupakan tampilan pertama kali muncul ketika  
program aplikasi penggajian pertama kali dijalankan. Aplikasi  
Login ini dirancang untuk pengamanan setiap aplikasi. Berikut  
tampilan perancangan Login.



Gambar 3.15 Rancangan Halaman Login

1. **Halaman Menu Utama**

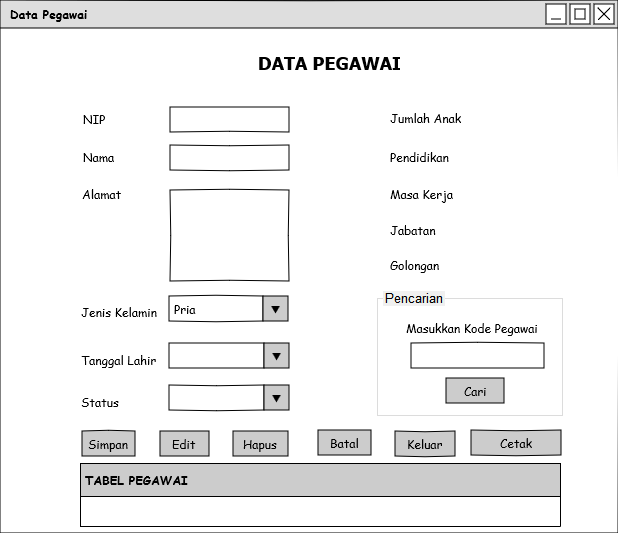
Halaman ini yang pertama akan dituju oleh user ketika login. Disini juga user dapat melihat menu utama.



Gambar 3.16 Rancangan Menu Utama

1. **Halaman Data Pegawai**

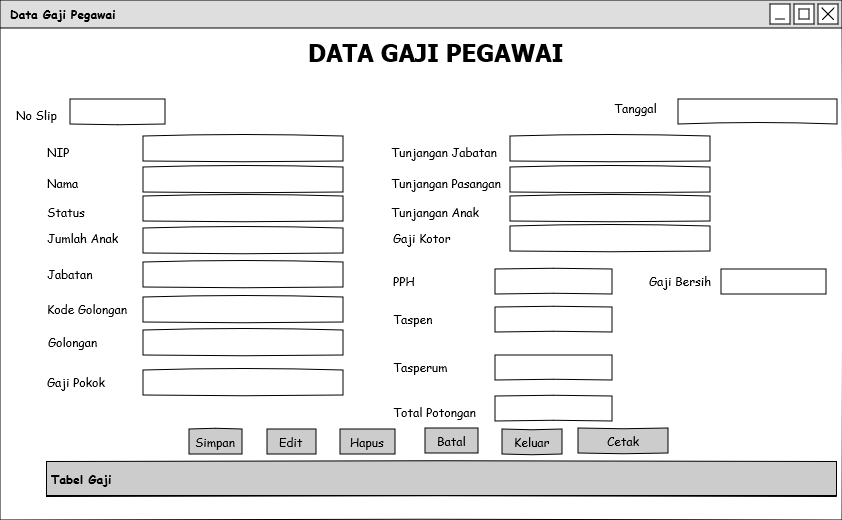
Perancangan form data pegawai merupakan rancangan yang dibuat  
untuk melakukan proses penginputan data pegawai. Berikut adalah  
rancangan form data pegawai.



Gambar 3.17 Rancangan Form Pegawai

1. **Halaman Form Gaji Pegawai**

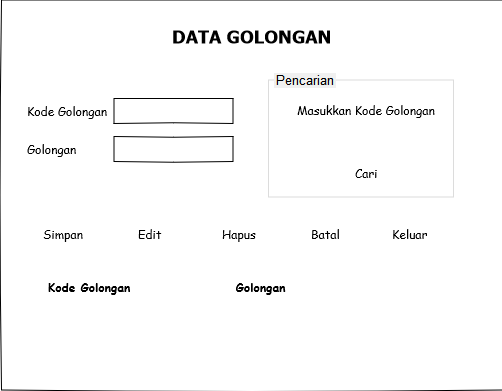
Perancangan form gaji pegawai merupakan rancangan yang dibuat  
untuk melakukan proses penghitungan gaji pegawai. Berikut adalah  
rancangan form gaji pegawai.



Gambar 3.18 Rancangan Form Gaji Pegawai

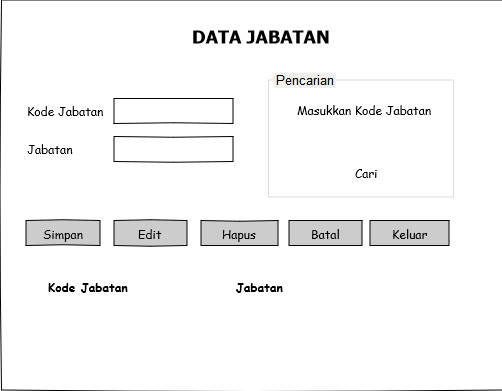
1. **Tampilan Form Golongan**

Perancangan form golongan dibuat untuk mengetahui daftar  
golongan yang ada. Berikut perancangannya.



Gambar 3.19 Rancangan Form Golongan

1. **Tampilan Form Jabatan**

****

Gambar 3.20 Rancangan Form Jabatan