

## **LAPORAN FINAL PROJECT**



### **RANCANGAN APLIKASI SISTEM INFORMASI MANAJEMEN KOST BERBASIS DEKSTOP**

Disusun Oleh :

1. Tazky Hafidzan E41231571
2. Nadila Filzah Widyawati E41231521
3. Annisa Ikrimatus Soleha E41231900
4. Muhammad Altaf Dirgantara E41231548
5. Mohammad Syahrur Rohman E41231687

**PROGRAM STUDI TEKNIK INFORMATIKA**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI JEMBER**

**2023**

## KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa berkat dan rahmat-Nya sehingga penulis dapat menyelesaikan Laporan Final Project yang berjudul “Aplikasi Manajemen Kos Berbasis Desktop” tepat pada waktunya. Adapun tujuan dari penulisan Laporan Final Project ini adalah untuk memenuhi tugas Mata Kuliah Workshop Basis Data dan Workshop Perangkat Lunak dalam Program Studi Teknik Informatika Politeknik Negeri Jember.

Pada kesempatan ini, penulis hendak menyampaikan terima kasih kepada semua pihak yang telah memberikan dukungan moril maupun materil, sehingga Laporan Final Project ini dapat selesai. Ucapan terimakasih ini penulis tujuhan kepada :

1. Bapak Syamsul Arifin, S.Kom, M.Cs selaku dosen pengampu Workshop Pengembangan Perangkat Lunak
2. Ibu Trismayanti Dwi P, S.Kom, M.Cs selaku dosen pengampu Workshop Pengembangan Perangkat Lunak
3. Ibu Elly Antika, ST, M.Kom. selaku dosen pengampu Workshop Pengembangan Perangkat Lunak
4. Bapak Prawidya Destarianto, S.Kom, M.T selaku dosen pengampu Workshop Basis Data
5. Ibu Bety Etikasari, S.Pd, M.Pd selaku dosen pengampu Workshop Basis Data
6. Ibu Damar Novtahaning, S.Tr.Kom., M.Sc selaku dosen pengampu Workshop Basis Data
7. Kepada teman-teman seperjuangan yang telah berkontribusi secara fisik maupun material dalam pembuatan Laporan Final Project ini.

Dengan segala kerendahan hati, penulis menyadari kekurangan dan kelemahan dalam penyajian Laporan Final Project. Hal ini terjadi karena keterbatasan ilmu pengetahuan dan kemampuan yang penulis miliki. Besar harapan penulis agar hal kecil ini dapat bermanfaat bagi perkembangan ilmu komputer, khususnya di lingkungan Politeknik Negeri Jember dan khalayak umum. Penulis mengharapkan saran dan kritik serta masukan yang membangun dari para pembaca guna menyempurnakan segala kekurangan dalam penyusunan Laporan Final Project ini.

Dalam penyusunan Laporan Final Project tidak sedikit penulis mengalami kesulitan, namun berkat bimbingan dari para dosen, teknisi, dan bantuan dari berbagai pihak maka kesulitan itu dapat diatasi. Akhir kata, penulis berharap semoga laporan ini dapat berguna bagi para pembaca dan pihak-pihak lain yang berkepentingan.

Jember, November 2023

Penyusun

## DAFTAR ISI

<b>KATA PENGANTAR.....</b>	<b>2</b>
<b>DAFTAR GAMBAR.....</b>	<b>5</b>
<b>BAB 1. PENDAHULUAN.....</b>	<b>8</b>
1.1 Latar Belakang.....	8
1.2 Rumusan Masalah.....	9
1.3 Tujuan dan Manfaat.....	9
<b>BAB 2. TINJAUAN PUSTAKA.....</b>	<b>10</b>
2.1 Aplikasi.....	10
2.2.2 Management.....	11
2.4 Desktop.....	11
2.4.1 Java.....	11
2.4.2 MySql.....	11
2.5 Flowchart.....	12
<b>BAB 3. METODE PENELITIAN.....</b>	<b>13</b>
3.1 UNDERSTAND.....	13
3.2 DIVERGE.....	19
3.3 DECIDE.....	24
3.4 PROTOTYPE.....	25
3.5 VALIDATE.....	36
3.5.1 Login.....	36
3.5.2 Beranda.....	36
3.5.3 Kamar.....	36
3.5.4 Profil.....	36
3.5.5 Keuangan.....	37
3.5.6 Transaksi.....	37
3.5.7 Laporan.....	37
3.5.8 LOGOUT.....	37
<b>BAB 4. PEMBAHASAN.....</b>	<b>38</b>
4.1 RANCANGAN DATABASE.....	38
4.2 FLOWCHART SYSTEM.....	42
4.3 DESIGN DAN KODE PROGRAM.....	43
5.1 KESIMPULAN.....	66
5.2 SARAN.....	66
<b>DAFTAR PUSTAKA.....</b>	<b>67</b>
<b>LAMPIRAN.....</b>	<b>68</b>

## DAFTAR GAMBAR

3.1 UNDERSTAND.....	13
Gambar 3.1 Hasil Understand Oleh Tazky.....	14
Gambar 3.2 Hasil Understand Oleh Dilla.....	15
Gambar 3.3 Hasil Understand Oleh Annisa.....	16
Gambar 3.4 Hasil Understand Oleh Altaf.....	17
Gambar 3.5 Hasil Understand Oleh Alung.....	18
3.2 DIVERGE.....	19
Gambar 3.6 Hasil Design Sprint Oleh Tazky.....	19
Gambar 3.7 Hasil Design Sprint Oleh Nadila.....	20
Gambar 3.8 Hasil Design Sprint Oleh Annisa.....	21
Gambar 3.9 Hasil Design Sprint Oleh Altaf.....	22
Gambar 3.10 Hasil Design Sprint Oleh Alung.....	23
3.4 PROTOTYPE.....	25
Gambar 3.12 Beranda.....	26
Gambar 3.13 Log In.....	26
Gambar 3.14 Sign Up.....	27
Gambar 3.15 Forgot Password.....	27
Gambar 3.16 Verification.....	28
Gambar 3.17 New Password.....	28
Gambar 3.18 Beranda.....	29
Gambar 3.19 Kamar.....	29
Gambar 3.20 Hapus Data Kamar.....	30
Gambar 3.21 Tambah Data Kamar.....	30
Gambar 3.22 Profil Tambah.....	31
Gambar 3.23 Profil Semua Penghuni.....	31
Gambar 3.24 Profil Penghuni.....	32
Gambar 3.25 Keuangan.....	32
Gambar 3.26 Tambah Keuangan.....	33
Gambar 3.27 Transaksi Keuangan.....	33
Gambar 3.28 Kwitansi Pembayaran Kauangan.....	34
Gambar 3.29 Laporan.....	34
Gambar 3.30 Data Laporan.....	35
Gambar 3.31 LOGOUT.....	35
4.1 RANCANGAN DATABASE.....	38
Gambar 4.1 Rancangan Database.....	38
Gambar 4.1.1 ERD Aplikasi.....	38
Gambar 4.1.2 Tabel Akun.....	39
Gambar 4.1.2 Tabel Pemilik.....	39
Gambar 4.1.2 Tabel Penyewa.....	40

Gambar 4.1.2 Tabel Kamar.....	40
Gambar 4.1.2 Tabel Transaksi.....	40
Gambar 4.1.2 Tabel Detail Bayar.....	41
Gambar 4.1.2 Tabel Tagihan.....	41
<b>4.2 FLOWCHART SYSTEM.....</b>	<b>42</b>
Gambar 4.2 Flowchart Admin.....	42
<b>4.3 DESIGN DAN KODE PROGRAM.....</b>	<b>43</b>
Gambar 4.3.1 Design Form Utama.....	43
Gambar 4.3.1 Design Form Login.....	43
Gambar 4.3.1 Design Form Register.....	44
Gambar 4.3.1 Design Form Beranda.....	44
Gambar 4.3.1 Design Form Profil.....	45
Gambar 4.3.1 Design Form Kamar.....	45
Gambar 4.3.1 Design Form Keuangan.....	46
Gambar 4.3.1 Design Form Pengeluaran.....	46
Gambar 4.3.1 Design Form Transaksi.....	47
Gambar 4.3.1 Design Form Riwayat Transaksi.....	47
Gambar 4.3.2 Kode Program Config.....	48
Gambar 4.3.2 Kode Form Login.....	48
Gambar 4.3.2 Kode Form Register.....	49
Gambar 4.3.2 Kode Form Utama.....	50
Gambar 4.3.2 Kode Form Load Table Kamar.....	51
Gambar 4.3.2 Kode Form Tampilan Jumlah Kamar.....	52
Gambar 4.3.2 Kode Form Cari Data Kamar.....	52
Gambar 4.3.2 Kode Form Tambah Data Kamar.....	53
Gambar 4.3.2 Kode Form Edit Data Kamar.....	54
Gambar 4.3.2 Kode Form Hapus Data Kamar.....	54
Gambar 4.3.2 Kode Form Reset Data Kamar.....	55
Gambar 4.3.2 Kode Load Table Profil.....	55
Gambar 4.3.2 Kode Combo Box Profil.....	56
Gambar 4.3.2 Kode Tampilan Profil.....	56
Gambar 4.3.2 Kode Tambah Profil.....	57
Gambar 4.3.2 Kode Edit Profil.....	57
Gambar 4.3.2 Kode Aksi Edit.....	58
Gambar 4.3.2 Kode Aksi Delete.....	58
Gambar 4.3.2 Kode Reset Profil.....	59
Gambar 4.3.2 Kode Load Table Keungan.....	59
Gambar 4.3.2 Kode Cari Data Keuangan.....	60
Gambar 4.3.2 Kode Tampilan Jumlah Pemasukan.....	60
Gambar 4.3.2 Kode Tampilan Jumlah Pengeluaran.....	61
Gambar 4.3.2 Kode Button Tambah Dan Reset Pengeluaran.....	61
Gambar 4.3.2 Kode Button Kembali dan Tambah Pengeluaran.....	61
Gambar 4.3.2 Kode Tampilan Jumlah Pengeluaran.....	62
Gambar 4.3.2 Kode Combo Box Transaksi.....	62

Gambar 4.3.2 Kode Bayar Transaksi.....	63
Gambar 4.3.2 Kode Lanjut Transaksi.....	63
Gambar 4.3.2 Kode Batal Transaks.....	64
Gambar 4.3.2 Kode Load Tabel Riwayat.....	64
Gambar 4.3.2 Kode Tampilan Nilai Jumlah Tagihan.....	65
Gambar 4.3.2 Kode Reset Riwayat.....	65

## BAB 1. PENDAHULUAN

### 1.1 Latar Belakang

Di zaman yang semakin modern, ilmu pengetahuan dan teknologi pun mengalami perkembangan secara dinamis. Secara tidak langsung dengan adanya kemajuan teknologi telah mempengaruhi segala aspek kehidupan, baik dalam bidang politik, ekonomi, sosial budaya, bahkan dalam bidang pendidikan. Peranan teknologi dapat memudahkan setiap pekerjaan yang pada awalnya manual menjadi terotomatisasi (Indiharto, 2016). Salah satu teknologi yang sering digunakan saat ini adalah Komputer/Laptop.

Perkembangan teknologi menjadikan kebutuhan akan penggunaan perangkat Teknologi Informasi dan Komunikasi (TIK) yang mobile seperti laptop menjadi sangat tinggi. Hal ini terkait dengan mobilitas dan kemudahan dalam penggunaan. Jika dilihat berdasarkan aktivitas penggunaan laptop saat tidak terhubung dengan internet, dalam survei dari 2.121 responden. Aktivitas tertinggi adalah menonton video dan mendengarkan musik, yakni 38,66 persen dan yang terendah membuat komputer dan coding hanya 3,68 persen. Dan untuk aktivitas penggunaan internet saat terhubung dengan internet dengan 4238 responden, aktivitas yang paling banyak dilakukan ialah web browsing, sekitar 74,69 persen. Selanjutnya untuk komunikasi melalui internet sebesar 42,63 persen.

Saat ini laptop tidak hanya digunakan sebagai komunikasi, melainkan sebagai alat untuk membantu kegiatan sehari-hari. Salah satunya adalah proses manajemen. Manajemen merupakan salah satu aspek penting dalam kehidupan manusia. Dengan adanya manajemen yang baik, maka kegiatan yang akan dilakukan dapat berjalan dengan baik. Salah satunya yakni dapat diterapkan dalam manajemen penyewaan kost.

Manajemen penyewaan kost bertujuan untuk memberi kemudahan bagi pemilik kost dalam mengelola penyewaan, pendataan, dan keuangan calon penghuni kost. Pada kost Omah Muslimah yang terletak di Jl. Mastrip Gg VII No. 7, Kecamatan Sumbersari, Jember yang memiliki 24 kamar kost dengan 2 tipe kamar dan memiliki harga yang berbeda. Manajemen keuangan penyewaan kost di kost ini masih menggunakan cara yang konvensional, seperti menggunakan catatan di kertas atau hanya sekadar mengingat saja. Dalam hal sewa kamar kost pun masih dilakukan secara manual. Jika sistem yang masih manual ini tetap digunakan, menyebabkan pemilik kesulitan dalam pengecekan data penyewa yang telah membayar atau belum. Hal serupa juga dengan data kamar yang memiliki fasilitas berbeda, tentunya memiliki harga yang berbeda pula. Seringkali pemilik kost dihadapkan pada masalah data penyewa yang kurang lengkap, sementara data penyewa tersebut wajib dilaporkan pada rukun tetangga yang bersangkutan.

Dengan adanya aplikasi manajemen kost di kost Omah Muslimah berbasis desktop ini diharapkan dapat mengatasi kendala yang dialami pemilik kost yakni pada permasalahan sistem yang sedang berjalan dan memudahkan proses manajemen kost kedepannya. Berdasarkan uraian di atas, maka penulis akan merancang sebuah sistem informasi manajemen kost berbasis desktop yang berguna bagi pemilik dalam mengelola kost.

## **1.2 Rumusan Masalah**

- a. Bagaimana rancangan database pada sistem informasi manajemen kost ?
- b. Bagaimana flowchart system pada sistem informasi manajemen kost ?
- c. Bagaimana desain dan kode program pada sistem informasi manajemen kost ?

## **1.3 Tujuan dan Manfaat**

- a. Menyediakan sistem informasi manajemen penyewaan kamar kost yang meliputi data penghuni kamar, status kamar, dan keuangan kost
- b. Memudahkan pemilik kos untuk mengelola kost dengan aplikasi tersebut dengan harapan dapat menghasilkan informasi yang valid

## BAB 2. TINJAUAN PUSTAKA

### 2.1 Aplikasi

Menurut Hengky (2021), aplikasi adalah suatu perangkat lunak yang dirancang untuk memenuhi berbagai jenis aktivitas dan pekerjaan manusia. Contohnya adalah aplikasi untuk periklanan, aktivitas bisnis, layanan masyarakat, permainan, dan sebagainya.

Istilah "aplikasi" berasal dari bahasa Inggris, yang berarti "penerapan" atau "penggunaan." Secara harfiah, aplikasi adalah aplikasi software atau perangkat lunak yang dirancang untuk melakukan fungsi tertentu.

Dalam pengembangannya, aplikasi dapat dikategorikan dalam tiga kelompok, diantaranya;

- a. **Aplikasi Desktop**, yaitu aplikasi yang hanya dijalankan di perangkat PC komputer atau laptop.
- b. **Aplikasi Web**, yaitu aplikasi yang dijalankan menggunakan komputer dan koneksi internet.
- c. **Aplikasi Mobile**, yaitu aplikasi yang dijalankan di perangkat mobile di mana untuk kategori ini penggunaannya sudah banyak sekali.

(<https://www.liputan6.com/hot/read/5287655/aplikasi-adalah-program-perangkat-lunak-ketahui-fungsi-dan-jenisnya>)

### 2.2 Aplikasi Management kost

#### 2.2.1 Aplikasi

Secara istilah pengertian aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanakan suatu fungsi bagi pengguna jasa aplikasi serta penggunaan aplikasi lain yang dapat digunakan oleh suatu sasaran yang akan dituju. Menurut kamus komputer eksekutif, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu teknik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputasi yang diinginkan atau diharapkan maupun pemrosesan data yang diharapkan.

### **2.2.2 Management**

Organisasi Standarisasi Internasional (ISO) menyatakan bahwa sistem manajemen adalah suatu teknik yang digunakan oleh suatu organisasi untuk mengelola berbagai bagian bisnis yang berhubungan satu sama lain agar mereka dapat mencapai tujuan mereka.

([Sistem Manajemen: Pengertian, Tujuan, dan Implementasinya \(ppmschool.ac.id\)](http://Sistem%20Manajemen:%20Pengertian,%20Tujuan,%20dan%20Implementasinya%20(ppmschool.ac.id)))

### **2.2.3 Kost**

Kost merupakan suatu tempat tinggal yang disewakan oleh pemilik kost dengan fasilitas-fasilitas dan harga tertentu. Kost sendiri lebih banyak dikenal sebagai domisili karena ditempati dalam waktu yang cukup lama dimana istilah kost sendiri sudah sangat melekat pada mahasiswa dikarenakan banyak pemilik kost yang lebih memilih untuk menyewakan kost miliknya kepada mahasiswa yang berdomisili jauh dari kampus tempat ia belajar. Kost juga merupakan investasi jangka panjang yang menjanjikan untuk zaman sekarang karena keuntungan yang didapat bisa dihasilkan setiap bulannya.

## **2.4 Desktop**

Menurut Tech Target, desktop adalah bagian tampilan komputer yang terletak di halaman utama dan biasanya berisi aplikasi atau objek tertentu, seperti folder dan file. Desktop juga sering digunakan sebagai tempat untuk menyimpan dan mengakses file atau item yang sering digunakan. menjadikannya lebih mudah untuk diakses saat digunakan daripada harus mencari melalui fitur pencarian Windows.

([Pengertian Desktop, Fitur-fitur, dan Fungsinya di Windows \(kompas.com\)](http://Pengertian%20Desktop,%20Fitur-fitur,%20dan%20Fungsinya%20di%20Windows%20(kompas.com)))

### **2.4.1 Java**

Java merupakan bahasa pemrograman yang digunakan secara luas untuk pengkodean aplikasi web. Bahasa ini telah menjadi pilihan populer di antara developer selama lebih dari dua dekade, dengan jutaan aplikasi Java yang digunakan saat ini. Java merupakan bahasa multiplatform yang berorientasi pada objek dan berpusat pada jaringan yang dapat digunakan sebagai platform di dalamnya. Java merupakan bahasa pemrograman yang cepat, aman, dan andal untuk mengodekan segala sesuatu mulai dari aplikasi seluler dan perangkat lunak korporasi hingga aplikasi *big data* dan teknologi sisi server.

([Apa itu Java? - Penjelasan tentang Bahasa Pemrograman Java - AWS \(amazon.com\)](http://Apa%20itu%20Java?%20-%20Penjelasan%20tentang%20Bahasa%20Pemrograman%20Java%20-%20AWS%20(amazon.com)))

### **2.4.2 MySql**

MySQL adalah salah satu jenis database yang banyak digunakan untuk membuat aplikasi berbasis web yang dinamis. MySQL termasuk jenis RDBMS (Relational Database Management System). MySQL ini mendukung Bahasa pemrograman PHP. MySQL juga mempunyai query atau

bahasa SQL(Structured Query Language) yang simple dan menggunakan escape character yang sama dengan PHP. MySQL adalah sebuah implementasi dari sistem manajemen basisdata relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (General Public License). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial.

(JTIM: Jurnal Teknik Informatika Mahakarya 2 (2), (2019))

## 2.5 Flowchart

Flowchart adalah grafik yang menggambarkan prosedur program dalam urutan langkah-langkah. Flowchart berguna dalam proses analisis, perancangan, dan pengkodean karena mereka membagi masalah ke dalam bagian yang lebih kecil untuk digunakan. Pada evaluasi lebih lanjut, flowchart biasanya membantu menyelesaikan masalah. Sebuah flowchart juga dapat didefinisikan sebagai sebuah diagram yang menggunakan simbol grafis untuk menunjukkan aliran proses yang menampilkan beberapa langkah. Selain itu, flowchart juga dapat dianggap sebagai gambaran grafik dari langkah-langkah atau urutan langkah dari suatu prosedur program yang melakukan fungsi tertentu. Untuk membuat proses produksi mudah dipahami dan mudah dilihat, fungsi flowchart digunakan. Flowchart disusun dalam urutan langkah-langkah proses. Selain itu, membuat rangkaian prosedur lebih mudah dipahami pengguna. (Rochaety, dkk. 2023)

## 2.6 Canva

Canva adalah program desain online yang menawarkan berbagai alat seperti presentasi, resume, poster, pamflet, brosur, grafik, infografis, spanduk, penanda buku, bulletin, dan lain-lain. Jenis presentasi yang tersedia di Canva termasuk presentasi kreatif, bisnis, teknologi, dan periklanan. Menurut Tanjung dan Faiza (2019), beberapa kelebihan aplikasi Canva adalah memiliki banyak desain yang menarik, memungkinkan guru dan siswa untuk lebih kreatif dalam mendesain media pembelajaran karena banyak fitur yang tersedia, secara praktis menghemat waktu dalam media pembelajaran, dan desain dapat dilakukan melalui ponsel.

## **BAB 3. METODE PENELITIAN**

### **3.1 UNDERSTAND**

1. Siapa pengguna aplikasi tersebut
  - Pemilik kost
  - Admin
2. Apa yang mereka butuhkan
  - Laporan keuangan
  - Data penghuni
  - Laporan pembayaran
  - List data kamar
  - Nota Pembayaran
3. Apa masalah utama pengguna utama yang harus diselesaikan
  - Laporan keuangan yang tidak tercatat secara digital
  - Kesulitan mengetahui data penghuni setiap kamar
  - Kesulitan mendata ketersediaan kamar
4. Kompetitor
  - Kosanku
  - Pak Kos
  - Juragan Kost
5. Formulasi strategi penyelesaian

Berdasarkan permasalahan yang dialami oleh pemilik kost maka dibuatkan aplikasi berbasis desktop untuk management kost dengan fitur :

- Laporan keuangan
- Profil penghuni
- List data kamar

Berikut merupakan hasil understand setiap anggota kelompok :

### \* Understanding \*

1. Siapa pengguna aplikasi tersebut?  
Pemilik kost
2. Apa yang mereka butuhkan?
  - Laporan keuangan
  - List kamar
  - Profil penghuni
  - Laporan pembayaran
  - Nota pembayaran
3. Apa masalah utama pengguna yang harus diselesaikan?
  - Kesulitan untuk melihat profil penghuni
  - Laporan keuangan tidak tercatat secara digital
  - Setiap ingin mengetahui kamar tersedia, harus cek kamar satu-satu
4. Kompetitor
  - Kosanaku
  - Pakkos
  - Juragankost
5. Formulasikan strategi penyelesaian!Berdasarkan permasalahan di atas, maka akan dibuat sebuah aplikasi berbasis desktop dengan fitur:
  - Dapat menampilkan laporan keuangan
  - Dapat menampilkan data profil penghuni
  - Dapat menampilkan list kamar yang tersedia (termasuk jenis kamar dan harganya)

Gambar 3.1 Hasil Understand Oleh Tazky

### \* Understanding

1. Siapa pengguna aplikasi tersebut ?

pemilik Kost (Admin)

2. Apa yg mereka butuhkan ?

✓ List Kamar

✓ Data penyewa

✓ Laporan Keuangan

✓ Laporan Pembayaran

✓ Nota Pembayaran

3. Apa masalah utama pengguna yang harus diselesaikan ?

Kesulitan untuk mengetahui laporan keuangan secara digital, melihat profil penghuni, dan untuk mengetahui kamar yang tersedia tanpa harus memeriksa satu-satu.

4. Kompetitor

a) KosAnku

b) POKKOS

c) Juragan Kost

5. Formulasikan strategi penyelesaian !

Berdasarkan permasalahan di atas, maka akan dibuat sebuah aplikasi berbasis desktop dg fitur,

- dpt menampilkan laporan keuangan
- dpt menampilkan data profil penghuni
- dpt menampilkan list kamar yg tersedia  
(termasuk jenis kamar dan harganya)

Gambar 3.2 Hasil Understand Oleh Dilla

-Understanding

1. Siapa pengguna aplikasi tersebut?

Jawab: peneliti fos / admin  
penghuni fos

2. Apa yang mereka butuhkan?

Jawab: data penghuni fos  
Laporan pengeluaran dan pemasukan

3. Apa masalah utama pengguna yang harus  
di selesaikan?

Jawab: - peneliti fos / admin  
kesesuaian dalam pendataan laporan  
keuangan dan dalam pengelolaan  
fos, tiap hari dan bulannya

4. Kompetitor

- fosantu
- pak fos
- juragan kost

5. Formulirkan Strategi penyelesaian

Berdasarkan masalah yang terjadi pada  
peneliti fos terhadap susahnya penyelesaian  
kost maka dibuatkan aplikasi berbasis  
desktop dengan fitur home, laporan, catatan,  
keluhan, pengaturan

Gambar 3.3 Hasil Understand Oleh Annisa

### Understanding

1. Siapa Pengguna aplikasi tersebut ?  
: Pemilik Kost
2. Apa yang mereka butuhkan ?
  - Laporan keuangan
  - List kamar
  - Profil Penghuni
  - Laporan pembayaran
  - Nota pembayaran
3. Apa masalah utama pengguna yang harus diselesaikan ?
  - Kesulitan untuk melihat profil penghuni
  - Laporan keuangan tidak tercatat secara digital.
  - Setiap ingin mengetahui kamar tersedia harus cek kamar satu-satu.
4. Kompetitor
  - Kosanku
  - Pakkos
  - Juragan kost
5. Formulasikan strategi penyelesaian ?
  - : Berdasarkan permasalahan di atas, maka akan dibuat sebuah aplikasi berbasis desktop dengan fitur :
    - Dapat menampilkan laporan keuangan
    - Dapat menampilkan data profil penghuni
    - Dapat menampilkan list kamar yang tersedia (termasuk jenis kamar dan harganya).

Gambar 3.4 Hasil Understand Oleh Altaf

### Under Standing

1. Siapa Pengguna Aplikasi tersebut ?
  - a. Pemilik kost
2. Apa yang mereka butuhkan ?
  - a. Laporan Keuangan
  - b. List kamar tersedia beserta Penghuni nya
  - c. Laporan dan Nota Pembayaran
3. Apa Masalah Utama Pengguna Yang harus di selesaikan ?
  - a. Sedikit kendala saat melihat Profil Penghuni
  - b. Laporan keuangan tidak tercetak secara digital
  - c. Saya ingin mengetahui kamar yang tersedia harus mengecek kamar satu-satu
4. Kompetitor ~
  - a. kostku
  - b. pak kos
  - c. Juragan Kost

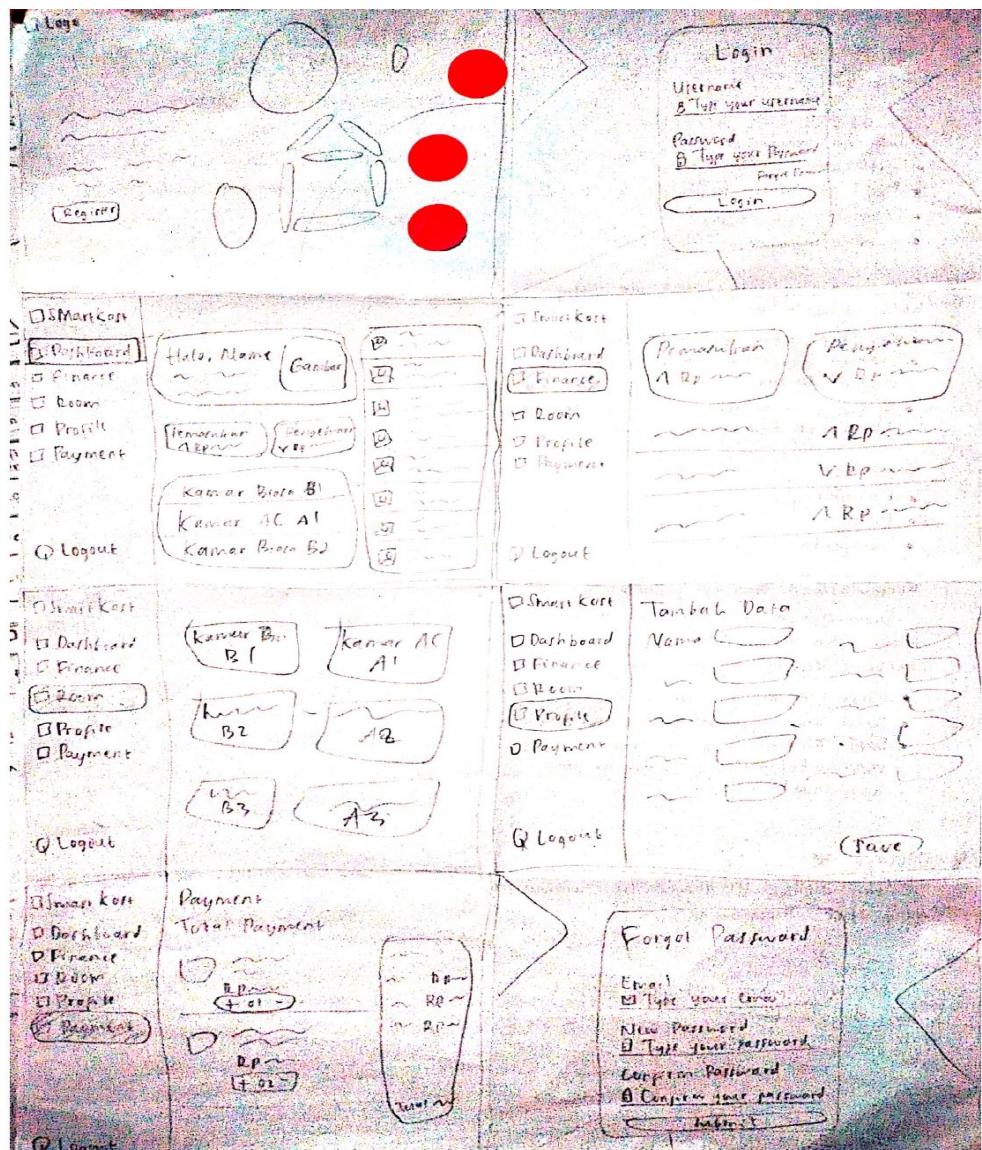
### 5. Formulasikan Strategi Penyelesaian !

- a. berdasarkan permasalahan diatas, maka akan di buat sebuah aplikasi berbasis Desktop dengan fitur :
  - Dapat Menampilkan Laporan Keuangan
  - Dapat Menampilkan data Profil penghuni
  - Dapat Menampilkan List kamar yang tersedia, termasuk jenis kamar dan harganya

Gambar 3.5 Hasil Understand Oleh Alung

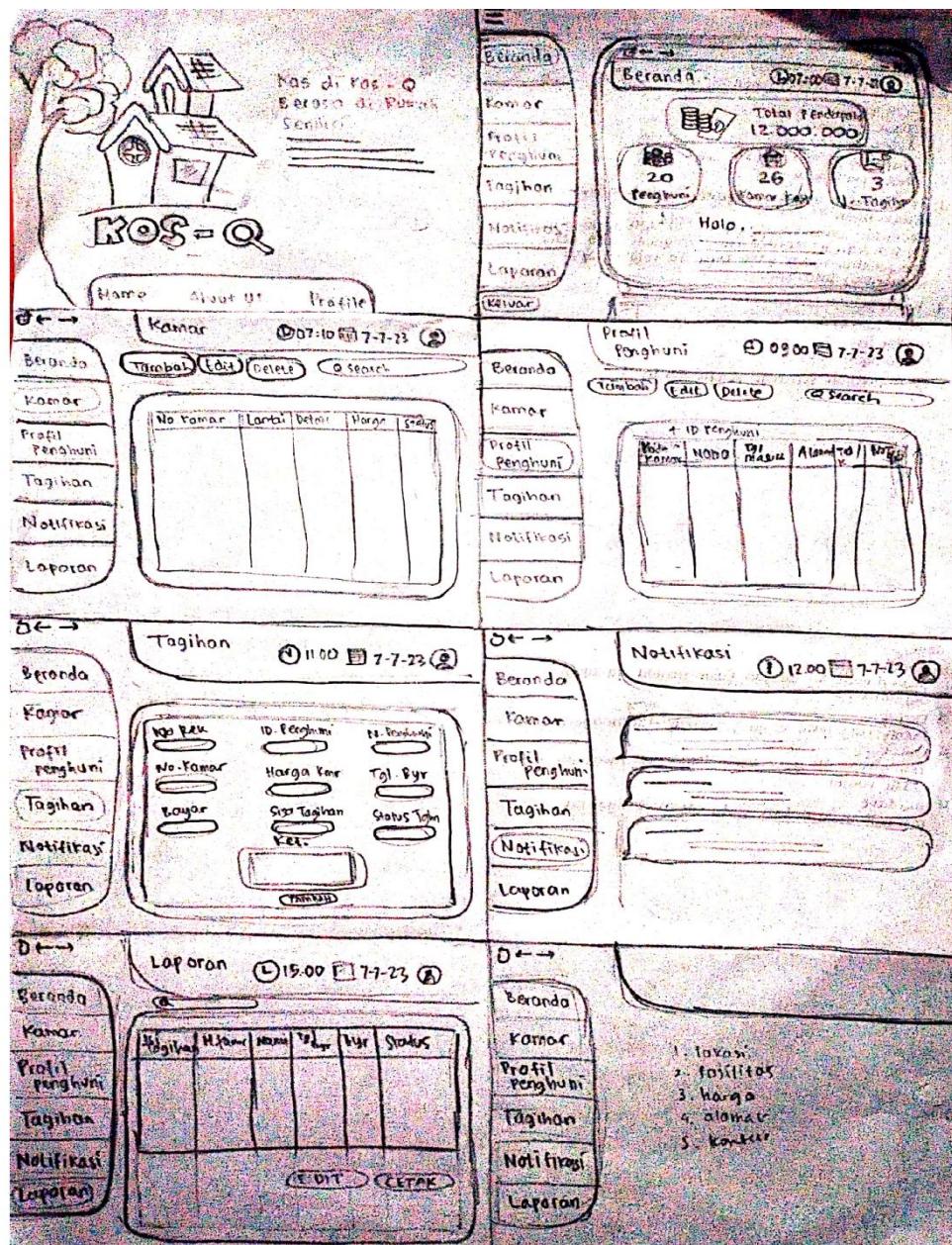
### 3.2 DIVERGE

Storyboard Tazky Hafidzan



Gambar 3.6 Hasil Design Sprint Oleh Tazky

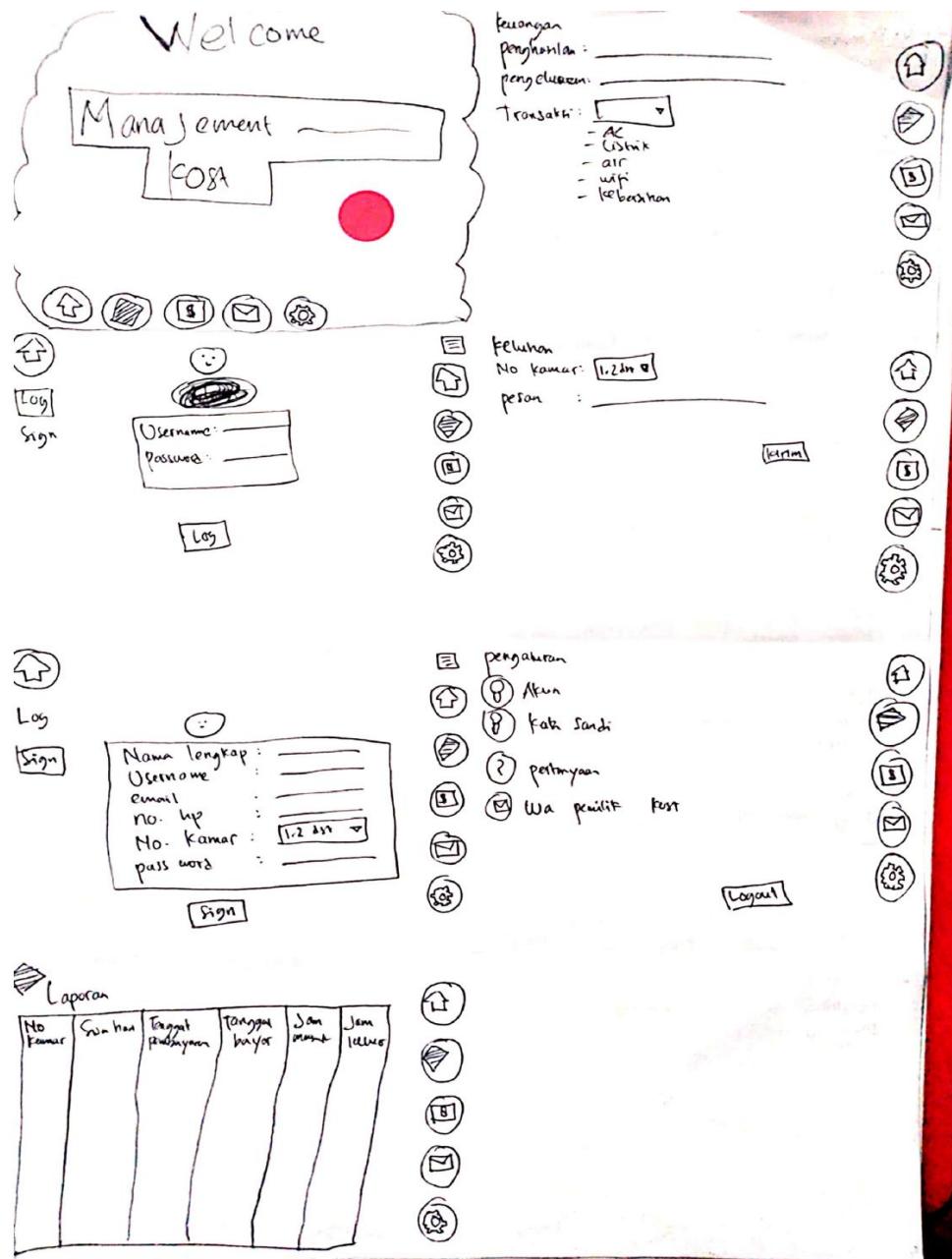
## Storyboard Nadila Filzah Widyawati



Dipindai dengan CamScanner

Gambar 3.7 Hasil Design Sprint Oleh Nadila

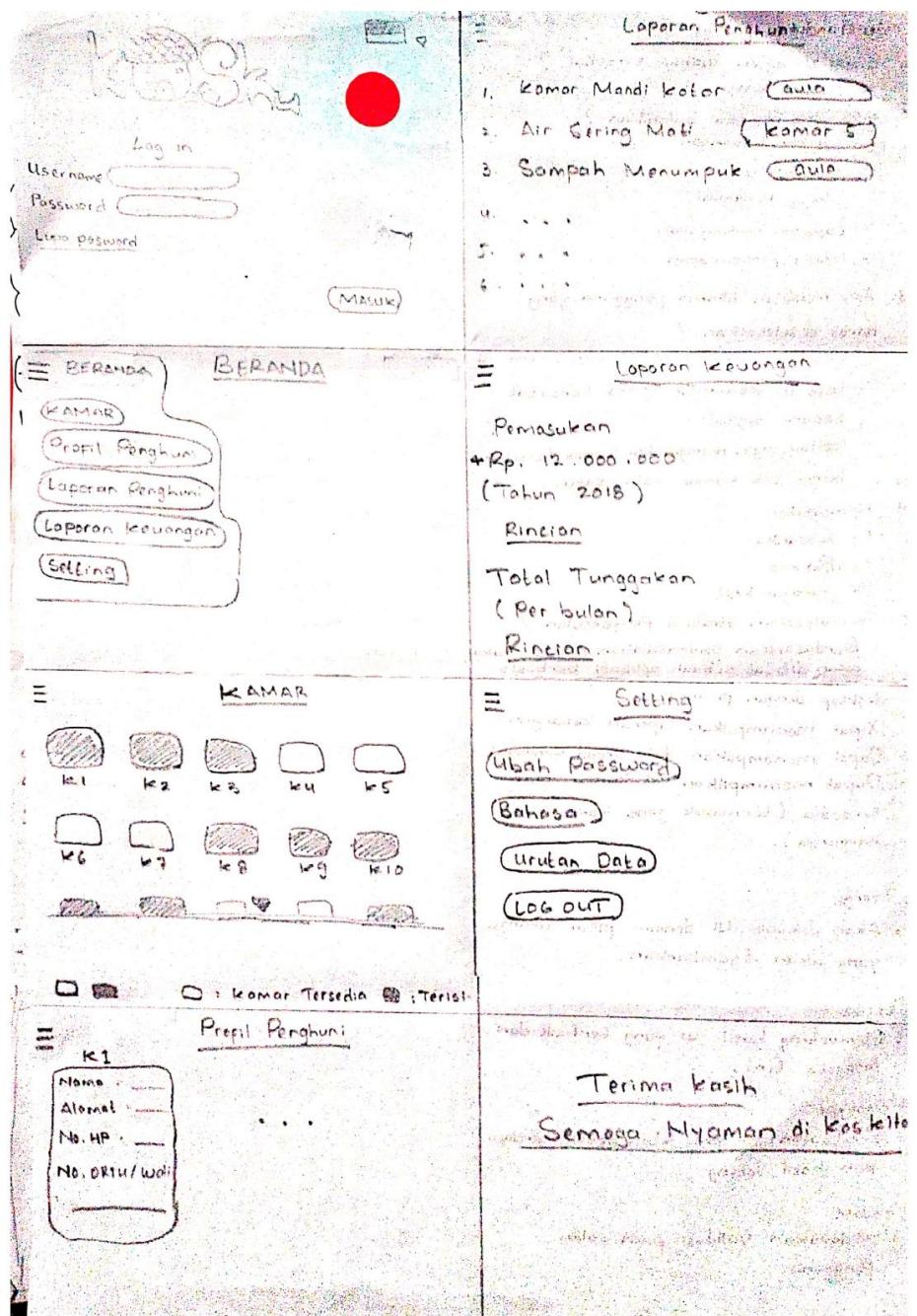
## Storyboard Annisa Ikrimatus Soleha



Dipindai dengan CamScanner

Gambar 3.8 Hasil Design Sprint Oleh Annisa

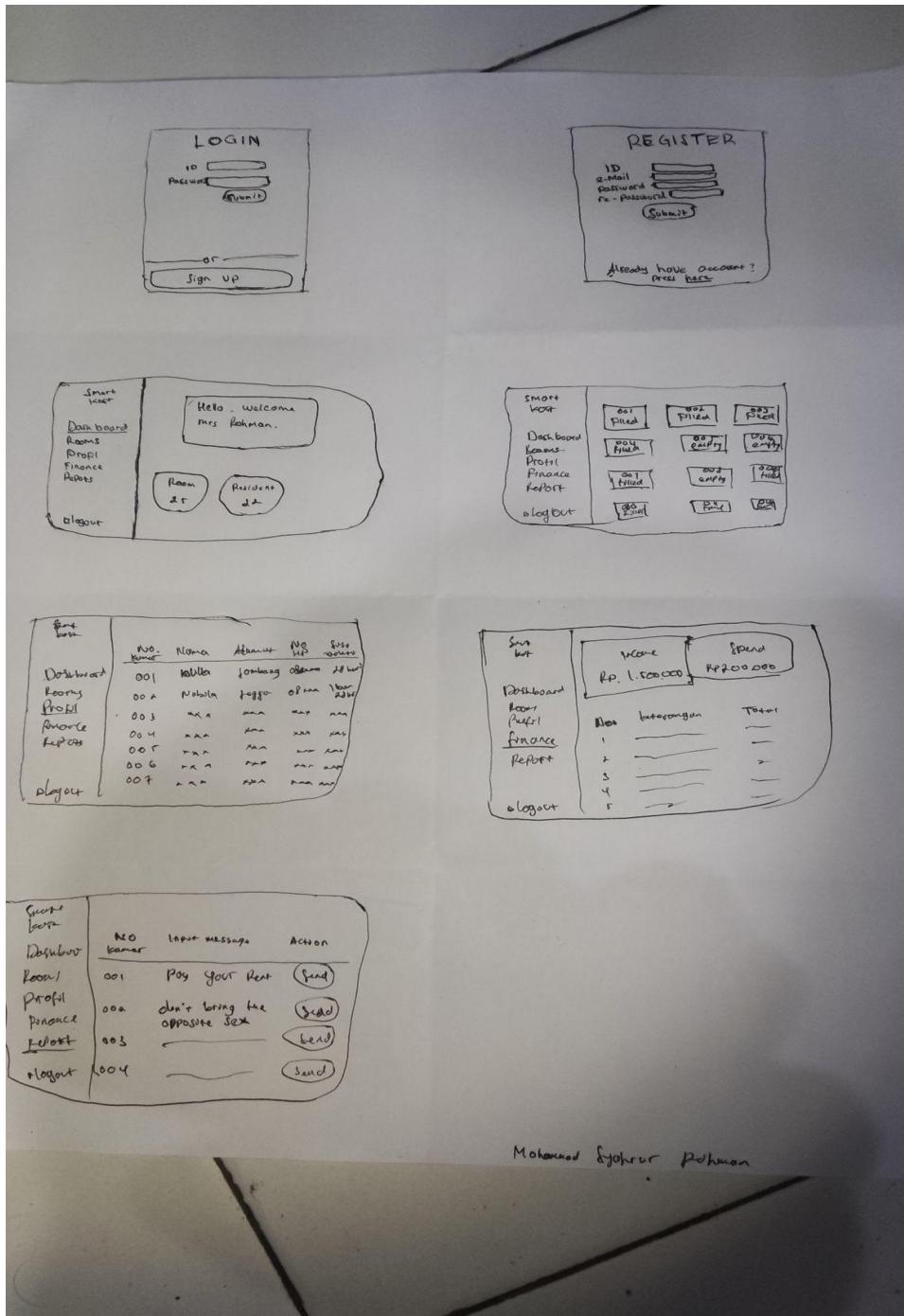
## Storyboard Muhammad Altaf Dirgantara



Dipindai dengan CamScanner

Gambar 3.9 Hasil Design Sprint Oleh Altaf

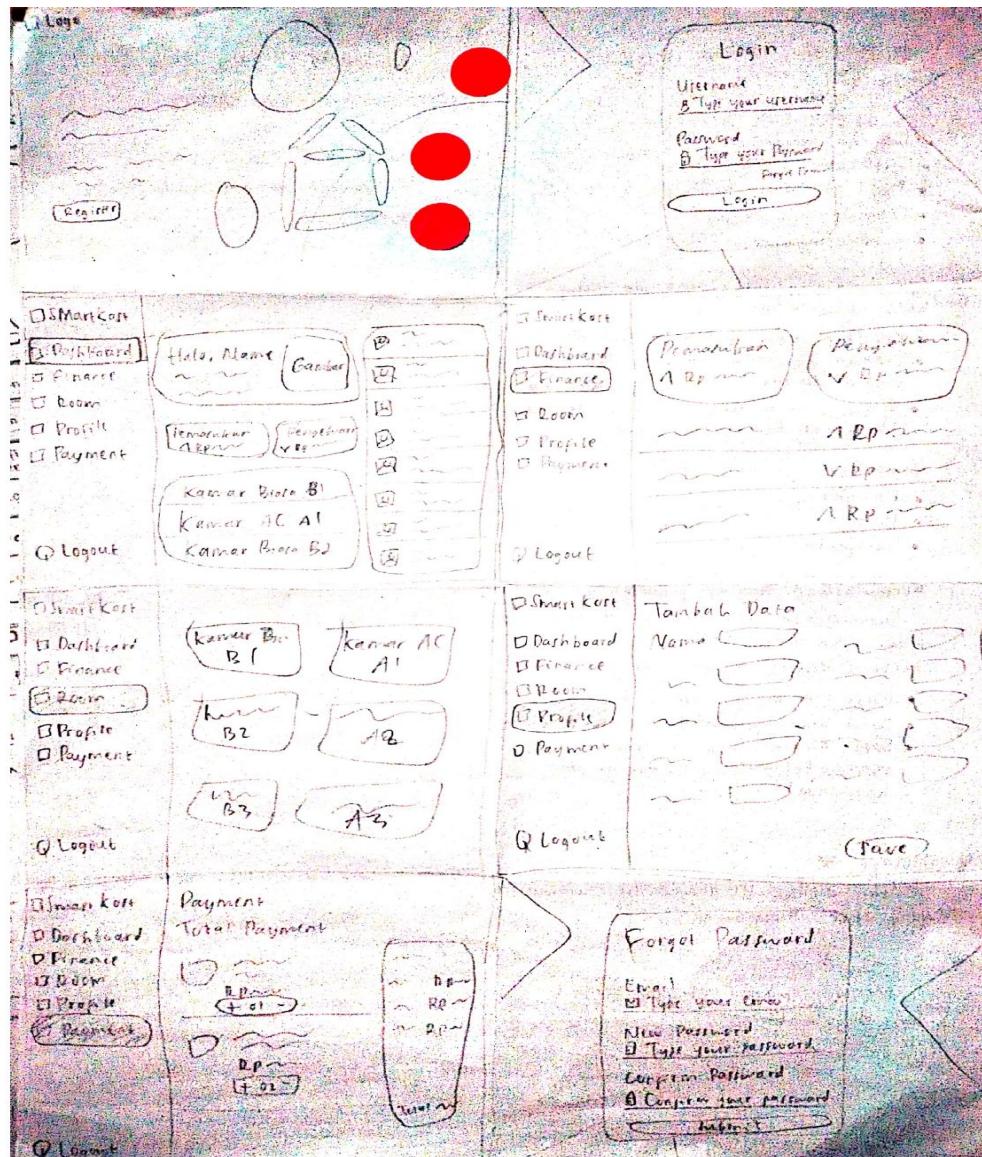
## Storyboard Mohammad Syahrur Rohman



Gambar 3.10 Hasil Design Sprint Oleh Alung

### 3.3 DECIDE

Berdasarkan dari hasil voting dari para anggota kelompok, yang mendapatkan voting terbanyak adalah desain milik Tazky sebagai berikut.

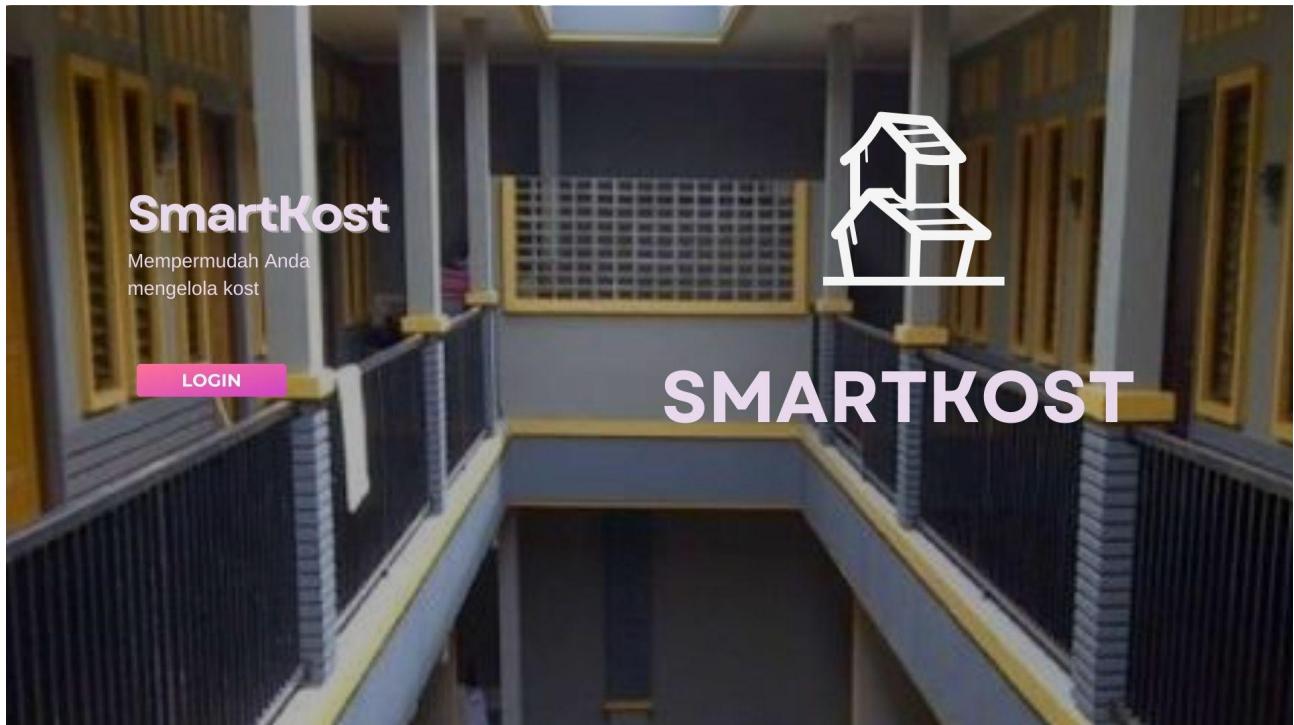


CS Dipindai dengan CamScanner

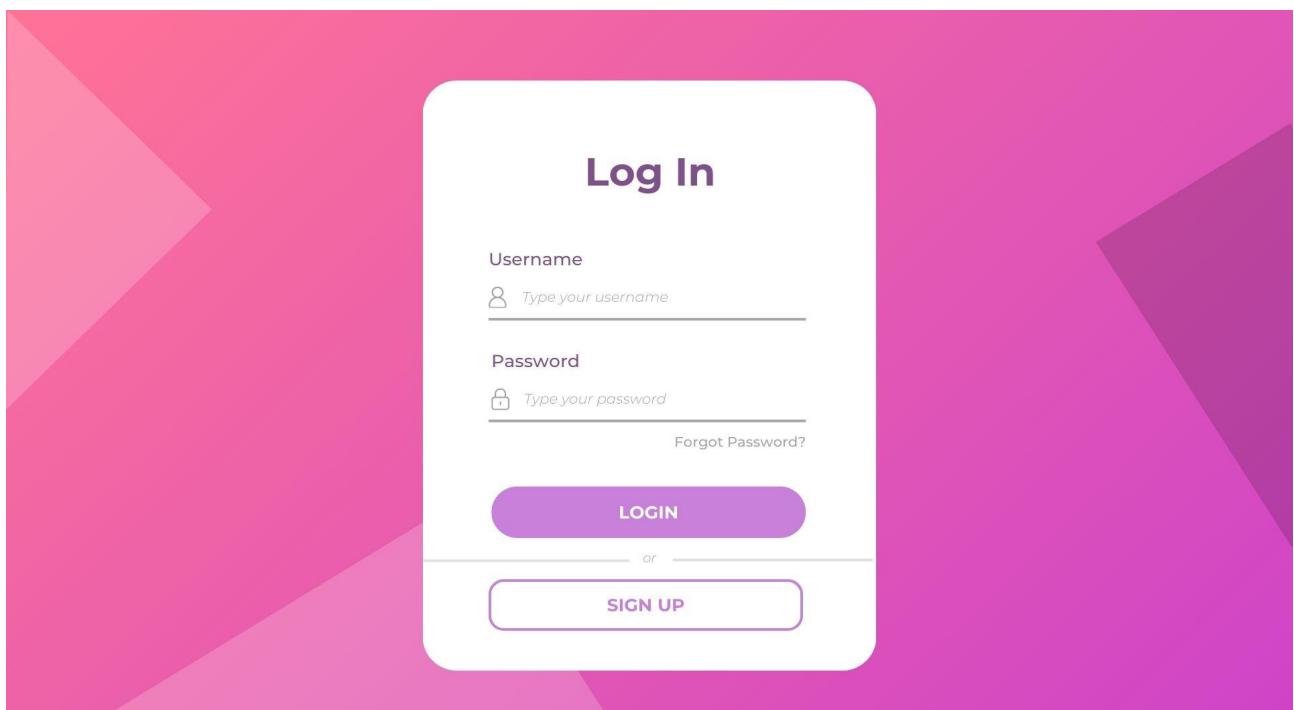
Gambar 3.11 Hasil Design Sprint Oleh Tazky

### **3.4 PROTOTYPE**

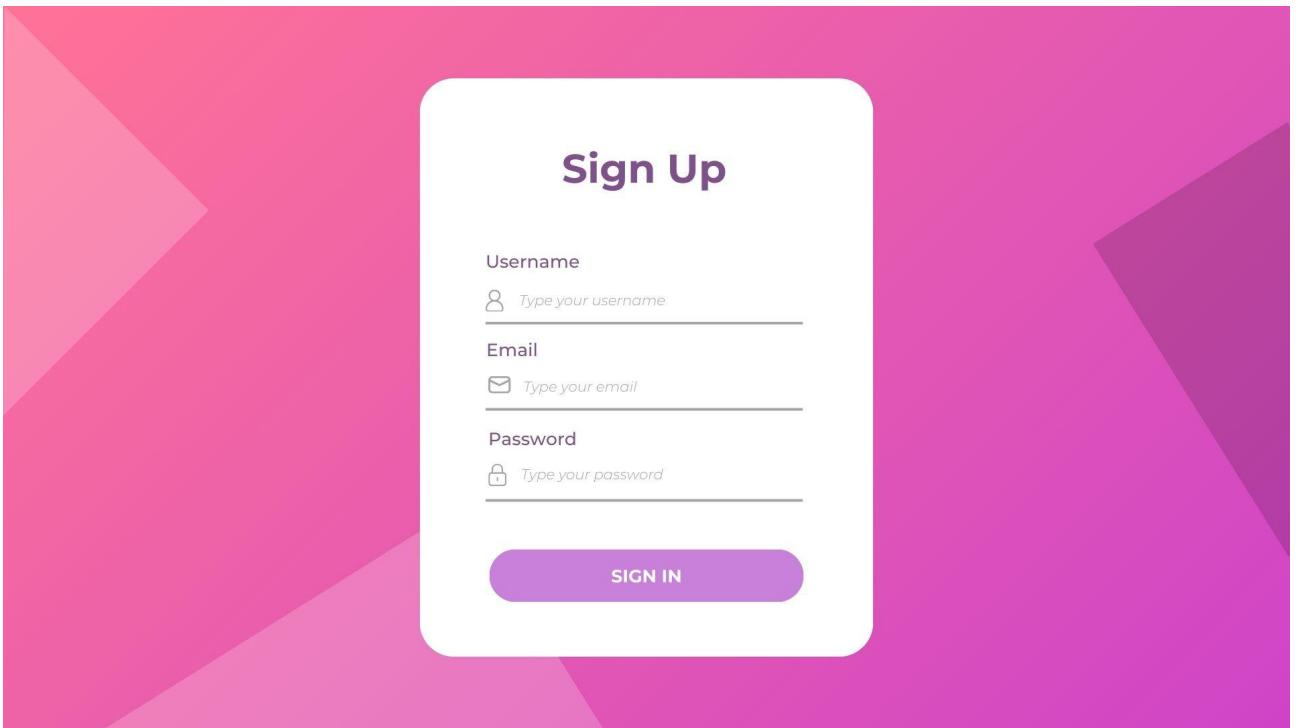
Berikut ini beberapa hasil design *User Interface* dari sistem manajemen kost untuk akses admin. Halaman beranda ditunjukkan seperti pada gambar.



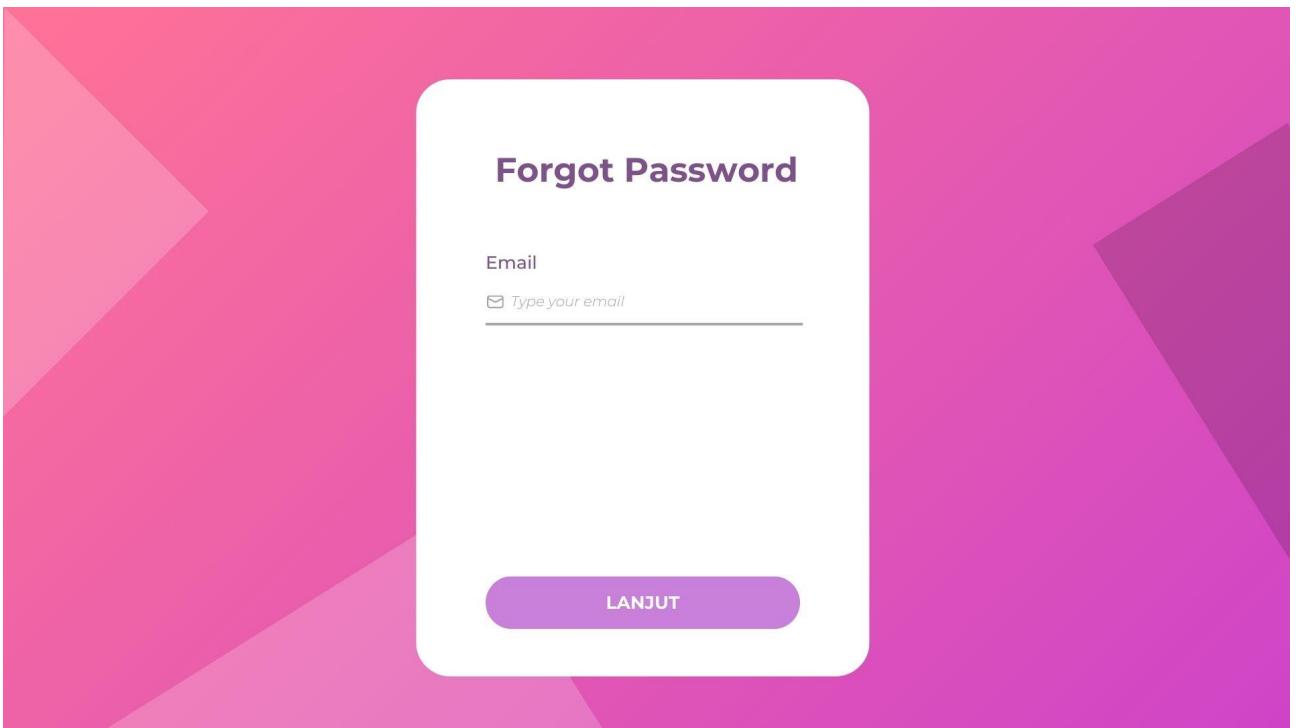
Gambar 3.12 Beranda



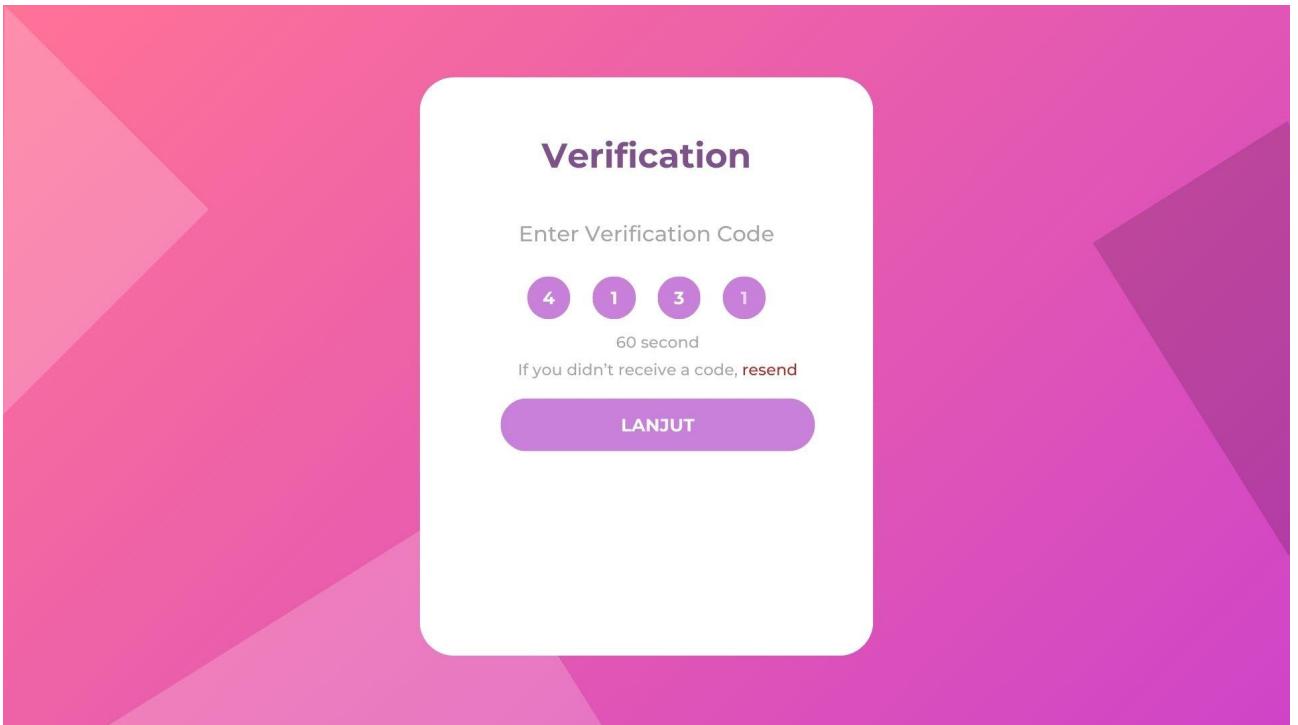
Gambar 3.13 Log In



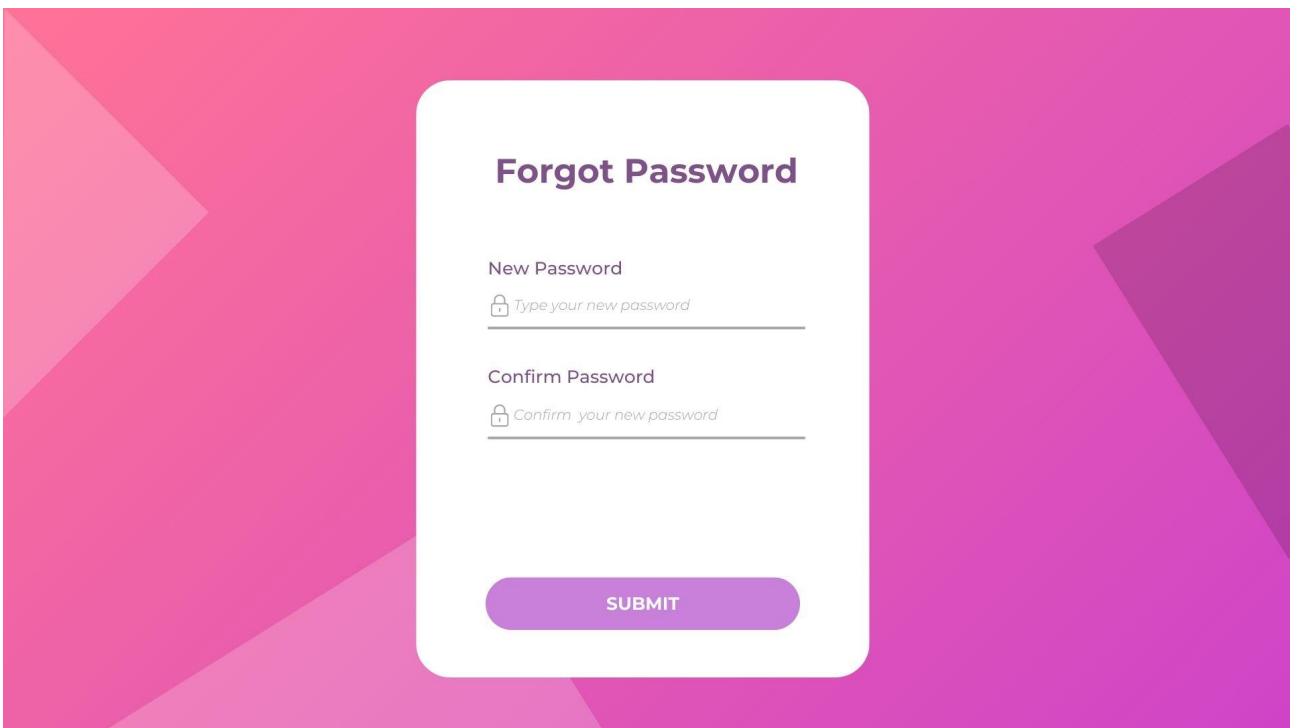
Gambar 3.14 Sign Up



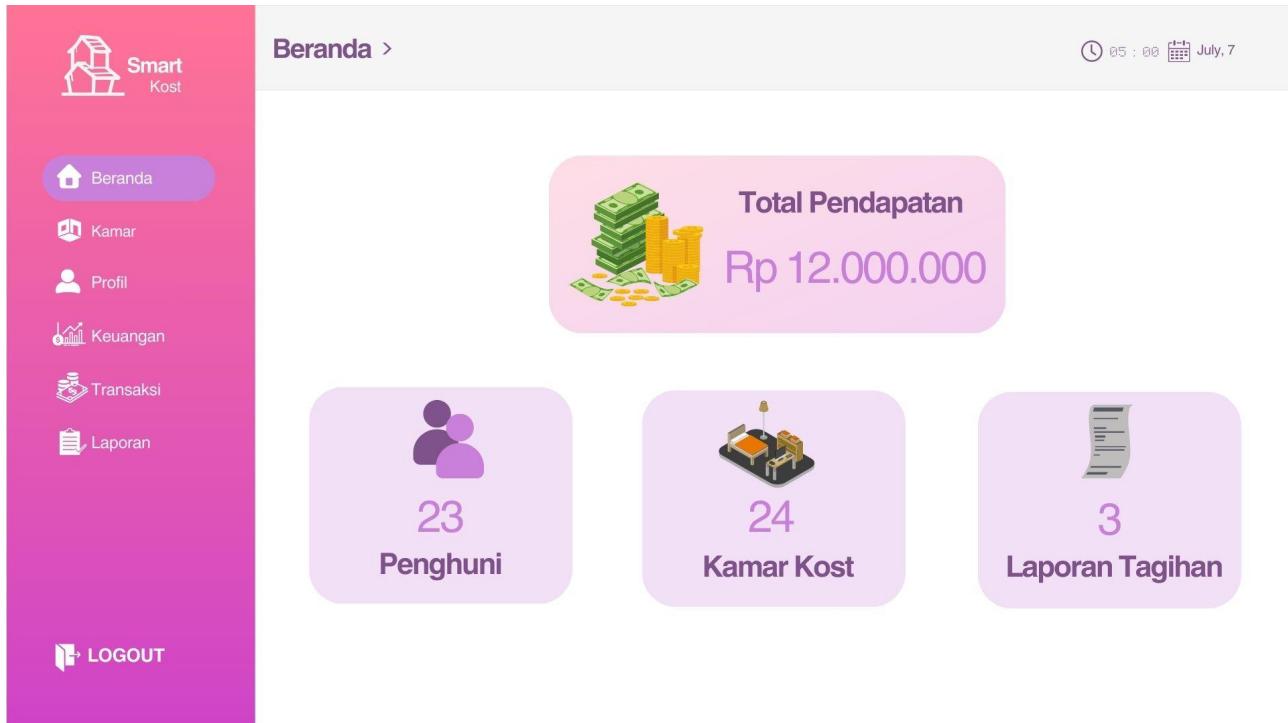
Gambar 3.15 Forgot Password



Gambar 3.16 Verification



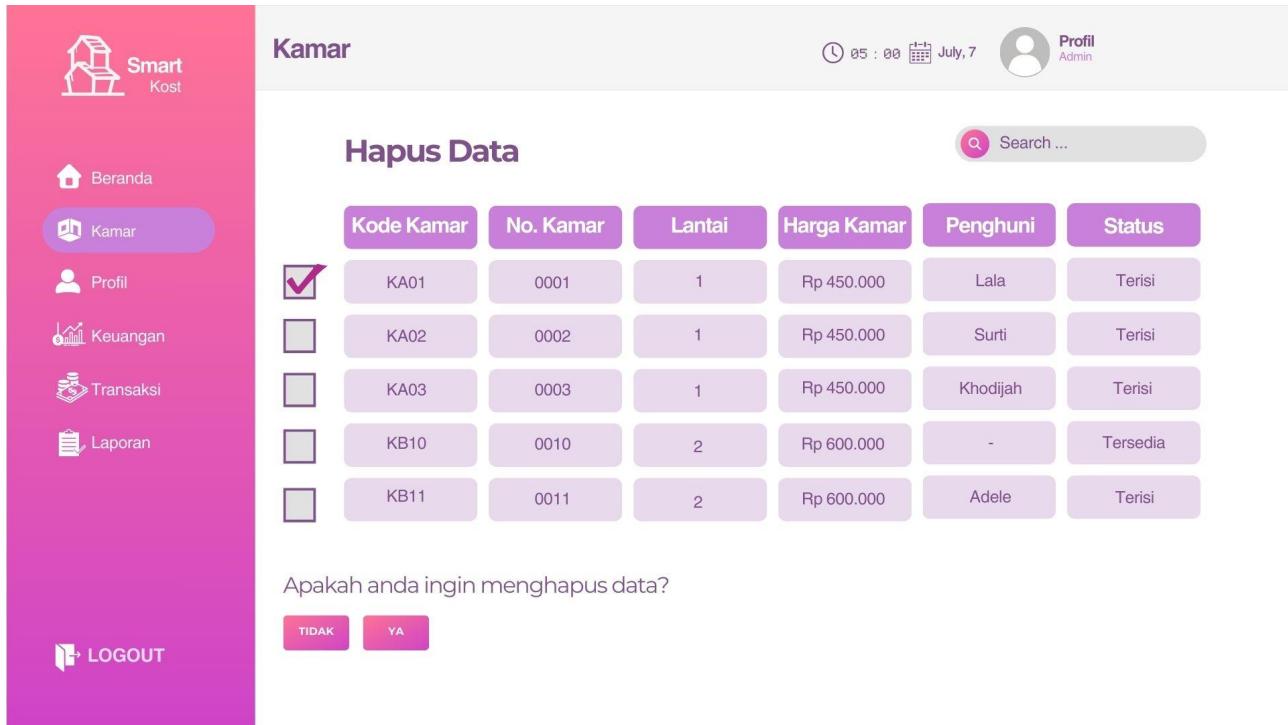
Gambar 3.17 New Password



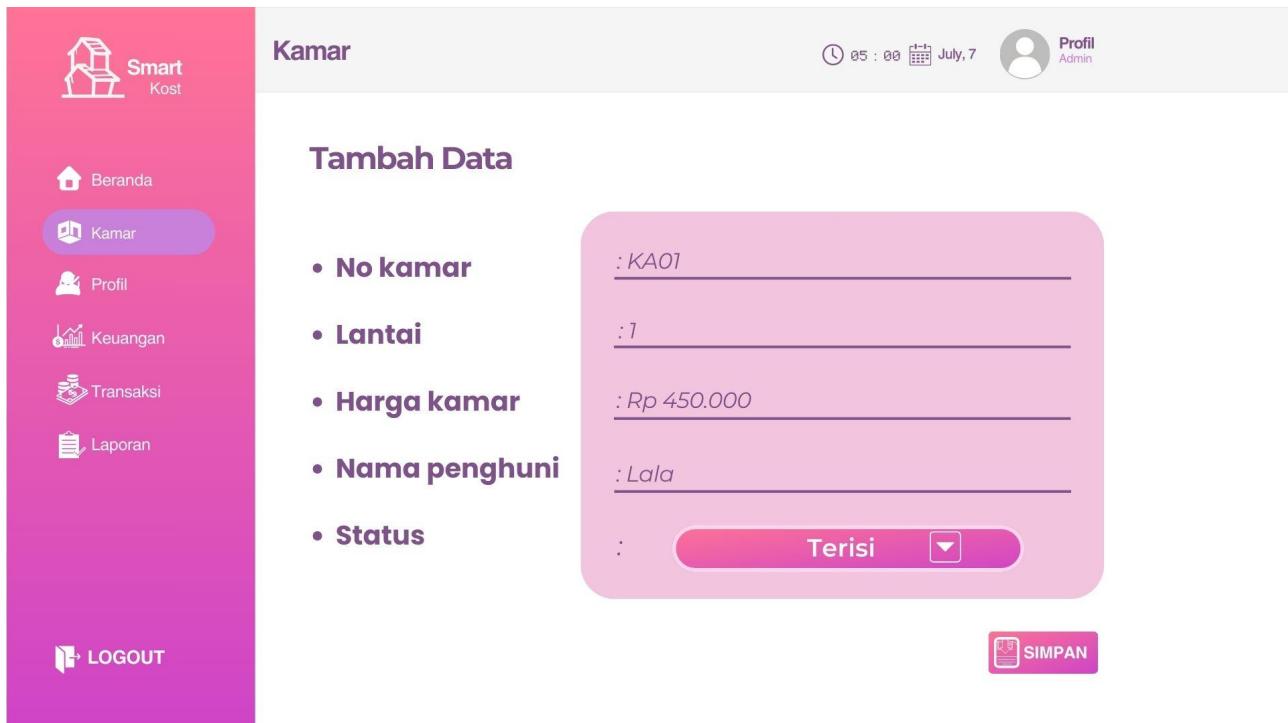
Gambar 3.18 Beranda

Kamar					
<a href="#">+ TAMBAH DATA</a>		<a href="#">EDIT</a>	<a href="#">HAPUS</a>	Search ...	
Kode Kamar	No. Kamar	Lantai	Harga Kamar	Penghuni	Status
KA01	0001	1	Rp 450.000	Lala	Terisi
KA02	0002	1	Rp 450.000	Surti	Terisi
KA03	0003	1	Rp 450.000	Khodijah	Terisi
KB10	0010	2	Rp 600.000	-	Tersedia
KB11	0011	2	Rp 600.000	Adele	Terisi

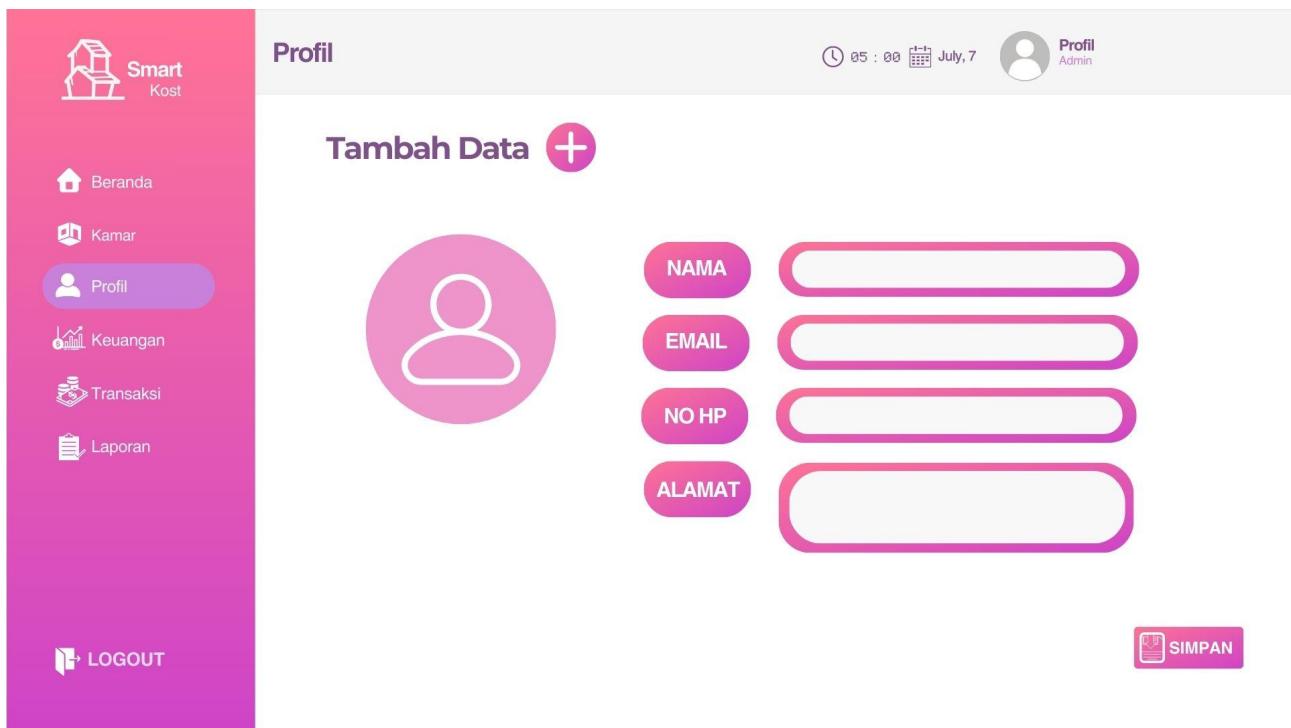
Gambar 3.19 Kamar



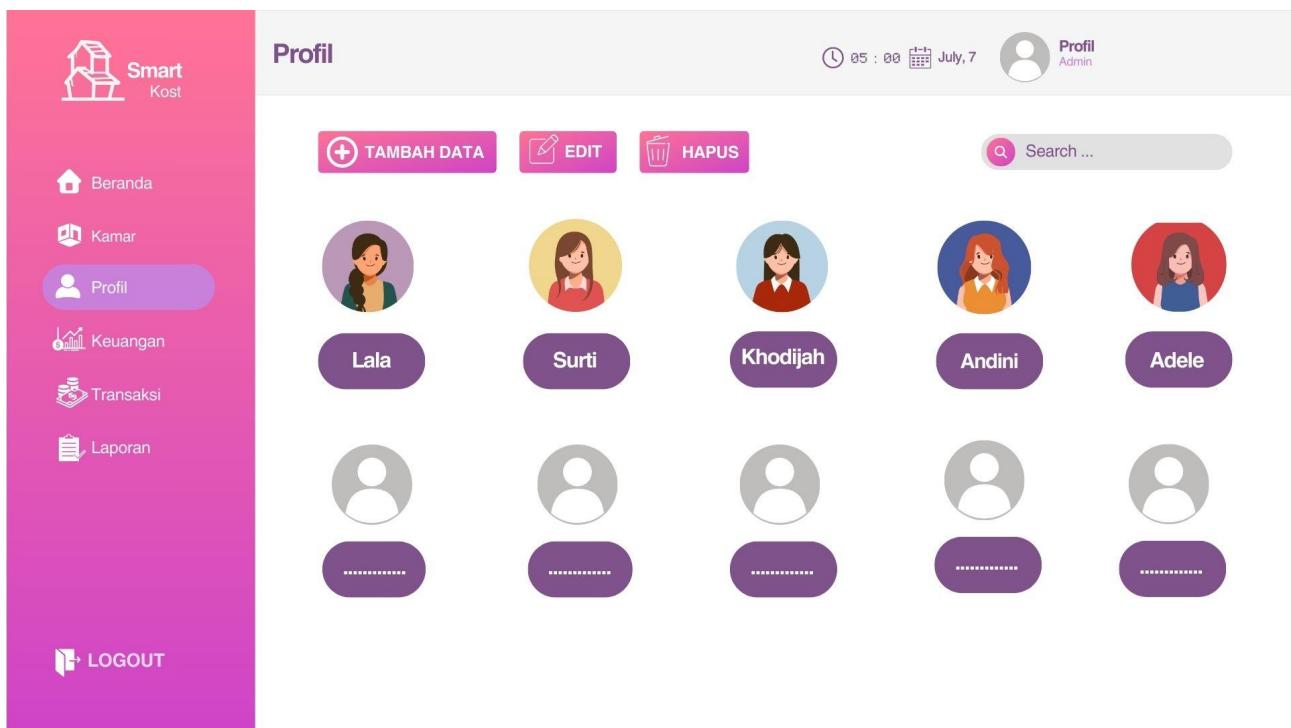
Gambar 3.20 Hapus Data Kamar



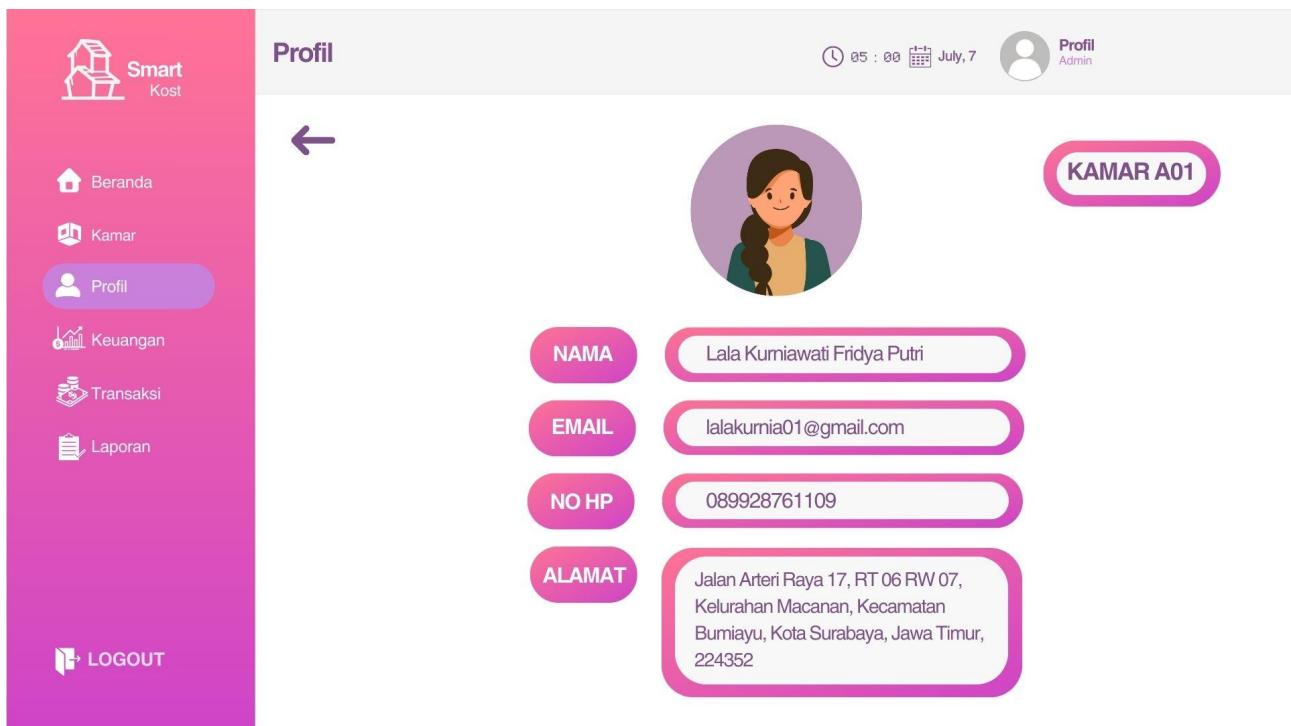
Gambar 3.21 Tambah Data Kamar



Gambar 3.22 Profil Tambah



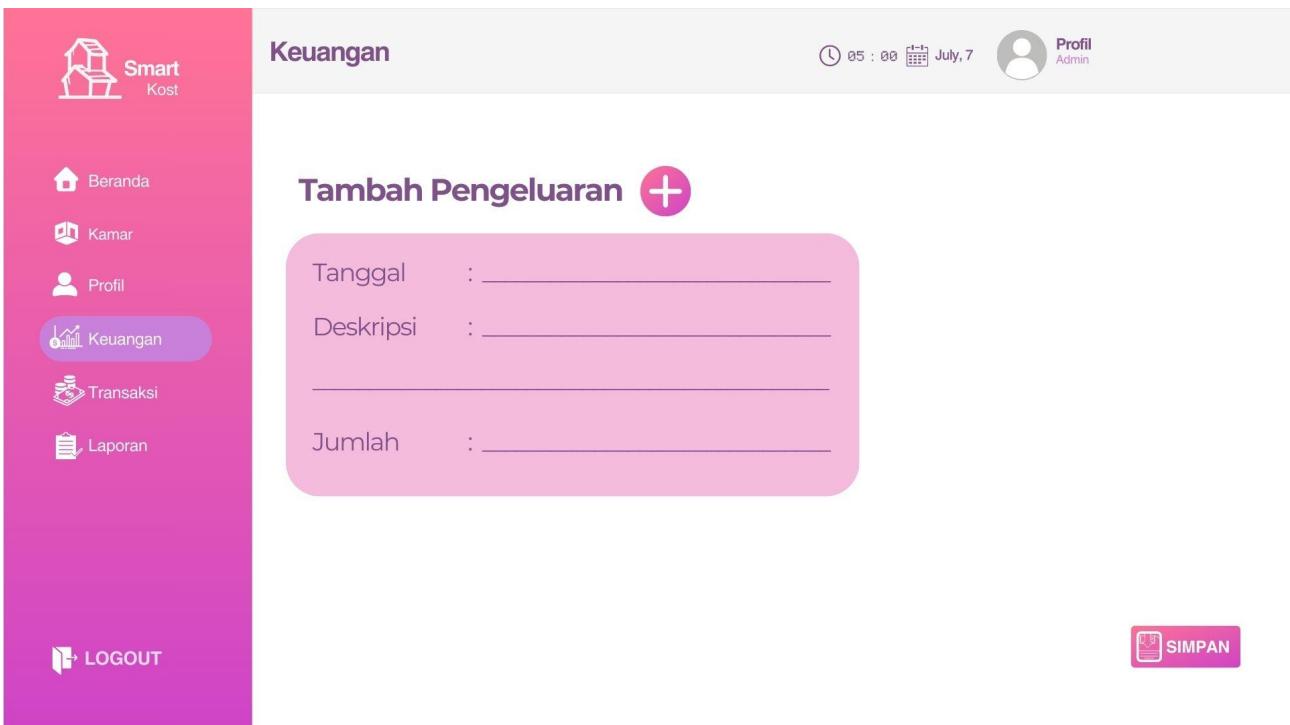
Gambar 3.23 Profil Semua Penghuni



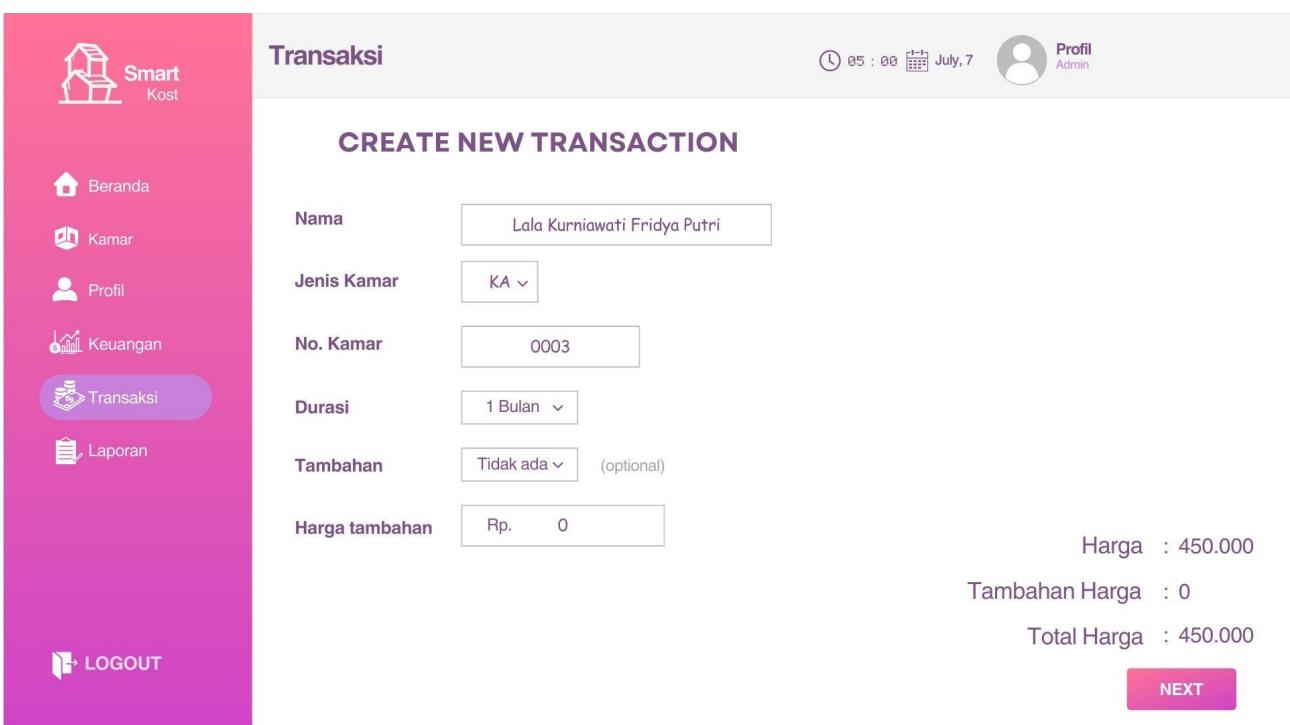
Gambar 3.24 Profil Penghuni



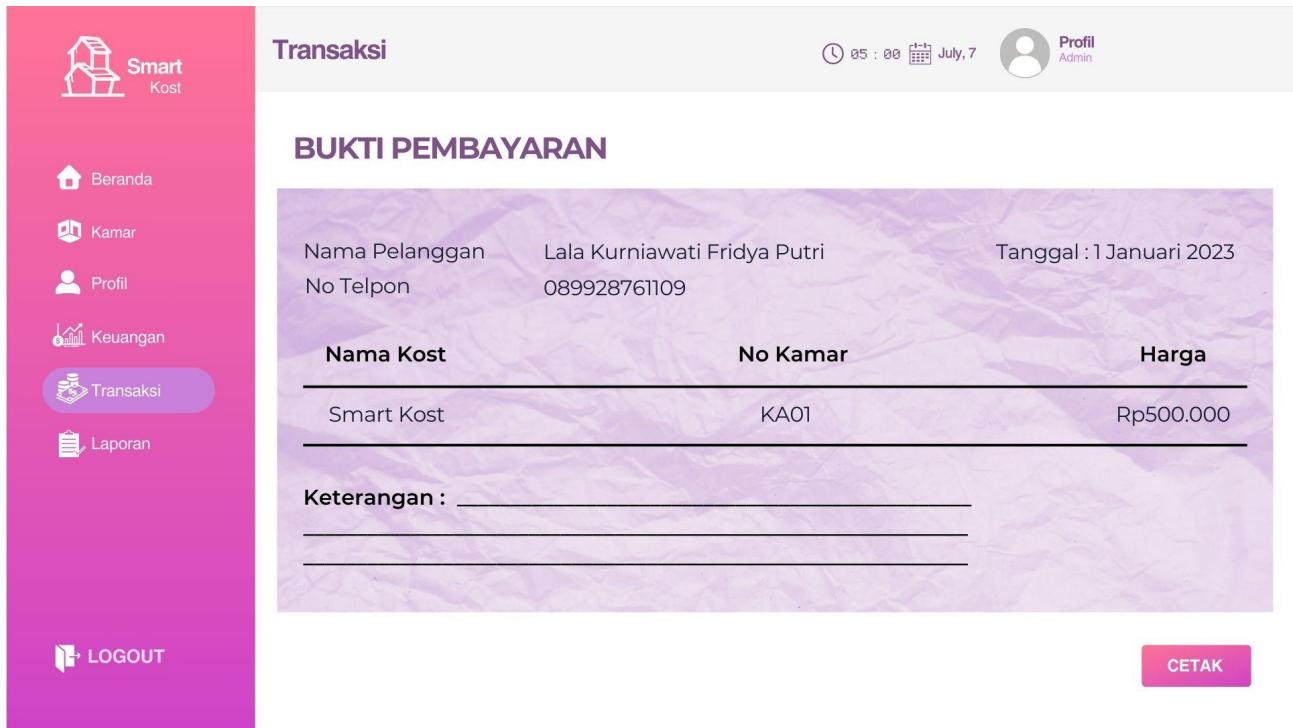
Gambar 3.25 Keuangan



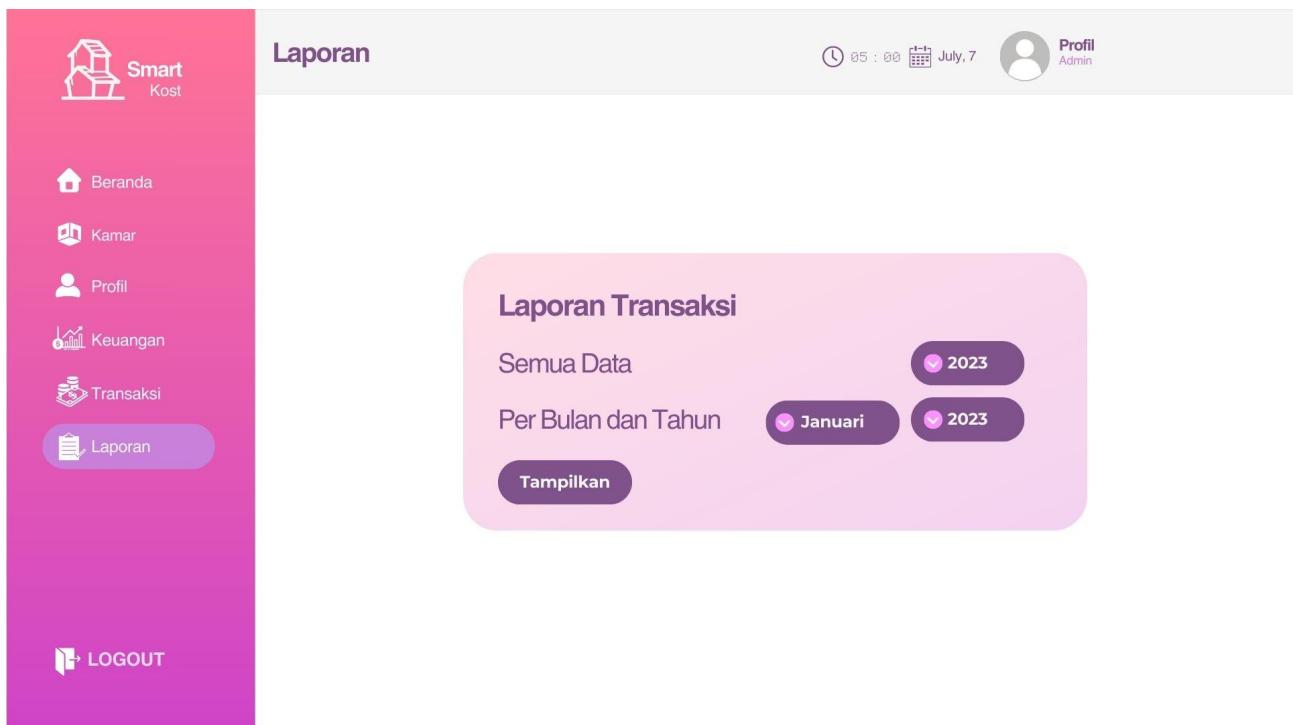
Gambar 3.26 Tambah Keuangan



Gambar 3.27 Transaksi Keuangan



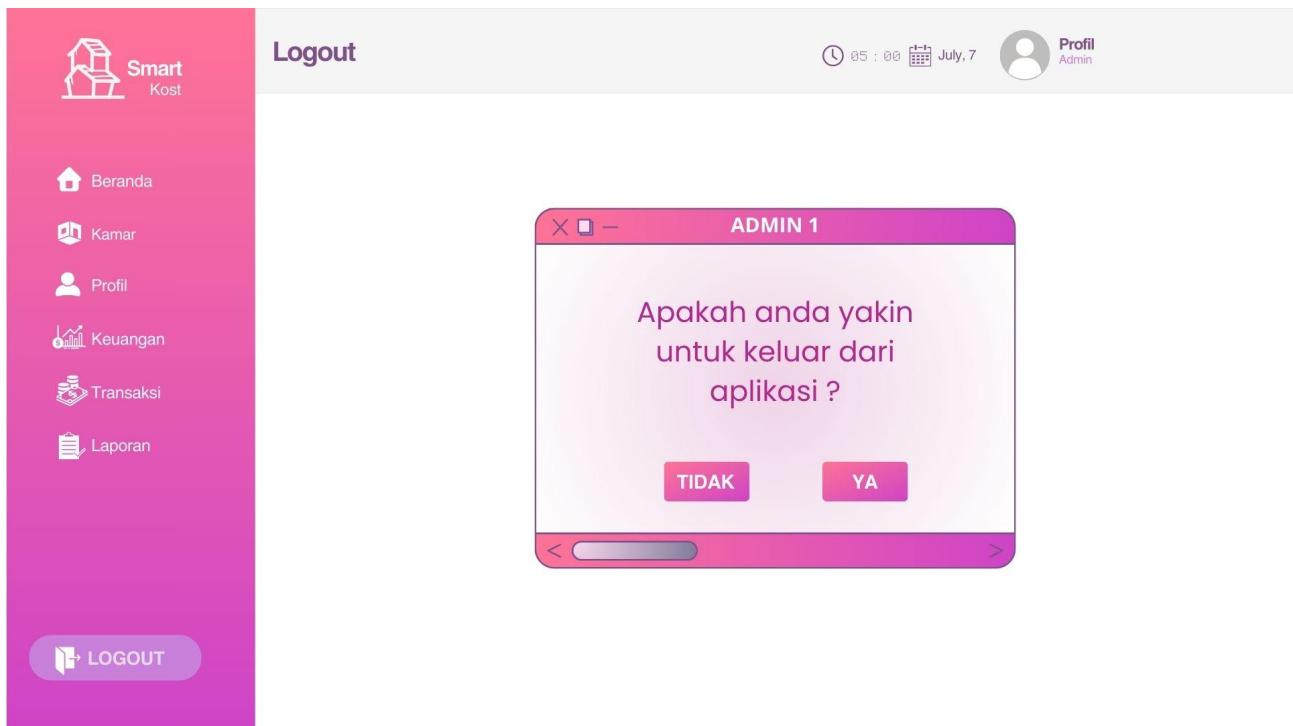
Gambar 3.28 Kwitansi Pembayaran Kauangan



Gambar 3.29 Laporan

No	No Kamar	Tagihan Bulan	Penghuni Aktif	Tanggal Bayar	Total	Status
1	KAO1	Januari	Lala	01-01-2023	Rp500.000	Lunas
2	KAO3	Januari	Andini	-	-	Belum Bayar
3	-	-	-	-	-	-
4	-	-	-	-	-	-

Gambar 3.30 Data Laporan



Gambar 3.31 LOGOUT

### **3.5 VALIDATE**

Melakukan uji pada sistem baru untuk memastikan bahwa peralihan ke sistem baru berfungsi dan memenuhi kebutuhan pengguna. Untuk memastikan bahwa peralihan ke sistem baru berhasil, perencanaan, pengendalian, dan implementasi yang tepat dari fasilitas baru harus dipastikan.

#### **3.5.1 Login**

Pada form Login akan diminta username dan password yang telah dibuat sebelumnya atau apabila belum memiliki akun maka dapat melakukan sign up dan juga dapat melakukan reset password apabila diperlukan, lalu admin akan diarahkan menuju form beranda.

#### **3.5.2 Beranda**

Pada form Beranda akan menampilkan total pendapatan, total penghuni, kamar kost, dan laporan tagihan yang valid sesuai data yang telah diinput sebelumnya dan pada pojok sebelah kanan atas terdapat waktu jam dan tanggal.

#### **3.5.3 Kamar**

Pada form Kamar akan menampilkan data penghuni kamar setiap penghuni antara lain Kode Kamar, No.Kamar, Lantai, Harga Kamar, Nama Penghuni, dan Status Kamar. Pemilik juga dapat menambahkan, mengganti bahkan menghapus data penghuni kamar. Dan terdapat kolom Search di sebelah kanan untuk memudahkan dalam pencarian data penghuni yang diinginkan.

#### **3.5.4 Profil**

Pada saat pemilik kost ingin menambahkan data maka akan menuju pada form Profil Tambah Data yang berisi Nama, Email, No Hp, dan Alamat dari calon penghuni kamar. Setelah selesai pemilik dapat menyimpan dengan menekan tombol Simpan. Pada form Profil ini juga pemilik dapat mencari, mengedit dan menghapus data penghuni pada tombol yang telah disediakan.

### **3.5.5 Keuangan**

Pada form keuangan ini pemilik dapat mencatat pemasukan dan pengeluaran kost beserta keterangan waktu dan keterangan keuangan. Pemilik juga dapat mencari data keuangan pada bulan bulan sebelumnya dengan menekan nama bulan diatas data transaksi dan untuk menambahkan data catatan pemilik dapat menekan tanda + pada bagian bawah data transaksi.

### **3.5.6 Transaksi**

Pada form Transaksi ini bertujuan untuk menambahkan data transaksi antar pemilik kost dengan penghuni kamar kost, contohnya pemilik kost membuat laporan pembayaran kost pada setiap bulannya dan dapat dicetak berupa kwitansi yang kemudian diberikan kepada penghuni kost untuk mengingatkan akan pembayaran disetiap bulannya. Data transaksi ini tersimpan di menu laporan.

### **3.5.7 Laporan**

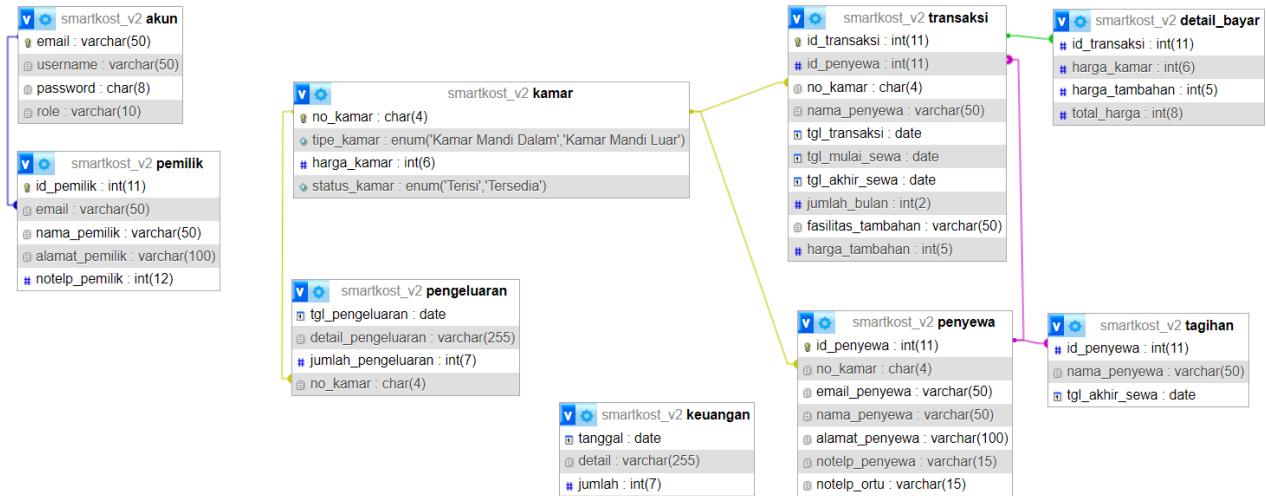
Pada form Laporan terdapat opsi untuk mencari laporan keuangan pada tanggal, bulan dan tahun yang akan di cari. Menu laporan ini juga terdapat data penghuni kost secara lengkap, dan terdapat keterangan berupa “Lunas” dan “Belum Bayar”, dimana pemilik kost dapat mengetahui siapa saja penghuni kamar yang belum membayar kost pada bulan tersebut.

### **3.5.8 LOGOUT**

Pada pojok kiri bawah terdapat LOGOUT yang berfungsi untuk keluar dari akun pemilik kost, ini bertujuan agar privasi data kost tetap aman tidak dapat di akses oleh orang lain.

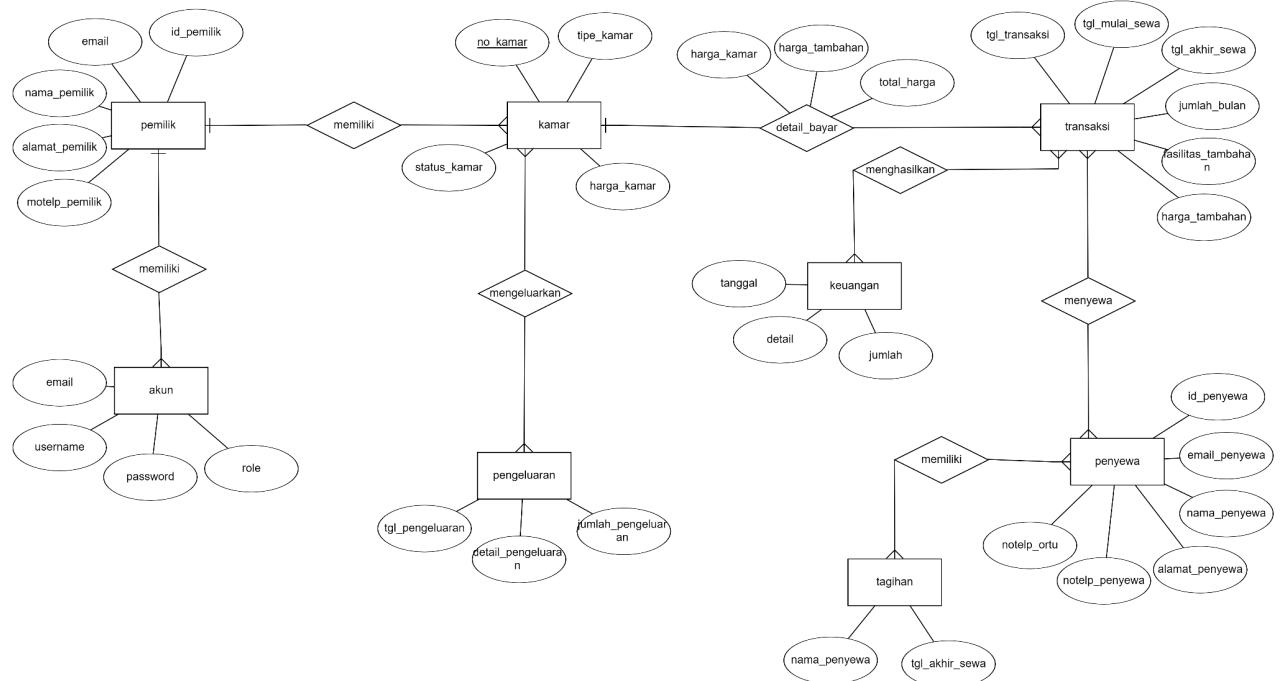
## BAB 4. PEMBAHASAN

### 4.1 RANCANGAN DATABASE



Gambar 4.1 Rancangan Database

#### 4.1.1 ERD



Gambar 4.1.1 ERD Aplikasi

#### 4.1.2 TABEL

##### 1. Tabel Akun

Tabel akun menyimpan informasi pengguna. Ini terdiri dari email, username, password, dan peran (role) pengguna. Email digunakan untuk berkomunikasi dan verifikasi, username adalah nama pengguna yang sebenarnya, password digunakan untuk otentikasi, dan peran (role) menentukan hak akses pengguna ke sistem.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action	
<input type="checkbox"/>	1 <b>email</b> 	varchar(50)	utf8mb4_general_ci		No	None		 Change	 Drop	<a href="#">More</a>
<input type="checkbox"/>	2 <b>username</b>	varchar(50)	utf8mb4_general_ci		No	None		 Change	 Drop	<a href="#">More</a>
<input type="checkbox"/>	3 <b>password</b>	char(8)	utf8mb4_general_ci		No	None		 Change	 Drop	<a href="#">More</a>
<input type="checkbox"/>	4 <b>role</b>	varchar(10)	utf8mb4_general_ci		Yes	Admin		 Change	 Drop	<a href="#">More</a>

Gambar 4.1.2 Tabel Akun

##### 2. Tabel Pemilik

Semua informasi tentang pemilik kost disimpan dalam tabel pemilik, yang terdiri dari id\_pemilik, email, nama\_pemilik, alamat\_pemilik, dan notelp\_pemilik.

Id\_pemilik adalah identitas asli pemilik, email digunakan untuk berkomunikasi, dan nama\_pemilik adalah nama lengkap pemilik, alamat\_pemilik adalah alamat tempat tinggal pemilik, dan notelp\_pemilik adalah nomor telepon pemilik.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action	
<input type="checkbox"/>	1 <b>id_pemilik</b> 	int(11)			No	None	AUTO_INCREMENT	 Change	 Drop	<a href="#">More</a>
<input type="checkbox"/>	2 <b>email</b> 	varchar(50)	utf8mb4_general_ci		No	None		 Change	 Drop	<a href="#">More</a>
<input type="checkbox"/>	3 <b>nama_pemilik</b>	varchar(50)	utf8mb4_general_ci		No	None		 Change	 Drop	<a href="#">More</a>
<input type="checkbox"/>	4 <b>alamat_pemilik</b>	varchar(100)	utf8mb4_general_ci		No	None		 Change	 Drop	<a href="#">More</a>
<input type="checkbox"/>	5 <b>notelp_pemilik</b>	int(12)			No	None		 Change	 Drop	<a href="#">More</a>

Gambar 4.1.2 Tabel Pemilik

##### 3. Tabel Penyewa

Tabel penyewa menyimpan informasi tentang penyewa kamar kost. Tabel ini terdiri dari id\_penyewa, no\_kamar, email\_penyewa, nama\_penyewa, alamat\_penyewa, notelp\_penyewa, dan notelp\_ortu. Id\_penyewa adalah identitas unik penyewa, no\_kamar adalah nomor kamar yang disewa, email\_penyewa digunakan untuk berkomunikasi dan memverifikasi, nama\_penyewa adalah nama lengkap penyewa, dan alamat\_penyewa adalah alamat

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1 <b>id_penyewa</b> 🔑	int(11)			No	None		AUTO_INCREMENT	Change  Drop  More
□	2 <b>no_kamar</b> 🔑	char(4)	utf8mb4_general_ci		No	None			Change  Drop  More
□	3 <b>email_penyewa</b>	varchar(50)	utf8mb4_general_ci		No	None			Change  Drop  More
□	4 <b>nama_penyewa</b>	varchar(50)	utf8mb4_general_ci		No	None			Change  Drop  More
□	5 <b>alamat_penyewa</b>	varchar(100)	utf8mb4_general_ci		No	None			Change  Drop  More
□	6 <b>notelp_penyewa</b>	varchar(15)	utf8mb4_general_ci		No	None			Change  Drop  More
□	7 <b>notelp_ortu</b>	varchar(15)	utf8mb4_general_ci		No	None			Change  Drop  More

Gambar 4.1.2 Tabel Penyewa

#### 4. Tabel Kamar

Tabel kamar adalah sebuah struktur data yang digunakan untuk menyimpan informasi tentang kamar yang tersedia dalam suatu kost. Isi dari tabel kamar yaitu no\_kamar, tipe\_kamar, harga\_kamar, dan status\_kamar. No\_kamar adalah nomor identitas unik untuk setiap kamar, tipe\_kamar adalah jenis kamar yang tersedia, harga\_kamar adalah harga sewa kamar, dan status\_kamar adalah status ketersediaan kamar.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1 <b>no_kamar</b> 🔑	char(4)	utf8mb4_general_ci		No	None			Change  Drop  More
□	2 <b>tipe_kamar</b>	enum('Kamar Mandi Dalam', 'Kamar Mandi Luar')	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
□	3 <b>harga_kamar</b>	int(6)			No	None			Change  Drop  More
□	4 <b>status_kamar</b>	enum('Terisi', 'Tersedia')	utf8mb4_general_ci		Yes	NULL			Change  Drop  More

Gambar 4.1.2 Tabel Kamar

#### 5. Tabel Transaksi

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1 <b>id_transaksi</b> 🔑	int(11)			No	None		AUTO_INCREMENT	Change  Drop  More
□	2 <b>id_penyewa</b> 🔑	int(11)			Yes	NULL			Change  Drop  More
□	3 <b>no_kamar</b> 🔑	char(4)	utf8mb4_general_ci		No	None			Change  Drop  More
□	4 <b>nama_penyewa</b>	varchar(50)	utf8mb4_general_ci		No	None			Change  Drop  More
□	5 <b>tgl_transaksi</b>	date			Yes	curdate()			Change  Drop  More
□	6 <b>tgl_mulai_sewa</b>	date			Yes	NULL			Change  Drop  More
□	7 <b>tgl_akhir_sewa</b>	date			Yes	NULL			Change  Drop  More
□	8 <b>jumlah_bulan</b>	int(2)			Yes	NULL			Change  Drop  More
□	9 <b>fasilitas_tambahan</b>	varchar(50)	utf8mb4_general_ci		Yes	NULL			Change  Drop  More
□	10 <b>harga_tambahan</b>	int(5)			Yes	0			Change  Drop  More

Gambar 4.1.2 Tabel Transaksi

#### 6. Tabel Detail Bayar

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1 <b>id_transaksi</b> 🔑	int(11)			Yes	NULL			Change  Drop  More
□	2 <b>harga_kamar</b>	int(6)			No	None			Change  Drop  More
□	3 <b>harga_tambahan</b>	int(5)			Yes	NULL			Change  Drop  More
□	4 <b>total_harga</b>	int(8)			No	None			Change  Drop  More

Gambar 4.1.2 Tabel Detail Bayar

## 7. Tabel Tagihan

Semua informasi tentang tagihan yang harus dibayar oleh penyewa setelah transaksi atau perniatan disimpan dalam tabel tagihan, yang terdiri dari id\_penyewa, nama\_penyewa, dan tgl\_akhir\_sewa.

- id\_penyewa adalah identitas unik pengguna yang harus membayar tagihan,
- nama\_penyewa adalah nama lengkap pengguna yang harus membayar tagihan.
- Tanggal akhir sewa kamar atau perniatan

Untuk memudahkan pelacakan dan rekonsiliasi pembayaran, tabel tagihan digunakan untuk mencatat informasi tentang tagihan yang harus dibayar oleh penyewa. Informasi tentang tagihan dapat disimpan secara terstruktur dan akurat, yang memudahkan proses keuangan dan pelaporan.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	<b>id_penyewa</b>	int(11)			Yes	NULL			Change  Drop  More
2	<b>nama_penyewa</b>	varchar(50)	utf8mb4_general_ci		No	None			Change  Drop  More
3	<b>tgl_akhir_sewa</b>	date			No	None			Change  Drop  More

Gambar 4.1.2 Tabel Tagihan

## 8. Tabel Keuangan

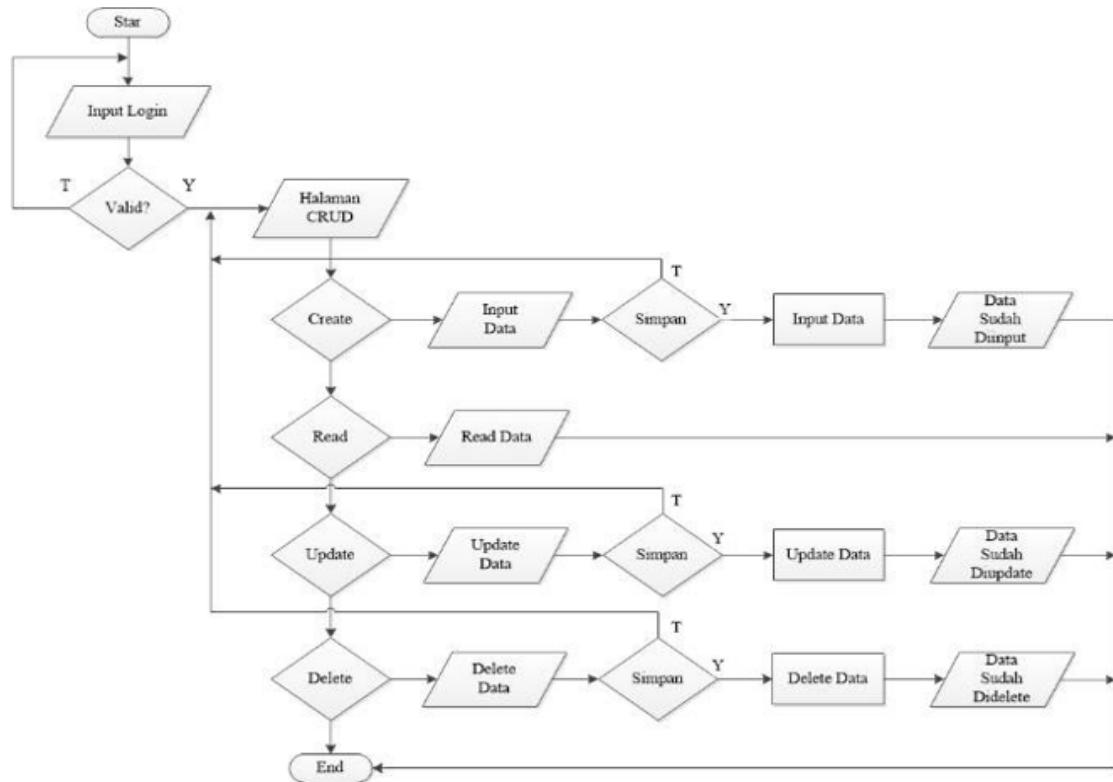
Tabel keuangan menyimpan informasi tentang pengelolaan keuangan kost dan Tabel keuangan menyimpan informasi tentang pengelolaan keuangan kost dan mencakup tanggal, detail, dan jumlah.

- Tanggal: Menyimpan informasi tentang tanggal kedatangan penyewa, seperti tanggal hotel atau tanggal kosong.
- Detail: Berisi informasi lebih rinci tentang keuangan, seperti jenis kamar yang disewa, layanan yang dipilih, atau biaya tambahan yang terkait dengan transaksi.
- Jumlah: menyimpan informasi tentang berapa banyak yang harus dibayar penyewa, termasuk harga kamar, biaya tambahan, dan biaya lainnya.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action	
□ 1	<b>tanggal</b>	date			No	None			Change	Drop
□ 2	<b>detail</b>	varchar(255)	utf8mb4_general_ci		Yes	Pembayaran Kost			Change	Drop
□ 3	<b>jumlah</b>	int(7)			No	None			Change	Drop

Gambar 4.1.2 Tabel Keuangan

## 4.2 FLOWCHART SYSTEM

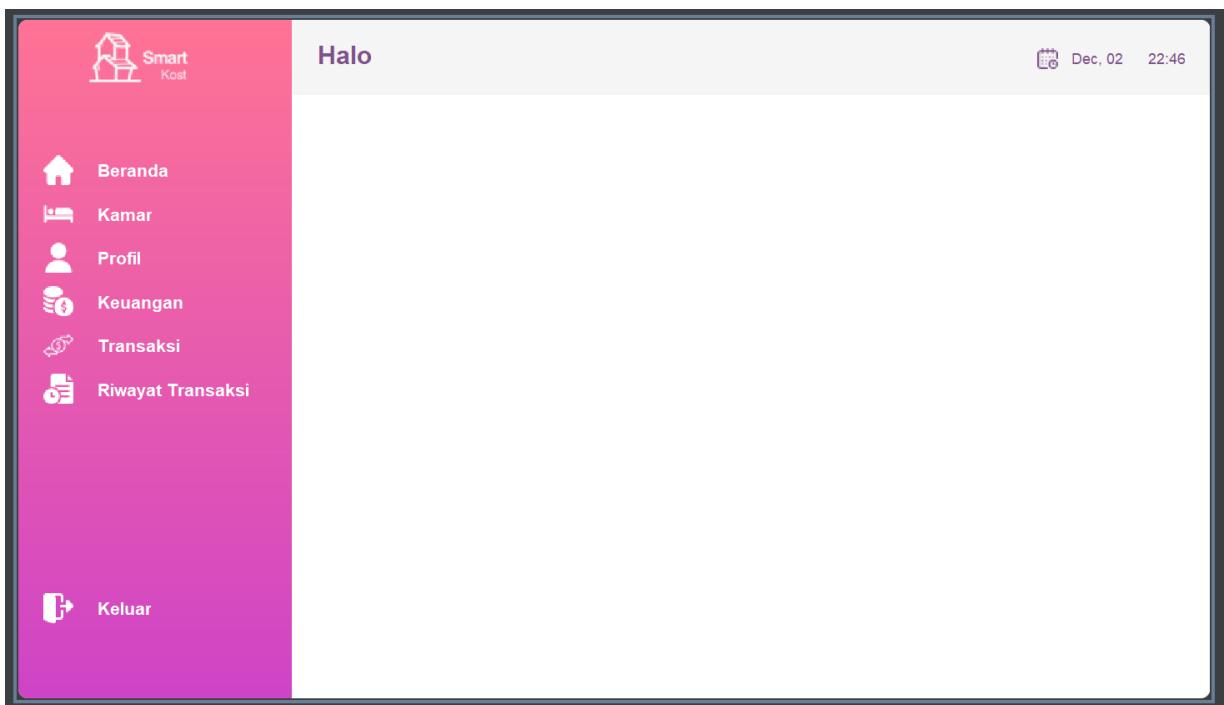


Gambar 4.2 Flowchart Admin

## 4.3 DESIGN DAN KODE PROGRAM

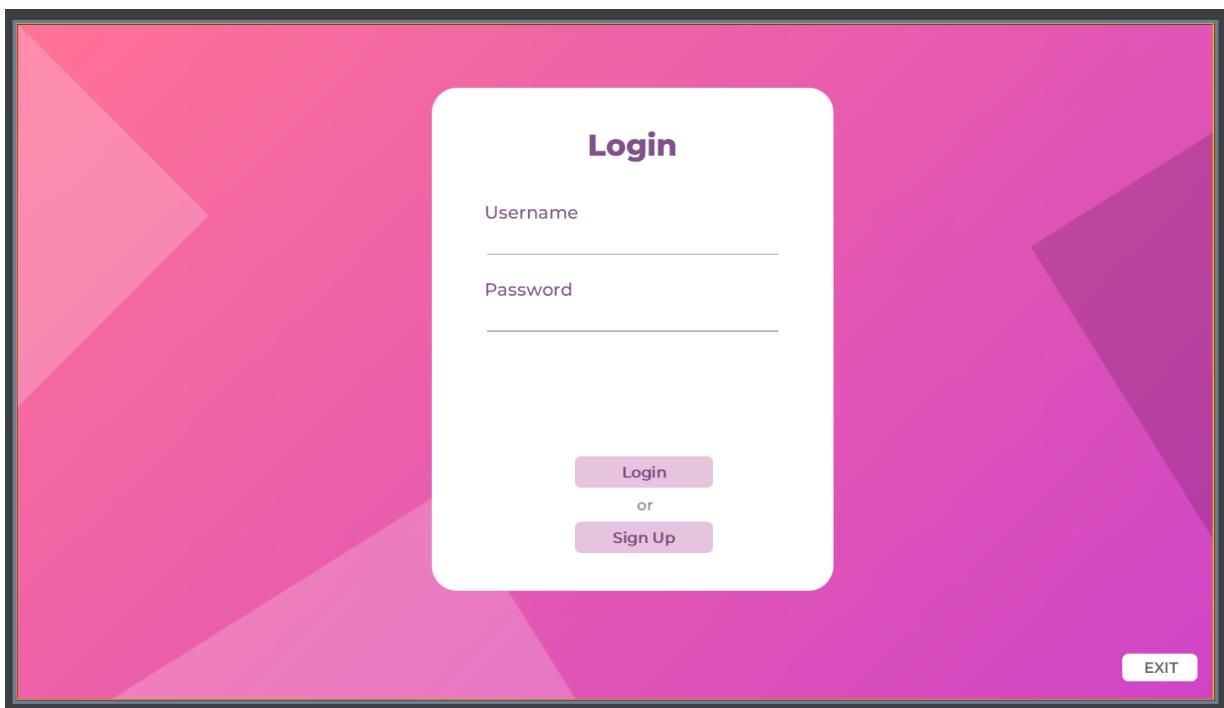
### 4.3.1 DESIGN

#### 1. Form Utama



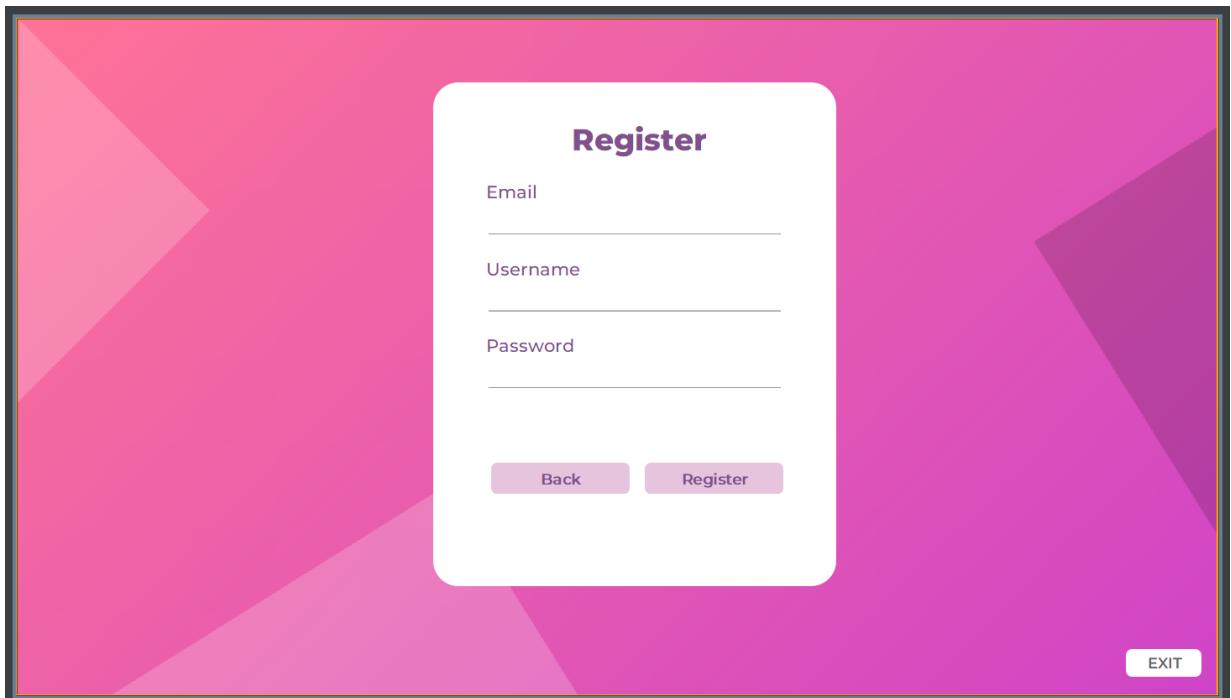
Gambar 4.3.1 Design Form Utama

#### 2. Form Login



Gambar 4.3.1 Design Form Login

### 3. Form Register



Gambar 4.3.1 Design Form Register

### 4. Form Beranda



Gambar 4.3.1 Design Form Beranda

### 5. Form Profil

## Profil

No. Kamar	:	<input type="text"/>	Alamat	:	<input type="text"/>		
Nama Penghuni	:	<input type="text"/>			No. HP	:	<input type="text"/>
Email	:	<input type="text"/>			No. HP Ortu	:	<input type="text"/>

**Edit** **Reset** **Tambah**

No. Kamar	Email	Nama	Alamat	No. HP	No. Ortu	Aksi
[Redacted]						

Gambar 4.3.1 Design Form Profil

### 6. Form Kamar

## Kamar

No. Kamar	:	<input type="text"/>	Tipe Kamar	:	<input type="text"/>
Harga Kamar	:	<input type="text"/>	Status Kamar	:	<input type="text"/>

filter by :  **Cari** **Edit** **Reset** **Tambah**

No. Kamar	Tipe	Harga	Status	Aksi
[Redacted]				

Gambar 4.3.1 Design Form Kamar

## 7. Form Keuangan

The screenshot shows a financial management application interface. At the top, there are two pink rounded rectangular boxes. The left box contains a green icon of coins with an upward arrow and the text "Pemasukan Rp18090000". The right box contains a green icon of a money bag with a downward arrow and the text "Pengeluaran Rp700000". Below these boxes are three buttons: "Reset" (pink), "Tambah" (pink), and "Cari" (pink). To the right of the "Cari" button is a dropdown menu set to "January". A large dark grey rectangular area below the buttons is currently empty, representing a table or list.

Gambar 4.3.1 Design Form Keuangan

## 8. Form Tambah Pengeluaran

The screenshot shows a form titled "Tambah Pengeluaran" (Add Expense) in a large purple header. The form consists of three input fields: "Tanggal" (Date) with a placeholder "\_\_\_\_\_" and a pink "...", "Detail" (Description) with a placeholder "\_\_\_\_\_", and "Jumlah" (Amount) with a placeholder "\_\_\_\_\_.\_\_\_\_\_". At the bottom right are two pink buttons: "Kembali" (Back) and "Tambah" (Add).

Gambar 4.3.1 Design Form Pengeluaran

## 9. Form Transaksi

### Transaksi

Detail Sewa		Detail Bayar	
No. Kamar	:	<input type="text"/>	<input type="button" value="▼"/>
Nama Penghuni	:	<input type="text"/>	
Tanggal Mulai Sewa	:	<input type="text"/>	<input type="button" value="..."/>
Tanggal Akhir Sewa	:	<input type="text"/>	<input type="button" value="..."/>
Jumlah Bulan	:	<input type="text"/>	
Fasilitas Tambahan	:	<input type="text"/>	
Harga Tambahan	:	<input type="text"/>	

Gambar 4.3.1 Design Form Transaksi

## 10. Form Riwayat Transaksi

### Riwayat Transaksi

Tanggal	Kamar	Nama	Tgl Mulai	Tgl Akhir	Jml Bulan	Total Harga

Gambar 4.3.1 Design Form Riwayat Transaksi

### 4.3.2 KODE PROGRAM

#### 1. Config

Source code dibawah ini digunakan untuk menyediakan konfigurasi dan koneksi ke database MySQL

```
public class Config {  
  
    private static Connection mysqlconfig;  
  
    public static Connection configDB() throws SQLException {  
        try {  
            String url = "jdbc:mysql://localhost:3306/smartkost_v2";  
            String user = "root";  
            String pass = "";  
            DriverManager.registerDriver(new com.mysql.jdbc.Driver());  
            mysqlconfig = DriverManager.getConnection(url, user, password: pass);  
        } catch (Exception e) {  
            System.err.println("Koneksi Gagal " + e.getMessage());  
        }  
        return mysqlconfig;  
    }  
}
```

Gambar 4.3.2 Kode Program Config

#### 2. Form Login

##### a. Button Login

Digunakan untuk mengimplementasikan proses autentikasi atau login pada suatu aplikasi menggunakan data akun yang disimpan dalam database

```
private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) {  
    Connection conn = null;  
    try {  
        conn = (Connection) Config.configDB();  
        String sql = "SELECT username, password FROM akun WHERE username = '" + txtUsername.getText() + "' AND  
password = '" + txtPassword.getText() + "'";  
        PreparedStatement pstm = conn.prepareStatement(string: sql);  
        ResultSet rs = pstm.executeQuery(string: sql);  
  
        if (rs.next()) {  
            if (txtUsername.getText().equals(anObject: rs.getString(string: "username")) && txtPassword.getText().equals(  
anObject: rs.getString(string: "password"))) {  
                this.setVisible(b: false);  
                new Main().setVisible(b: true);  
            }  
        } else {  
            JOptionPane.showMessageDialog(parentComponent: null, message: "Username atau Password salah!");  
            txtUsername.setText(t: "");  
            txtPassword.setText(t: "");  
        }  
  
        rs.close();  
        pstm.close();  
    } catch (Exception e) {  
        JOptionPane.showMessageDialog(parentComponent: null, message: "Login Gagal\nTelah Terjadi Kesalahan");  
        e.printStackTrace();  
    } finally {  
        if (conn != null) {  
            try {  
                conn.close();  
            } catch (SQLException ex) {  
                JOptionPane.showMessageDialog(parentComponent: null, message: "Terjadi kesalahan saat menutup koneksi");  
            }  
        }  
    }  
}
```

Gambar 4.3.2 Kode Form Login

##### b. Button Back dan Exit

Digunakan untuk kembali dan keluar pada halaman utama login

```

    private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
        this.setVisible(b: false);
        new Login().setVisible(b: true);
    }

    private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
        System.exit(status: 0);
    }

```

### 3. Form Register

#### a. Button Register

Digunakan untuk menyambungkan data pada database akun, dan memeriksa kata sandi yang dimasukkan harus sebanyak 8 huruf jika data yang dimasukkan sesuai dengan source code maka akan tampil Register berhasil dan jika data yang dimasukkan tidak sesuai maka akan muncul Register gagal.

```

    private void btnRegisterActionPerformed(java.awt.event.ActionEvent evt) {
        Connection conn = null;
        try {
            conn = (Connection) Config.configDB();
            String sql = "INSERT INTO akun ('email', 'username', 'password') VALUES "
                + "(" + txtEmail.getText() + "", "" + txtUsername.getText() + "", "" + txtPassword.getText() + ")";
            PreparedStatement pst = conn.prepareStatement(string: sql);
            if (txtPassword.getText().length() == 8) {
                pst.execute();
                JOptionPane.showMessageDialog(parentComponent:null, message:"Register Berhasil\nData Anda Berhasil Disimpan");
                this.setVisible(b: false);
                new Login().setVisible(b: true);
            } else {
                JOptionPane.showMessageDialog(parentComponent:null, message:"Password harus sebanyak 8 huruf atau angka");
                txtPassword.setText(t: "");
            }
            pst.close();
        } catch (Exception e) {
            JOptionPane.showMessageDialog(parentComponent:null, message:"Register Gagal\nTelah Terjadi Kesalahan");
            e.printStackTrace();
        } finally {
            if (conn != null) {
                try {
                    conn.close();
                } catch (SQLException ex) {
                    Logger.getLogger(Register.class.getName()).log(level: Level.SEVERE, msg:null, thrown: ex);
                }
            }
        }
    }

```

Gambar 4.3.2 Kode Form Register

#### b. Button Sign dan Exit

Digunakan untuk kembali dan keluar pada halaman utama sign up

```

    private void btnSignupActionPerformed(java.awt.event.ActionEvent evt) {
        this.setVisible(b: false);
        new Register().setVisible(b: true);
    }

    private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
        System.exit(status: 0);
    }

```

### 4. Form Utama

#### a. Form Utama

Source code dibawah ini digunakan untuk membuat frame utama aplikasi dengan berbagai formulir yang dapat ditampilkan di dalamnya, juga mengatur menu dan menanggapi

pemilihan menu dengan menampilkan formulir yang sesuai. Metode setForm digunakan untuk mengganti formulir yang ditampilkan di dalam frame.

```
public class Main extends javax.swing.JFrame {

    private Form_Home beranda;
    private Form_Kamar kamar;
    private Form_Profil profil;
    private Form_Keuangan keuangan;
    private Form_Transaksi transaksi;
    private Form_Riwayat riwayat;

    public Main() {
        initComponents();
        ImageIcon image = new ImageIcon(filename: "/com/coba/icon/logo.png");
        this.setIconImage(image: image.getImage());
        GlassPanePopup.install(frame: this);
        setBackground(new Color(r: 0, g: 0, b: 0, a: 0));
        beranda = new Form_Home();
        kamar = new Form_Kamar();
        profil = new Form_Profil();
        keuangan = new Form_Keuangan();
        transaksi = new Form_Transaksi();
        riwayat = new Form_Riwayat();

        menu.initMoving(fram: Main.this);
        menu.addEventMenuSelected(new EventMenuSelected() {
            @Override
            public void selected(int index) {
                if (index == 1) {
                    setForm(com:beranda);
                } else if (index == 2) {
                    setForm(com:kamar);
                } else if (index == 3) {
                    setForm(com:profil);
                } else if (index == 4) {
                    setForm(com:keuangan);
                } else if (index == 5) {
                    setForm(com:transaksi);
                } else if (index == 6) {
                    setForm(com:riwayat);
                } else if (index == 11) {
                    keluar();
                }
            }
        });
        setForm(new Form_Home());
    }

    private void keluar() {
        System.exit(0);
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    // End of variables declaration//GEN-END:variables
}
```

Gambar 4.3.2 Kode Form Utama

b. Set dan Keluar Form Utama

```
public void keluar() {
    int konfirmasiKeluar = JOptionPane.showConfirmDialog(parentComponent: null, message: "Apakah Anda ingin menutup aplikasi?", title: "Tutup Aplikasi", optionType: JOptionPane.YES_NO_OPTION);

    if (konfirmasiKeluar == JOptionPane.YES_OPTION) {
        this.dispose();
        System.exit(status: 0);
    }
}

public void setForm(JComponent com) {
    mainPanel.removeAll();
    mainPanel.add(comp: com);
    mainPanel.repaint();
    mainPanel.revalidate();
}
```

## 5. Form Kamar

a. Load table kamar

```
private void load_table() {
    Connection conn = null;
    try {
        int q;
        conn = (Connection) Config.configDB();
        String sql = "SELECT * FROM kamar";
        PreparedStatement pst = conn.prepareStatement(string: sql);
        ResultSet rs = pst.executeQuery(string: sql);
        ResultSetMetaData rss = rs.getMetaData();
        q = rss.getColumnCount();

        DefaultTableModel model = (DefaultTableModel) table.getModel();
        model.setRowCount(rowCount: 0);
        while (rs.next()) {
            Vector v2 = new Vector();
            for (int a = 1; a <= q - 1; a++) {
                v2.add(e: rs.getString(string: "no_kamar"));
                v2.add(e: rs.getString(string: "tipe_kamar"));
                v2.add(e: rs.getString(string: "harga_kamar"));
                v2.add(e: rs.getString(string: "status_kamar"));
            }
            model.addRow(rowData: v2);
        }

        rs.close();
        pst.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: null, message: "Data Gagal Ditampilkan");
        e.printStackTrace();
    } finally {
        if (conn != null) {
```

Gambar 4.3.2 Kode Form Load Table Kamar

## b. Tampilan Jumlah Kamar

```
private void fetchAndDisplayValue() {
    Connection conn = null;
    try {
        conn = (Connection) Config.configDB();
        PreparedStatement pstm = conn.prepareStatement(string: "SELECT COUNT(no_kamar) AS jumlah FROM kamar");
        ResultSet rs = pstm.executeQuery();

        if (rs.next()) {
            String valueFromDB = rs.getString(string: "jumlah");
            setText(text: valueFromDB);
        } else {
            setText(text: "NULL");
        }

        rs.close();
        pstm.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent:this, message:e.getMessage());
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {
                Logger.getLogger(name: ValuesCard2.class.getName()).log(level: Level.SEVERE, msg:null, thrown: ex);
            }
        }
    }
}
```

Gambar 4.3.2 Kode Form Tampilan Jumlah Kamar

## c. Cari Data Kamar

```
private void btnCariActionPerformed(java.awt.event.ActionEvent evt) {
    Connection conn = null;
    try {
        int q;
        conn = (Connection) Config.configDB();
        String sql = "SELECT * FROM kamar WHERE status_kamar = '" + cmbCariStatus.getSelectedItem() + "'";
        PreparedStatement pst = conn.prepareStatement(string: sql);
        ResultSet rs = pst.executeQuery(string: sql);
        ResultSetMetaData rss = rs.getMetaData();
        q = rss.getColumnCount();

        DefaultTableModel model = (DefaultTableModel) table.getModel();
        model.setRowCount(rowCount:0);
        while (rs.next()) {
            Vector v2 = new Vector();
            for (int a = 1; a <= q - 1; a++) {
                v2.add(e: rs.getString(string: "no_kamar"));
                v2.add(e: rs.getString(string: "tipe_kamar"));
                v2.add(e: rs.getString(string: "harga_kamar"));
                v2.add(e: rs.getString(string: "status_kamar"));
            }
            model.addRow(rowData:v2);
        }

        rs.close();
        pst.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent:null, message:"Data Gagal Ditampilkan");
        e.printStackTrace();
    } finally {
        if (conn != null) {
```

Gambar 4.3.2 Kode Form Cari Data Kamar

#### d. Tambah Data Kamar

```
private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {  
    Connection conn = null;  
    try {  
        conn = (Connection) Config.configDB();  
        String sql = "INSERT INTO kamar VALUES ('" + txtNoKamar.getText() + "', '" + cmbTipe.getSelectedItem() + "',  
        '' + txtHarga.getText() + "",'" + cmbStatus.getSelectedItem() + "')";  
        PreparedStatement pst = conn.prepareStatement(string: sql);  
        pst.execute();  
        JOptionPane.showMessageDialog(parentComponent: null, message: "Penambahan Kamar Berhasil");  
        pst.close();  
  
    } catch (Exception e) {  
        clear();  
        JOptionPane.showMessageDialog(parentComponent: null, message: "Tambah Data Gagal\nTelah Terjadi Kesalahan");  
        e.printStackTrace();  
    } finally {  
        if (conn != null) {  
            try {  
                conn.close();  
            } catch (SQLException ex) {  
                Logger.getLogger(name: Form_Kamar.class.getName()).log(level: Level.SEVERE, msg: null, thrown: ex);  
            }  
        }  
        load_table();  
        clear();  
    }  
}
```

Gambar 4.3.2 Kode Form Tambah Data Kamar

#### e. Edit Data Kamar

```
private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {  
    Connection conn = null;  
    try {  
        int konfirmasiHapus = JOptionPane.showConfirmDialog(parentComponent: null, message: "Apakah Anda ingin mengubah data  
ini?", title: "Ubah Data", optionType: JOptionPane.YES_NO_OPTION);  
  
        conn = (Connection) Config.configDB();  
        if (konfirmasiHapus == JOptionPane.YES_OPTION) {  
            String sql = "UPDATE kamar SET "  
            + "tipe_kamar = '" + cmbTipe.getSelectedItem() + "', "  
            + "harga_kamar = '" + txtHarga.getText() + "', "  
            + "status_kamar = '" + cmbStatus.getSelectedItem() + "' "  
            + "WHERE no_kamar = '" + txtNoKamar.getText() + "'";  
            PreparedStatement pst = conn.prepareStatement(string: sql);  
            pst.execute();  
            JOptionPane.showMessageDialog(parentComponent: this, message: "Perubahan Data Berhasil");  
            pst.close();  
        }  
  
    } catch (Exception e) {  
        JOptionPane.showMessageDialog(parentComponent: null, message: "Perubahan Data Gagal");  
        e.printStackTrace();  
    } finally {  
        if (conn != null) {  
            try {  
                conn.close();  
            } catch (SQLException ex) {  
                Logger.getLogger(name: Form_Kamar.class.getName()).log(level: Level.SEVERE, msg: null, thrown: ex);  
            }  
        }  
        load_table();  
    }  
}
```

Gambar 4.3.2 Kode Form Edit Data Kamar

```

TableActionEvent event = new TableActionEvent() {
    @Override
    public void onEdit(int row) {
        Connection conn = null;
        try {
            int pilihBaris = table.getSelectedRow();
            int pilihKolom = 0;
            Object nilai = table.getValueAt(row:pilihBaris, column: pilihKolom);

            conn = (Connection) Config.configDB();
            String sql = "SELECT * FROM kamar WHERE no_kamar = '" + nilai + "'";
            PreparedStatement pst = conn.prepareStatement(string: sql);
            ResultSet rs = pst.executeQuery();

            if (rs.next() == true) {
                txtNoKamar.setText(t: rs.getString(string: "no_kamar"));
                txtNoKamar.disable();
                cmbTipe.setSelectedItem(anObject: rs.getString(string: "tipe_kamar"));
                txtHarga.setText(t: rs.getString(string: "harga_kamar"));
                cmbStatus.setSelectedItem(anObject: rs.getString(string: "status_kamar"));
            }
            btnEdit.setVisible(aFlag: true);
            rs.close();
            pst.close();
        } catch (Exception e) {
            JOptionPane.showMessageDialog(parentComponent:null, message:"Terjadi Kesalahan");
            e.printStackTrace();
        } finally {
            if (conn != null) {
                try {
                    conn.close();
                } catch (SQLException ex) {

```

Gambar 4.3.2 Kode Form Edit Data Kamar

#### f. Hapus Data Kamar

```

    @Override
    public void onDelete(int row) {
        Connection conn = null;
        try {
            if (table.isEditing()) {
                table.getCellEditor().stopCellEditing();
            }

            int konfirmasiHapus = JOptionPane.showConfirmDialog(parentComponent:null, message:"Apakah Anda ingin
menghapus data ini?", title: "Hapus Kamar", optionType: JOptionPane.YES_NO_OPTION);

            if (konfirmasiHapus == JOptionPane.YES_OPTION) {
                int pilihBaris = table.getSelectedRow();
                int pilihKolom = 0;
                Object nilai = table.getValueAt(row:pilihBaris, column: pilihKolom);

                conn = (Connection) Config.configDB();
                String sql = "DELETE FROM kamar WHERE no_kamar = '" + nilai + "'";
                PreparedStatement pst = conn.prepareStatement(string: sql);
                pst.execute();
                DefaultTableModel model = (DefaultTableModel) table.getModel();
                model.removeRow(row);
                JOptionPane.showMessageDialog(parentComponent:null, message:"Data Berhasil Dihapus");
                pst.close();
            }
        } catch (Exception e) {
            JOptionPane.showMessageDialog(parentComponent:null, message:"Data Gagal Dihapus");
            e.printStackTrace();
        } finally {
            if (conn != null) {
                try {

```

Gambar 4.3.2 Kode Form Hapus Data Kamar

g. Reset Data Kamar

```
private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {  
    clear();  
    load_table();  
    btnEdit.setVisible(aFlag: false);  
    cmbCariStatus.setSelectedIndex(anIndex: -1);  
}
```

Gambar 4.3.2 Kode Form Reset Data Kamar

## 6. Form Profil

a. Load Table Profil

```
private void load_table() {  
    Connection conn = null;  
    try {  
        conn = (Connection) Config.configDB();  
        int q;  
        String sql = "SELECT no_kamar, email_penyewa, nama_penyewa, alamat_penyewa, notelp_penyewa, notelp_ortu FROM  
penyewa ORDER BY no_kamar ASC";  
        PreparedStatement pst = conn.prepareStatement(string: sql);  
        ResultSet rs = pst.executeQuery(string: sql);  
        ResultSetMetaData rss = rs.getMetaData();  
        q = rss.getColumnCount();  
  
        DefaultTableModel model = (DefaultTableModel) table.getModel();  
        model.setRowCount(rowCount: 0);  
        while (rs.next()) {  
            Vector v2 = new Vector();  
            for (int a = 1; a <= q - 1; a++) {  
                v2.add(e: rs.getString(string: "no_kamar"));  
                v2.add(e: rs.getString(string: "email_penyewa"));  
                v2.add(e: rs.getString(string: "nama_penyewa"));  
                v2.add(e: rs.getString(string: "alamat_penyewa"));  
                v2.add(e: rs.getString(string: "notelp_penyewa"));  
                v2.add(e: rs.getString(string: "notelp_ortu"));  
            }  
            model.addRow(rowData: v2);  
        }  
  
        rs.close();  
        pst.close();  
    }
```

Gambar 4.3.2 Kode Load Table Profil

## b. Mengisi Combo Box Profil

```
public void addValueModel() {
    Connection conn = null;
    try {
        cmbKamar.removeAllItems();
        conn = (Connection) Config.configDB();
        Statement stmt = conn.createStatement();
        ResultSet rs = stmt.executeQuery(string: "SELECT no_kamar FROM kamar");

        while (rs.next()) {
            String nilai = rs.getString(string: "no_kamar");
            cmbKamar.addItem(item: nilai);
        }

        rs.close();
        stmt.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: null, message: "Terjadi Kesalahan");
        e.printStackTrace();
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {
                Logger.getLogger(Form_Profil.class.getName()).log(level: Level.SEVERE, msg: null, thrown: ex);
            }
        }
    }
}
```

Gambar 4.3.2 Kode Combo Box Profil

## c. Tampilan Jumlah Profil

```
private void fetchAndDisplayValue() {
    Connection conn = null;
    try {
        conn = (Connection) Config.configDB();
        PreparedStatement pstm = conn.prepareStatement(string: "SELECT COUNT(id_penyewa) AS jumlah FROM penyewa");
        ResultSet rs = pstm.executeQuery();

        if (rs.next()) {
            String valueFromDB = rs.getString(string: "jumlah");
            setText(text: valueFromDB);
        } else {
            setText(text: "NULL");
        }

        rs.close();
        pstm.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: this, message: e.getMessage());
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {
                Logger.getLogger(ValuesCard.class.getName()).log(level: Level.SEVERE, msg: null, thrown: ex);
            }
        }
    }
}
```

Gambar 4.3.2 Kode Tampilan Profil

#### d. Tambah Profil

```
private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {
    Connection conn = null;
    try {
        conn = (Connection) Config.configDB();
        String sql = "INSERT INTO penyewa (no_kamar, email_penyewa, nama_penyewa, alamat_penyewa, notelp_penyewa, notelp_ortu) VALUES ('" +
                    + "'" + cmbKamar.getSelectedItem() + "', " +
                    + "'" + txtEmail.getText() + "', " +
                    + "'" + txtNama.getText() + "', " +
                    + "'" + txtAlamat.getText() + "', " +
                    + "'" + txtNoPribadi.getText() + "', " +
                    + "'" + txtNoOrtu.getText() + "')";
        PreparedStatement pst = conn.prepareStatement(string: sql);
        pst.execute();
        JOptionPane.showMessageDialog(parentComponent: null, message: "Penambahan Data Berhasil");
        pst.close();
    } catch (Exception e) {
        clear();
        JOptionPane.showMessageDialog(parentComponent: null, message: "Tambah Data Gagal\nTelah Terjadi Kesalahan");
        e.printStackTrace();
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {
                Logger.getLogger(Form_Profil.class.getName()).log(level: Level.SEVERE, msg: null, thrown: ex);
            }
        }
    }
    load_table();
    clear();
}
```

Gambar 4.3.2 Kode Tambah Profil

#### e. Edit Profil

```
private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {
    Connection conn = null;
    try {
        int konfirmasiHapus = JOptionPane.showConfirmDialog(parentComponent: null, message: "Apakah Anda ingin mengubah data ini?", title: "Ubah Data", optionType: JOptionPane.YES_NO_OPTION);

        conn = (Connection) Config.configDB();
        if (konfirmasiHapus == JOptionPane.YES_OPTION) {
            String sql = "UPDATE penyewa SET " +
                        + "email_penyewa = '" + txtEmail.getText() + "', " +
                        + "nama_penyewa = '" + txtNama.getText() + "', " +
                        + "alamat_penyewa = '" + txtAlamat.getText() + "', " +
                        + "notelp_penyewa = '" + txtNoPribadi.getText() + "', " +
                        + "notelp_ortu = '" + txtNoOrtu.getText() + "' " +
                        + "WHERE no_kamar = '" + cmbKamar.getSelectedItem() + "'";
            PreparedStatement pst = conn.prepareStatement(string: sql);
            pst.execute();
            JOptionPane.showMessageDialog(parentComponent: this, message: "Perubahan Data Berhasil");
            pst.close();
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: null, message: "Perubahan Data Gagal");
        e.printStackTrace();
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {
                Logger.getLogger(Form_Profil.class.getName()).log(level: Level.SEVERE, msg: null, thrown: ex);
            }
        }
    }
}
```

Gambar 4.3.2 Kode Edit Profil

## f. Aksi Edit

```
TableActionEvent event = new TableActionEvent() {
    @Override
    public void onEdit(int row) {
        Connection conn = null;
        try {
            int pilihBaris = table.getSelectedRow();
            int pilihKolom = 0;
            Object nilai = table.getValueAt(row:pilihBaris, column: pilihKolom);

            conn = (Connection) Config.configDB();
            String sql = "SELECT * FROM kamar WHERE no_kamar = '" + nilai + "'";
            PreparedStatement pst = conn.prepareStatement(string: sql);
            ResultSet rs = pst.executeQuery();

            if (rs.next() == true) {
                txtNoKamar.setText(t: rs.getString(string: "no_kamar"));
                txtNoKamar.disable();
                cmbTipe.setSelectedItem(anObject: rs.getString(string: "tipe_kamar"));
                txtHarga.setText(t: rs.getString(string: "harga_kamar"));
                cmbStatus.setSelectedItem(anObject: rs.getString(string: "status_kamar"));
            }
            btnEdit.setVisible(aFlag: true);
            rs.close();
            pst.close();
        } catch (Exception e) {
            JOptionPane.showMessageDialog(parentComponent:null, message:"Terjadi Kesalahan");
            e.printStackTrace();
        } finally {
            if (conn != null) {
                try {
                    conn.close();
                } catch (SQLException ex) {

```

Gambar 4.3.2 Kode Aksi Edit

## g. Aksi Delete

```
@Override
public void onDelete(int row) {
    Connection conn = null;
    try {
        if (table.isEditing()) {
            table.getCellEditor().stopCellEditing();
        }

        int konfirmasiHapus = JOptionPane.showConfirmDialog(parentComponent:null, message:"Apakah Anda ingin
menghapus data ini?", title: "Hapus Profil", optionType: JOptionPane.YES_NO_OPTION);

        if (konfirmasiHapus == JOptionPane.YES_OPTION) {
            int pilihBaris = table.getSelectedRow();
            int pilihKolom = 0;
            Object nilai = table.getValueAt(row:pilihBaris, column: pilihKolom);

            conn = (Connection) Config.configDB();
            String sql = "DELETE FROM penyewa WHERE no_kamar = '" + nilai + "'";
            PreparedStatement pst = conn.prepareStatement(string: sql);
            pst.execute();
            DefaultTableModel model = (DefaultTableModel) table.getModel();
            model.removeRow(row);
            JOptionPane.showMessageDialog(parentComponent:null, message:"Data Berhasil Dihapus");
            pst.close();
        }

    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent:null, message:"Data Gagal Dihapus");
        e.printStackTrace();
    } finally {
        if (conn != null) {
```

Gambar 4.3.2 Kode Aksi Delete

#### h. Reset Profil

```
private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {  
    load_table();  
    addValueModel();  
    clear();  
}
```

Gambar 4.3.2 Kode Reset Profil

### 7. Form Keuangan

#### a. Load Tabel Keuangan

```
private void load_table() {  
    Connection conn = null;  
    try {  
        int q;  
        conn = (Connection) Config.configDB();  
        String sql = "SELECT DATE_FORMAT(keuangan.tanggal, '%d %b %y') AS tanggal, keuangan.detail, keuangan.jumlah  
FROM keuangan";  
        PreparedStatement pst = conn.prepareStatement(string: sql);  
        ResultSet rs = pst.executeQuery(string: sql);  
        ResultSetMetaData rss = rs.getMetaData();  
        q = rss.getColumnCount();  
  
        DefaultTableModel model = (DefaultTableModel) table.getModel();  
        model.setRowCount(rowCount: 0);  
        while (rs.next()) {  
            Vector v2 = new Vector();  
            for (int a = 1; a <= q; a++) {  
                v2.add(rs.getString(string: "tanggal"));  
                v2.add(rs.getString(string: "detail"));  
                v2.add(rs.getString(string: "jumlah"));  
            }  
            model.addRow(rowData:v2);  
        }  
  
        rs.close();  
        pst.close();  
    } catch (Exception e) {  
        JOptionPane.showMessageDialog(parentComponent:null, message:"Data Gagal Ditampilkan");  
        e.printStackTrace();  
    } finally {  
        if (conn != null) f
```

Gambar 4.3.2 Kode Load Table Keungan

## b. Cari Data Keuangan

```
private void btnCariActionPerformed(java.awt.event.ActionEvent evt) {
    Connection conn = null;
    try {
        int q;
        conn = (Connection) Config.configDB();
        String sql = "SELECT * FROM keuangan WHERE MONTHNAME(tanggal) = '" + cmbBulan.getSelectedItem() + "'";
        PreparedStatement pst = conn.prepareStatement(string: sql);
        ResultSet rs = pst.executeQuery(string: sql);
        ResultSetMetaData rss = rs.getMetaData();
        q = rss.getColumnCount();

        DefaultTableModel model = (DefaultTableModel) table.getModel();
        model.setRowCount(rowCount: 0);
        while (rs.next()) {
            Vector v2 = new Vector();
            for (int a = 1; a <= q; a++) {
                v2.add(e: rs.getString(string: "tanggal"));
                v2.add(e: rs.getString(string: "detail"));
                v2.add(e: rs.getString(string: "jumlah"));
            }
            model.addRow(rowData:v2);
        }

        rs.close();
        pst.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent:null, message:"Data Gagal Ditampilkan");
        e.printStackTrace();
    } finally {
        if (conn != null) {
            try {
                conn.close();
            }
        }
    }
}
```

Gambar 4.3.2 Kode Cari Data Keuangan

## c. Tampilan Jumlah Pemasukan

```
private void fetchAndDisplayValue() {
    Connection conn = null;
    try {
        conn = (Connection) Config.configDB();
        PreparedStatement pstm = conn.prepareStatement(string: "SELECT tgl_transaksi, SUM(total_harga) AS jumlah FROM transaksi JOIN detail_bayar ON transaksi.id_transaksi = detail_bayar.id_transaksi WHERE MONTH(tgl_transaksi) = MONTH(CURDATE()) GROUP BY tgl_transaksi");
        ResultSet rs = pstm.executeQuery();

        if (rs.next()) {
            String valueFromDB = rs.getString(string: "jumlah");
            setText("Rp" + valueFromDB);
        } else {
            setText(text: "NULL");
        }

        rs.close();
        pstm.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent:this, message:e.getMessage());
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {
                Logger.getLogger(ValuesPemasukan.class.getName()).log(level: Level.SEVERE, msg:null, thrown: ex);
            }
        }
    }
}
```

Gambar 4.3.2 Kode Tampilan Jumlah Pemasukan

d. Tampilan Tambah Pengeluaran

```
private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {  
    String inputDate = txtDate.getText();  
    SimpleDateFormat inputFormat = new SimpleDateFormat(pattern:"dd-MMM-yyyy");  
    SimpleDateFormat outputFormat = new SimpleDateFormat(pattern:"yyyy-MM-dd");  
  
    Connection conn = null;  
    try {  
        Date parsedDate = inputFormat.parse(source: inputDate);  
        String formattedDate = outputFormat.format(date: parsedDate);  
  
        conn = (Connection) Config.configDB();  
        String sql = "INSERT INTO pengeluaran (tgl_pengeluaran, detail_pengeluaran, jumlah_pengeluaran) VALUES ('" +  
        formattedDate + "', '" + txtDetail.getText() + "', '" + txtJumlah.getText() + "')";  
        String sql2 = "INSERT INTO keuangan VALUES ('" + formattedDate + "', '" + txtDetail.getText() + "', '" +  
        txtJumlah.getText() + "')";  
        PreparedStatement pst = conn.prepareStatement(string: sql);  
        PreparedStatement pstm = conn.prepareStatement(string: sql2);  
        pst.execute();  
        pstm.execute();  
        JOptionPane.showMessageDialog(parentComponent:null, message:"Pengeluaran Berhasil Ditambahkan");  
        pst.close();  
        pstm.close();  
  
    } catch (Exception e) {  
        JOptionPane.showMessageDialog(parentComponent:null, message:"Error!\nTidak Dapat Melanjutkan");  
        e.printStackTrace();  
    } finally {  
        if (conn != null) {  
            try {  
                conn.close();  
            } catch (SQLException ex) {  
                Logger.getLogger(name:TambahPengeluaran.class.getName()).log(level:Level.SEVERE, msg:null, thrown: ex);  
            }  
        }  
    }  
}
```

Gambar 4.3.2 Kode Tampilan Jumlah Pengeluaran

e. Button Tambah dan Reset Pengeluaran

```
private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {  
    new TambahPengeluaran().setVisible(b: true);  
}  
  
private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {  
    load_table();  
}
```

Gambar 4.3.2 Kode Button Tambah Dan Reset Pengeluaran

f. Button Kembali Tambah Pengeluaran

```
private void btnKembaliActionPerformed(java.awt.event.ActionEvent evt) {  
    try {  
        int konfirmasiHapus = JOptionPane.showConfirmDialog(parentComponent:null, message:"Apakah Anda ingin membatalkan  
tambah pengeluaran?", title: "Batal Tambah Pengeluaran", optionType: JOptionPane.YES_NO_OPTION);  
  
        if (konfirmasiHapus == JOptionPane.YES_OPTION) {  
            this.setVisible(b: false);  
        }  
    } catch (Exception e) {  
        JOptionPane.showMessageDialog(parentComponent:null, message:"Error!\nTidak Dapat Kembali");  
        e.printStackTrace();  
    }  
}
```

Gambar 4.3.2 Kode Button Kembali dan Tambah Pengeluaran

#### g. Tampilan Jumlah Pengeluaran

```
private void fetchAndDisplayValue() {
    Connection conn = null;
    try {
        conn = (Connection) Config.configDB();
        PreparedStatement pstm = conn.prepareStatement(string: "SELECT tgl_pengeluaran, SUM(jumlah_pengeluaran) AS jumlah FROM pengeluaran WHERE MONTH(tgl_pengeluaran) = MONTH(CURDATE())");
        ResultSet rs = pstm.executeQuery();

        if (rs.next()) {
            String valueFromDB = rs.getString(string: "jumlah");
            setText("Rp" + valueFromDB);
        } else {
            setText(text: "NULL");
        }

        rs.close();
        pstm.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent:this, message: e.getMessage());
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {
                Logger.getLogger(ValuesPengeluaran.class.getName()).log(level: Level.SEVERE, msg:null, thrown: ex);
            }
        }
    }
}
```

Gambar 4.3.2 Kode Tampilan Jumlah Pengeluaran

## 8. Form Transaksi

#### a. Combo Box Transaksi

```
private void addValueModel() {
    Connection conn = null;
    try {
        cmbKamar.removeAllItems();
        conn = (Connection) Config.configDB();
        Statement stmt = conn.createStatement();
        ResultSet rs = stmt.executeQuery(string: "SELECT no_kamar FROM kamar");

        while (rs.next()) {
            String nilai = rs.getString(string: "no_kamar");
            cmbKamar.addItem(item: nilai);
        }

        rs.close();
        stmt.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent:null, message:"Terjadi Kesalahan");
        e.printStackTrace();
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {
                Logger.getLogger(Form_Transaksi.class.getName()).log(level: Level.SEVERE, msg:null, thrown: ex);
            }
        }
    }
}
```

Gambar 4.3.2 Kode Combo Box Transaksi

## b. Bayar Transaksi

```
private void btnBayarActionPerformed(java.awt.event.ActionEvent evt) {
    Connection conn = null;
    try {
        conn = (Connection) Config.configDB();
        String idTransaksi = "SELECT MAX(id_transaksi) AS id, tgl_transaksi FROM transaksi";
        PreparedStatement pstm = conn.prepareStatement(string: idTransaksi);
        ResultSet rs = pstm.executeQuery();
        rs.next();
        String nilaiID = rs.getString(string: "id");
        String nilaiTgl = rs.getString(string: "tgl_transaksi");

        String sql = "INSERT INTO detail_bayar VALUES ('" + nilaiID + "', '" + txtHargaKamar2.getText() + "', '" +
        txtHargaTambah2.getText() + "', '" + txtTotalHarga2.getText() + "')";
        String sql3 = "INSERT INTO keuangan (tanggal, jumlah) VALUES ('" + nilaiTgl + "', '" + txtTotalHarga2.
        getText() + "")";
        PreparedStatement pst = conn.prepareStatement(string: sql);
        PreparedStatement pstmmt = conn.prepareStatement(string: sql3);
        pst.execute();
        pstmmt.execute();
        JOptionPane.showMessageDialog(parentComponent: null, message: "Transaksi Berhasil");
        rs.close();
        pst.close();
        pstmmt.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: null, message: "Terjadi Kesalahan");
        e.printStackTrace();
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {

```

Gambar 4.3.2 Kode Bayar Transaksi

## c. Lanjut Transaksi

```
private void btnLanjutActionPerformed(java.awt.event.ActionEvent evt) {
    String inputDate1 = txtDate.getText();
    String inputDate2 = txtDate2.getText();
    SimpleDateFormat inputFormat = new SimpleDateFormat(pattern: "dd-MMM-yyyy");
    SimpleDateFormat outputFormat = new SimpleDateFormat(pattern: "yyyy-MM-dd");

    Connection conn = null;
    try {
        Date parsedDate1 = inputFormat.parse(source: inputDate1);
        Date parsedDate2 = inputFormat.parse(source: inputDate2);
        String formattedDate1 = outputFormat.format(date: parsedDate1);
        String formattedDate2 = outputFormat.format(date: parsedDate2);

        conn = (Connection) Config.configDB();
        String sql = "INSERT INTO transaksi (no_kamar, nama_penyewa, tgl_mulai_sewa, tgl_akhir_sewa, jumlah_bulan,
        fasilitas_tambahan, harga_tambahan) VALUES (" +
                    + "" + cmbKamar.getSelectedItem() + "", "
                    + "" + txtNama.getText() + "", "
                    + "" + formattedDate1 + "", "
                    + "" + formattedDate2 + "", "
                    + "" + txtJumlah.getText() + "", "
                    + "" + txtFasilitas.getText() + "", "
                    + "" + txtHargaTambah.getText() + ");";
        String sql2 = "SELECT harga_tambahan FROM transaksi WHERE id_transaksi = (SELECT MAX(id_transaksi) FROM
        transaksi);";
        String sql3 = "UPDATE tagihan SET tagihan.tgl_akhir_sewa = '" + formattedDate2 + "' WHERE tagihan.
        nama_penyewa = '" + txtNama.getText() + "'";
        PreparedStatement pst = conn.prepareStatement(string: sql);
        PreparedStatement pstm = conn.prepareStatement(string: sql2);
        PreparedStatement pstmmt = conn.prepareStatement(string: sql3);
        pst.execute();
        pstmmt.execute();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: null, message: "Terjadi Kesalahan");
        e.printStackTrace();
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {

```

Gambar 4.3.2 Kode Lanjut Transaksi

#### d. Batal Transaksi

```
private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {
    Connection conn = null;
    try {
        int konfirmasiHapus = JOptionPane.showConfirmDialog(parentComponent: null, message: "Apakah Anda ingin membatalkan transaksi?", title: "Batal Transaksi", optionType: JOptionPane.YES_NO_OPTION);

        conn = (Connection) Config.configDB();
        if (konfirmasiHapus == JOptionPane.YES_OPTION) {
            String sql = "DELETE FROM transaksi WHERE id_transaksi = (SELECT MAX(id_transaksi) FROM transaksi)";
            PreparedStatement pst = conn.prepareStatement(string: sql);
            pst.execute();
            pst.close();
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: null, message: "Terjadi Kesalahan");
        e.printStackTrace();
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {
                Logger.getLogger(Form_Transaksi.class.getName()).log(level: Level.SEVERE, msg: null, thrown: ex);
            }
        }
    }

    btnBatal.setVisible(aFlag: false);
    btnBayar.setVisible(aFlag: false);
    clear();
}
```

Gambar 4.3.2 Kode Batal Transaks

### 9. Form Riwayat

#### a. Load Tabel Riwayat

```
private void load_table() {
    Connection conn = null;
    try {
        int q;
        conn = (Connection) Config.configDB();
        String sql = "SELECT DATE_FORMAT(transaksi.tgl_transaksi, '%d %b %y') AS tanggal, transaksi.no_kamar, transaksi.nama_penyewa, DATE_FORMAT(transaksi.tgl_mulai_sewa, '%d %b %y') AS tgl_mulai, DATE_FORMAT(transaksi.tgl_akhir_sewa, '%d %b %y') AS tgl_akhir, transaksi.jumlah_bulan, detail_bayar.total_harga FROM transaksi JOIN detail_bayar ON transaksi.id_transaksi = detail_bayar.id_transaksi";
        PreparedStatement pst = conn.prepareStatement(string: sql);
        ResultSet rs = pst.executeQuery(string: sql);
        ResultSetMetaData rss = rs.getMetaData();
        q = rss.getColumnCount();

        DefaultTableModel model = (DefaultTableModel) table.getModel();
        model.setRowCount(rowCount: 0);
        while (rs.next()) {
            Vector v2 = new Vector();
            for (int a = 1; a <= q; a++) {
                v2.add(e: rs.getString(string: "tanggal"));
                v2.add(e: rs.getString(string: "no_kamar"));
                v2.add(e: rs.getString(string: "nama_penyewa"));
                v2.add(e: rs.getString(string: "tgl_mulai"));
                v2.add(e: rs.getString(string: "tgl_akhir"));
                v2.add(e: rs.getString(string: "jumlah_bulan"));
                v2.add(e: rs.getString(string: "total_harga"));
            }
            model.addRow(rowData: v2);
        }
        rs.close();
        pst.close();
    }
```

Gambar 4.3.2 Kode Load Tabel Riwayat

b. Tampilan Nilai Jumlah Tagihan

```
private void fetchAndDisplayValue() {
    Connection conn = null;
    try {
        conn = (Connection) Config.configDB();
        PreparedStatement pstm = conn.prepareStatement(string: "SELECT COUNT(*) AS jumlah_tagihan FROM tagihan WHERE tagihan.tgl_akhir_sewa < CURDATE()");
        ResultSet rs = pstm.executeQuery();

        if (rs.next()) {
            String valueFromDB = rs.getString(string: "jumlah_tagihan");
            setText(text: valueFromDB);
        } else {
            setText(text: "NULL");
        }

        rs.close();
        pstm.close();
    } catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: this, message: e.getMessage());
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException ex) {
                Logger.getLogger(ValuesCard3.class.getName()).log(level: Level.SEVERE, msg: null, thrown: ex);
            }
        }
    }
}
```

Gambar 4.3.2 Kode Tampilan Nilai Jumlah Tagihan

c. Reset Riwayat

```
private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {
    load_table();
}
```

Gambar 4.3.2 Kode Reset Riwayat

## **BAB 5.**

### **KESIMPULAN DAN SARAN**

#### **5.1 KESIMPULAN**

Dari hasil penelitian dapat disimpulkan bahwa Aplikasi manajemen kost dapat meningkatkan efisiensi operasional dengan mengotomatiskan banyak tugas administratif, seperti pencatatan pembayaran sewa, pemeliharaan data penyewa, dan pelacakan pembayaran. Dengan adanya sistem manajemen kost, pemilik atau pengelola properti dapat memiliki visibilitas yang lebih baik terhadap keuangan. Aplikasi ini dapat membantu dalam manajemen penyewa, seperti melacak riwayat penyewa, memonitor ketersediaan unit, dan mengelola proses perekrutan penyewa baru.

Jika aplikasi berhasil mengurangi waktu yang dibutuhkan untuk tugas-tugas administratif dan mengurangi potensi kesalahan, hal ini dapat menghasilkan penghematan waktu dan biaya bagi pengelola properti. Kesimpulan positif dapat ditarik jika aplikasi berhasil mendorong adopsi teknologi di dalam industri manajemen properti, membantu para pemilik properti untuk tetap terkini dengan perkembangan teknologi

#### **5.2 SARAN**

Sebaiknya sebelum menggunakan aplikasi Smart Kost tersebut pemilik kost perlu mengetahui prosedur penggunaannya agar dapat meminimalisir kesalahan yang dapat terjadi. Seperti contoh, pemilik kost harus mempelajari fitur - fitur yang terdapat dalam aplikasi tersebut sehingga pada saat penggunaan secara langsung dapat berjalan dengan lancar. Pemilik kost perlu mengetahui cara penggunaan fitur yang tersedia dengan dibimbing oleh pembuat Aplikasi hingga dipastikan bahwa pemilik kost dapat mengelola dan management kost dengan data dan hasil yang valid. Oleh karena itu dengan penulisan makalah ini dimana membahas tentang aplikasi Smart Kost semoga dapat bermanfaat bagi pemilik kost dalam mengelola kost. Tidak lupa penulis mengharapkan kritik dan saran yang bersifat membangun agar penulisan makalah - makalah selanjutnya dapat lebih baik lagi.

## **DAFTAR PUSTAKA**

- Eka, I. (2017). Perancangan Sistem Aplikasi. *Jurnal TEKNOIF*, 71-76.
- Ermatita. (2016). Sistem dan Perancangan Sistem Informasi. *Jurnal Sistem Informasi*, 968- 972
- Ismael. (2017). Rancangan Bangun Sistem Informasi. *Jurnal Edik Informatika*.
- Mustika, Euis, & Sukarti. (2017). Sistem Informasi Pencarian Tempat Kos Berbasis Geografis di Bandar Lampung. *Jurnal Cendikia Vol. 14 No. 1*.
- Noertjahyana, A. (2016). Metode RAD (Rapid Application Development) Beserta Tahapannya. *Metode Pengembang Sistem*, 33-42

## LAMPIRAN

