**SKRIPSI**

**SISTEM INFORMASI PENGOLAHAN DATA NASABAH BANK SAMPAH BERBASIS WEB**



Oleh:

Bagas Kurniawan

2016140234

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS PAMULANG**

**PAMULANG**

**2020**

# KATA PENGANTAR

Segala puji dan syukur penulis panjatkan kehadirat Allah SWT yang telah memberikan rahmat serta hidayahnya sehingga penulis dapat menyelesaikan penulisan ilmiah yang berjudul **“SISTEM INFORMASI PENGOLAHAN DATA NASABAH BANK SAMPAH BERBASIS *WEB* ”.** Sholawat serta salam tak lupa penulis haturkan kepada suri tauladan, manusia paling sempurna di alam semesta Nabi Muhammad SAW. Kepada keluarga, sahabat, dan para pengikutnya.

Penulis menyadari bahwa dalam penyelesaian skripsi ini, tentunya tidak terlepas dari pertolongan dan dukungan pihak lain. Maka dari itu, penulis ingin menyampaikan terima kasih sebesar – besarnya kepada:

1. Bapak Dr. (H.C.), H. Darsono, selaku Ketua Yayasan Sasmita Jaya Group
2. Bapak Dr. H. Dayat Hidayat, M.M., selaku Rektor Universitas Pamulang.
3. Bapak DR. Ir. Sewaka.,M.M. selaku Dekan Fakultas Teknik dan Ketua Program Studi Teknik Informatika Universitas Pamulang.
4. Kedua Orang tua beserta Adik yang telah memberikan doa dan dukungan selama proses pembuatan skripsi.
5. Seluruh Dosen Teknik Informatika Universitas Pamulang yang telah memberikan ilmunya kepada penulis selama menempuh pendidikan di Universitas Pamulang.

Pamulang, Februari 2020

Bagas Kurniawan

# DAFTAR ISI

[KATA PENGANTAR 2](#_Toc53749540)

[DAFTAR ISI 3](#_Toc53749541)

[BAB 1 1](#_Toc53749542)

[PENDAHULUAN 1](#_Toc53749543)

[1.1. Latar Belakang 1](#_Toc53749544)

[1.2. Identifikasi Masalah 2](#_Toc53749545)

[1.3. Rumusan Masalah 2](#_Toc53749546)

[1.4. Tujuan Penelitian 3](#_Toc53749547)

[1.5. Batasan Masalah 3](#_Toc53749548)

[1.6. Manfaat Penelitian 3](#_Toc53749549)

[1.7 Metodologi Penelitian 3](#_Toc53749550)

[1.7.1. Metode Pengumpulan Data 4](#_Toc53749551)

[1.7.2. Metode Analisis Data 4](#_Toc53749552)

[1.7.3. Metode Pengukuran Penelitian 4](#_Toc53749553)

[1.8. Sistematika Penulisan 4](#_Toc53749554)

[BAB II 6](#_Toc53749555)

[LANDASAN TEORI 6](#_Toc53749556)

[2.1. Tinjauan Studi 6](#_Toc53749557)

[2.3. Sistem Informasi 8](#_Toc53749558)

[2.3.1. Pengolahan Data Nasabah 9](#_Toc53749559)

[2.3.2 Pengertian Bank Sampah 9](#_Toc53749560)

[2.3.3. Klasifikasi Sampah 10](#_Toc53749561)

[2.4. Unifield Modeling Language (UML) 11](#_Toc53749562)

[2.4.1. Use Case Diagram 12](#_Toc53749563)

[2.4.2. Activity Diagram 13](#_Toc53749564)

[2.4.3. Sequence Diagram 15](#_Toc53749565)

[2.5. Percancagan Basis Data 16](#_Toc53749566)

[2.5.1. Basis Data 16](#_Toc53749567)

[2.5.2. Database Management Sistem (DBMS) 16](#_Toc53749568)

[2.5.3. Normalisasi 17](#_Toc53749569)

[2.5.4. Entity Relationship Diagram (ERD) 18](#_Toc53749570)

[2.5.5. Kardinalitas/Derajat Kardinalitas 19](#_Toc53749571)

[2.5.6. Transformasi 20](#_Toc53749572)

[2.5.7. Logical Record Structure (LRS) 21](#_Toc53749573)

[2.6. Bahasa Pemrograman 22](#_Toc53749574)

[2.6.1 HTML 22](#_Toc53749575)

[2.6.2. PHP 23](#_Toc53749576)

[2.6.3. MySQL 24](#_Toc53749577)

[2.6.4. XAMPP 25](#_Toc53749578)

[2.6.5. Code Igneter 25](#_Toc53749579)

[2.7. System Development Life Cycle (SDLC) 26](#_Toc53749580)

[2.8. Model Prototype 28](#_Toc53749581)

[2.9. Pengujian Perangkat Lunak 29](#_Toc53749582)

[2.9.1. Black Box Testing 29](#_Toc53749583)

[BAB III 31](#_Toc53749584)

[ANALISA DAN PEMBAHASAN 31](#_Toc53749585)

[3.1. Analisa Sistem Berjalan 31](#_Toc53749586)

[3.2. Analisa Sistem Usulan 32](#_Toc53749587)

[3.2.1. Use Case 32](#_Toc53749588)

[3.2.2. Activity Diagram 33](#_Toc53749589)

[3.2.3. Sequence Diagram 38](#_Toc53749590)

[3.2.4. Class Diagram 43](#_Toc53749591)

[3.3. Rancangan Data 44](#_Toc53749592)

[3.3.1. *Entity* *Relationship* *Diagram* (ERD) 44](#_Toc53749593)

[3.3.2. *Transformasi* ERD ke LRS 45](#_Toc53749594)

[3.3.3. *Logical* *Record* *Structure* (LRS) 45](#_Toc53749595)

[3.3.4. Spesifikasi Tabel 46](#_Toc53749596)

[3.4. Rancangan Tampilan 46](#_Toc53749597)

BAB I

# PENDAHULUAN

## 1.1. Latar Belakang

Sampah merupakan suatu bahan yang terbuang atau dibuang dari sumber hasil aktivitas manusia maupun proses alam yang belum memiliki nilai ekonomi. Masalah sampah timbul dengan adanya peningkatan timbunan sampah  per hari, namun tak di imbangi dengan dukungan sarana dan prasarana penunjang yang memenuhi persyaratan teknis, sehingga banyak sampah yang tidak ditangani dengan maksimal. Masalah lingkungan sekarang ini bukan hanya tanggung jawab sekelompok orang, tetapi sudah menjadi tugas dan kewajiban semua manusia untuk menjaga dan memelihara nya agar tetap asri.

Di Sawah Baru, Ciputat, Tangerang Selatan, terdapat Bank Sampah yang bergerak dalam bidang pengolahan sampah. Melihat dari permasalahan yang mengakibatkan pencemaran lingkungan sekitar. Masih adanya masyarakat yang membuang sampah bukan pada tempatnya di sungai/saluran yang menyebabkan lingkungan menjadi kotor, timbulnya berbagai macam penyakit, pencemaran lingkungan dan rusaknya ekosistem.

Bank Sampah disini mengelola sampah yang berjenis (non-organik) seperti plastik, kertas, botol dsb. Sebelum sampah yang disetor warga, warga diberikan rincian daftar harga sampah per barang. Sampah lalu dikumpulkan digudang penyimpanan dan ditabung sehingga dapat menghasilkan nilai ekonomis untuk nasabah. Untuk mendaftar menjadi nasabah, masyarakat hanya perlu KTP dan No Telpon yang dapat dihubungi, Penabung pun disebut dengan nasabah mereka memiliki buku tabungan yang nantinya dapat diambil. Nama tempat itu adalah Bank Sampah Jalan Cendrawasih Raya (JAWARA) yang terletak di Sawah Baru, Ciputat, Tangerang Selatan.

Dari segi keseimbangan lingkungan, kesehatan dan kebersihan apabila sampah tidak dikelola dengan baik dapat menimbulkan berbagai gangguan-gangguan antara lain pencemaran udara, pencemaran air dan pencemaran tanah. Berdirinya bank sampah karena adanya permasalahan lingkungan. Bank Sampah tersebut mengumpulkan sampah dari masyarakat (nasabah) maupun warga luar yang ingin menabung sampah nya di bank sampah yang nantinya akan di timbang tempat gudang.

Maka dari itu, penulis tertarik untuk membuat **Sistem Informasi Pengolahan Data Nasabah Bank Sampah Berbasis Web** yang diharapkan dapat memberikan proses pengolahan data secara cepat, tepat dan efisien serta mampu memberikan hasil laporan yang sesuai secara tepat dan tepat.

## 1.2. Identifikasi Masalah

Berdasarkan dari latar belakang di atas, dapat di identifikasi beberapa permasalahan sebagai berikut:

1. Nasabah masih kesulitan mengakses informasi tabungan sampah yang ada di bank sampah.
2. Admin bank sampah kesulitan dalam mencari data nasabah serta pengolahan data yang masih manual.
3. Admin bank sampah masih kesulitan dalam dalam pembuatan laporan hasil kegiatan.

## 1.3. Rumusan Masalah

Berdasarkan dari latar belakang di atas, maka perumusan masalahnya adalah sebagai berikut:

1. Bagaimana merancang system bank sampah agar masyarakat dapat mengakses informasi tabungan sampah.
2. Bagaimana memudahkan admin bank sampah dalam mencari dan mengelola data nasabah di bank sampah.
3. Bagaimana memudahkan admin bank sampah dalam pembuatan hasil laporan kegiatan.

## 1.4. Tujuan Penelitian

Tujuan penelitian ini dengan diharapkan mempermudah serta mempercepat pengolahan data, mampu mencari informasi yang lengkap dan informasi lainnya dari data tetap yang dimiliki, dan dengan adanya penelitian mampu menampilkan data secara cepat dan akurat. Sesuai dengan permasalahan yang diuraikan diatas, maka tujuan dari penelitian adalah sebagai berikut:

1. Untuk mempermudah masyarakat dalam mengakses informasi tentang tabungan sampah mereka
2. Untuk mempermudah admin bank sampah dalam pengelolaan data tabungan sampah masyarakat.
3. Untuk mempermudah admin bank sampah dalam hasil pembuatan laporan hasil kegiatan

## 1.5. Batasan Masalah

Agar pembahasan penelitian ini lebih terarah, berikut batasan masalah pada penelitian ini:

1. Perancangan rekayasa perangkat Sistem Informasi Pengolahan Data Nasabah Bank Sampah Berbasis Web
2. Penelitian ini digunakan untuk mempermudah dalam mengelola data nasabah serta mempercepat dalam mencari informasi yang tepat dan akurat.

## 1.6. Manfaat Penelitian

Manfaat penelitian ini dengan diharapkan mempermudah pengolahan dan pencarian data tetap yang dimiliki serta mempermudah mendapatkan informasi yang detail terhadap data tetap yang dimiliki dan informasi lain nya.

## 1.7 Metodologi Penelitian

Berikut metodologi penelitian yang digunakan:

### 1.7.1. Metode Pengumpulan Data

1. Wawancara

Mempelajari dan menganalisis sitem yang sedang berjalan serta mendapatkan data secara langsung dari sumbernya dengan mengajukan pertanyaan kepada pengurus bank sampah

1. Observasi

Mempelajari proses dari data diterima dan diproses serta menemukan permasalahan yang ada pada siklus tersebut untuk kemudian dicarikan solusinya.

### 1.7.2. Metode Analisis Data

Analisa data yang penulis gunakan adalah analisis data kualitatif, yaitu suatu analisa yang didasari atas data yang diperoleh, kemudian diproses menjadi sebuah kesimpulan.

### 1.7.3. Metode Pengukuran Penelitian

Penulis menerapkan pengujian dengan metode *black box testing,* karena untuk *white box testing* sudah dilakukan pada saat pengkodingan perangkat lunak.

## 1.8. Sistematika Penulisan

Guna mempermudah dalam memahami bahasan-bahasan setiap bab dan memberikan gambaran yang jelas terhadap materi yang disampaikan, maka dalam sub bab ini, penulis akan memperjelas pembahasan dalam bentuk pembahasan sistematik dan menguraikan pokok bahasan sebagai berikut:

**BAB I PENDAHULUAN**

Bab ini mendeskripsikan gambaran umum yang meliputi latar belakang, identifikasi masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metode penelitian dan sistematika penulisan.

**BAB II LANDASAN TEORI**

Bab ini menjelaskan tentang tinjauan studi dari beberapa penelitian dan tinjauan pustaka yang menguraikan teori-teori yang berhubungan dengan materi penelitian.

**BAB III ANALISIS DAN PERANCANGAN SISTEM**

Bab ini berisi tentang analisis serta desain perancangan rekayasa perangkat lunak, dimulai dari analisis kebutuhan, fasilitas, merancang basis data dan *design* antar muka.

**BAB II**

# LANDASAN TEORI

## 2.1. Tinjauan Studi

Penelitian ini memiliki tujuan mempermudah mengelola data dengan lebih efektif dan efisien bagi lembaga bank sampah. Dengan system informasi yang berbasis web juga memberikan kemudahan bagi masyarakat umum mengetahui informasi tentang bank sampah yang ada di Tangerang Selatan. Selain itu pihak kecamatan bahkan pihak pemerintah kabupaten Banten dapat mudah mengawasi perkembangan bank sampah.

Beberapa penelitian tentang pembangunan system informasi bank sampah telah dilakukan seperti pembangunan bank sampah Ceria di Purwokerto (Riyanto & Kusumastuti, 2015) dimana hanya admin yang melakukan pengolahan data bank sampah. Sistem informasi bank sampah Bangkit yang berbasis web ada dua pengguna yaitu admin dan nasabah (Kusuma & Astuti, 2017)

System diartikan sebagai sekelompok komponen yang saling berhubungan, bekerja sama untuk mencapai tujuan bersama dengan menerima input serta menghasilkan output dalam proses transformasi yang teratur. (Agus Mulyanto 2009: )

Pengertian system menurut Jerry Fith Gerald, system adalah suatu jaringan kerja dari prosedur- prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau menyelesaikan suatu sasaran tertentu. (Agus Mulyanto 2009: )

System Informasi merupakan suatu komponen yang terdiri dari manusia, teknologi informasi, dan prosedur kerja yang memproses, menyimpan, menganalisis, dan penyebaran informasi untuk mencapai suatu tujuan. (Agus Mulyanto 2009:29 )

Tujuan dari penelitian ini adalah merancang system informasi yang lengkap dalam pengolahan data nasabah sekarang karena masih menggunakan cara manual dengan menuliskannya dalam list buku. Dengan dibuatnya system informasi pengolahan data nasabah tersebut mampu mempermudah dalam pencarian data tetap serta mempercepat dalam menyediakan laporan data tetap setiap dibutuhkan.

Sehingga dapat disimpulkan dengan adanya system informasi pengolahan data nasabah bank sampah yang menggunakan teknologi membantu organisasi bank sampah yang berada pada bagian administrator atau staff untuk mempermudah mencatat, mengelola serta membuatkan laporan data tetap tersebut menjadi mudah.

**2.2. Tinjauan Pustaka**

Sampah adalah sesuatu bahan atau benda padat yang sudah tidak dipakai lagi oleh manusia, atau benda padat yang sudah tidak digunakan lagi dalam kegiatan manusia dan dibuang. Para ahli kesehatan Amerika membuat batasan sampah (*waste*) adalah sesuatu yang tidak digunakan, tidak dipakai, tidak disenangi, atau sesuatu yang dibuang yang berasal dari kegiatan manusia, dan tidak terjadi dengan sendirinya (Notoatmodjo, 2011: 190).

**2.2.1. Pengertian Sampah**

Sampah ialah suatu bahan yang terbuang atau dibuang, merupakan hasil aktivitas manusia maupun alam yang sudah tidak digunakan lagi karena sudah diambil unsur atau fungsi utamanya. Pengelolaan sampah menurut sumbernya yang ada di permukaan bumi ini dapat berasal dari beberapa sumber berikut ini:

1. Pemukiman penduduk

Sampah disuatu pemukiman biasanya dihasilkan oleh satu atau beberapa keluarga yang tinggal dalam suatu bangunan atau asrama yang terdapat di desa atau di kota. Jenis sampah yang dihasilkan biasanya sisa makanan dan bahan sisa proses pengolahan makanan atau sampah basah (*garbage*), sampah kering (*rubbish*), abu, atau sampah sisa tumbuhan.

1. Tempat umum dan tempat perdagangan

Tempat umum adalah tempat yang memungkinkan banyak orang berkumpul dan melakukan kegiatan, termasuk juga tempat perdagangan. Jenis sampah yang dihasilkan dari tempat semacam itu berupa sisa-sisa makanan (*garbage*), sampah kering, abu, sisa-sisa bahan bangunan, sampah khusus dan terkadang sampah berbahaya

## 2.3. Sistem Informasi

Sistem informasi terdiri dari dua kata, yakni sistem dan informasi. Menurut (Romney dan Steinbart, 2015:3) sistem adalah rangkaian dari dua atau lebih komponen-komponen yang saling berhubungan, yang berinteraksi untuk mencapai suatu tujuan. Sebagian besar sistem terdiridari subsistem yang lebih kecil yang mendukung sistem yang lebih besar. Sistem mempunyai karakteristik atau sifat – sifat tertentu, yaitu : Komponen Sistem, Batasan Sistem, Lingkungan Luar Sistem, Penghubung Sistem, Masukan Sistem, Keluaran Sistem, Pengolahan Sistem dan Sasaran Sistem

Masih menurut (Rommey dan Steinbart, 2015:4) informasi adalah data yang telah dikelola dan di proses untuk memberikan arti dan memperbaiki proses pengambilan keputusan. Sedangkan (Krismaji, 2015:14) informasi adalah “data yang telah diorganisasi dan telah memiliki kegunaan dan manfaat”

Pengertian menurut (Krismaji, 2015:15) sistem informasi adalah cara-cara yang diorganisasi untuk mengumpulkan, memasukkan, dan mengolah serta menyimpan data, dan cara-cara yang diorganisasi untuk menyimpan, mengelola, mengendalikan, dan melaporkan informasi sedemikian rupa sehingga sebuah organisasi dapat mencapai tujuan yang telah ditetapkan.

Secara umum sistem informasi dapat didefinisikan sebagai suatu sistem di dalam suatu organisasi yang merupakan kombinasi dari orang-orang, fasilitas, teknologi, media prosedur-prosedur dan pengendalian yang ditujukan untuk mendapatkan jalur komunikasi penting, memproses tipe transaksi rutin tertentu, memberi sinyal kepada manajemen dan yang lainnya terhadap kejadian-kejadian internal dan eksternal yang penting dan menyediakan suatu dasar informasi untuk pengambilan keputusan.

### 2.3.1. Pengolahan Data Nasabah

Data yang telah diolah akan berubah menjadi sebuah informasi yang dapat digunakan untuk menambah pengetahuan bagi yang menerimanya. Maka dalam hal ini data dapat dianggap sebagai obyek dan informasi adalah suatu subyek yang bermanfaat bagi penerimanya. Informasi juga bias disebut sebagai hasil pengolahan atau pemrosesan data. Bedasarkan jenisnya data dikelompokan menjadi dua macam yaitu, data kualitatif dan data kuantitatif.

1. Data Kualitatif

Merupakan data yang menunjukan mutu atau kualitas yang ada, baik proses, keadaan, peristiwa, kejadian dan lainnya yang dinyatakan ke dalam bentuk pertanyaan atau berupa kata-kata.

1. Data Kuantitatif

Merupakan data yang berbentuk angka-angka sebagai hasil pengukuran ataupun hasil observasi. Data kuantitatif di dapatkan dari pengukuran langusng dan dari angka-angka yang diperoleh dengan mengubah data kualitatif menjadi data kuantitatif yang mempunyai sifat objektif dan dapat ditafsirkan oleh semua orang.

### 2.3.2 Pengertian Bank Sampah

Bank sampah adalah suatu tempat yang digunakan untuk mengumpulkan sampah yang sudah dipilah – pilah. Hasil dari pengumpulan sampah yang sudah dipilah akan disetorkan ke tempat pembuatan kerajinan dari sampah atau ke tempat pengepul sampah. Bank sampah dikelola menggunakan sistem seperti perbankan. Penyetor adalah warga yang tinggal disekitar lokasi bank sampah serta mendapat buku tabungan seperti menabung di bank (Rozak, 2014).

Bank sampah adalah tempat menabung sampah yang telah terpilah menurut jenis sampah. Sampah yang ditabung pada bank sampah adalah sampah yang mempunyai nilai ekonomis. Cara kerja bank sampah pada umumnya hampir sama dengan bank lainnya, ada nasabah, pencatatan pembukuan dan manajemen pengelolaannya, apabila dalam bank yang biasa kita kenal yang disetorkan nasabah adalah uang akan tetapi dalam bank sampah yang disetorkan adalah sampah yang mempunyai nilai ekonomis, sedangkan pengelola bank sampah harus orang yang kreatif dan inovatif serta memiliki jiwa kewirausahaan agar dapat meningkatkan pendapatan masyarakat. Sistem kerja bank sampah pengelolaan sampahnya berbasis rumah tangga, dengan memberikan penghargaan kepada yang berhasil memilah dan menyetorkan sejumlah sampah. Konsep bank sampah mengadopsi menajemen bank pada umumnya. Selain bisa sebagai sarana untuk melakukan gerakan penghijauan, pengelolaan sampah juga bisa menjadi sarana pendidikan gemar menabung untuk masyarakat dan anak-anak. Metode bank sampah juga berfungsi untuk memberdayakan masyarakat agar peduli terhadap kebersihan (Novianty, 2012).

### 2.3.3. Klasifikasi Sampah

1. Berdasarkan sifat

Berdasarkan sifatnya sampah dapat digolongkan sebagai berikut :

1. Sampah organik-dapat diurai (degradable)

Sampah organic yaitu sampah yang mudah membusuk seperti sisa makanan, sayuran, daun kering dan sebagainya. Sampah ini dapat diolah lebih lanjut menjadi kompos.

1. Sampah anongarnik-tidak terurai (undegradable)

Sampah anongarnik yaitu sampah yang tidak mudah membusuk, seperti plastic wadah pembungkus makanan, kertas, plastic, botol dan gelas minuman, kaleng, kayu, dan sebagainya.

1. Berdasarkan sumber

Menurut sumbernya sampah dapat digolongkan sebagai berikut :

1. Sampah alam
2. Sampah manusia
3. Sampah konsumsi
4. Sampah nuklir
5. Sampah industri
6. Sampah pertambangan (Notoatmodjo, 2011:194).

**2.3.4. Kinerja bank sampah**

Kinerja (prestasi kerja) adalah suatu hasil karya yang dicapai oleh seseorang dalam melaksanakan tugas-tugas yang dibebankan kepadanya yang didasarkan atas kecakapan, pengalaman dan kesungguhan waktu yang diukur dengan mempertimbangkan kuantitas, kualitas dan ketepatan waktu. kinerja (prestasi kerja) dapat diukur melalui pengukuran tertentu (standar) dimana kualitas adalah berkaitan dengan mutu kerja yang dihasilkan, sedangkan kwantitas adalah jumlah hasil kerja yang dihasilkan dalam kurun waktu tertentu, dan ketepatan waktu adalah kesesuaian waktu yang telah direncanakan. Karakteristik yang membedakan kinerja auditor dengan kinerja manajer adalah pada output yang dihasilkan (Trisnaningsih, 2007).

Menurut Indra Bastian, kinerja adalah suatu gambaran mengenai tingkat pencapaian pelaksanaan suatu kegiatan/program/kebijaksanaan dalam mewujudkan sasaran, tujuan, misi dan visi organisasi yang tertuang dalam perumusan skema strategi (strategic planning) suatu organisasi (Fahmi, 2013).

## 2.4. Unifield Modeling Language (UML)

*Unifield Modeling Language* merupakan bahasa yang mendefinisikan notasi dan syntax/semantik. Notasi *Unifield Modeling Language* menggambarkan berbagai *diagram* perangkat lunak dan syntax *Unifield Modeling Language* mendefinisikan bagaimana mengkombinasikan bentuk-bentuk tersebut.

Menurut (Muslihudin dan Oktafianto, 2016:59), *Unified Modeling Language* yang berarti bahasa pemodelan standar. setiap model-model yang dibuat harus saling berhubungan satu dengan lainnya dengan mengikuti standar yang ada, sehingga menceritakan konteks yang ingin dibuat. *Unified Modeling Language* diaplikasikan untuk:

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisis dan mencari apa yang diperlukan sistem.
4. Mendokumentasi sistem yang ada, proses-proses dan organisasinya.

### 2.4.1. Use Case Diagram

*Use* *case* *diagram* mendukung penyusunan requirement sebuah sistem, proses komunikasi saat perancangan dengan klien, dan perancangan test *case* semua feature pada sistem.

Menurut (Mulyani, 2016:48), use case model adalah sekumpulan *diagram* yang digunakan untuk menggambarkan sistem kedalam notasi grafis. Use case model lebih terfokus pada penggambaran tuuan dari suatu sistem. Use case model terdiri dari tiga notasi, yaitu:

1. Use case *diagram*, yaitu *diagram* yang digunakan untuk menggambarkan hubungan antara sistem dengan aktor.
2. Use case narative, yaitu uraian deskripsi dari use case *diagram* sehingga pengguna UML bisa mengetahui detail dari proses yang ada pada use case *diagram*
3. Use case scenario, yaitu *diagram* yang menggambarkan logika-logika dari usecase narative.

*Use case diagram* menggunakan simbol-simbol berikut:

Tabel 2. Simbol - simbol ERD

|  |  |  |
| --- | --- | --- |
| No. | Simbol | Deskripsi |
| 1 |  | *Use case* mendeskripsikan fungsionalitas yang ada pada sistem sebagai unit-unit yang saling bertukar pesan dengan aktor dan biasanya dinyatakan menggunakan kata kerja pada nama awal *use case.* |
| 2 |  | Aktor adalah *abtraction* orang atau sistem lain yang mengaktifkan fungsi target sistem. Identifikasi aktor memerlukan penentuan pembagian tenaga kerja dan tugas-tugas yang berhubungan dengan perannya sesuai konteks target sistem. Orang atau sistem tersebut dapat hadir pada beberapa peran. Aktor hanya berinteraksi dengan *use case*, namun tidak dapat mengontrol *use case.* |
| 3 |  | Asosiasi *actor* dengan *use case* yang menggunakan garis tanpa panah adalah untuk mengidentifikasi apa atau siapa yang meminta interaksi secara langsung dan bukan mengidentifikasikan aliran data. |
| 4 |  | Asosisasi *actor* dan *use case* yang digambarkan dengan panah terbuka dimaksudkan untuk mengidentifikasi apabila *actor* berinteraksi dengan sistem secara pasif. |
| 5 |  | *Include*, merupakan pemanggilan *use case* yang dilakukan *use case* lainnya. Contohnya yakni pemanggilan suatu fungsi program. |
| 6 |  | Extend berarti perluasan *use case* lain bila syarat atau kondisi telah terpenuhi. |

### 2.4.2. Activity Diagram

Berikut pendefinisian yang banyak menggunakan *diagram* aktivitas:

* 1. Rancangan *sistem* proses bisnis yang setiap urutan aktivitas didefinisikan.
  2. Pengelompokkan maupun urutan tampilan sistem/*user interface* dimana setiap aktivitas dianggap mempunyai tampilan rancangan antar muka.
  3. Setiap aktivitas rancangan pengujian yang kasus ujinya memerlukan definisi.
  4. Rancangan menu yang akan ditampilkan dalam perangkat lunak.

Simbol-simbol yang digunakan pada *activity diagram*, yaitu:

Tabel 2. Simbol - simbol Activity D

|  |  |  |
| --- | --- | --- |
| No. | Simbol | Keterangan |
| 1 |  | Status awal dalam aktivitas sistem. Setiap *diagram* aktivitas memiliki status awal. |
| 2 |  | Aktifitas yang biasanya diawali kata kerja. |
| 3 |  | Asosiasi percabangan yang memiliki lebih dari satu pilihan aktivitas. |
| 4 |  | Asosiasi penggabungan yang memiliki penggabungan lebih dari satu aktivitas. |
| 5 |  | Status akhir *sistem*. Tiap *diagram* aktivitas memiliki status akhir. |
| 6 |  | Menggolongkan organisasi bisnis yang bertanggungjawab pada setiap aktifitas yang terjadi. |
| 7 | Fork | Berfungsi sebagai petunjuk kegiatan paralel yang dilakukan. |
| 8 | Fork | Berguna sebagai petunjuk penggabungan kegiatan. |

### 2.4.3. Sequence Diagram

Guna *Sequence* *diagram* biasanya adalah untuk mendeskripsikan skenario atau rangkaian langkah-langkah, mulai dari apa yang menjadi dasar aktivitas tersebut, apa proses dan perubahan internal yang terjadi dan apa hasil *output* yang muncul.

Menurut (Sukamto dan Shalahuddin, 2015:165), *Diagram* sekuen menjelaskan tingkah laku objek dalam *use case*, mendeskripsikan masa waktu objek dan isi yang antar objek kirim dan terima.

Berikut ini adalah simbol – simbol yang digunakan dalam *sequence diagram*, yaitu sebagai berikut:

Tabel 2. Simbol - simbol Sequence Diagram

|  |  |  |
| --- | --- | --- |
| No. | Simbol | Deskripsi |
| 1 |  | *Entity class*, bagian sistem yang terdiri atas sekumpulan kelas berupa entitas – entitas yang menampilkan gambaran awal sistem dan menjadi dasar penyusunan basis data. |
| 2 |  | *Boundary class*, berisi sekumpulan kelas yang berupa *interface* atau interaksi satu atau lebih *actor* dengan sistem, misalnya tampilan *form* cetak dan *form* *entry*. |
| 3 |  | *Control class*, sebuah objek yang berisi logika aplikasi dan tidak bertanggung jawab pada entitas, misalnya kalkulasi dan aturan bisnis yang melibatkan beragam objek. *Control object* melakukan koordinasi pesan antara entitas dan *boundary*. |
| 4 |  | Message, simbol mengirim pesan antar *class*. |
| 5 |  | *Recursive*, simbol mengirim isi pesan antar *class.* |
|  |  | *Activation*, mewakili suatu eksekusi operasi objek, ukuran panjang kotak menyesuaikan dengan lama aktivasi suatu operasi. |
| 7 |  | *Lifeline*, garis titik-titik dan terhubung dengan objek, sepanjang *lifeline* terdapat aktivasi. |

## 2.5. Percancagan Basis Data

### 2.5.1. Basis Data

Sebagai satu kesatuan istilah, basis data (*database*) sendiri dapat didefinisikan dalam sejumlah sudut pandang (Fathansyah, 2012:2), seperti:

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redundansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.

Kumpulan *file*/*tabel*/arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronik.

### 2.5.2. Database Management Sistem (DBMS)

*Database* secara sederhana pengertiannya adalah kumpulan data seperti kumpulan nomor telpon, daftar pegawai dan lain sebagainya yang diberi kode tertentu untuk memudahkan dalam pencarian maupun pengelolaannya (Nurcholish, 2018:5). Sebuah sistem perangkat lunak yang mendukung pengguna untuk membuat, mengontrol, memelihara dan mudah mengakses *database* adalah DBMS.

DBMS adalah sebuah sistem perangkat lunak yang mengizinkan pengguna untuk mendefinisikan, membuat, memelihara, dan mengontrol akses ke dalam basis data (Connoly dan Begg, 2010:66). *Database* merupakan kumpulan datanya, dan pengelola program berdiri sendiri untuk membaca data, mengisi data, menghapus data dan melaporkan data pada *database*.

### 2.5.3. Normalisasi

Normalisasi merupakan proses pengorganisasian file untuk menghilangkan grup elemen yang berulang menjadi tabel-tabel yang menunjukkan entitas dan relasinya. Pada proses ini selalu dituju pada beberapa kondisi apakah ada kesulitan pada saat menambah (insert), menghapus (delete), mengubah (update), membaca (retrieve) pada suatu database. Dalam pembuatan normalisasi terdapat beberapa tahapan yaitu:

1. Bentuk Tidak Normal (*Unnormalized* *Form*)

Bentuk tidak normal merupakan kumpulan data yang akan direkam, tidak ada keharusan untuk mengikuti format tertentu, dapat saja data tidak lengkap atau terduplikasi. Data dikumpulkan apa adanya sesuai dengan saat menginput.

1. Bentuk Normal Kesatu (1 NF/*First Normal Form*)

Bentuk normal kesatu mempunyai ciri yaitu setiap data dibentuk dalam flat file, data dibentuk dalam satu record demi satu record nilai dari field berupa ‘Atomatic value’. Tidak ada set atribut yang berulang atau atribut bernilai ganda (multivalue). Tiap field hanya satu pengertian.

1. Bentuk Normal Kedua (2 NF/*Second Normal Form*)

Bentuk normal kedua mempunyai syarat atau bentuk data telah memenuhi kriteria bentuk normal kesatu. Atribut bukan kunci haruslah bergantung secara fungsi pada kunci utama (primery key). Sehingga untuk membentuk normal kedua haruslah sudah ditentukan kunci field. Kunci field haruslah unik dan dapat mewakili atribut lain yang menjadi anggotanya.

1. Bentuk Normal Ketiga (3 NF/ *Thrid Normal Form*)

Relasi harus dalam bentuk normal kedua dan semua atribut bukan primer tidak punya hubungan yang transitif.

1. *Boyce-Codd Normal Form* (BCNF)

Mempunyai paksaan yang lebih kuat dari bentuk normal ketiga. Untuk menjadi BCNF, Relasi harus dalam bentuk normal kesatu dan setiap atribut harus bergantung fungsi pada atribut super key.

### 2.5.4. Entity Relationship Diagram (ERD)

*Entity Relationship Diagram* merupakan tools yang digunakan untuk memodelkan struktur data dengan menggambarkan entitas dan hubungan antara entitas (*relationship*) secara abstrak (Mulyani, 2016:100). 3 fungsi utama *Entity Relationship Diagram* yaitu:

1. Sebagai alat untuk memodelkan hasil dari analisis data.
2. Sebagi alat untuk memodelkan data konseptual.
3. Sebagai alat untuk memodelkan objek-objek dalam suatu sistem.

*Entity Relationship Diagram* terdiri dari 3 komponen utama, yaitu :

1. Entitas (*entity*) adalah suatu objek di dunia nyata yang dapat dibedakan dengan objek lainnya.
2. Atribut (*attribute*) merupakan semua informasi yang berkaitan dengan entitas, atau disebut juga properti dari suatu objek (entitas).
3. Relasi (*relationship*) adalah garis yang menghubungkan antar entitas.

Menurut (Sukamto dan Shalahuddin, 2015:50) ERD digunakan sebagai pemodelan awal basis data dan pemodelan basis data relasional. Tujuan *entity relationship* adalah menunjukan objek data dan *relationship* pada objek tersebut.

|  |  |  |
| --- | --- | --- |
| No. | Simbol | Deskripsi |
| 1 | Nama\_entitas | Entitas adalah data inti yang disimpan. Pada tabel basis data, data pada benda harus disimpan agar aplikasi komputer dapat mengakses data. Pemberian nama entitas biasanya bersifat kata benda. |
| 2 |  | kolom data yang perlu disimpan pada suatu entitas (*Field*). |
| 3 | Nama\_relasi | Relasi penghubung antar entitas diawali kata kerja. |
| 4 |  | Penghubung antara relasi dengan entitas. Kedua ujungnya memiliki *multiplicity* kemungkinan kuantiti pemakai. |

### 2.5.5. Kardinalitas/Derajat Kardinalitas

Kardinalitas relasi menunjukkan jumlah maksimum entitas yang dapat berelasi dengan entitas pada himpunan entitas yang lain (Fathansyah, 2012:78). Derajat kardinalitas dibagi menjadi 3, yaitu :

1. Satu ke satu (*One to One*)

Yang berarti setiap entitas pada himpunan entitas A berhubungan dengan paling banyak dengan satu entitas pada himpunan entitas B, dan begitu juga sebaliknya setiap entitas pada himpunan entitas B berhubungan dengan paling banyak dengan satu entitas pada himpunan entitas A (Fathansyah, 2012:79).

Manager

mengepalai

1

1

Departemen

Gambar 2. *Diagram* E-R untuk relasi satu ke satu

1. Satu ke banyak (*One to Many*)

Yang berarti setiap entitas pada himpunan entitas A dapat berhubungan dengan paling banyak dengan satu entitas pada himpunan entitas B, dan begitu juga sebaliknya setiap entitas pada himpunan entitas B berhubungan dengan paling banyak dengan satu entitas pada himpunan entitas A (Fathansyah, 2012:79).

Guru

mengajar

1

M

Murid

Gambar 2. *Diagram* E-R untuk relasi satu ke banyak

1. Banyak ke banyak (*Many to Many*)

Yang berarti setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B dan demikian juga sebaliknya, dimana setiap entitas pada himpunan entitas B dapat berhubungan dengan banyak entitas pada himpunan entitas A (Fathansyah, 2012:81).

Murid

Mempelajari

N

M

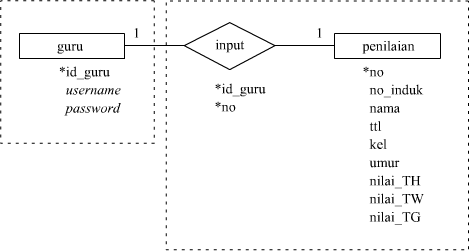
Mata Pelajaran

Gambar 2. *Diagram* E-R untuk relasi banyak ke banyak

### 2.5.6. Transformasi

Menurut (Tabrani, 2014:33-41), Transformasi dengan ERD ke LRS merupakan suatu kegiatan untuk membentuk data-data dari diagram hubungan entitas ke suatu LRS. Dalam kaitannya dengan transformasi dari ERD ke LRS, perubahan yang terjadi mengikuti aturan–aturan sebagai berikut :

1. Jika hubungan yang terjadi antara set entitas yang satu dengan yang lainnya adalah satu ke satu (*one to one*), maka atribut dari *relationship* set diambil dan dimasukan ke set entitas yang tidak memiliki atribut tersebut.



Gambar 2. Contoh transformasi *one to one*

1. Jika hubungan yang terjadi antara set entitas yang satu dengan set entitas yang lainnya adalah satu ke banyak (*one to many*) maka atribut dari *relationship* set digabungkan dengan set entitas yang mempunyai kardinalitas yang banyak.

AGT

agt

ajukan

1

M

FPK

\*no\_anggota

Nm\_anggota

Al\_rumah

Al\_kantor

No\_tlp

No\_hp

\*no\_fpk

Tgl\_fpk

Jm\_kredit

Lm\_angsuran

\*no\_anggota

\*no\_fpk

Gambar 2. Contoh transformasi *one to many*

1. Jika hubungan yang terjadi antara set entitas yang satu dengan set yang lainnya adalah banyak ke banyak (*many to many*) maka atribut dari *relationship* set dibentuk menjadi satu set entitas.

mahasiswa

ikut

M

M

mata\_kuliah

\*NIM

Tgl\_lahir

nama

alamat

jen\_kel

\*kd\_matkul

Deskripsi

SKS

\*nim

Kd\_matkul

nilai

Gambar 2. Contoh transformasi *many to many*

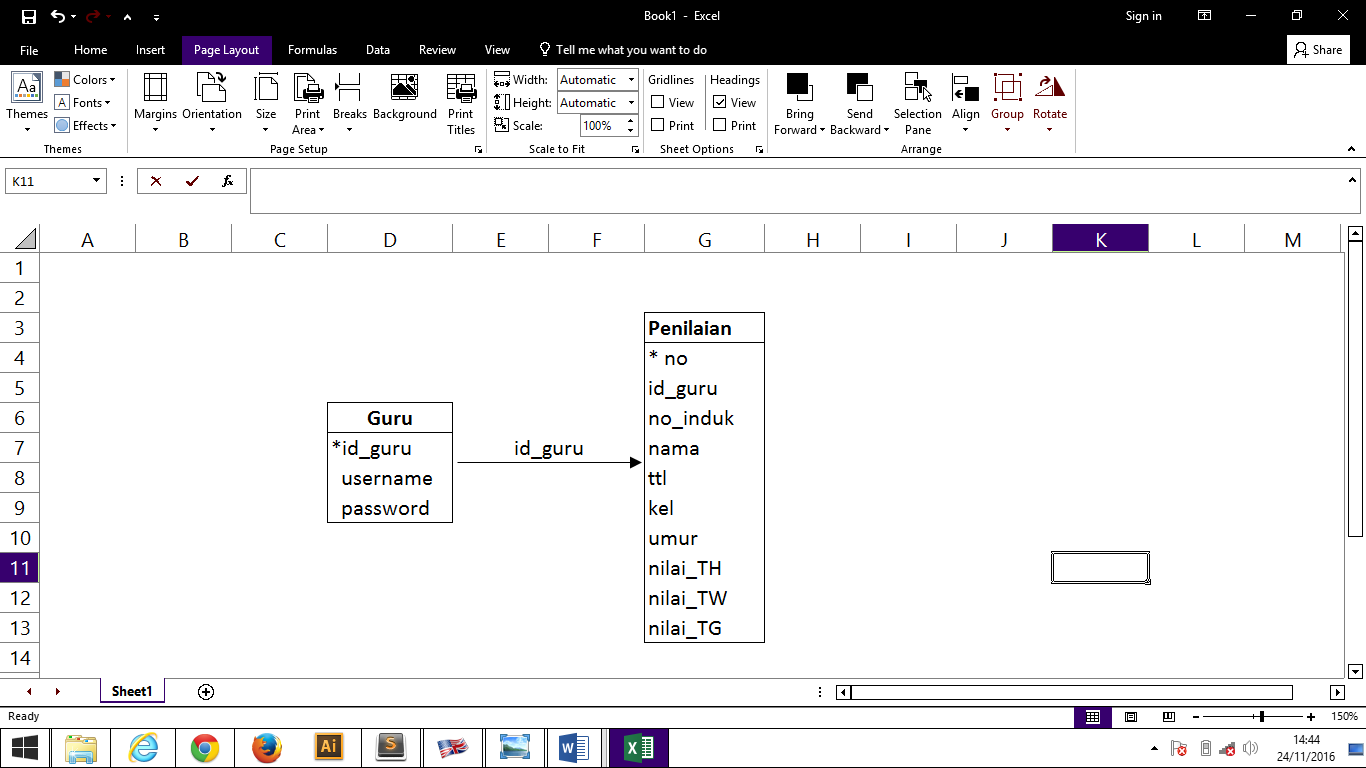
### 2.5.7. Logical Record Structure (LRS)

LRS (*Logical Record Structure*) adalah representasi dari struktur *record–record* pada tabel–tabel yang terbentuk dari hasil relasi antara humpunan entitas. Menurut (Hasugian dan Shidiq, 2012:608) berpendapat bahwa LRS adalah sebuah model sistem yang digambarkan dengan sebuah diagram ER akan mengikuti pola atau aturan permodelan tertentu dalam kaitannya dengan konveksi ke LRS.

Sedangkan menurut (Ladjamudin, 2013:159) LRS merupakan hasil transformasi ERD ke LRS yang memulai proses kardinalitas dan

menghilangkan atribut-atribut yang saling berelasi”.

Berikut contoh dari LRS :



Contoh Logical Record Structure

## 2.6. Bahasa Pemrograman

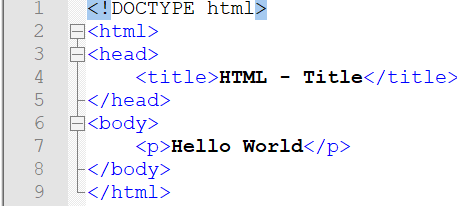
### 2.6.1 HTML

HTML adalah bahasa dasar untuk menampilkan halaman web pada web browser. HTML adalah kependekan dari *Hypertext Markup Language* yang berarti bahasa *markup* (penanda) berbasis *text* atau bisa juga disebut sebagai *formatting language* (bahasa untuk memformat).

Menurut (Enterprise Jubliee, 2016:9), HTML tidak bisa disebut bahasa pemrograman walaupun menganduk beragam “kode” (tanda/*mark*/*tag*), karena HTML tidak memiliki stuktur tertentu seperti logika *if*, pengulangan, variabel, *debugger* dan penulisan *syntax* yang cermat. Di dunia HTML, kesalahan penulisan kode tidak akan menghasilkan pesan kesalahan, hanya tampilan *website* yang anda buat tampak tidak seperti yang diharapkan. Struktur dasar HTML terdiri dari:

* 1. Tag DTD atau DOCTYPE
  2. Tag HTML
  3. Tag HEAD
  4. Tag BODY

Contoh *script* dasar dari stuktur HTML yaitu:



Gambar 2. Contoh *script* dasar HTML

### 2.6.2. PHP

R. Lerdorf membuat *PHP* pada tahun 1995 dengan nama FI (*Form* *Interpreted*), yakni skrip pengolah data form dalam web. Kemudian Lerdorf merilis kode sumber menjadi sumber terbuka sehingga pengembangan *PHP* mengalami kemajuan pesat. Seiring dengan perkembangan *PHP*, perusahaan Zend membuat interpreter baru dan meresmikannya sebagai *PHP* 3.0 serta merubah singkatan *PHP* menjadi *Hypertext Preprocessor* (Yanto, 2016:211).

Bahasa pemprograman PHP merupakan bahasa pemrograman yang paling banyak digunakan, tentu karena berbagai alasan, salah satunya adalah mempunyai beberepa kelebihan dibandingkan dengan bahasa pemrograman lainnya yang sejenis. Berikut ini kelebihan pemrograman PHP:

1. PHP adalah bahasa multiplatform yang artinya dapat berjalan di berbagai mesin dan sistem operasi (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara runtime melalui console serta juga dapat menjalankan perintah-perintah sistem lainnya.
2. PHP bersifat Open Source yang berarti dapat digunakan oleh siapa saja secara gratis.
3. Web server yang mendukung PHP dapat ditemukan dimana-mana dari mulai apache, IIS, Lighttpd, nginx, hingga Xitami dengan konfigurasi yang relatif mudah dan tidak berbelit-belit, bahkan banyak yang membuat dalam bentuk paket atau package (PHP, MySQL dan Web Server).
4. Dalam misi pengembangan lebih mudah, karena banyaknya komunitas dan developer yang siap membantu dalam pengembangan.
5. Dalam sisi pemahaman, PHP adalah bahasa scripting yang paling mudah karena memiliki referensi yang banyak
6. Banyak bertebaran aplikasi dan program PHP yang gratis dan siap dipakai seperti wordpress, prestashop, dan lain-lain.
7. Dapat mendukung banyak *database*, seperti MySQL, oracle, MS-SQL, dan lain-lain.

Sedangkan kekurangannya adalah PHP tidak mengenal package, jika tidak di-*encoding*, maka kode PHP dapat dibaca semua orang dan untuk meng-endcodingnya dibutuhkan tool dari zend yang mahal sekali biayanya, PHP juga memiliki kelemahan keamanan jadi programmer harus jeli dan berhati-hati dalam melakukan pemrograman dan konfirgurasi PHP.

### 2.6.3. MySQL

Pengembangan *MySql* pertama kali diciptakan Michael M. Widenius menggunakan tool *database* UNIREG untuk perusahaan Swedia (TcX) pada tahun 1979. Kemudian tahun 1994, TcX melakukan pengembangan selanjutnya untuk aplikasi pada web. *MySql* 3.11.1 mulai diperkenalkan pada dunia dan diterapkan pada Linux pada 1996 (Yanto, 2016:72).

Kelebihan dari MySQL adalah:

1. Source MySQL dapat diperoleh dengan mudah dan gratis.
2. Sintaksnya lebih mudah dipahami dan tidak rumit.
3. Pengaksesan *database* dapat dilakukan dengan mudah.
4. MySQL merupakan program yang multithreaded, sehingga dapat dipasang pada server yang memiliki multiCPU.
5. Didukung program-program umum seperti C, C++, Java, Perl, PHP, Python, dsb.
6. Bekerja pada berbagai platform.
7. Memiliki jenis kolom yang cukup banyak sehingga memudahkan konfigurasi sistem *database*.
8. Memiliki sistem sekuriti yang cukup baik dengan verfikasi host.
9. mendukung recored yang memiliki kolom dengan panjang tetap atu panjang bervariasi.

(Yanto, 2016:73) Beberapa ketentuan pemberian perintah dalam *MySql* adalah:

1. Perintah *MySql* bersifat *case insensitive* yakni penulisan huruf kecil maupun besar.
2. Penggunaan tanda titik koma pada setiap akhir perintah.

### 2.6.4. XAMPP

Menurut Riyanto (2015:1), XAMPP merupakan paket PHP dan MySQL berbasis *open sourse*, yang dapat digunakan sebagai *tool* pembantu pengembangan apliaksi berbasis PHP. XAMPP merupakan kompilasi beberapa program. XAMPP berfungsi sebagai server mandiri (*localhost)* dan terdiri atas HTTP Server, program Apache, *MySql* *Database* serta penerjemah bahasa menggunakan *PHP* dan Perl. XAMPP adalah singkatan dari X (dijalankan di banyak sistem operasi seperti Linux, Windows), Apache, *MySql*, *PHP* dan Pearl.

### Code Igneter

Bagi Anda yang sudah akrab menguasai dasar-dasar PHP, maka Anda mungkin sudah mengenal beberapa framework yang ada di dalamnya. Framework secara sederhana dapat diartikan sebagai kumpulan dari fungsi-fungsi atau prosedur-prosedur dan class-class untuk tujuan tertentu yang sudah siap digunakan sehingga bisa lebih mempermudah dan mempercepat pekerjaan seorang programer, tanpa harus membuat fungsi atau class dari awal.  
Salah satu framework yang bisa digunakan adalah codeigniter. CodeIgniter adalah aplikasi open source yang berupa framework dengan model MVC (Model, View, Controller) untuk membangun website dinamis dengan menggunakan PHP. CodeIgniter memudahkan developer untuk membuat aplikasi web dengan cepat dan mudah dibandingkan dengan membuatnya dari awal.

Model View Controller merupakan suatu konsep yang cukup populer dalam pembangunan aplikasi web, berawal pada bahasa pemrograman Small Talk, MVC memisahkan pengembangan aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi seperti manipulasi data, user interface, dan bagian yang menjadi kontrol aplikasi. Terdapat 3 jenis komponen yang membangun suatu MVC pattern dalam suatu aplikasi yaitu :

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 merepresentasikan 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 yang 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.

## 2.7. System Development Life Cycle (SDLC)

Istilah *System Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dengan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya (Rossa A.S dan M. Shalahudin 2013:24).

Secara umum, tahapan-tahapan pada SDLC adalah:

1. Inisiasi (*initiation*)

Pembuatan proposal perangkat lunak.

1. Pengembangan konsep sistem (*sistem concept development*) yakni mendeskripsikan ruang lingkup konsep seperti dokumen dalam lingkup sistem, manajemen rencana, analisa manfaat biaya, dan pembelajaran kemudahan sistem.
2. Perencanaan (*planning*)

Pengembangan perencanaan manajemen proyek beserta dokumen perencanaan lainya. Menyiapkan cara memperoleh sumber daya untuk memperoleh solusi.

1. Analisis kebutuhan (*requirements analysis*)

Menganalisa kebutuhan dan mengembangkan kebutuhan pengguna sistem, termasuk mempersiapkan kebutuhan fungsional.

1. Desain (*design*)

Men*transformasi* *detail* kebutuhan menjadi kebutuhan yang sudah lengkap. Desain sistem berfokus pada pemenuhan fungsi-fungsi yang dibutuhkan.

1. Pengembangan (*development*)

Mengkonversi desain menjadi sistem informasi yang lengkap, misalnya bagaimana melakukan dan menyiapkan instalasi lingkungan sistem yang sesuai, menyiapkan file atau berkas pengujian, melakukan pengkodean, mengkompilasi, membersihkan dan memperbaiki program, peninjauan pengujian pembuatan basis data dan menyiapkan prosedur kasus pengujian.

1. Integrasi dan pengujian (*integration and test*)

Pendemonstrasian sistem perangkat lunak guna mengetahui apakah sistem telah memenuhi kebutuhan yang telah dijelaskan pada data kebutuhan fungsional. Proses ini biasanya diarahkan staf penjamin kualitas dan *user* sehingga menghasilkan laporan analisa pengujian.

1. Implementasi (*implementation*)

Berisi persiapan penerapan peranti lunak dalam lingkungan produksi dan menerapkan resolusi atas masalah yang teridentifikasi pada tahap integrasi dan pengujian.

1. Operasi dan pemeliharaan (*operations and maintenance*)

Tahap implementasi dan pemeliharaan sistem dalam lingkungan produksi, hingga pengimplementasian akhir dan memulai proses peninjauan.

1. Disposisi (*disposition*)

Tahap penjelasan aktifitas akhir dalam pengembangan sistem dan pembangunan data yang sebenarnya sesuai aktifitas *user.*

## 2.8. Model Prototype

Prototipe adalah salah satu versi dari sebuah sistem potensial yang memberikan ide bagi para pengembang dan calon pengguna, bagaimana sistem akan berfungsi dalam bentuk yang telah selesai (Darmawan, 2013:229). Sedangkan menurut (Simarmata, 2010:62) teknik Prototyping adalah bagian dalam produk yang mengekspresikan logika maupun tampilan fisik antar muka eksternal.

Menurut (Mulyani, 2016:29), Tahapan langkah requirement *prototype*, adalah sebagai berikut:

1. Analisis kebutuhan user, pengembang dan pengguna atau pemilik sistem melakukan diskusi dimana pengguna atau pemilik sistem menjelaskan kepada pengembang tentang kebutuhan sistem yang mereka inginkan.
2. Membuat *prototype*, pengembang membuat *prototype* dari sistem yang telah dijelaskan oleh pengguna atu pemilik sistem.
3. Menyesuaikan *prototype* dengan keinginan user, pengembang menanyakan kepada pengguna atau pemilik sistem tentang *prototype* yang sudah dibuat, apakah sesuai atau tidak dengan kebutuhan sistem.
4. Membuat sistem baru, pengembang menggunakan *prototype* yang sudah dibuat untuk membuat sistem baru.
5. Melakukan *testing* sistem, pengguna atau pemilik sistem melakukan uji coba terhadap sistem yang dikembangkan.
6. Menyesuaikan dengan keinginan user, sistem disesuaikan dengan keinginan user dan kebutuhan sistem, jika sudah sesuai sistem siap digunakan.

Kelebihan dari teknik pengembangan *prototype* yaitu:

1. Menghemat biaya pengembangan.
2. Menghemat waktu pengembangan.
3. Kualitas sistem yang dihasilkan baik.
4. Implementasi akan menjadi mudah, karena pengguna atau pemilik sistem sudah mempunyai gambaran tentang sistem.
5. Memungkinkan tim pengembang sistem memprediksi dan memperkirakan pengembangan-pengembangan sistem selanjutnya.
6. Pengguna atau pemilik sistem ikut terlibat dalam pengembangan, sehingga kemungkinan-kemungkinan terjadinya kesalahpahaman dalam sistem bisa diminimalisir.

Sedangkan kelemahannya adalah pengguna / pemilik sistem bisa terus menerus menambah kompleksitas sistem hingga sistem menjadi sangat kompleks, hal ini bisa menyebabkan pengembang meninggalkan pekerjaannya sehingga sistem yang dikerjakan tidak akan pernah terselesaikan.

## 2.9. Pengujian Perangkat Lunak

Setiap perangkat lunak memerlukan pengujian. Pengujian akan membantu menemukan kesalahan atau error ketika perangkat lunak di uji. Hal ini akan mempermudah pembuat kode program mengetahui dan menetapkan langkah selanjutnya dengan lebih jelas jika kesalahan atau error tersebut muncul.

### 2.9.1. Black Box Testing

*Black box testing* (pengujian kotak hitam) yaitu pengujian perangkat lunak dari sisi spesifikasi fungsional, tanpa melibatkan desain dan kode program. Pengujian ini juga bertujuan untuk mengetahui kesesuaian fungsi, masukan hingga keluaran perangkat lunak dengan kebutuhan spesifikasi (Sukamto dan Shalahuddin, 2015:277).

Pada *black box* kegiatan yang akan di lakukan oleh *tester* selama pengujian adalah sebagai berikut:

1. Membuat *test case* untuk menguji fungsi-fungsi yang ada pada perangkat lunak.
2. Membuat *test case* untuk menguji kesesuaian alur kerja suatu fungsi di perangkat lunak dengan requirement yang dibutuhkan *customer* untuk fungsi tersebut.
3. Mencari *bugs* atau e*rror* dari tampilan (*interface*) perangkat lunak.

BAB III

# ANALISA DAN PEMBAHASAN

## 3.1. **Analisa Sistem Berjalan**

Berdasarkan hasil pengamatan dan observasi pengolahan data nasabah bank sampah di data dengan mendaftarkan data diri (KTP, No telfon) melalui tatap muka atau datang ketempat penimbangan sehingga tidak efisien dalam ketika mencari sebuah informasi data yang tidak valid. Hal ini menyulitkan staff ketika mencari data nasabah yang tetap.

Ketika pembuatan data nasabah baru, staff yang bertanggung jawab terhadap pengolahan data nasabah bank sampah tersebut selalu melakukan re-list. Selain itu ketika suatu nasabah ingin mengambil atau mencairkan uang yang telah ditabung staff selalu menghitung dengan cara manual untuk mengetahui nilai tabungan dari nasabah tersebut.

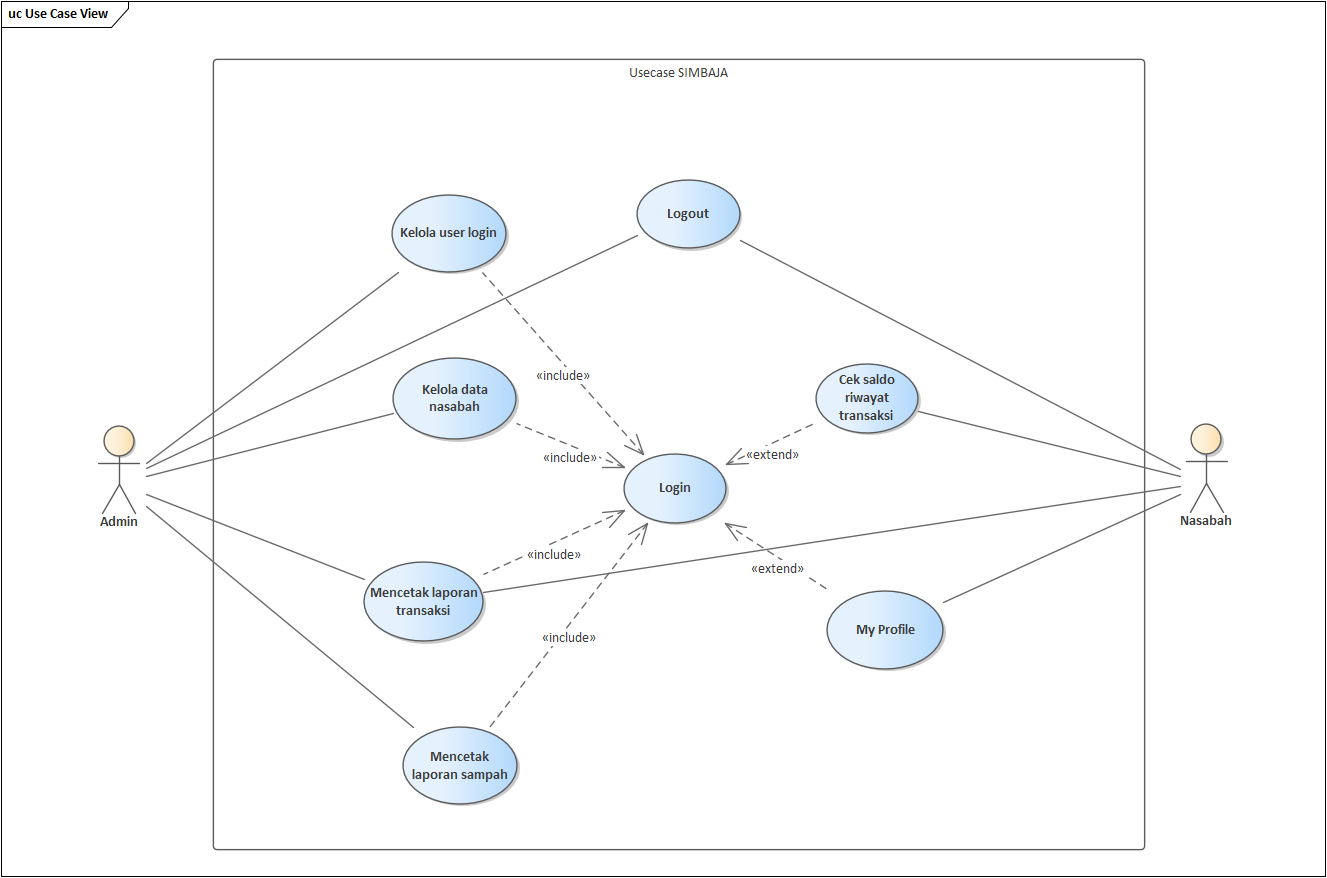


Gambar 3. Alur analisa system berjalan

## 3.2. Analisa Sistem Usulan

Berdasarkan system yang telah berjalan diatas dan mengetahui permasalahan yang ada, maka disusunlah solusi untuk menyelesaikan masalah diatas. System yang diusulkan memiliki tujuan untuk mendapatkan hasil yakni berupa efisiensi dalam melakukan proses dari pendaftaran nasabah baru, laporang tabungan nasabah, data informasi nasabah. Hal ini dapat membantu organisasi bank sampah untuk mengetahui secara pasti data nasabah yang dimiliki.

### 3.2.1. Use Case

Use case diagram ialah diagram yang mendeskripsikan kebutuhan system dari cara piker user, dan menunjukan relasi-relasi yang terjadi antara actor dengan use case dalam system.

Gambar 3. *Use case* Sistem Informasi Pengolahan Data Nasabah Bank Sampah

### 3.2.2. Activity Diagram

Activity diagram memudahkan dalam memahami langkah-langkah aliran kerja. Diagram ini menggambarkan langkah kerja. Sehingga dapat diketahui siapa yang bertanggung jawab atas masing-masing aktivitas seperti pada gambar dibawah ini:

* + - 1. *Activity* *Diagram Login*

Pada *activity* *diagram* ini, merupakan activy diagram untuk login ke dalama sistem dimana pengguna harus memasukan *user* dan *password*.



Gambar 3.2 Activity Diagram Login

* + - 1. *Activity Diagram kelola User Login*

Pada *activity* *diagram* ini, seorang *Admin* dapat mengelola *user* yang dapat mengakses aplikasi dari mendaftarkan, mengedit, menghapus *user.*



Gambar 3.3 Activity Diagram Kelola User Login

* + - 1. *Activity Diagram kelola Data Nasabah*

Pada *activity* *diagram* ini, seorang *Admin* dapat mengelola *user* yang dapat mengakses aplikasi dari mendaftarkan, mengedit, menghapus *user.*



Gambar 3.4 *Activity* *diagram* Kelola *Data Nasabah*

* + - 1. *Activity Diagram kelola Informasi Berita*

Pada *activity* *diagram* ini, seorang *Admin* dapat mengelola *informasi berita* mengenai bank sampah dan sebagainya, *Admin* dapat menambahkan, mengedit, dan menghapus *berita.*



Gambar 3.5 Activity Diagram kelola Informasi Berita

* + - 1. *Activity Diagram kelola Data Rekening Nasabah*

Pada *activity* *diagram* ini, seorang *Admin* dapat mengelola *Data Rekening Nasabah* bank sampah, *Admin* dapat menambahkan, mengedit, dan menghapus *Data Rekening Nasabah.*



Gambar 3.6 Activity Diagram kelola Data Rekeing Nasabah

* + - 1. *Activity Diagram Laporan Transaksi*

*Admin* dan Pengguna aplikasi dapat melakukan pembuatan laporan transaksi yang dimana sistem akan men*generate* secara otomatis laporan tersebut ke dalam format *pdf.*



Gambar 3.7 Activity Diagram laporan Transaksi

* + - 1. *Activity Diagram Laporan Sampah*

*Admin* aplikasi dapat melakukan pembuatan laporan sampah yang dimana sistem akan men*generate* secara otomatis laporan tersebut ke dalam format *pdf.*



Gambar 3.8 Activity Diagram Laporan Sampah

* + - 1. *Activity Diagram Logout*

*Activity diagram logout* ini menjalaskan aktifitas *pengguna* yangdapat melakukan aktivitas *logout.*



Gambar 3.9 Activity Diagram Logout

### 3.2.3. Sequence Diagram

1. *Sequence* *Diagram* *Login*
2. Nasabah memasukkan *email* dan *password* pada *form* *login*
3. *Interface* akan mengirimkan isian dari *form* *login* ke *authentification* (yang berfungsi sebagai kontrol) untuk mengecek validasi *login*
4. *Authentification* mengecek validasi *login* dengan mengirimkan data dari *interface* *login* kepada entitiy *user* yang terdapat dalam *database* sistem.
5. Jika *login* vaild maka sistem *user* akan masuk ke menu utama sesuai dengan *level user* yang digunakan oleh pengguna jika *login* tidak valid maka sistem akan menampilkan pesan kesalahan *error* kepada pengguna sistem.



Gambar 3.10 Sequence Diagram Login

1. *Sequence* *Diagram* – *User Login*
2. *Admin* meng*input* data *user* pada *form* tambah data *user*
3. Sistem akan menyimpan data *user* yang telah di*input*.
4. *Admin* juga dapat merubah data *user* ataupun menghapus data yang telah dibuat.



Gambar 3.11 Sequence Diagram kelola User Login

1. *Sequence* *Diagram* – *Kelola Data Nasabah*
2. *Admin* meng*input* data *user* pada *form* tambah data *user*
3. Sistem akan menyimpan data *user* yang telah di*input*.
4. *Admin* juga dapat merubah data *user* ataupun menghapus data yang telah dibuat.

****

Gambar 3.12 Sequence Diagram kelola Data Nasabah

1. *Sequence* *Diagram* – Cetak Transaksi
2. Penggunamengklik halaman cetak laporan
3. Penggunamelakukan pencarian terhadap transaksi tetap yang ingin di cetak laporannya.
4. Penggunamengklik tombol *print.*
5. Laporan transaksi tetap tercetak.



Gambar 3.14 Sequence Diagram Laporan Transaksi

1. *Sequence* *Diagram* – Cetak Laporan Sampah
2. Penggunamengklik halaman cetak laporan
3. Penggunamelakukan pencarian terhadap transaksi tetap yang ingin di cetak laporannya.
4. Penggunamengklik tombol *print.*
5. Laporan transaksi tetap tercetak.



Gambar 3.15 Sequence Diagram Laporan sampah

1. *Sequence* *Diagram* – *Kelola Informasi Berita*
   * + 1. *Admin* meng*input* berita pada *form* tambah data berita.
       2. Sistem akan menyimpan data informasi berita yang telah di*input*.
       3. *Admin* juga dapat merubah atau memperbaharui data berita ataupun menghapus data yang telah dibuat.



Gambar 3.16 Sequence Diagram Laporan sampah

1. *Sequence* *Diagram* –*Logout*
2. Pengguna mengklik tombol *logout*
3. Sistem mengakhiri *session*



Gambar 3.17 Sequence Diagram Laporan sampah

### 3.2.4. Class Diagram

*Class diagram* adalah model statis yang menggambarkan struktur dan deskripsi class serta hubunganya antara class. Class diagram mirip ER-Diagram pada perancangan database, bedanya pada ER-diagram tidak terdapat operasi tapi hanya atribut.

Gambar 3.18 Sequence Diagram Laporan sampah

## 3.3. Rancangan Data

### *Entity* *Relationship* *Diagram* (ERD)

Entitiy *Relationship* *Diagram* (ERD) adalah model teknik pendekatan yang menyatakan hubungan suatu model. Hubungan tersebut menggambarkan objek dan relasi pada *entity*.

Gambar 3.19 *Entity Relationship Diagram (ERD)*

Gambar diatas merupakan *Entity Diagram Relationship* pada sistem informasi manajemen aset tetap berbasis *web*, yang berguna untuk mempermudah dalam rekayasa model struktur data, alur dan relasi tiap data.

### *Transformasi* ERD ke LRS

Gambar 3.20 *Transformasi* ERD ke LRS

Gambar diatas merupakan *Transformasi* *Entity Relation Diagram* ke *Logical Record Structure*, transformasi merupakan bentuk perubahan dari bentuk awal ke bentuk berikutnya. Data pada *Entity Relation Diagram* akan dikelompokan dalam bentuk basis data untuk menentukan *entity* pada *diagram* *Logical Record Structure,* yang berguna dalam pembangunan suatu *database*.

### *Logical* *Record* *Structure* (LRS)

Hasil dari transformasi ERD ke LRS adalah sebagai berikut:

Gambar 3.21 *Logical* *Record* *Structure* (LRS)

Gambar diatas merupakan *Logical Relation Structure* dari sistem informasi manajemen aset tetap berbasis *web* dalam bentuk tabel-tabel pada *database* yang dihasilkan dari hubungan antar entitas.

### 3.3.4. Spesifikasi Tabel

Spesifikasi basis data ini memberikan penjelasan secara detail tentang tabel – tabel yang ada pada *database* yang digunakan untuk sistem yang diusulkan.

1. Tabel *user*

Nama file : *user*

Isi : Data *User*

*Primary* *Key* : *id*

Struktur Tabel :

Tabel 3. 1 *Tabel* user

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama** | **Tipe** | **Ukuran** | **Kunci** | **Keterangan** |
| Id | Int | 11 | *Primary Key* | Nomor identitas Admin dan Nasabah |
| Nik | Varchar | 16 |  | Nomor ktp dari register |
| Nama | Varchar | 128 |  |  |
| Tanggal\_lahir | Date |  |  | Tanggal\_lahir user |
| Jenis\_kelamin | Varchar | 50 |  | Jenis\_kelamin user |
| No\_telpon | Varchar | 13 |  | No\_telpon user |
| Email | Varchar | 50 |  | Email user |
| Password | Varchar | 50 |  | Password user |
| Agama | Varchar | 50 |  | Agama user |
| Kecamatan | Varchar | 50 |  | Kecamatan user |
| Kelurahan | Varchar | 50 |  | Kelurahan user |
| Status | Varchar | 50 |  | Status user |
| Pekerjaan | Varchar | 50 |  | Pekerjaan user |
| Role\_id | Int | 1 |  | Role\_id user |
| Date\_created | Int | 11 |  | Hari/Tanggal pembuatan user |
| Image | Varchar | 13 |  | Lokasi gambar |
| Saldo | Int | 11 |  | Isi dari transaksi penyetoran / penarikan |
| Status\_akun | Int | 1 |  | Status akun aktif / tidak aktif |
| File | Varchar | 128 |  |  |

1. Tabel *tb\_transaksi*

Nama file : *tb\_transaksi*

Isi : Data *Transaksi*

*Primary* *Key* : *id*

Struktur Tabel :

Tabel 3.2 *Tabel* tb\_transaksi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama** | **Tipe** | **Ukuran** | **Kunci** | **Keterangan** |
| Id | Int | 11 | *Primary Key* | Nomor identitas transaksi |
| Id\_user | Int | 11 | *Foreign Key* | Id dari nasabah terdaftar |
| Jumlah\_penarikan | Int | 11 |  | Saldo yang ada pada nasabah |
| Keterangan | Text |  |  | Withdraw / Setor |
| Tanggal | Date |  |  | Tanggal withdraw / setor |
| Verifikasi\_penarikan | Varchar | 50 |  | Status withdraw |

1. Tabel *user*

Nama file : *tb\_harga*

Isi : Data *Harga Sampah*

*Primary* *Key* : *id*

Struktur Tabel :

Tabel 3. 3 *Tabel* tb\_harga

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama** | **Tipe** | **Ukuran** | **Kunci** | **Keterangan** |
| Id | Int | 11 | *Primary Key* | Nomor identitas harga sampah |
| Jenis\_sampah | Varchar | 50 |  | Jenis sampah |
| Harga | Varchar | 50 |  | Harga sampah |
| Satuan | Varchar | 16 |  | Satuan berat sampah |
| Tanggal | Date |  |  | Tanggal input jenis sampah |

1. Tabel *tb\_logs*

Nama file : *tb\_logs*

Isi : Data *Logs User*

*Primary* *Key* : *id*

Struktur Tabel :

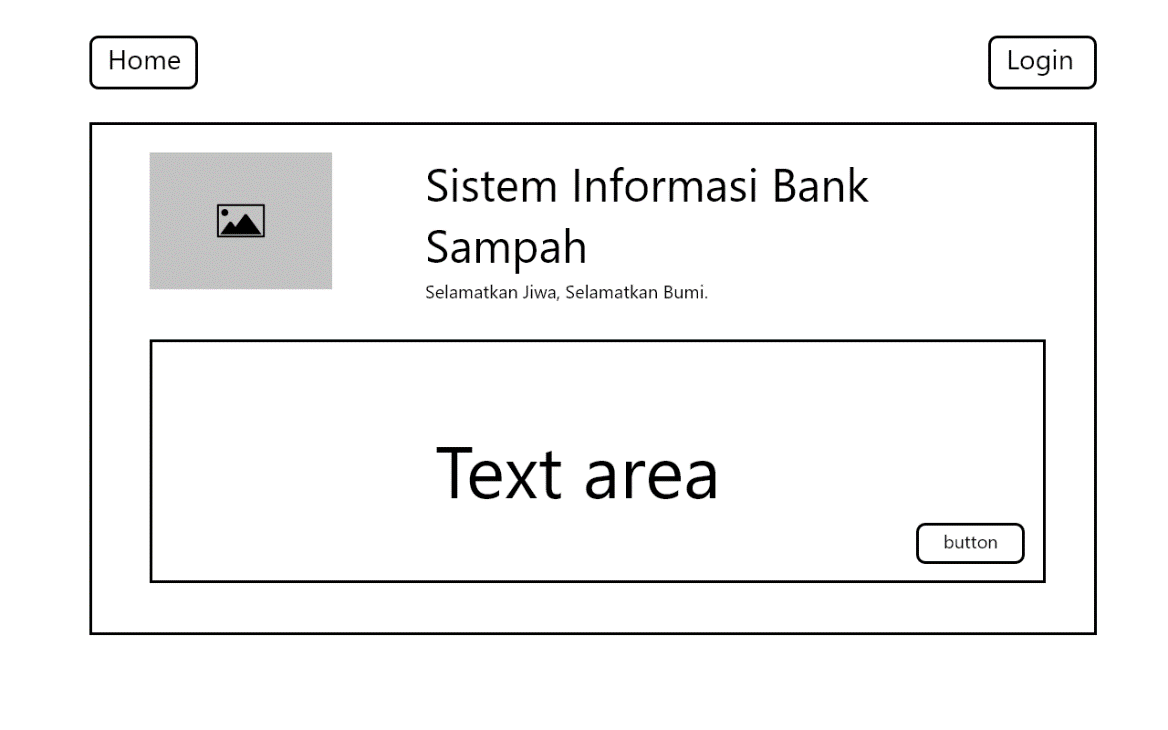
Tabel 3. 4 *Tabel* tb\_logs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama** | **Tipe** | **Ukuran** | **Kunci** | **Keterangan** |
| Id | Int | 11 | *Primary Key* | Nomor identitas logs |
| Id\_user | Int | 11 | *Foreign Key* | Nomor identitas user |
| Datetime | Datetime |  |  | Waktu user mengakses sistem |
| Keterangan | Text |  |  | History user |

## 3.4. Rancangan Tampilan

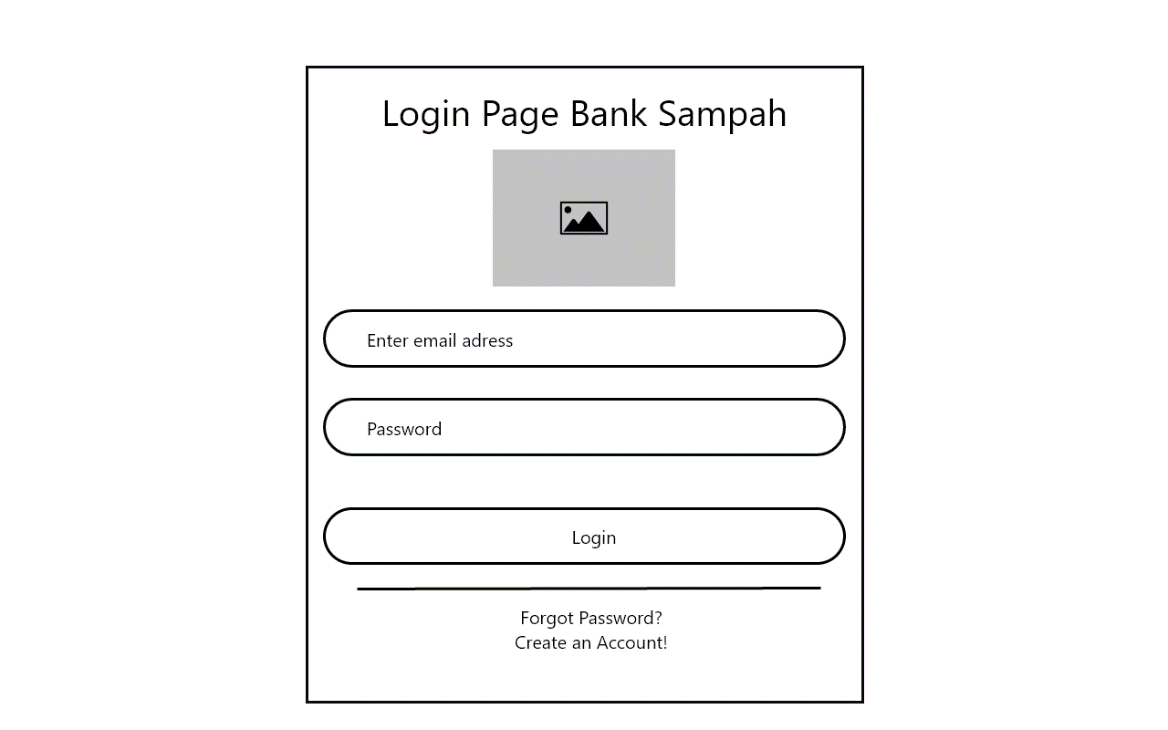
Merancang *user* *interface* merupakan bagian penting dalam perancangan aplikasi, karena dapat mengetahui kesesuaian aplikasi dengan kebutuhan. Berikut ini adalah rancangan *user* *interface* dalam perancangan aplikasi yang akan dibuat.

* 1. Halaman *Form Utama*



Gambar 3.22 Rancangan Tampilan *Form* *Utama*

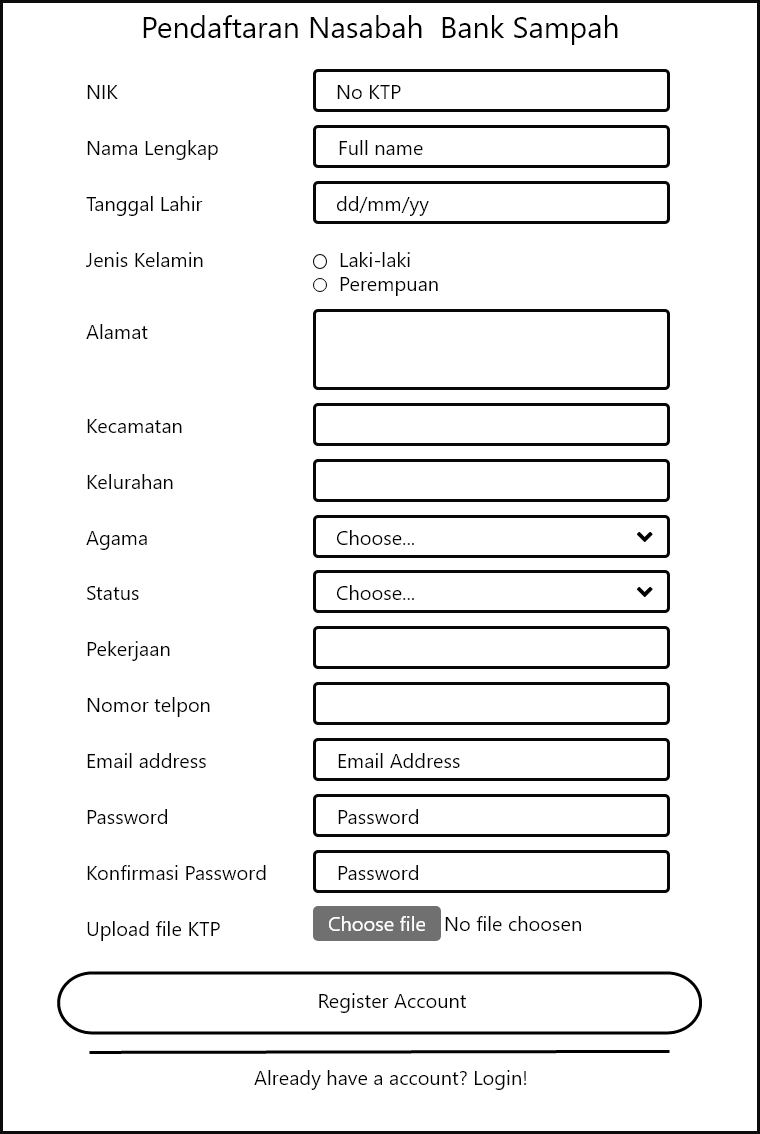
1. Halaman *Form Login*



Gambar 3.23 Rancangan Tampilan *Form* *Login*

Rancangan halaman pada form *login*, dibuat sederhana untuk memudahkan pengguna dalam melakukan proses *login* kedalam sistem.

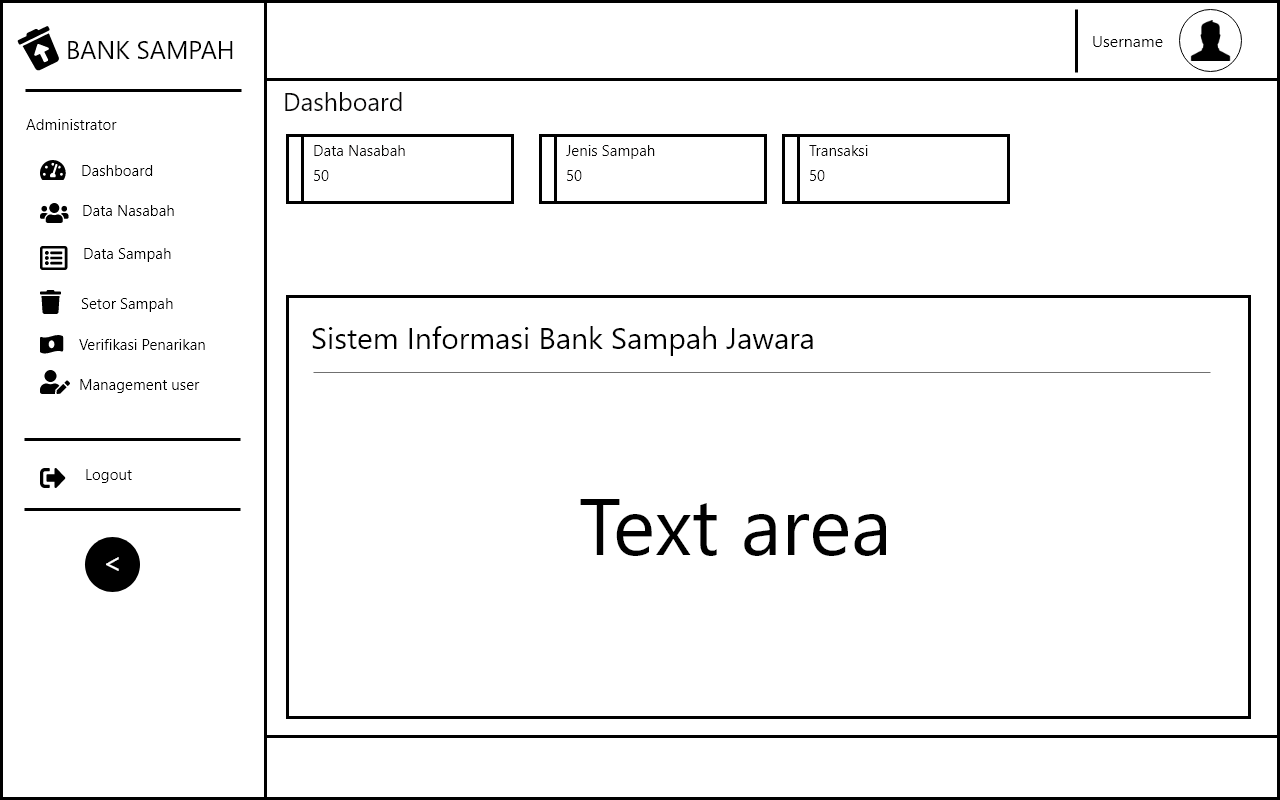
1. Halaman *Form Registrasi*



Gambar 3. Rancangan Tampilan *Form* *Registrasi*

Rancangan halaman pada form *registrasi*, dibuat sederhana untuk memudahkan pengguna dalam melakukan pendaftaran nasabah.

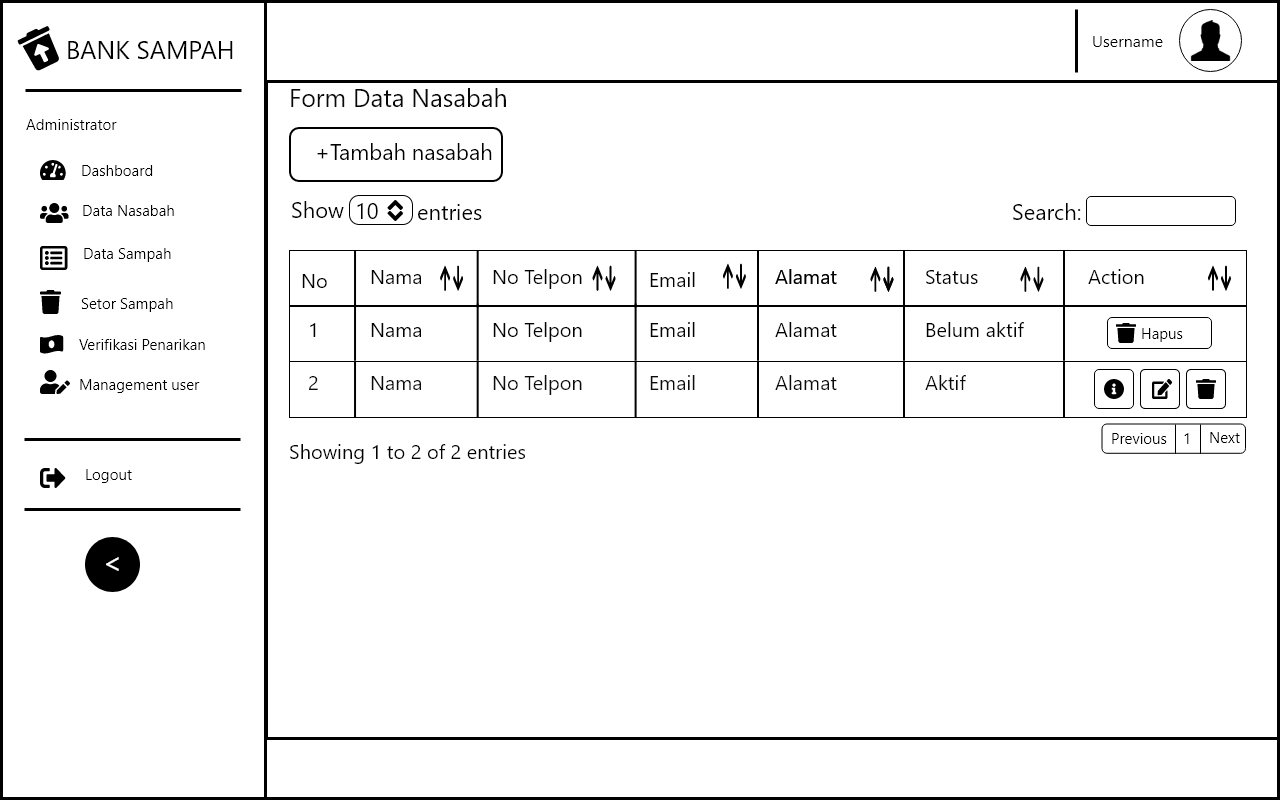
1. Halaman *Form Dashboard admin*



Gambar 3. Rancangan Tampilan *Form* *Dashboard admin*

Rancangan halaman pada form *Dashboard Admin* , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

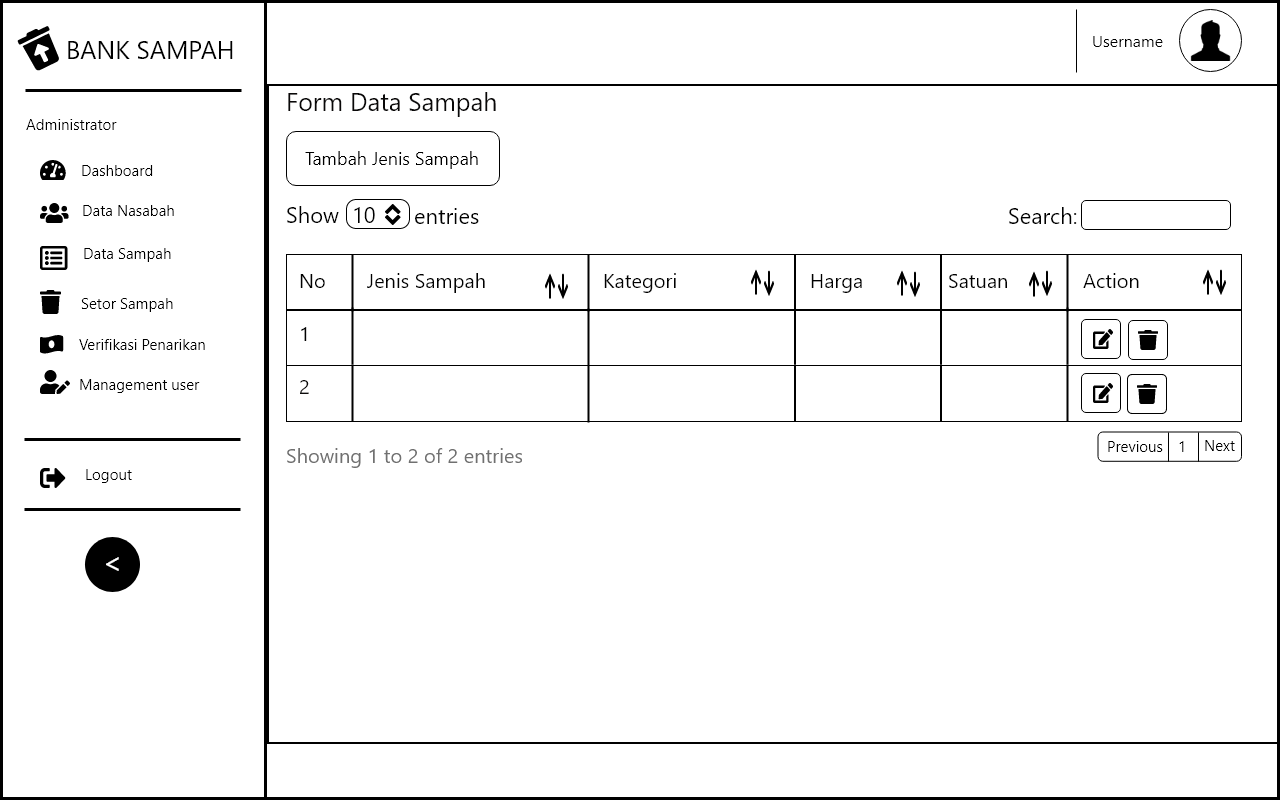
1. Halaman *Form Data Nasabah Admin*



Gambar 3. Rancangan Tampilan *Form* *Data Nasabah Admin*

Rancangan halaman pada form *Data Nasabah Admin*, dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

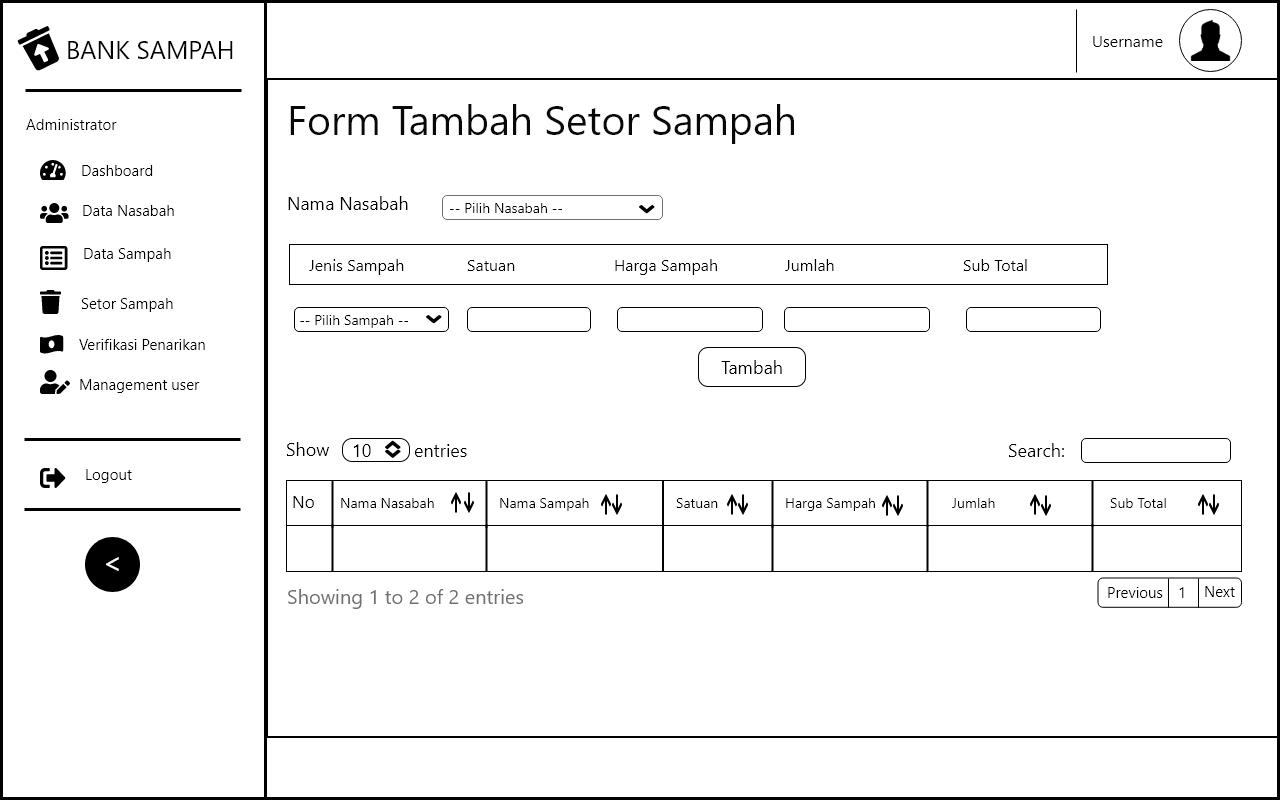
1. Halaman *Form Data Sampah Admin*



Gambar 3. Rancangan Tampilan *Form* *Data Sampah Admin*

Rancangan halaman pada form *Data Sampah Admin*, dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

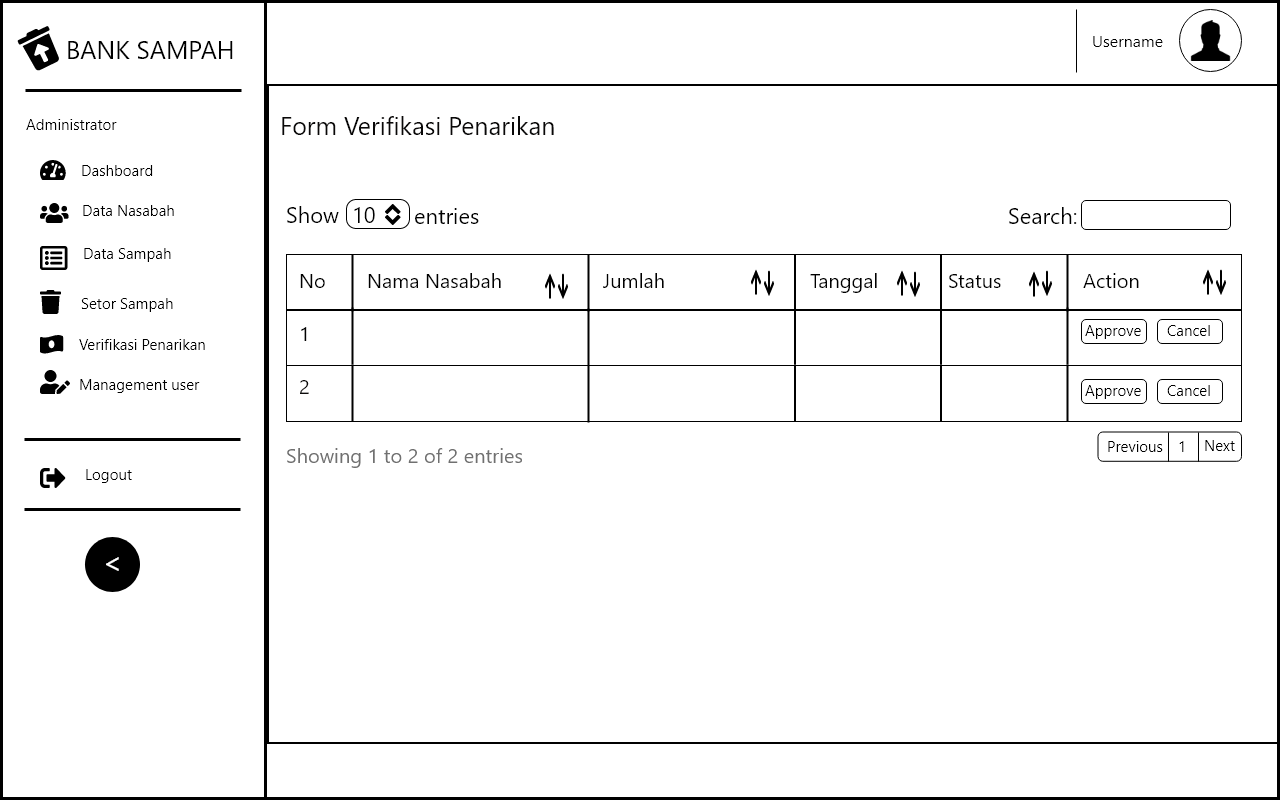
1. Halaman *Form Data Setor Sampah Admin*



Gambar 3. Rancangan Tampilan *Form* *Data Setor Sampah Admin*

Rancangan halaman pada form *Data Setor Sampah Admin*, dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

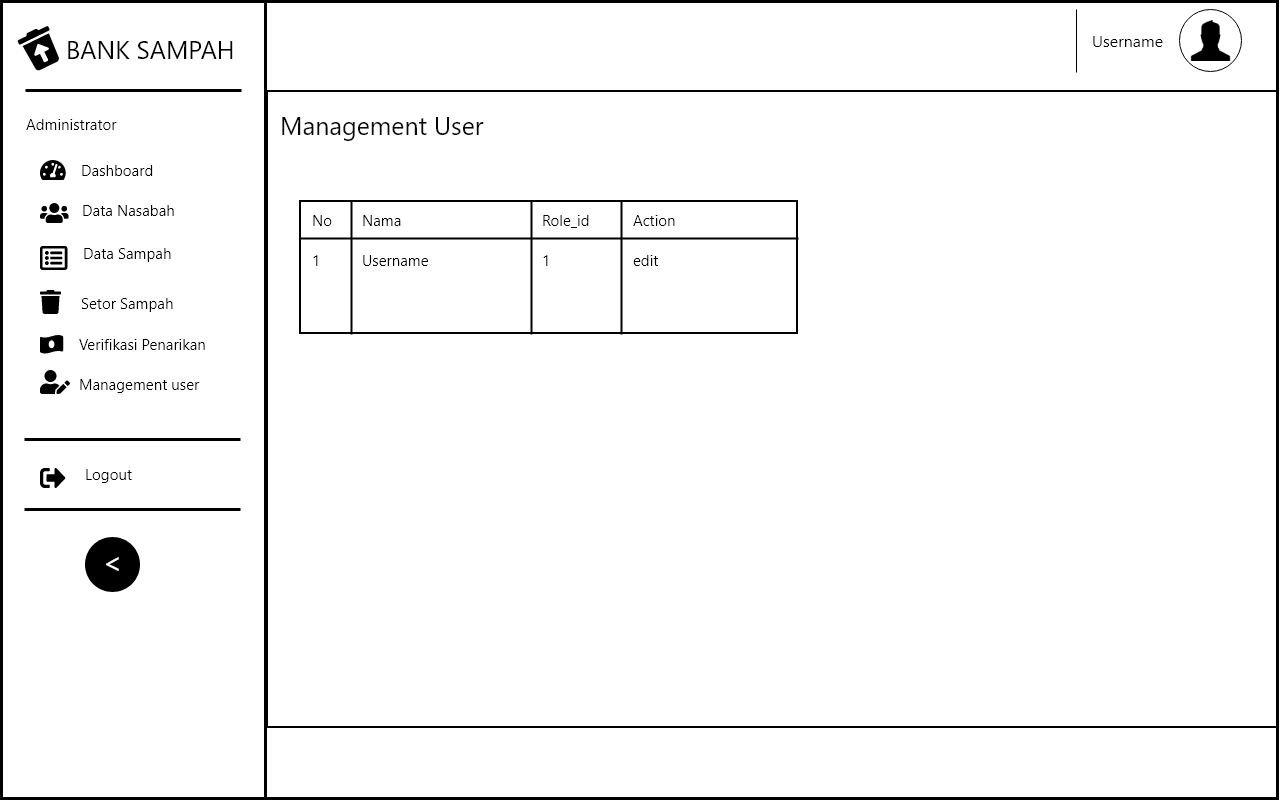
1. Halaman *Form Verifikasi Penarikan*



Gambar 3. Rancangan Tampilan *Form* *Verifikasi Penarikan*

Rancangan halaman pada form *Verifikasi Penarikan*, dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

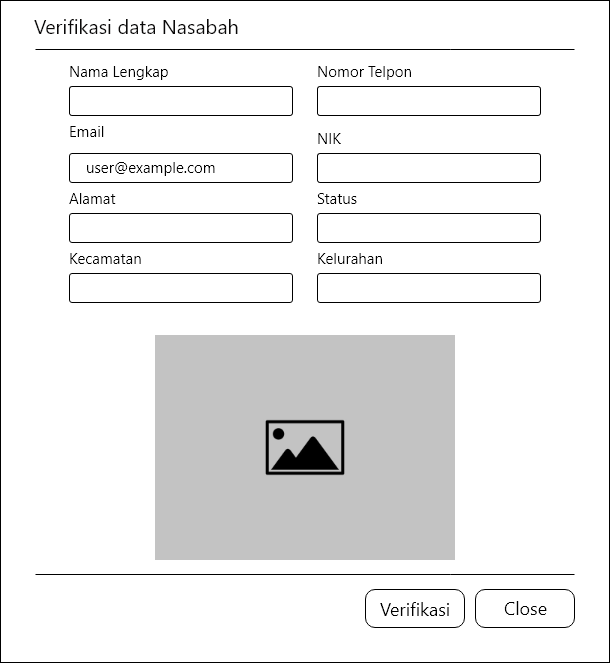
1. Halaman *Form Management User*



Gambar 3. Rancangan Tampilan *Form* *Management User*

Rancangan halaman pada form *Management User*, dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

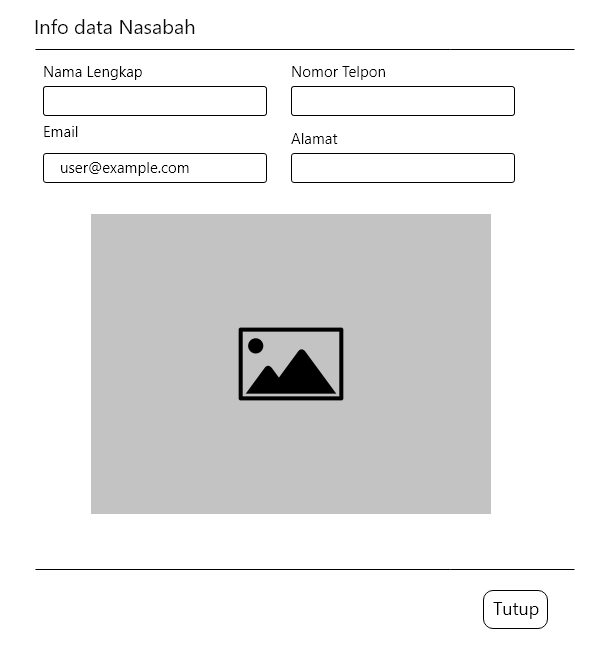
1. Halaman *Form Verifikasi Akun*



Gambar 3. Rancangan Tampilan *Form* *Verifikasi Akun*

Rancangan halaman pada form *Verifikasi Akun*, dengan adanya tampilan verifikasi admin dapat dengan mudah menyesuaikan data nasabah.

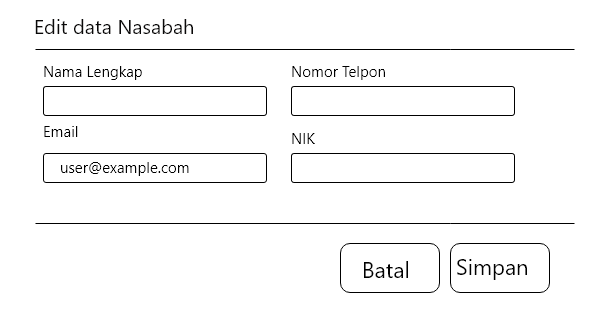
1. Halaman *Form Info Nasabah*



Gambar 3. Rancangan Tampilan *Form Info Nasabah*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

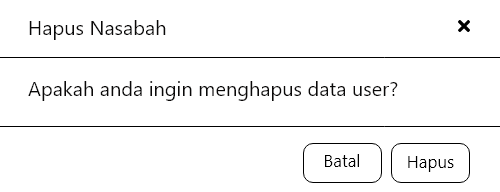
1. Halaman *Form Edit Nasabah*



Gambar 3. Rancangan Tampilan *Form Info Nasabah*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

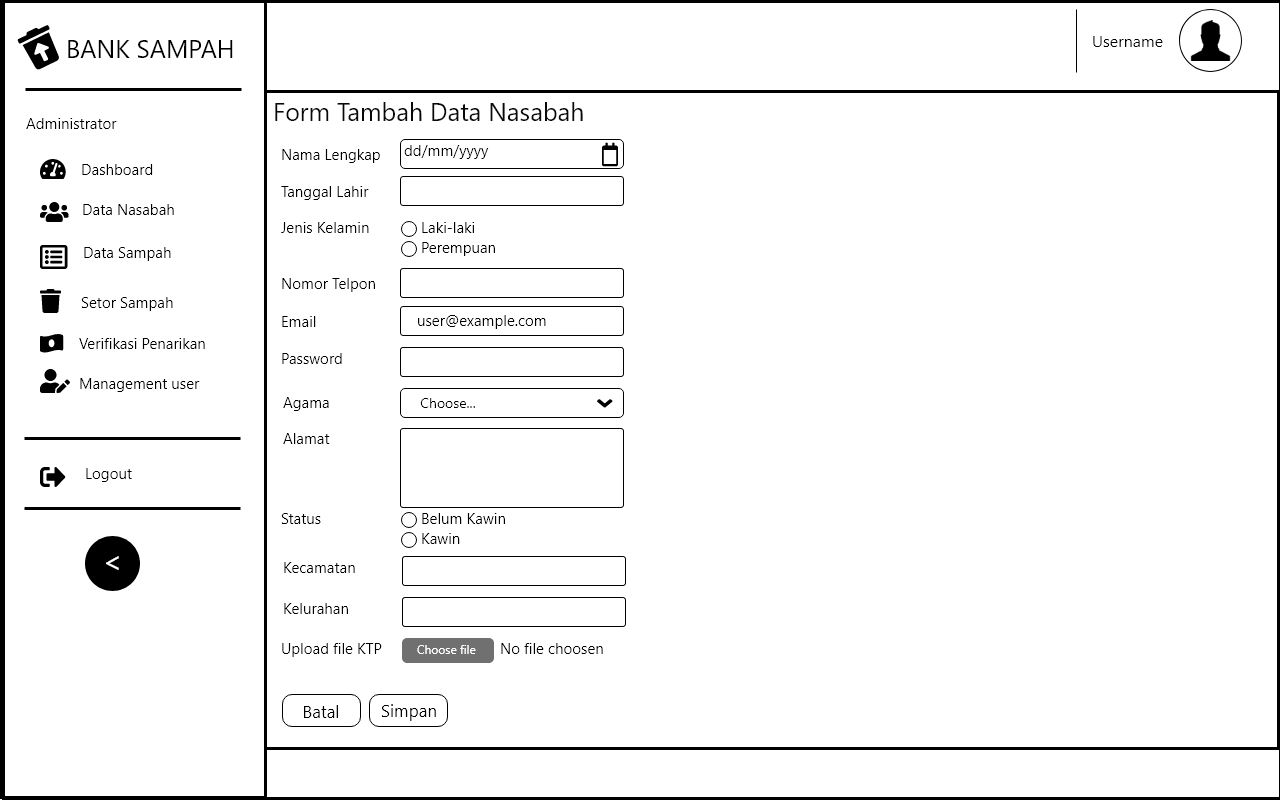
1. Halaman *Form Hapus Nasabah*



Gambar 3. Rancangan Tampilan *Form Hapus Nasabah*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

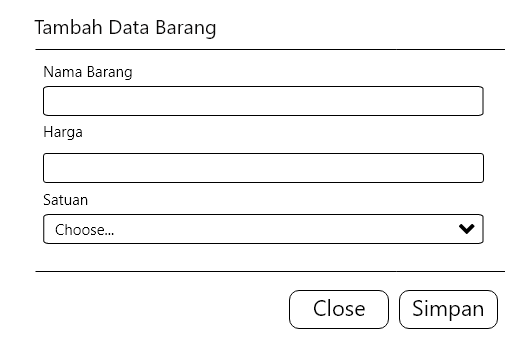
1. Halaman *Form Tambah Data Nasabah*



Gambar 3. Rancangan Tampilan *Form Tambah Data Nasabah*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

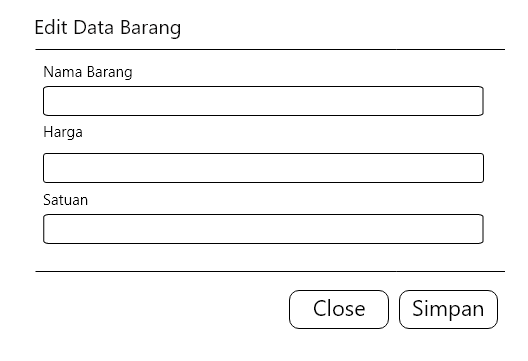
1. Halaman *Form Tambah Data Barang Sampah*



Gambar 3. Rancangan Tampilan *Form Tambah Data Barang Sampah*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

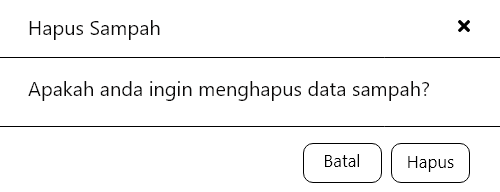
1. Halaman *Form Edit Data Barang Sampah*



Gambar 3. Rancangan Tampilan *Form Edit Data Barang Sampah*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

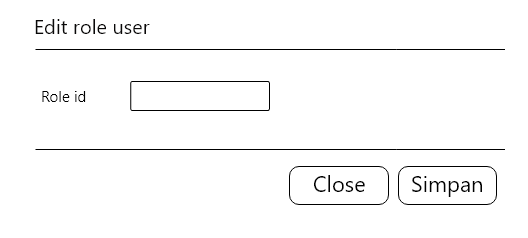
1. Halaman *Form Hapus Data Sampah*



Gambar 3. Rancangan Tampilan *Form Hapus Data Sampah*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

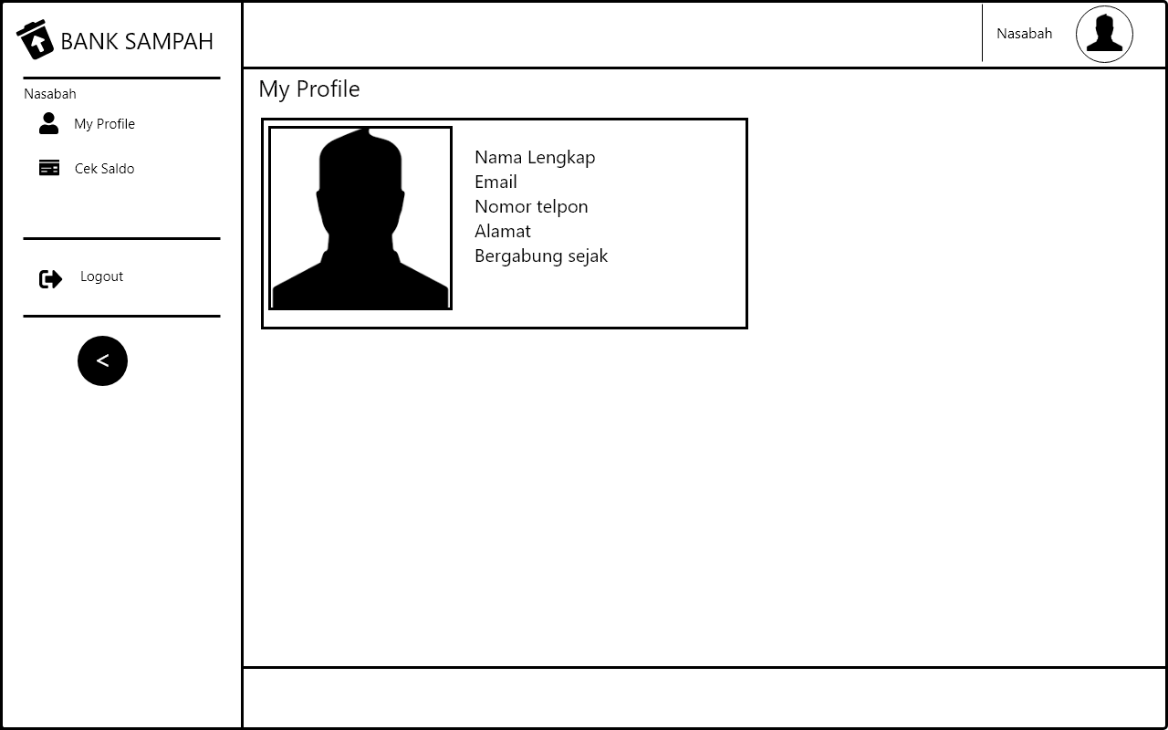
1. Halaman *Form Edit Role id*



Gambar 3. Rancangan Tampilan *Form Edit Role id*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

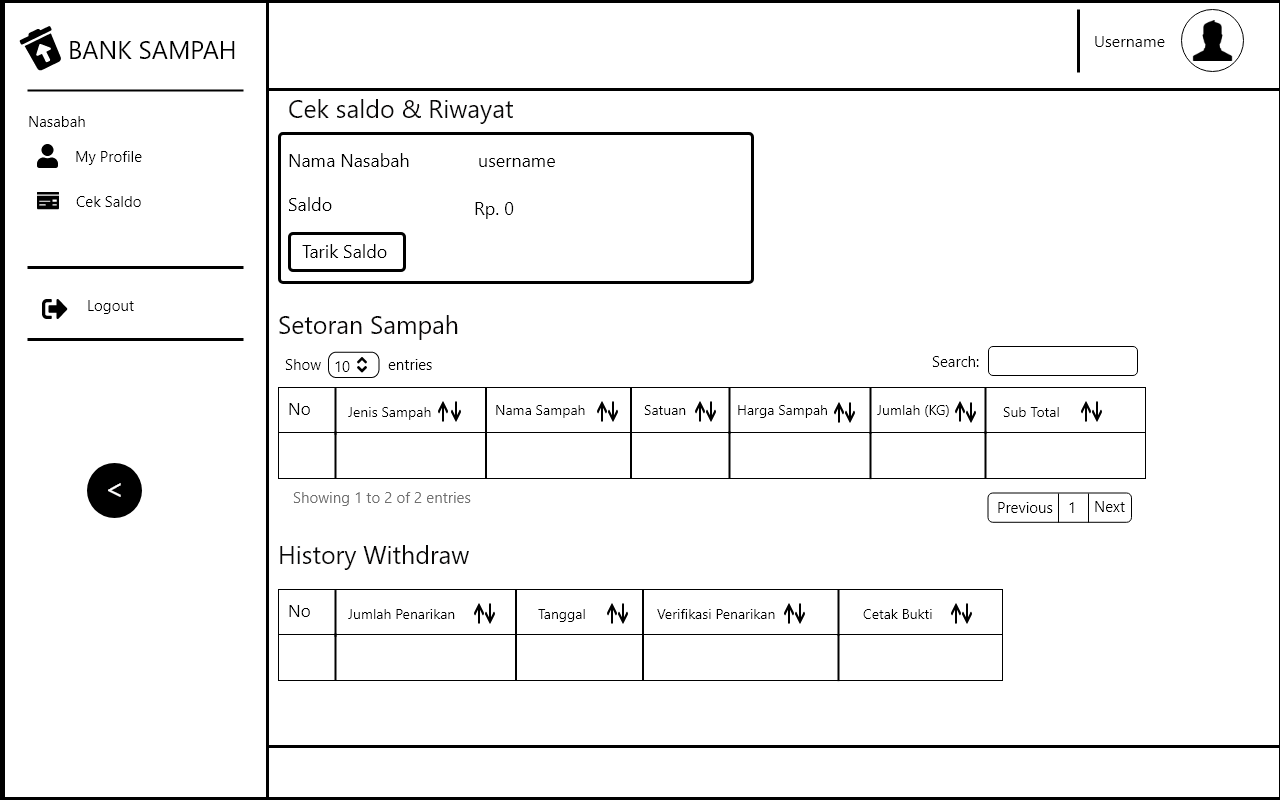
1. Halaman *Form Index Nasabah*



Gambar 3. Rancangan Tampilan *Form Index Nasabah*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

1. Halaman *Form Cek saldo dan Transaksi*



Gambar 3. Rancangan Tampilan *Form Cek saldo dan Transaksi*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

1. Halaman *Form*

Gambar 3. Rancangan Tampilan *Form*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

1. Halaman *Form*

Gambar 3. Rancangan Tampilan *Form*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

1. Halaman *Form*

Gambar 3. Rancangan Tampilan *Form*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

1. Halaman *Form*

Gambar 3. Rancangan Tampilan *Form*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

1. Halaman *Form*

Gambar 3. Rancangan Tampilan *Form*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.

1. Halaman *Form*

Gambar 3. Rancangan Tampilan *Form*

Rancangan halaman pada form , dibuat sederhana untuk memudahkan pengguna dalam melakukan proses.