**LAPORAN KERJA PRAKTIK**

**Pengembangan Aplikasi Mobile Pencatatan Inventory pada CV Wijaya Lesmana Sejahtera**

**Diajukan sebagai syarat untuk menyelesaikan**

**Mata Kuliah Kerja Praktik**

**Oleh:**

**Fernando 2125250014**

**Dep’ Niel Sinaga 2125250058**

Logo, company name

Description automatically generated

**Program Studi Informatika**

**Fakultas Ilmu Komputer dan Rekayasa**

**Universitas Multi Data Palembang**

**2024**

**LEMBAR PENGESAHAN**

**LAPORAN KERJA PRAKTIK**

**Pengembangan Aplikasi Mobile Pencatatan Inventory pada CV Wijaya Lesmana Sejahtera**

**Oleh:**

**Fernando 2125250014**

**Dep’ Niel Sinaga 2125250058**

Palembang, 26 Juni 2024

Menyetujui:

Penyelia Perusahaan Dosen Pembimbing

Indra Wijaya, S.T. M. Ezar Al Rivan, M.Kom

Direktur NIK. 131092

Mengetahui,

Ketua Program Studi Informatika

Derry Alamsyah, S.Si., M.Kom., M.Pd

NIK. 111069

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS ILMU KOMPUTER DAN REKAYASA**

**UNIVERSITAS MULTI DATA PALEMBANG**

**PALEMBANG**

**2024**

**Fakultas Ilmu Komputer dan Rekayasa  
Universitas Multi Data Palembang**

Program Studi Informatika

Laporan Kerja Praktik

Semester Gasal/Genap\* Tahun 20XX/20XX

**Pengembangan Aplikasi Mobile Pencatatan Inventory pada CV Wijaya Lesmana Sejahtera**

Fernando 2125250014

Dep’ Niel Sinaga 2125250058

**Abstrak**

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.

**Kata kunci**: xxxxxxxxxx xxxxxxxxxxxxxx xxxxxxxxxx xxxxxxxxxx xxxxxxxx xxxxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxxxx xxxxxxxxx xxxxxxx

**Faculty of Computer Science and Engineering  
Universitas Multi Data Palembang**

Undergraduate Program in Informatics

Internship Report

2nd Semester of Academic Year 2023/2024

**Development of a Mobile Application for Inventory Recording at CV Wijaya Lesmana Sejahtera**

Fernando 2125250014

Dep’ Niel Sinaga 2125250058

**Abstract**

*xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx*xxx.

**Keywords**: xxxxxxxxxx xxxxxxxxxxxxxx xxxxxxxxxx xxxxxxxxxx xxxxxxxx xxxxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxxxx xxxxxxxxx xxxxxxx

**KATA PENGANTAR**

Puji syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas segala rahmat, karunia, serta hidayah-Nya yang selalu memberikan kemudahan dan keberkahan dalam setiap langkah kami dalam menyelesaikan Laporan Kerja Praktik ini yang berjudul "Pengembangan Aplikasi Mobile Pencatatan Inventory pada CV Wijaya Lesmana Sejahtera".

Selama proses penulisan laporan ini, penulis mendapat berbagai ilmu, bantuan serta informasi dari berbagai pihak, baik secara langsung ataupun tidak langsung. Oleh karena itu, penulis ingin menyampaikan ucapan terima kasih kepada pihak yang telah membantu dalam pembuatan laporan ini, yaitu sebagai berikut:

1. Bapak Johannes Petrus, S.Kom., M.T.I., selaku Rektor Universitas Multi Data Palembang yang telah memberikan kesempatan untuk pelaksanaan kerja. praktik ini.
2. Ibu Yulistia, S.Kom, M.TI, selaku Wakil Rektor I Universitas Multi Data Palembang.
3. Ibu Kathryn Sugara, S.E., M.Si., selaku Wakil Rektor II Universitas Multi Data Palembang.
4. Bapak Dedy Hermanto, S.Kom., M.T.I, selaku Wakil Rektor III Universitas Multi Data Palembang.
5. Charisma Ayu P, M.HRM, selaku Wakil Rektor IV Universitas Multi Data Palembang.
6. Bapak Dr. Wijang Widhiarso, M.Kom., selaku Dekan Fakultas Ilmu Komputer dan Rekayasa Universitas Multi Data Palembang.
7. Ibu Yoannita, M.Kom., selaku Wakil Dekan Fakultas Ilmu Komputer dan Rekayasa Universitas Multi Data Palembang.
8. Bapak Derry Alamsyah, S.Si, M.Kom., M.Pd., selaku Ketua Program Studi Informatika yang telah memberikan kesempatan dan persetujuan untuk pelaksanaan kerja praktik ini.
9. Bapak M. Ezar Al Rivan, M.Kom, sebagai pembimbing yang telah banyak membimbing penulis dalam menulis laporan kerja praktik ini.
10. Orang Tua dan keluarga penulis yang telah memberikan dukungan dan semangat kepada penulis.
11. Bapak Indra Wijaya, S.T., selaku Direktur dan Penyelia CV Wijaya Lesmana Sejahtera
12. Segenap Dosen Universitas Multi Data Palembang yang telah memberikan bimbingan akademik kepada penulis.
13. Staf Administrasi Universitas Multi Data Palembang yang telah membantu penulis dalam menyelesaikan urusan akademis.
14. Staf Perpustakaan Universitas Multi Data Palembang yang telah membantu penulis memeriksa kesalahan penulisan laporan kerja praktik ini.
15. Semua pihak yang telah terlibat dalam proses pembuatan laporan ini yang tidak bisa disebutkan satu-persatu.

Penulis menyadari bahwa dalam penyusunan Laporan Kerja Praktik ini, masih banyak kekurangan dan kelemahan yang perlu penulis perbaiki kedepannya. Oleh karena itu, kritik dan saran yang membangun dari pembaca sangat penulis harapkan guna meningkatkan kualitas Laporan Kerja Praktik ini.

Palembang, Juni 2024

Penulis

**DAFTAR ISI**

[**LEMBAR PENGESAHAN iv**](#_heading=h.gjdgxs)

[**Abstrak v**](#_heading=h.30j0zll)

[**Abstract vi**](#_heading=h.1fob9te)

[**KATA PENGANTAR vii**](#_heading=h.3znysh7)

[**DAFTAR ISI viii**](#_heading=h.2et92p0)

[**DAFTAR TABEL 1**](#_heading=h.tyjcwt)

[**DAFTAR GAMBAR 2**](#_heading=h.3dy6vkm)

[**BAB 1**](#_heading=h.1t3h5sf) **PENDAHULUAN 3**

[1.1](#_heading=h.4d34og8) Latar Belakang 3

[1.2](#_heading=h.2s8eyo1) Rumusan Masalah 4

[1.3](#_heading=h.17dp8vu) Profil Tempat Kerja Praktik 5

[1.3.1](#_heading=h.3rdcrjn) Denah Lokasi Kerja Praktik 5

[1.3.2](#_heading=h.44sinio) Visi Misi 8

[1.3.3](#_heading=h.2jxsxqh) Struktur Organisasi 9

[1.3.4](#_heading=h.3j2qqm3) Tugas dan Wewenang 10

[1.4](#_heading=h.1y810tw) Tujuan dan Manfaat Kerja Praktik 11

[1.5](#_heading=h.4i7ojhp) Ruang Lingkup 11

[1.6](#_heading=h.2xcytpi) Metodologi 12

[1.7](#_heading=h.3whwml4) Sistematika Penulisan 16

[**BAB 2**](#_heading=h.2bn6wsx) **PELAKSANAAN KERJA PRAKTIK 17**

[2.1](#_heading=h.3as4poj) Tinjauan Pustaka 17

[2.1.1](#_heading=h.1pxezwc) Judul Subbab 17

[2.1.2](#_heading=h.49x2ik5) Judul Subbab 18

[2.1.3](#_heading=h.2p2csry) Judul Subbab 19

[2.1.4](#_heading=h.147n2zr) Judul Subbab 20

[2.1.5](#_heading=h.3o7alnk) Judul Subbab 20

[2.2](#_heading=h.23ckvvd) Analisis Kebutuhan 21

[2.3](#_heading=h.1hmsyys) Deskripsi Pekerjaan 24

[2.4](#_heading=h.2grqrue) Kesesuaian Ilmu Teoritis dan Praktis 26

[**BAB 3**](#_heading=h.vx1227) **HASIL DAN PEMBAHASAN 28**

[3.1](#_heading=h.1v1yuxt) Perancangan Sistem 28

[3.1.1](#_heading=h.4f1mdlm) subbab 28

[3.1.2](#_heading=h.19c6y18) subbab 29

[3.1.3](#_heading=h.28h4qwu) subbab 30

[3.1.4](#_heading=h.nmf14n) subbab 30

[3.2](#_heading=h.37m2jsg) Pengujian Program 31

[3.3](#_heading=h.2lwamvv) Hasil Kuesioner 32

[3.4](#_heading=h.3l18frh) Judul Subbab 34

[3.5](#_heading=h.2zbgiuw) Judul Subbab 39

[**BAB 4**](#_heading=h.2dlolyb) **PENUTUP 43**

[4.1](#_heading=h.sqyw64) Kesimpulan 43

[4.2](#_heading=h.3cqmetx) Saran 43

[**DAFTAR PUSTAKA 45**](#_heading=h.1rvwp1q)

[**LAMPIRAN 47**](#_heading=h.4bvk7pj)

[Lembar Konsultasi 48](#_heading=h.2r0uhxc)

[Kode Program 49](#_heading=h.1664s55)

[<judul lampiran> 50](#_heading=h.3q5sasy)

[<judul lampiran> 51](#_heading=h.25b2l0r)

**DAFTAR TABEL**

[Tabel 2.1 Jadwal Kerja Praktik 24](#_heading=h.41mghml)

[Tabel 3.1 Waktu dan Ruangan Perkuliahan 29](#_heading=h.2u6wntf)

[Tabel 3.2 Uji Coba *Form Login* 31](#_heading=h.1mrcu09)

[Tabel 3.3 Uji Coba *xxxx xxxxx* 32](#_heading=h.46r0co2)

[Tabel 3.4 Dosen Pengajar Mata Kuliah Semester Gasal 33](#_heading=h.111kx3o)

[Tabel 3.5 FGA 10 Kromosom & 2 Individu Setiap Kromosom 35](#_heading=h.4k668n3)

[Tabel 3.6 Hasil Pengujian FGA Skenario 20 Individu Semester Gasal 39](#_heading=h.1egqt2p)

[Tabel 3.7 Hasil Pengujian GA Skenario 20 Individu Semester Gasal 40](#_heading=h.3ygebqi)

**DAFTAR GAMBAR**

[**BAB 1 PENDAHULUAN 14**](#_heading=h.1t3h5sf)

[1.1 Latar Belakang 14](#_heading=h.4d34og8)

[1.2 Rumusan Masalah 16](#_heading=h.2s8eyo1)

[1.3 Profil Lokasi Kerja Praktik 16](#_heading=h.u0h2nekfo4nq)

[1.3.1 Denah Lokasi Kerja Praktik 16](#_heading=h.3rdcrjn)

[1.3.2 Visi Misi 17](#_heading=h.44sinio)

[1.3.3 Struktur Organisasi 18](#_heading=h.2jxsxqh)

[1.3.4 Tugas dan Wewenang 18](#_heading=h.3j2qqm3)

[1.4 Tujuan dan Manfaat Kerja Praktik 20](#_heading=h.1y810tw)

[1.5 Ruang Lingkup 21](#_heading=h.4i7ojhp)

[1.6 Metodologi 21](#_heading=h.2xcytpi)

[1.7 Sistematika Penulisan 23](#_heading=h.3whwml4)

[**BAB 2 PELAKSANAAN KERJA PRAKTIK 25**](#_heading=h.2bn6wsx)

[2.1 Tinjauan Pustaka 25](#_heading=h.3as4poj)

[2.1.1 Android 25](#_heading=h.1pxezwc)

[2.1.2 Java 26](#_heading=h.49x2ik5)

[2.1.4.1 Realtime Database 28](#_heading=)

[2.2 Teknologi yang Telah Dimanfaatkan 29](#_heading=h.23ckvvd)

[2.3 Analisis Permasalahan 30](#_heading=h.mkxy2n15jsu5)

[2.4 Analisis Kebutuhan 30](#_heading=h.jtqiq392vij8)

[2.5 Deskripsi Pekerjaan 32](#_heading=h.1hmsyys)

[2.6 Kesesuaian Ilmu Teoritis dan Praktis 35](#_heading=h.2grqrue)

[**BAB 3 HASIL DAN PEMBAHASAN 36**](#_heading=h.vx1227)

[3.1 Perancangan Sistem 36](#_heading=h.1v1yuxt)

[3.1.1 Use Case Diagram 36](#_heading=h.4f1mdlm)

[3.1.2 Tampilan Aplikasi 38](#_heading=h.19c6y18)

[3.2 Pengujian Program 55](#_heading=h.37m2jsg)

[3.3 Hasil Kuesioner 61](#_heading=h.2lwamvv)

[3.4 Judul Subbab 62](#_heading=h.3l18frh)

[3.5 Judul Subbab 70](#_heading=h.2zbgiuw)

[**BAB 4 PENUTUP 75**](#_heading=h.2dlolyb)

[4.1 Kesimpulan 75](#_heading=h.sqyw64)

[4.2 Saran 75](#_heading=h.3cqmetx)

[**LAMPIRAN 81**](#_heading=h.4bvk7pj)

[Lembar Konsultasi 82](#_heading=h.2r0uhxc)

[Kode Program 84](#_heading=h.1664s55)

[<judul lampiran> 85](#_heading=h.3q5sasy)

[<judul lampiran> 86](#_heading=h.25b2l0r)

# PENDAHULUAN

Pada Bab 1 akan membahas Latar Belakang, Rumusan Masalah, Profil CV Wijaya Lesmana Sejahtera yang mencakup Denah Lokasi, Visi dan Misi, Struktur Organisasi, Tugas dan Wewenang dari CV Wijaya Lesmana Sejahtera, Tujuan dan Manfaat Kerja Praktik, Ruang Lingkup Kerja Praktik, dan Metodologi yang berisikan metode yang penulis gunakan dalam dalam membangun Aplikasi

## Latar Belakang

Aplikasi merupakan hal yang penting dalam suatu organisasi atau perusahaan (Alda, 2021). Dalam era digital yang semakin berkembang pesat, penggunaan teknologi aplikasi telah menjadi kunci dalam meningkatkan efisiensi, produktivitas, dan akurasi dalam berbagai aspek bisnis, termasuk manajemen inventaris. Dengan memanfaatkan aplikasi yang tepat, perusahaan dapat mengoptimalkan proses pencatatan, pemantauan, dan manajemen inventaris secara efisien, serta meminimalkan kesalahan manusia yang mungkin terjadi dalam proses manual. Oleh karena itu, pengembangan aplikasi mobile pencatatan inventaris menjadi sebuah langkah strategis bagi CV Wijaya Lesmana Sejahtera untuk meningkatkan daya saing dan efektivitas operasionalnya di tengah persaingan bisnis yang semakin ketat.

CV Wijaya Lesmana Sejahtera bergerak di bidang jasa sebagai Kontraktor Interior bagi orang-orang yang ingin mendesain interior di bangunannya. Kontraktor Interior bertanggung jawab dalam mendesain, merencanakan, mengatur, dan membuat berbagai jenis kebutuhan interior yang ada. Perusahaan akan bekerja sama dengan pihak yang terkait dalam perancangan desain interior yang nanti akan dibuat.

CV Wijaya Lesmana Sejahtera dihadapkan pada berbagai kendala dalam pengelolaan pergudangan. Salah satu permasalahan utama yang dihadapi adalah kecenderungan melakukan kesalahan dalam input atau pencatatan data yang akurat terkait dengan barang-barang yang masuk dan keluar dari pergudangan. Ketidakakuratan informasi terkait inventaris dapat mengganggu efisiensi operasional dan manajemen stok barang secara efektif di perusahaan.

Perusahaan ini kesulitan mengetahui secara pasti informasi tentang barang masuk dan keluar akibat dari kurangnya disiplin dalam memperbaharui data yang ada menyebabkan informasi mengenai status barang seringkali tidak akurat atau tidak terkini (*real time*). Hal ini dapat mengganggu efisiensi operasional serta menghambat proses pengambilan keputusan yang berkaitan dengan manajemen pergudangan.

Selain itu, dengan adanya beberapa proyek interior yang berlangsung secara bersamaan, proses pencatatan barang di pergudangan CV Wijaya Lesmana Sejahtera menjadi semakin rumit dan kompleks. Proyek-proyek tersebut memerlukan manajemen stok yang cermat dan terkoordinasi, namun pengelolaan data yang tidak terstruktur menyulitkan upaya tersebut. Keterbatasan dalam proses pencatatan barang juga menyebabkan pengambilan barang menjadi tidak dapat diawasi, sehingga menghambat perencanaan dan eksekusi proyek secara efisien di CV Wijaya Lesmana Sejahtera.

## Rumusan Masalah

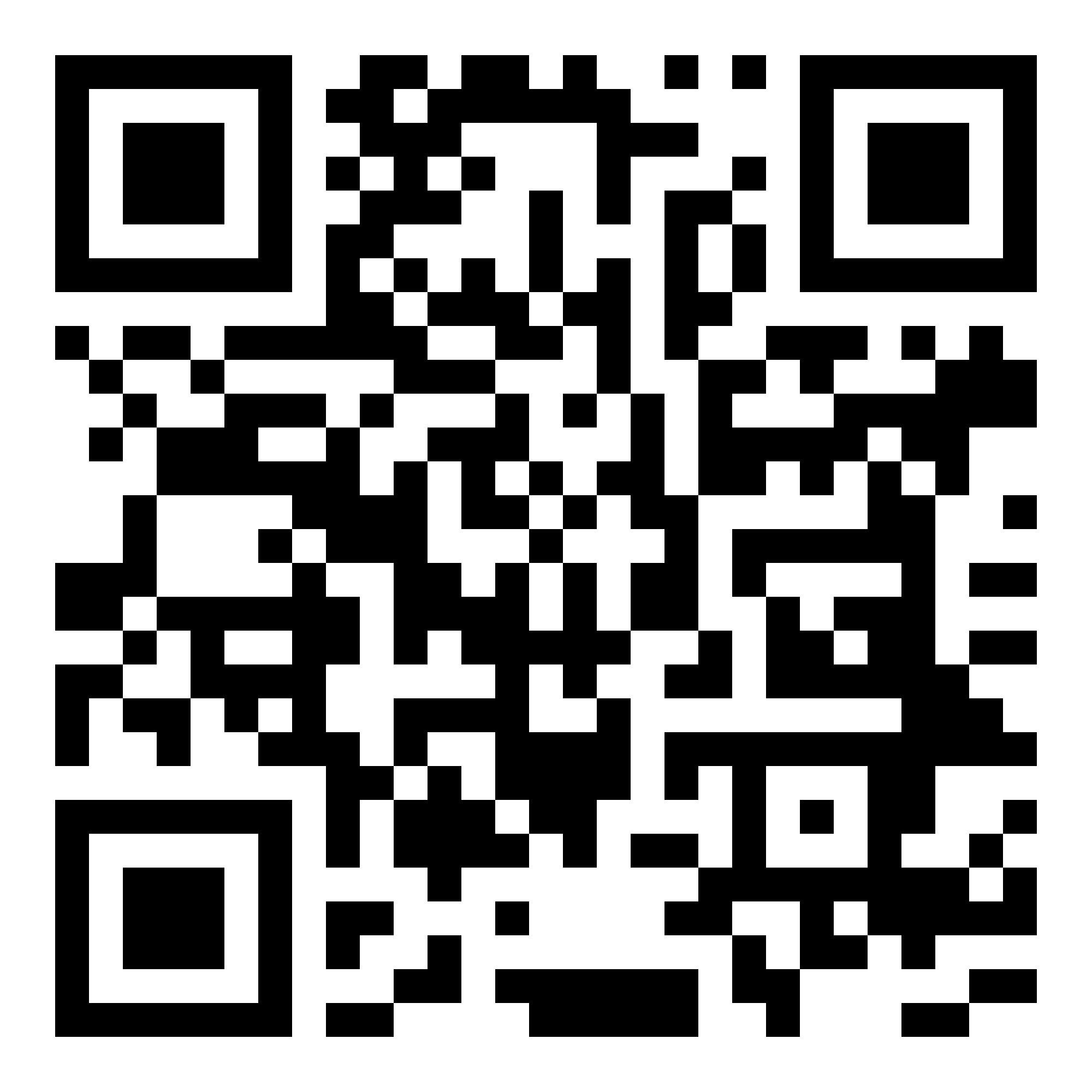
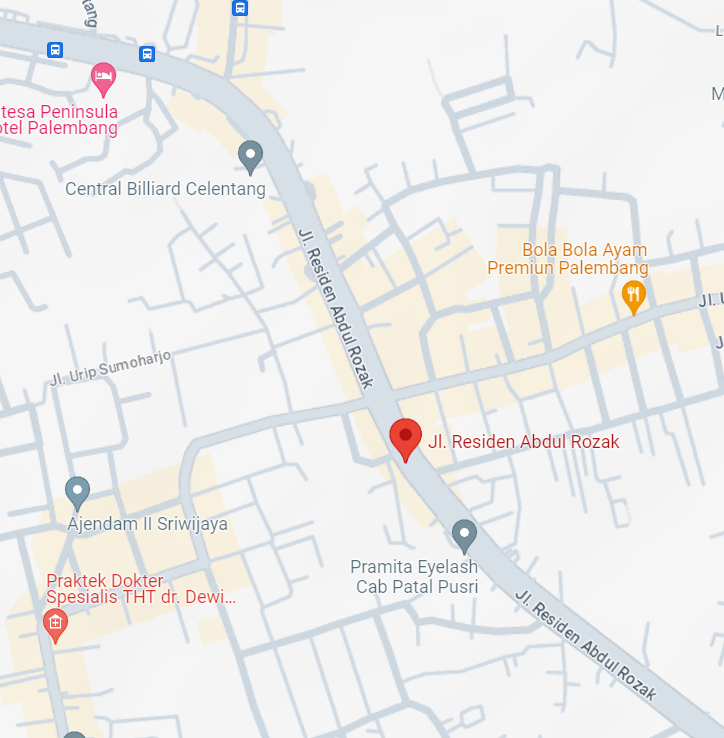
CV Wijaya Lesmana Sejahtera memiliki kendala dalam pengelolaan pergudangan. Salah satu permasalahan utama yang dihadapi adalah kecenderungan melakukan kesalahan dalam input atau pencatatan data yang akurat terkait dengan barang-barang yang masuk dan keluar dari pergudangan. Ketidakakuratan informasi terkait inventaris dapat mengganggu efisiensi operasional dan manajemen stok barang secara efektif di perusahaan.

### Profil Lokasi Kerja Praktik

CV Wijaya Lesmana Sejahtera bergerak di bidang jasa sebagai Kontraktor Interior bagi orang-orang yang ingin mendesain interior di bangunannya. Kontraktor Interior bertanggung jawab dalam mendesain, merencanakan, mengatur, dan membuat berbagai jenis kebutuhan interior yang ada. Perusahaan akan bekerja sama dengan pihak yang terkait dalam perancangan desain interior yang nanti akan dibuat.

### Denah Lokasi Kerja Praktik

CV Wijaya Lesmana Sejahtera beralamat di Lr. Tembusan No.1828-A, Kel. Bukit Lama, Kec. Ilir Barat I, Kota Palembang, Provinsi Sumatera Selatan. CV Wijaya Lesmana Sejahtera berdiri pada tanggal 27 November 2015 dengan NIB Nomor 1208000220212. Perusahaan ini merupakan penyedia jasa layanan *interior contractor* yang didirikan dan dikelola oleh Indra Wijaya sebagai pemilik sekaligus direktur dari CV Wijaya Lesmana Sejahtera. CV Wijaya Lesmana Sejahtera memiliki pusat penjualan dan perencanaan yang berada di Jalan Residen Haji Abdul Rozak, Desa/Kelurahan Bukit Sangkal, Kec. Kalidoni, Kota Palembang, Provinsi Sumatera Selatan. Denah lokasi pusat penjualan dan perencanaan diperlihatkan pada Gambar 1.1 yang dapat diakses melalui link <https://maps.app.goo.gl/jhm3kCU81nmH71Jf7>.

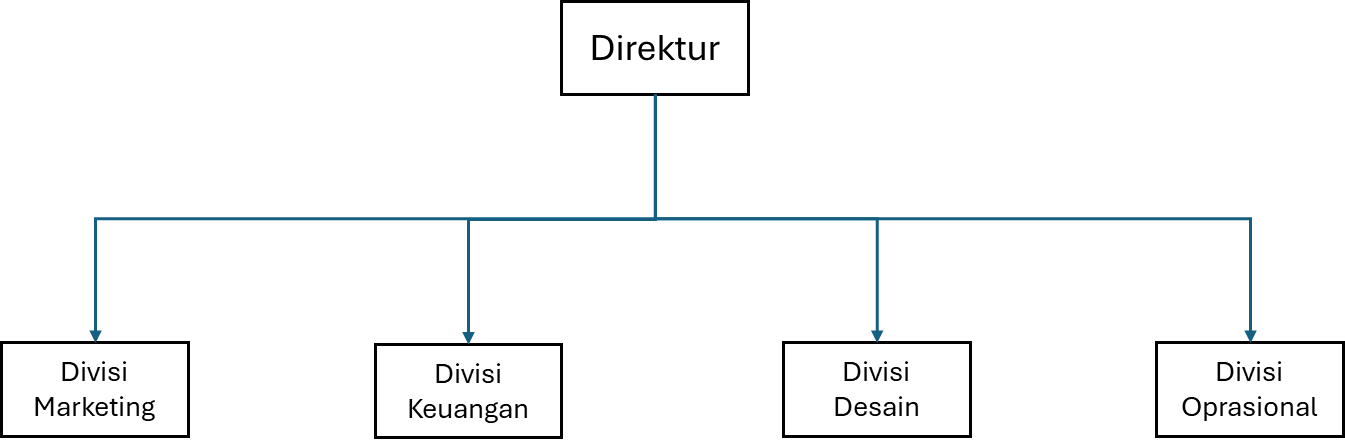


**Gambar 1.1 Denah Lokasi Kerja Praktik CV Wijaya Lesmana Sejahtera**

### Visi Misi

CV Wijaya Lesmana Sejahter**a** mempunyai visi yakni menjadi sebuah perusahaan yang berkompeten di bidang Kontraktor Interior berbagai produk *Kitchen Set*, Plafon, Karaoke, Kamar Set, *Interior*, *Eksterior*, dll. Dengan meningkatkan kualitas dari produk yang ditawarkan serta Desain yang menarik. Sedangkan misi yang terus akan dikembangkan adalahmenjalin hubungan yang baik dengan pelanggan serta mengutamakan kepuasan pelanggan dan memberikan hasil yang memuaskan dan berkualitas pada konsumen.

### Struktur Organisasi



**Gambar 1.2 Struktur Organisasi**

CV Wijaya Lesmana Sejahtera ini juga memiliki struktur organisasi pegawai yang digunakan untuk mengetahui bagian-bagian apa saja yang merupakan tanggung jawab masing-masing pekerjaan. CV Wijaya Lesmana Sejahteramemiliki beberapa karyawan dengan 4 (empat) divisi, yaitu Divisi Keuangan, Divisi Desain, Divisi *Marketing*, dan Divisi Operasional. Dimana Direktur sebagai pemimpin dari keempat divisi.

### Tugas dan Wewenang

Sebagai pemimpin perusahaan, Direktur bertanggung jawab atas pengambilan keputusan strategis dan pengawasan seluruh operasional perusahaan. Direktur memastikan setiap divisi bekerja sesuai dengan visi dan misi perusahaan, serta menjalin hubungan baik dengan klien dan mitra bisnis seperti bertemu dengan klien, membuat kontrak dengan klien untuk proyek yang akan dikerjakan. Selain itu, Direktur bertugas untuk memantau perkembangan proyek, dan mengarahkan kebijakan perusahaan untuk mencapai tujuan jangka panjang.

Divisi Keuangan bertanggung jawab untuk mengelola semua aspek keuangan perusahaan, termasuk akuntansi, pengelolaan anggaran, dan penyusunan laporan keuangan. Mereka memastikan semua transaksi keuangan dicatat dengan benar dan transparan. Selain itu, divisi ini bertugas untuk mengelola arus kas, memonitor pengeluaran dan pemasukan, serta memastikan ketersediaan dana yang cukup untuk operasional perusahaan. Divisi Keuangan juga bertanggung jawab untuk mengembangkan strategi keuangan yang mendukung pertumbuhan perusahaan.

Divisi Desain memiliki tugas utama dalam perancangan dan pengembangan konsep interior. Mereka bekerja mulai dari membuat desain awal berdasarkan kebutuhan dan preferensi klien, hingga mempresentasikan konsep kepada klien untuk mendapatkan persetujuan. Setelah desain disetujui, divisi ini mengembangkan desain detail yang siap untuk diimplementasikan. Mereka juga berkoordinasi dengan divisi operasional untuk memastikan bahwa desain yang telah dibuat dapat direalisasikan sesuai dengan rencana dan standar kualitas yang ditetapkan.

Divisi Marketing bertanggung jawab untuk mempromosikan layanan perusahaan dan mengembangkan strategi pemasaran yang efektif. Tugas mereka meliputi riset pasar, serta mengelola kampanye pemasaran dan promosi. Divisi ini juga mengurus branding perusahaan, menjalin dan memelihara hubungan dengan klien, serta mengelola komunikasi publik dan media sosial. Tujuannya adalah untuk meningkatkan visibilitas dan reputasi perusahaan di pasar, serta menarik lebih banyak klien.

Divisi Operasional mengatur dan mengelola pelaksanaan proyek desain interior di lapangan. Tugas mereka mencakup koordinasi dengan tim desain untuk memahami detail proyek, pengawasan tenaga kerja, dan pengelolaan logistik bahan dan peralatan yang dibutuhkan. Divisi ini memastikan bahwa setiap proyek berjalan sesuai dengan jadwal, dan standar kualitas yang ditetapkan. Mereka juga bertanggung jawab untuk menangani masalah yang muncul selama proses implementasi dan memastikan bahwa proyek selesai dengan memuaskan bagi klien.

## Tujuan dan Manfaat Kerja Praktik

Tujuan Kerja Praktik ini adalah untuk mengembangkan aplikasi mobile demi meningkatkan efisiensi dan efektivitas pengelolaan pergudangan di CV Wijaya Lesmana Sejahtera. Manfaat dari Kerja Praktik ini adalah sebagai berikut:

1. Memberikan pengalaman berharga dalam mengembangkan aplikasi yang berkualitas dan berorientasi pada solusi bagi penyelia.
2. Memperluas dan memperdalam keterampilan dalam mengembangkan aplikasi mobile, manajemen proyek, dan komunikasi dengan penyelia.

## Ruang Lingkup

Dalam Kerja Praktik ini, penulis memberikan beberapa batasan masalah agar pengembangan aplikasi sesuai tujuan, antara lain sebagai berikut:

1. Aplikasi dapat melakukan proses pengelolaan barang, seperti input data barang, hapus data barang, edit data barang, pencarian data barang, dan pembuatan laporan stok barang.
2. Aplikasi dapat melakukan proses pengelolaan data klien seperti input klien, edit klien, hapus klien dan pencarian data klien.
3. Aplikasi dapat mengontrol stok barang, dan melakukan pengecekan stok barang yang akan habis. Aplikasi dapat mengedit data pembelian barang, menghapus data pembelian barang, dan melihat laporan pemakaian barang berdasarkan tempat/klien, data barang masuk, dan data barang keluar berdasarkan periode tertentu, *supplier*, dan jenis barang.
4. Aplikasi dapat menyimpan data klien tetap, mengedit data klien, menghapus data klien, mencari data klien, dan membuat laporan daftar klien tetap, serta laporan pemakaian barang berdasarkan klien masing-masing.

## Metodologi

Metodologi yang digunakan dalam Kerja Praktik ini adalah. *System Development Life Cycle* (SDLC), yaitu model *Waterfall*. Metode waterfall merupakan pengembangan perangkat lunak dengan penyelesaian satu tahap secara lengkap sebelum melangkah ke tahap berikutnya secara sistematis. Metode *waterfall* ini harus dilakukan secara berurutan sesuai dengan tahap yang ada pada Gambar 1.3.



**Gambar 1.3 Tahapan Metode *Waterfall***

Adapun beberapa tahapan di dalam *Waterfall* sebagai berikut :

1. Tahap *Analysis*

Pada tahap ini, wawancara dilakukan untuk mengumpulkan informasi kebutuhan apa saja yang diperlukan untuk pengembangan aplikasi dan permasalahan yang ada

1. Tahap *Design*

Pada tahap ini dilakukan pembuatan *Use Case Diagram* sebagai desain perangkat lunak yang mencakup keseluruhan sistem, desain *database*, desain antarmuka pengguna, dan juga spesifikasi teknis.

1. Tahap *Development*

Pada tahap ini dilakukan implementasi perangkat lunak berdasarkan desain yang telah ditetapkan dengan mengubah konsep desain menjadi kode yang dapat dieksekusi oleh komputer.

1. Tahap *Testing*

Setelah tahap pengembangan selesai, perangkat lunak diuji untuk memastikan bahwa setiap komponen yang ada berfungsi sesuai yang diharapkan. Hal yang dapat diuji yakni pengujian fungsionalitas, integrasi, kinerja, dan keamanan.

1. Tahap *Deployment*

Setelah pengujian selesai dilakukan dan tidak ada kendala yang muncul maka perangkat lunak akan diterapkan ke Pengguna akhir. Dalam proses ini dapat dilakukan instalasi perangkat lunak, migrasi data, dan juga pelatihan bagi pengguna.

## Sistematika Penulisan

Laporan kerja praktik ini disusun secara sistematik dari bab pendahuluan sampai dengan bab akhir. Adapun penjelasan terkait dari laporan kerja praktik ini adalah :

**BAB 1 PENDAHULUAN**

Pada bab ini dijelaskan mengenai latar belakang masalah dari pihak penyelia, profil tempat kerja praktik, tujuan dan manfaat kerja praktik, ruang lingkup, metodologi pengembangan sistem informasi yang digunakan, dan sistematika penulisan laporan kerja praktik.

**BAB 2 PELAKSANAAN KERJA PRAKTIK**

Pada bab ini menjelaskan tentang tinjauan pustaka mengenai elemen-elemen apa saja yang digunakan selama pelaksanaan kerja praktik. Selain itu, terdapat juga analisis kebutuhan, deskripsi pekerjaan, serta kesesuaian antara ilmu teoritis dan praktis.

**BAB 3 PELAKSANAAN KERJA PRAKTIK**

Pada bab ini membahas hasil dari pelaksanaan kerja praktik, yang terdiri dari perancangan sistem, pengujian program, dan hasil kuisioner.

**BAB 4 PENUTUP**

Pada bab ini menjelaskan tentang kesimpulan yang dapat diambil dari hasil kerja praktik yang telah terlaksana dan saran.

# PELAKSANAAN KERJA PRAKTIK

Kerja praktik adalah salah satu mata kuliah wajib bagi mahasiswa di Universitas MDP. Tujuan dari kerja praktik tersebut diantaranya adalah dapat mengembangkan ilmu yang telah didapatkan dalam proses perkuliahan, dan tentunya memberikan pengalaman kerja bagi mahasiswa agar tujuan dari pelaksanaan kerja praktik ini dapat memberikan wawasan serta kemampuan dalam dunia kerja bidang teknologi informasi. Pada bab ini akan dibahas mengenai tinjauan pustaka, analisis kebutuhan, dan deskripsi pekerjaan, serta kesesuaian ilmu teoritis dan praktis.

## Tinjauan P ustaka

Pada bab ini akan dibahas berbagai literatur yang relevan dengan pembuatan aplikasi. Tinjauan pustaka ini bertujuan untuk memberikan dasar teoritis yang kuat mengenai elemen-elemen yang dipakai dalam pembuatan aplikasi kerja praktik ini. adapun elemen-elemen tersebut yaitu:

### Android

Android (sistem operasi) –OS Android –Merupakan sebuah sistem operasi yang berbasis Linux untuk telepon seluler seperti telepon pintar dan komputer tablet (Deybi et al., 2015). Keunggulan utama Android terletak pada sifatnya yang terbuka dan fleksibel. Sebagai sistem operasi open-source, Android memberikan kebebasan kepada pengembang untuk memodifikasi dan menyesuaikan sistem sesuai dengan kebutuhan mereka. Ini memungkinkan kompatibilitas yang luas dengan berbagai jenis perangkat keras dari berbagai produsen. Android juga terintegrasi dengan berbagai layanan Google seperti Gmail, Google Maps, dan Google Drive, yang menambah nilai fungsionalitasnya. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak (Novita dan Zahri, 2018)

### Java

Java adalah sebuah teknologi yang diperkenalkan oleh Sun Microsystems pada pertengahan tahun 1990 (Budiyanto dan Sultonuddin, 2021). Java adalah selaku teknologi yang baik dibanding hanya sebuah bahasa pemrograman konvensional yang lain, karena java lebih lengkap. Java dirancang untuk memiliki portabilitas tinggi, memungkinkan kode yang ditulis di satu platform untuk dijalankan di berbagai platform lainnya tanpa perubahan berarti. Bahasa ini memiliki sintaks yang mirip dengan C++ namun lebih sederhana, dengan fokus pada menghilangkan beberapa fitur kompleks dan berpotensi membingungkan dari C++, seperti operator overloading dan pointer aritmetik. Java juga memiliki manajemen memori otomatis melalui Garbage Collection, yang membantu mengelola alokasi dan dealokasi memori secara otomatis, sehingga mengurangi risiko kebocoran memori. Dengan pustaka standar yang kaya dan ekosistem yang luas, Java banyak digunakan dalam berbagai aplikasi mulai dari pengembangan aplikasi Android, aplikasi web, sistem perusahaan, hingga aplikasi embedded. Kelebihan lainnya adalah keamanannya yang tinggi dan kemampuannya untuk diintegrasikan dengan teknologi lain, menjadikannya pilihan yang populer di kalangan pengembang dan perusahaan di seluruh dunia.

* + 1. **Android Studio**

Android Studio merupakan lingkungan pengembangan perangkat lunak terpadu – Integrated Development Environment (IDE) untuk pengembangan aplikasi android, berdasarkan IntelliJ IDEA (Budiyanto dan Sultonuddin, 2021). Fitur utamanya meliputi editor kode yang cerdas dengan kemampuan pelengkapan otomatis dan analisis kode, desain layout visual yang memungkinkan pengembang merancang antarmuka pengguna secara intuitif, serta sistem build berbasis Gradle yang kuat untuk mengelola dependensi dan otomatisasi build. Android Studio juga dilengkapi dengan emulator Android yang dapat digunakan untuk menguji aplikasi pada berbagai perangkat virtual dan konfigurasi. Selain itu, integrasi yang mendalam dengan berbagai layanan Google seperti Firebase memungkinkan pengembang menambahkan fitur backend dengan mudah, seperti autentikasi pengguna, penyimpanan data real-time, dan analitik.

* + 1. **Firebase**

Firebase adalah platform pengembangan aplikasi yang menyediakan berbagai layanan untuk membantu pengembang membangun, meningkatkan, dan mengelola aplikasi dengan lebih mudah. Firebase dapat menyimpan data dalam format JavaScript Object Notation (JSON)yang tidak menggunakan query untuk memasukan, memperbaharui, menghapus, atau menambah data ke dalamnya (Evandi et al., 2020).

#### *Realtime Database*

Realtime Database Adalah salah satu fasilitas yang menyimpan data ke databasedan mengambil data darinya dengan sangat cepat tetapi firebase bukan hanya realtime database, jauh lebih dari itu (Payara dan Tanone, 2018). Dirancang untuk menyimpan dan menyinkronkan data antara pengguna secara real-time, layanan ini memungkinkan setiap perubahan yang dilakukan pada data di salah satu klien langsung diperbarui di semua klien lain yang terhubung tanpa penundaan. Fitur ini sangat berguna untuk aplikasi yang memerlukan pembaruan data langsung, seperti aplikasi obrolan, kolaborasi dokumen, atau pelacakan lokasi secara real-time. Selain itu, Firebase Realtime Database memiliki kemampuan offline, di mana SDK-nya menyimpan data di perangkat pengguna sehingga tetap dapat diakses meskipun perangkat offline. Ketika perangkat kembali online, perubahan data yang dilakukan saat offline akan disinkronkan secara otomatis dengan basis data utama.

* + - 1. **Firebase Authentication**

Firebase Authentication, atau Firebase Auth, adalah layanan yang disediakan oleh platform Firebase untuk mengelola autentikasi pengguna dalam aplikasi. Layanan ini menyediakan berbagai metode autentikasi, termasuk email dan kata sandi, nomor telepon, serta autentikasi berbasis penyedia pihak ketiga seperti Google, Facebook, Twitter, dan GitHub. Ini juga mencakup sistem manajemen pengguna di mana pengembang dapat mengaktifkan otentikasi pengguna dengan email dan login kata sandi yang disimpan dengan Firebase (Khawas, 2018). Firebase Auth membantu pengembang mengelola autentikasi pengguna dengan aman dan efisien, memastikan aplikasi mereka aman dan memberikan pengalaman pengguna yang baik.

* + - 1. **Firebase *Storage***

Firebase Storage adalah layanan penyimpanan file berbasis cloud yang disediakan oleh platform Firebase. Layanan ini dirancang untuk membantu pengembang menyimpan dan mengelola konten buatan pengguna seperti foto, video, dan file lainnya dengan mudah dan aman. Ini memfasilitasi transfer file yang mudah dan aman terlepas dari kualitas jaringan untuk aplikasi Firebase (Khawas, 2018). Firebase Storage juga menyediakan SDK untuk berbagai platform termasuk Android, iOS, dan web, memudahkan pengembang untuk mengunggah dan mengunduh file langsung dari aplikasi mereka. Dengan kemampuan untuk menangani unggahan dan unduhan file besar secara efisien, Firebase Storage mendukung fitur seperti pengunggahan latar belakang, melanjutkan unggahan yang terhenti, dan monitoring progres unggahan.

## Teknologi yang Telah Dimanfaatkan

CV Wijaya Lesmana Sejahtera menggunakan Microsoft Excel yang berada pada komputer kantor untuk menyimpan data barang masuk, barang keluar dan data klien. Dimana proses penginputan data-data tersebut dilakukan oleh admin saja. Microsoft Excel adalah sebuah aplikasi yang digunakan oleh bisnis dalam kebutuhan mereka akan grafik, grafik, perhitungan data statistik, dan pembuatan formula dan diluncurkan oleh Microsoft pada tahun 1985 (Hermawati dan Primasetya, 2021).

## Analisis Permasalahan

CV Wijaya Lesmana Sejahtera memiliki kendala dalam pengelolaan pergudangan. Salah satu permasalahan utama yang dihadapi adalah kecenderungan lupa dalam melakukan input atau pencatatan data barang terkait dengan barang-barang yang masuk dan keluar dari pergudangan, para pegawai seringkali lupa melapor ke Admin bahwa mereka mengeluarkan barang. Ketidakakuratan informasi terkait inventaris dapat mengganggu efisiensi operasional dan manajemen stok barang secara efektif di perusahaan. Sehingga dapat menghambat pekerjaan yang ditangani oleh CV Wijaya Lesmana.

## Analisis Kebutuhan

CV Wijaya Lesmana Sejahtera menghadapi kendala dalam pengelolaan pergudangan, terutama dalam hal pencatatan dan input data barang yang masuk dan keluar karena Seringkali para pegawai lupa melapor ke Admin bahwa mereka memakai barang, sehingga menyebabkan Ketidakakuratan informasi inventaris mengganggu efisiensi operasional dan manajemen stok barang. Untuk mengatasi masalah ini, disarankan pengembangan aplikasi yang akan digunakan oleh pegawai dan admin sehingga ketika karyawan menggunakan atau mengambil barang untuk keperluan projek mereka bisa menginputnya sendiri barang apa saja yang mereka butuhkan. aplikasi ini memiliki dua role, yaitu Karyawan dan Admin.

Karyawan memiliki kemampuan terbatas karena untuk menjaga keamanan dan integritas data. pada role karyawan ini terdapat menu input barang masuk dimana barang tersebut telah terdaftar dimiliki pada stok barang, dan terdapat juga menu input barang keluar dimana menu ini berfungsi untuk menginput barang apa saja yang telah digunakan atau dikeluarkan dari gudang, ini butuh pengajuan barang keluar yang membutuhkan persetujuan dari Admin sebelum dikirim ke klien.

Pada role Admin, Admin memiliki kontrol penuh, mencakup manajemen pengguna, manajemen stok barang, persetujuan pengajuan barang keluar, riwayat persetujuan, manajemen klein, dan laporan barang keluar dari gudang dan diantar ke klien. Fitur untuk Admin meliputi manajemen pengguna (tambah user, pilih role, aktif/nonaktifkan user), manajemen stok barang (tambah barang baru, nonaktifkan barang habis/tidak diproduksi lagi), pengajuan barang keluar (ACC/tolak pengajuan), riwayat persetujuan, manajemen klien (input/edit data klien), dan laporan (lihat riwayat barang keluar).

Alur kerja aplikasi dimulai dengan Karyawan yang menginput data barang masuk dan mengajukan permintaan barang keluar. Admin yang nantinya Admin akan meninjau, dan memutuskan untuk menyetujui atau menolak pengajuan tersebut. Jika disetujui, barang keluar dapat digunakan atau dikirim ke klien. Dengan aplikasi ini, CV Wijaya Lesmana Sejahtera dapat meningkatkan akurasi data inventaris, sehingga efisiensi operasional dan manajemen stok barang menjadi lebih efektif.

## Deskripsi Pekerjaan

Pelaksanaan kerja praktik dilakukan selama 10 minggu, dimulai dari tanggal 8 Maret 2024 sampai 17 Mei 2024 dengan total durasi 280 jam. Jadwal pelaksanaan kegiatan kerja praktik adalah setiap hari Senin sampai Sabtu (Tabel 2.1).

**Tabel 2.1 Jadwal Kerja Praktik**

|  |  |  |
| --- | --- | --- |
| **Hari Pelaksanaan**  **Kerja Praktik** | **Fernando**  **(2125250014)** | **Dep’Niel Sinaga**  **(2125250058)** |
| Senin | 13:00 – 15:00 | 13:00 – 15:00 |
| Selasa | 16:00 – 19:00 | 16:00 – 19:00 |
| Rabu | 09:00 – 12:00 | 09:00 – 12:00 |
| Kamis | 13:00 – 16:00 | 13:00 – 16:00 |
| Jumat | 16:00 – 19:00 | 16:00 – 19:00 |
| Sabtu | 13:00 – 15:00 | 13:00 – 15:00 |

Pada minggu pertama pelaksanaan kerja praktik di CV Wijaya Lesmana tanggal 26 Februari 2024 sampai 2 Maret 2024. Di hari pertama dilakukan wawancara mengenai permasalahan dari tempat kerja praktik, serta dilakukan pengenalan lingkungan kerja praktik oleh penyelia, Bapak Indra Wijaya selaku Direktur CV Wijaya Lesmana. Dilanjuti dengan penyusunan KAKP dan analisis kebutuhan aplikasi yang akan dibuat, dan membuat desain awal menggunakan figma. Pada minggu ini semua kegiatan berjalan sesuai dengan KAKP.

Minggu kedua kerja praktik dilaksanakan dari tanggal 4 Maret 2024 sampai 9 Maret 2024. Kegiatan yang dilakukan adalah melanjutkan pembuatan desain awal menggunakan figma serta melanjutkan penyusunan KAKP dan melakukan revisi KAKP pada bimbingan kedua serta menyiapkan Android Studio sebagai IDE untuk pembuatan aplikasi. Selain itu, dilakukan juga persiapan DBMS, yaitu Firebase dengan menyiapkan *database* dan integrasi dengan aplikasi. Pada minggu ini semua kegiatan berjalan sesuai dengan KAKP.

Minggu ketiga kerja praktik dilaksanakan dari tanggal 11 Maret 2024 sampai 16 Maret 2024. Kegiatan yang dilakukan adalah menyelesaikan KAKP dan sistem *login, register,* dan lupa password. pada minggu ketiga ini juga dilakukan pembuatan sistem input klien dan serta melakukan desain pada tampilan aplikasi.Pembuatan sistem juga dihubungkan dengan *realtime database* dari Firebase.Pada minggu ini semua kegiatan berjalan lebih cepat dari KAKP.

Minggu keempat kerja praktik dilaksanakan dari tanggal 18 Maret 2024 sampai 23 Maret 2024. Kegiatan yang dilakukan adalah memperbaiki error code dan menyelesaikan input klien serta membuat fitur edit, hapus klien, membuat fitur input stock barang dan melakukan desain pada tampilan aplikasi.Pada minggu ini beberapa kegiatan berjalan lebih cepat dari KAKP.

Minggu kelima kerja praktik dilaksanakan dari tanggal 25 Maret 2024 sampai 30 Maret 2024. Kegiatan yang dilakukan pada minggu ini memperbaiki error. membuat dialog tampil stock, status barang berupa enable atau disable, menambahkan search bar pada stock barang dan client, membuat fitur input barang masuk, barang keluar dan serta mendesain tampilan aplikasi. Pada minggu ini beberapa kegiatan berjalan lebih cepat dari KAKP.

Minggu keenam kerja praktik dilaksanakan dari tanggal 1 April 2024 sampai 6 April 2024*.* Kegiatan yang dilakukan adalah memperbaiki error, membuat pengajuan barang masuk dan barang keluar untuk role karyawan, dan mendesain tampilan aplikasi. Pada minggu beberapa kegiatan tidak sesuai dengan KAKP dikarenakan penambahan fitur dan kebutuhan aplikasi.

Minggu ketujuh kerja praktik dilaksanakan dari tanggal 15 April 2024 sampai 20 April 2024. Kegiatan yang dilakukan adalah membuat laporan harian, mingguan dan bulanan pada laporan barang masuk dan laporan barang keluar, dan memperbaiki error pada laporan barang masuk dan laporan barang keluar Pada minggu ini, beberapa kegiatan tidak sesuai KAKP dikarenakan terdapat banyaknya error dan penambahan fitur kebutuhan aplikasi.

Minggu kedelapan kerja praktik dilaksanakan dari tanggal 22 April 2024 sampai 27 April 2024. Kegiatan yang dilakukan adalah melanjutkan pembuatan laporan barang keluar, memperbaiki error, menambahkan role karyawan dan admin, memperbarui halaman register fitur kirim email untuk mengganti password ketika melakukan aktivitas lupa password. Pembuatan laporan akhir kerja praktik juga dilanjutkan. Pada minggu, kegiatan tidak sesuai dengan KAKP dikarenakan penambahan fitur dan kebutuhan aplikasi.

Minggu kesembilan kerja praktik dilaksanakan dari tanggal 29 April 2024 sampai 4 Mei 2024. Kegiatan yang dilakukan adalah bertemu dengan penyelia untuk menyampaikan progres, menambahkan fitur kelola user, mendesain tampilan Pada minggu, beberapa kegiatan tidak berjalan sesuai dengan KAKP dikarenakan penambahan fitur dan kebutuhan aplikasi.

Minggu kesepuluh kerja praktik dilaksanakan dari tanggal 6 Mei 2024 sampai 11 Mei 2024. Kegiatan yang dilakukan adalah penyerahan aplikasi ke pihak penyelia, melakukan survei aplikasi dan penyelesaian laporan akhir kerja praktik. Pada minggu ini semua kegiatan berjalan sesuai dengan KAKP.

## Kesesuaian Ilmu Teoritis dan Praktis

Selama melaksanakan kerja praktik di CV Wijaya Lesmana Sejahtera, penulis mendapatkan kesesuaian ilmu teoritis dan praktis yaitu:

1. Telah diajarkan langkah – langkah pembuatan laporan pada mata kuliah Bahasa Indonesia.
2. Dasar – dasar dari bahasa pemrograman Java telah diajarkan di lingkungan akademik, sehingga mempermudah dalam pembuatan Aplikasi Presensi Karyawan pada CV Wijaya Lesmana Sejahtera.
3. Pada mata kuliah Pemrograman Aplikasi Bergerak 1 dan 2 yang diajarkan di lingkungan akademik, yaitu dengan menggunakan DBMS dan API dapat mempermudah penulis dalam pembuatan database aplikasi dari CV Wijaya Lesmana Sejahtera.
4. Dalam menghubungkan relasi antar tabel pada database aplikasi ini, diterapkan teori akademis dari mata kuliah Pemrograman Aplikasi Bergerak 1 dan 2 yang telah diajarkan.

# HASIL DAN PEMBAHASAN

Pada bab ini akan dibahas mengenai rancangan sistem berupa *use case diagram* dan tampilan program, serta hasil kuesioner dari karyawan terhadap aplikasi yang telah dikembangkan.

## Perancangan Sistem

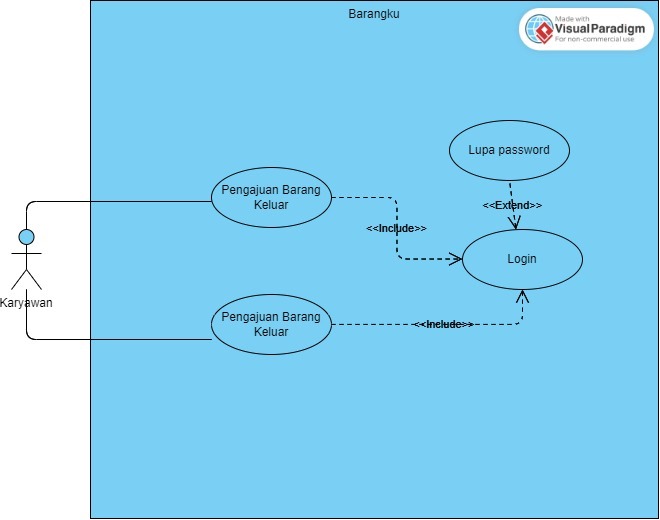
Perancangan sistem untuk CV Wijaya Lesmana Sejahtera akan dijelaskan mengenai sistem informasi yang dikembangkan dalam ruang lingkup CV Wijaya Lesmana Sejahtera

### Use Case Diagram

Interaksi antara pengguna dan sistem digambarkan dalam bentuk *use case diagram* dengan memiliki 2 sisi, yaitu admin dan karyawan. Interaksi dari Sisi Karyawan dengan sistem memiliki 2 menu utama, yaitu:

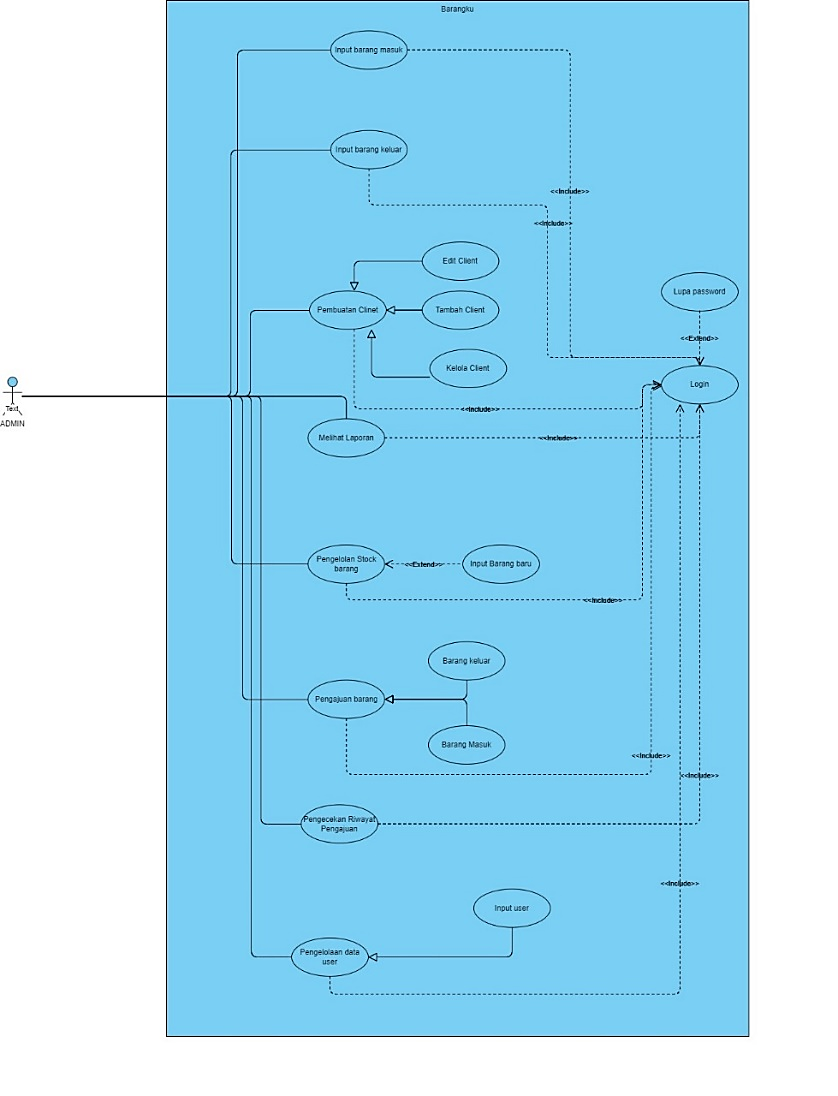
1. Barang Masuk yang memiliki tanggal dan data barang masuk.
2. Barang Keluar yang memiliki tanggal dan data barang keluar.

Interaksi karyawan dengan sistem yang dikembangkan dalam bentuk *use case diagram* dapat dilihat pada gambar 3.1.



**Gambar 3.1 Use Case Diagram Karyawan**

Duis aute irure dolor Lorem ipsum dolor sit amet, consecte. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor Lorem ipsum dolor sit amet, consecte. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor Lorem ipsum dolor sit amet, consecte.



**Gambar 3.2 Use Case Diagram Admin**

**Tabel 3.1 Waktu dan Ruangan Perkuliahan**

|  |  |
| --- | --- |
| **Ruangan Pengajaran** | **Waktu Perkuliahan** |
| Ruangan Praktikum:   * A200-A207 | Periode Waktu:   * 07:50 – 09:30 * 09:40 – 11:20 * 11:30 – 13:10 * 13:30 – 15:10 |
| Ruangan Teori:   * A301-A313 * A401-A412 * A414 |

### Tampilan Aplikasi

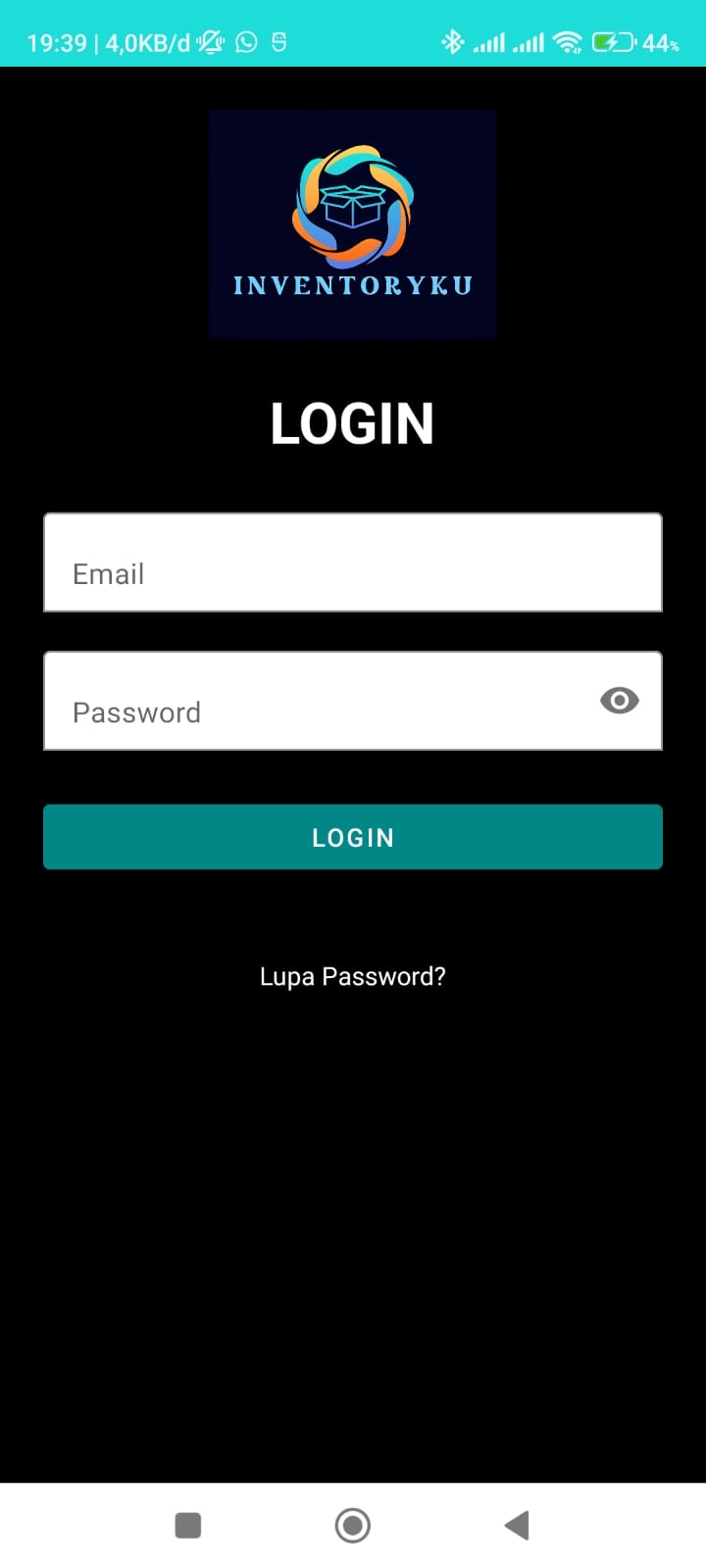
Aplikasi *Inventoryku* untuk CV Wijaya Lesmana memiliki dua tampilan utama, yaitu karyawan dan admin. Dari sisi karyawan, aplikasi menyediakan fitur untuk melakukan pengajuan barang masuk dan pengajuan barang keluar. Pada tampilan admin memiliki tampilan yang berbeda dengan karyawan dengan tambahan beberapa menu seperti pengajuan, riwayat pengajuan, klien, laporan, stock barang, dan admin.

pada bagian Splash Screen Activity muncul saat aplikasi dibuka. Splash Screen memiliki logo aplikasi yang dapat dilihat pada Gambar 3.3.

****

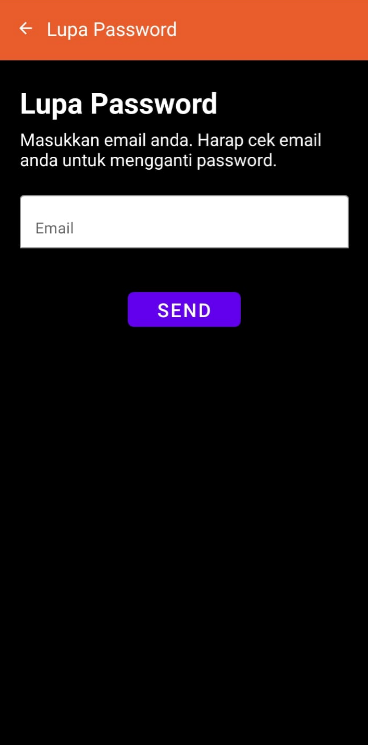
**Gambar 3.3 Tampilan Splash Screen**

Kemudian pada tampilan Login memiliki dua input yaitu input email dan password serta tombol Login untuk mengakses aplikasi setelah mengisi email dan password dengan benar maka akan mengakses menu yang ada di aplikasi sesuai dengan rolenya masing masing, serta terdapat Lupa password jika user lupa password untuk login aplikasi. Tampilan login dapat dilihat pada Gambar 3.4.

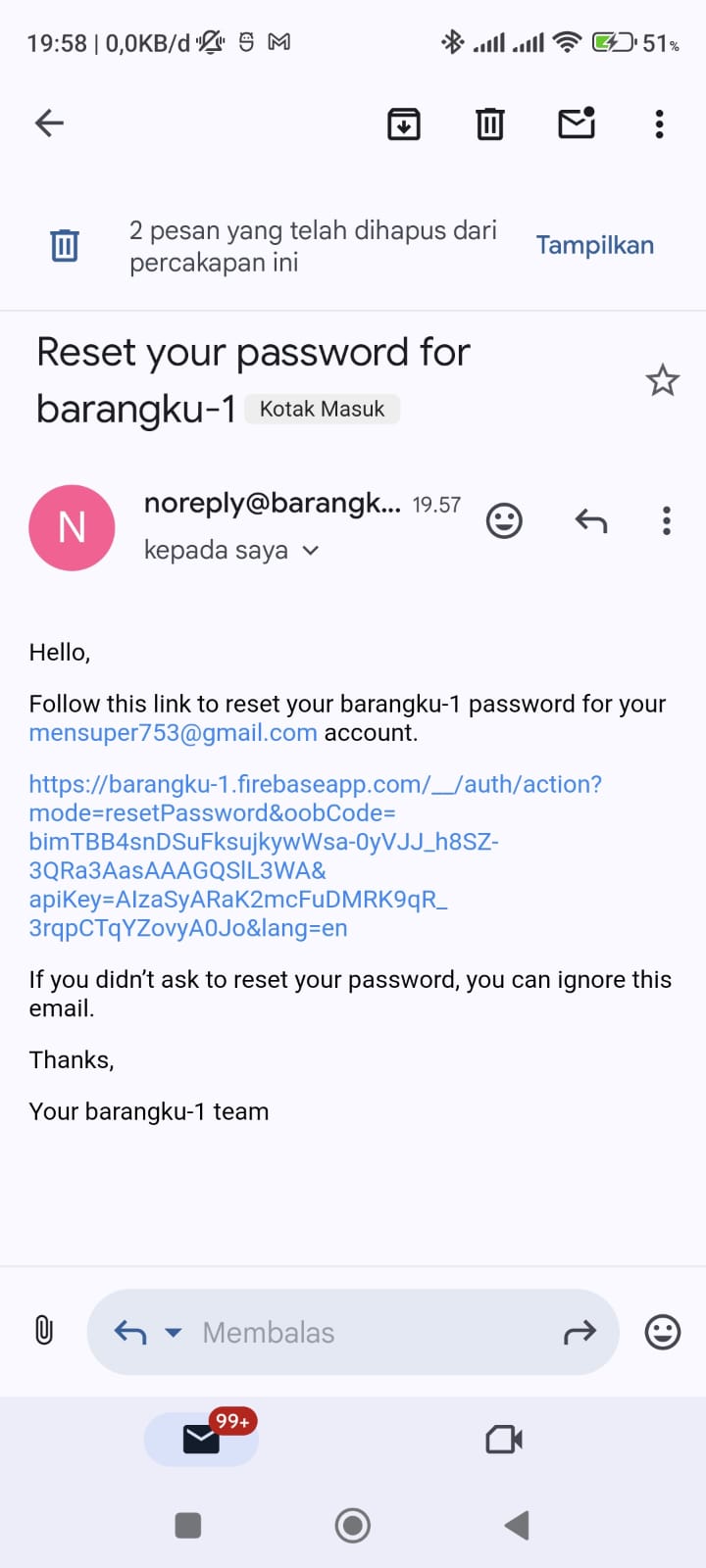
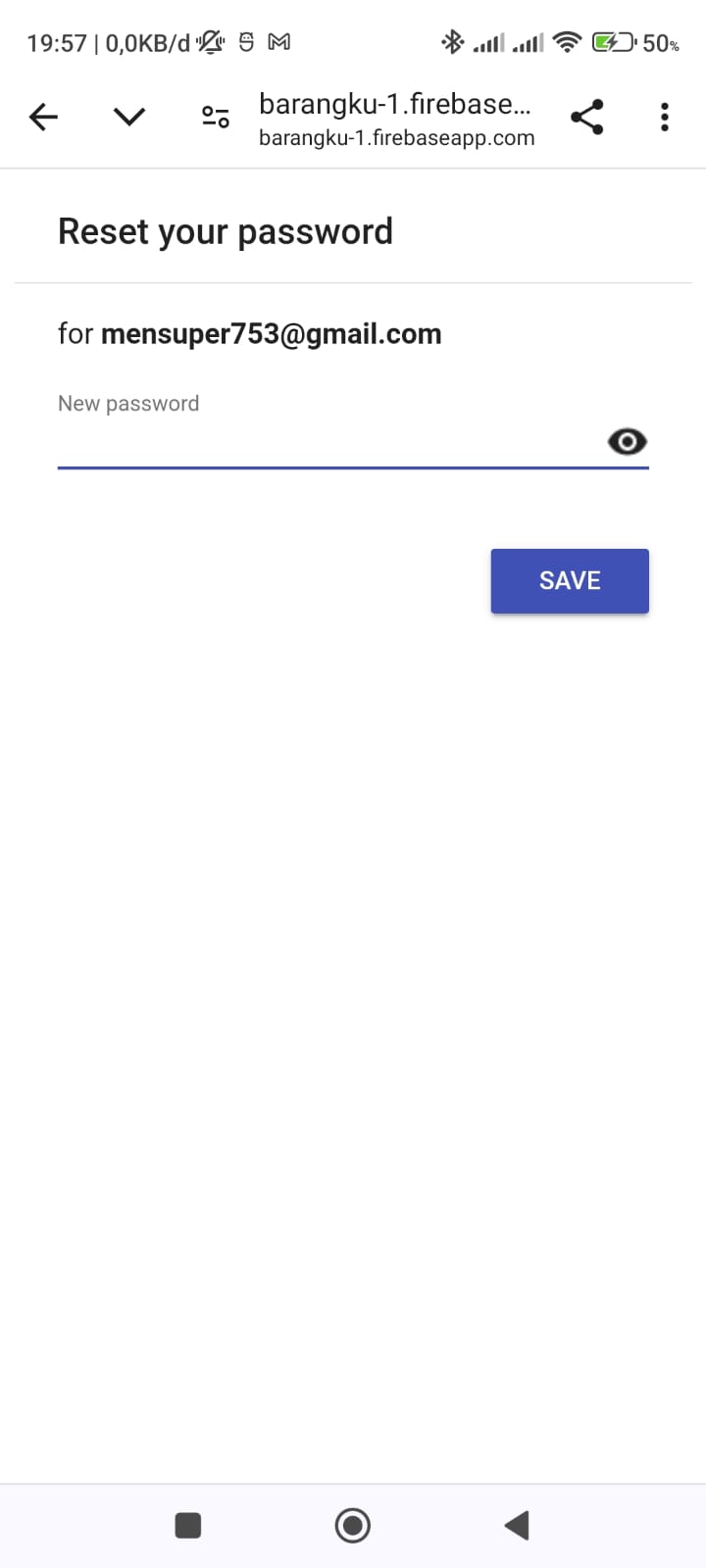


**Gambar 3.4 Tampilan Halaman Login**

Kemudian pada bagian Lupa Password terdapat input email yang telah terdaftar ini berfungsi untuk mereset password, dan terdapat tombol Send yang berfungsi untuk mengirimkan email yang berisikan link yang dapat digunakan untuk mereset password dari email yang telah terdaftar pada aplikasi ini. Tampilan Lupa Password dapat dilihat seperti gambar 3.5.



**Gambar 3.5 Tampilan Lupa Password**

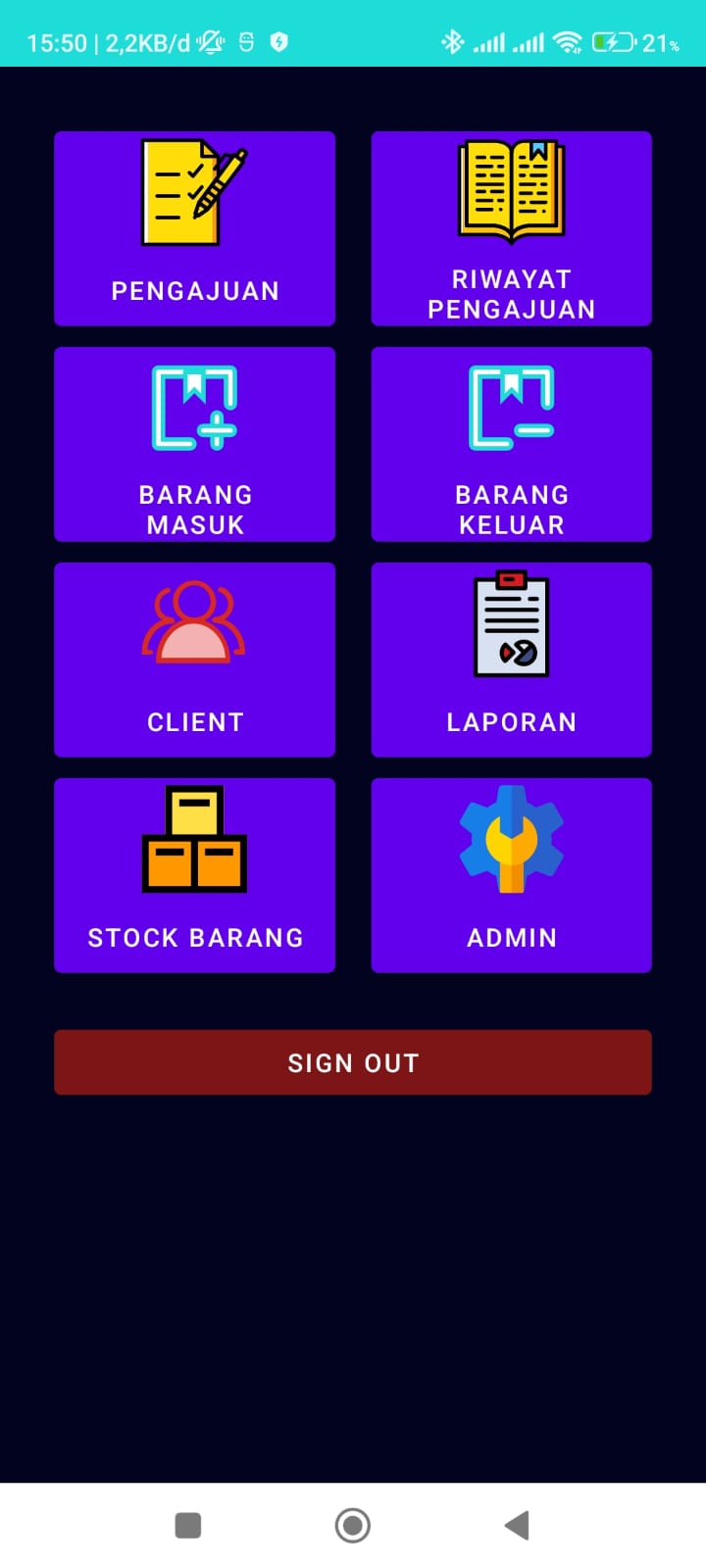
** **

**Gambar 3.6 Tampilan Email dari Lupa Password**

Tampilan Main Activity memiliki 2 tampilan yaitu tampilan dari sisi karyawan dan dari sisi admin dimana pada tampilan dari sisi karyawan hanya terdapat 2 menu saja dan tombol sign out, dan pada tampilan admin memiliki 8 menu dan 1 tombol sign out. Tampilan untuk Main Activity karyawan dan admin dapat dilihat pada gambar 3.7 dan gambar 3.8.



**Gambar 3.7 Tampilan Main Activity Karyawan**

****

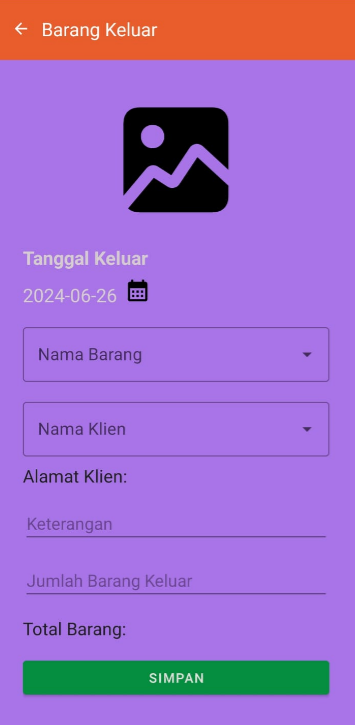
**Gambar 3.8 Tampilan Main Activity Admin**

Pada tampilan Barang Masuk terdapat input nama barang, keterangan dan jumlah barang. pada tampilan ini dapat melakukan input nama barang dapat melakukan input barang yang masuk ke gudang dengan melakukan menginput nama barang yang masuk ke gudang, keterangan barang tersebut dan jumlah barang tersebut, dimana tanggal yang ada pada tampilan ini adalah tanggal melakukan input barang masuk mengikuti tanggal terkini dan tidak dapat diubah, ketika pemilihan barang dan pemilihan client maka pada saat memilih salah satu barang maupun klien maka akan menampilkan jumlah barang yang berada di stock barang dan klien akan menampilkan alamat yang sesuai dengan nama client tersebut. Dan terdapat tombol Simpan dimana ketika menekan tombol Simpan inputan barang masuk ini akan mengirimkan pengajuan barang tersebut ke admin dimana nantinya admin akan melakukan validasi untuk memastikan benar atau tidaknya barang tersebut dipesan. sedangkan bila di input langsung oleh admin maka tidak membutuhkan pengajuan seperti yang seperti karyawan ketika admin melakukan input barang masuk dan menekan tombol Simpan maka stok barang tersebut akan bertambah. Tampilan barang masuk dapat dilihat pada Gambar 3.9.

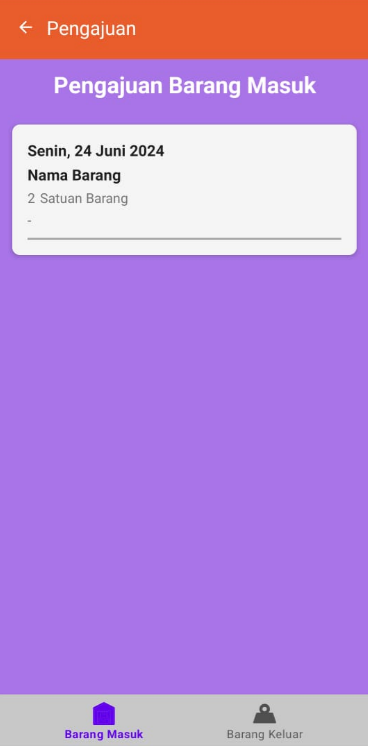
**Gambar 3.9 Tampilan Barang Masuk**

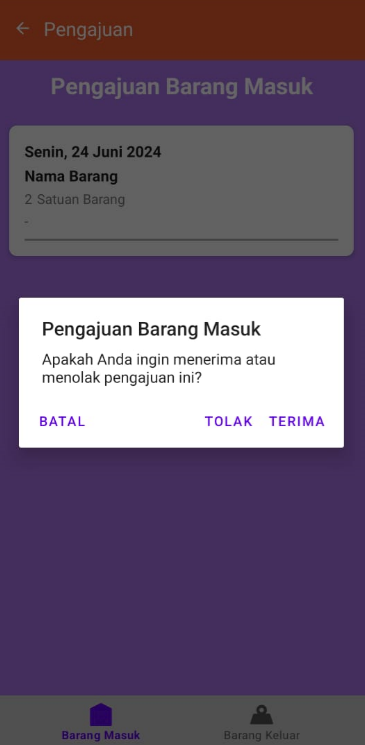
Tampilan barang keluar terdapat input nama barang, nama klien, keterangan dan jumlah barang. pada tampilan ini dapat melakukan input barang yang akan keluar dari gudang, dengan menginput nama barang yang keluar dari gudang, nama klien sebagai tempat barang tersebut dikirim, alamat klien yang akan terisi otomatis sesuai dengan data klien, keterangan yang harus diisi guna untuk memberikan keterangan barang tersebut digunakan untuk apa, dan menampilkan jumlah barang yang tersedia dan satuan barang tersebut, dan terdapat tombol Simpan dimana ketika menekan tombol Simpan inputan barang keluar ini akan mengirimkan pengajuan barang tersebut ke admin dimana nantinya admin akan memberikan izin mengeluarkan barang tersebut. sedangkan bila di input langsung oleh admin maka tidak membutuhkan pengajuan seperti yang seperti karyawan ketika admin melakukan input barang masuk dan menekan tombol Simpan maka stok barang tersebut akan bertambah. Tampilan barang keluar dapat dilihat pada gambar 3.10.

**Gambar 3.10 Tampilan Barang Keluar**

Pada tampilan pengajuan terdapat 2 navbar di bawah untuk menampilkan pengajuan barang masuk dan barang keluar. Recyclerview yang menampilkan tanggal pengajuan, nama barang yang diajukan , jumlah barang yang diajukan, satuan barang, dan keterangan. saat recyclerview tersebut ditekan maka akan keluar toast untuk menerima atau menolak pengajuan barang tersebut saat pengajuan barang tersebut diterima maka stok barang yang ada akan berubah mengikuti transaksi yang dilakukan sedangkan bila ditolak maka stok barang tersebut tidak akan berubah. Tampilan Pengajuan dapat dilihat pada gambar 3.11.

** **

** **

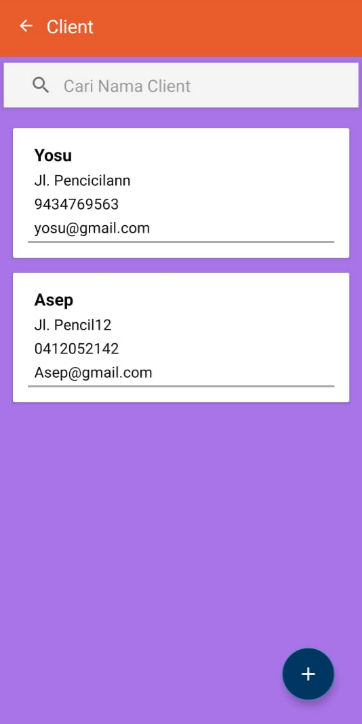
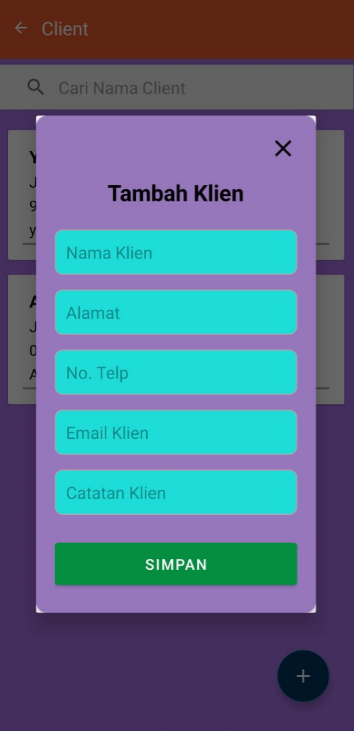
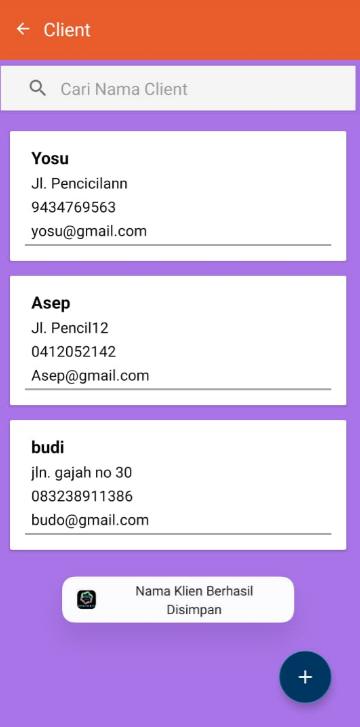
**Gambar 3.11 Tampilan Pengajuan**

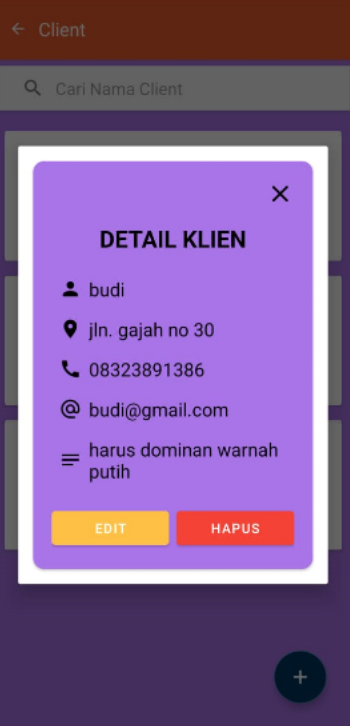
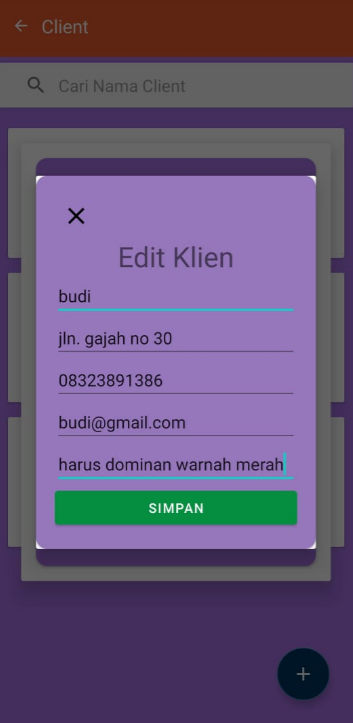
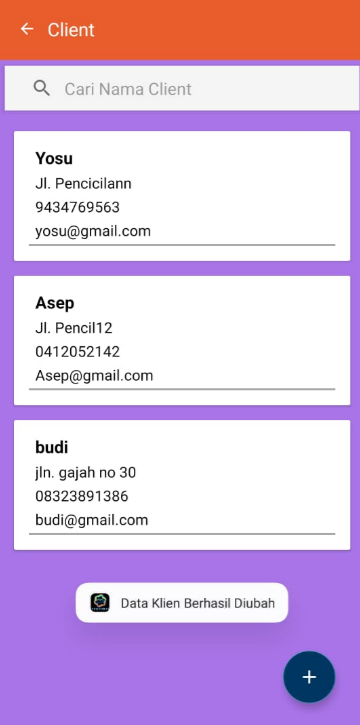
Pada tampilan Riwayat pengajuanterdapat recyclerview yang menampilkan tanggal pengajuan barang masuk maupun barang keluar nama barang, jumlah barang, satuan barang, keterangan dan, status pengajuan barang diterima atau ditolak pada bagian riwayat pengajuan terdapat 2 navbar di bagian bawah yang berfungsi untuk menunjukan riwayat pengajuan barang masuk dan riwayat pengajuan barang keluar

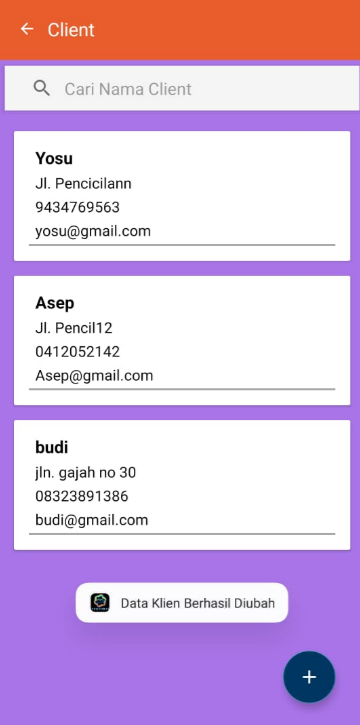
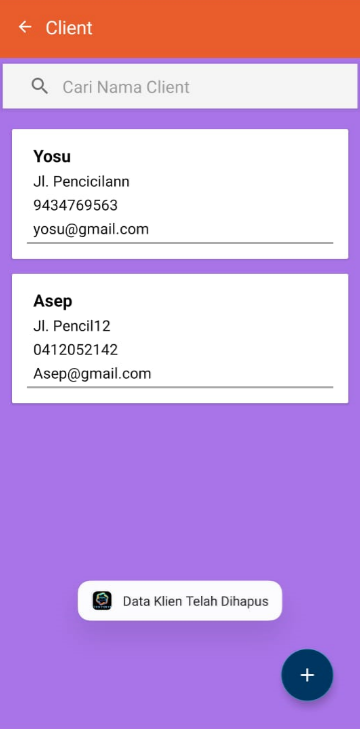
** **

**Gambar 3.12 Tampilan Riwayat Pengajuan**

Pada tampilan Client terdapat recyclerview, search bar dan floating action button. pada recyclerview menampilkan nama klien, alamat klien, nomor telepon klien. Ketika recyclerview ditekan akan menampilkan dialog yang menampilkan nama klien, alamat klien, nomor telepon klien, email klien dan catatan khusus dari klien dan terdapat 2 tombol yaitu tombol edit dan hapus. ketika tombol edit di tekan maka akan keluar dialog lain yang berisikan input nama klien, alamat klien, nomor telepon klien, email klien dan catatan khusus dan terdapat tombol simpan pada dialog tersebut yang berfungsi untuk menyimpan perubahan data sehingga menungkinkan untuk dapat melakukan perubahan data sewaktu-waktu bila terjadi kesalahan informasi pada data klien. dan fungsi tombol hapus adalah untuk menghapus data klien jika dirasa data tersebut tidak perlu digunakan lagi atau ketika kerjasama antar klien tersebut telah selesai. pada tampilan awal menu terdapat search bar yang berfungsi untuk mencari klien berdasarkan nama klien tersebut. Floating action button pada tampilan awal berfungsi untuk menampilkan dialog yang berguna untuk menambahkan data klien baru dengan memasukan beberapa input seperti nama klien, alamat klien, nomor telepon klien, email klien dan catatan khusus, dan terdapat juga tombol Simpan pada dialog ini berguna untuk menyimpan data klien baru. pada tampilan ini akan memberikan toast untuk kegiatan tambah klien, edit klien dan hapus klien bila berhasil. Tampilan Client dapat dilihat pada gambar 3.11.

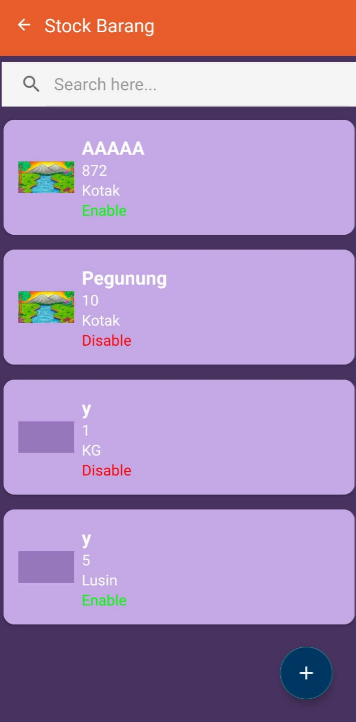
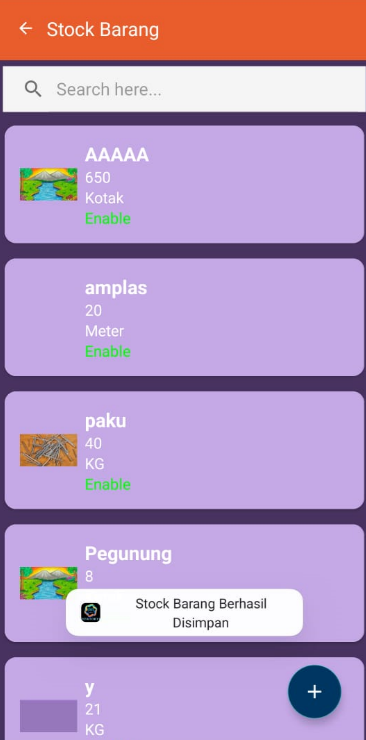
  

**Gambar 3.13 Tampilan Client**

Padat Tampilan Stock Barang terdapat Search bar yang berfungsi untuk mencari barang dengan menggunakan nama barang. Floating action button yang berfungsi untuk menambahkan barang baru. Recyclerview yang berfungsi menampilkan barang-barang yang ada.ketika recyclerview ditekan maka akan akan keluar dialog yang menampilkan foto barang, nama barang, jumlah barang dan satuan barang serta terdapat tombol enable dan disable yang digunakan sebagai status dari barang tersebut. Tombol Enable berfungsi untuk memberitahu bahwa barang tersebut masih tersedia atau bisa dipakai sedangkan Tombol Disable berfungsi untuk memberi tahu bahwa barang tersebut tidak bisa digunakan keduanya ini berpengaruh pada proses barang masuk dan keluar. barang yang bisa gunakan hanya barang yang memiliki status enable dan pada yang memiliki status disable barang tersebut tidak bisa dilakukan transaksi pada barang masuk dan barang keluar. Tampilan Stock Barang dapat dilihat pada gambar 3.14.

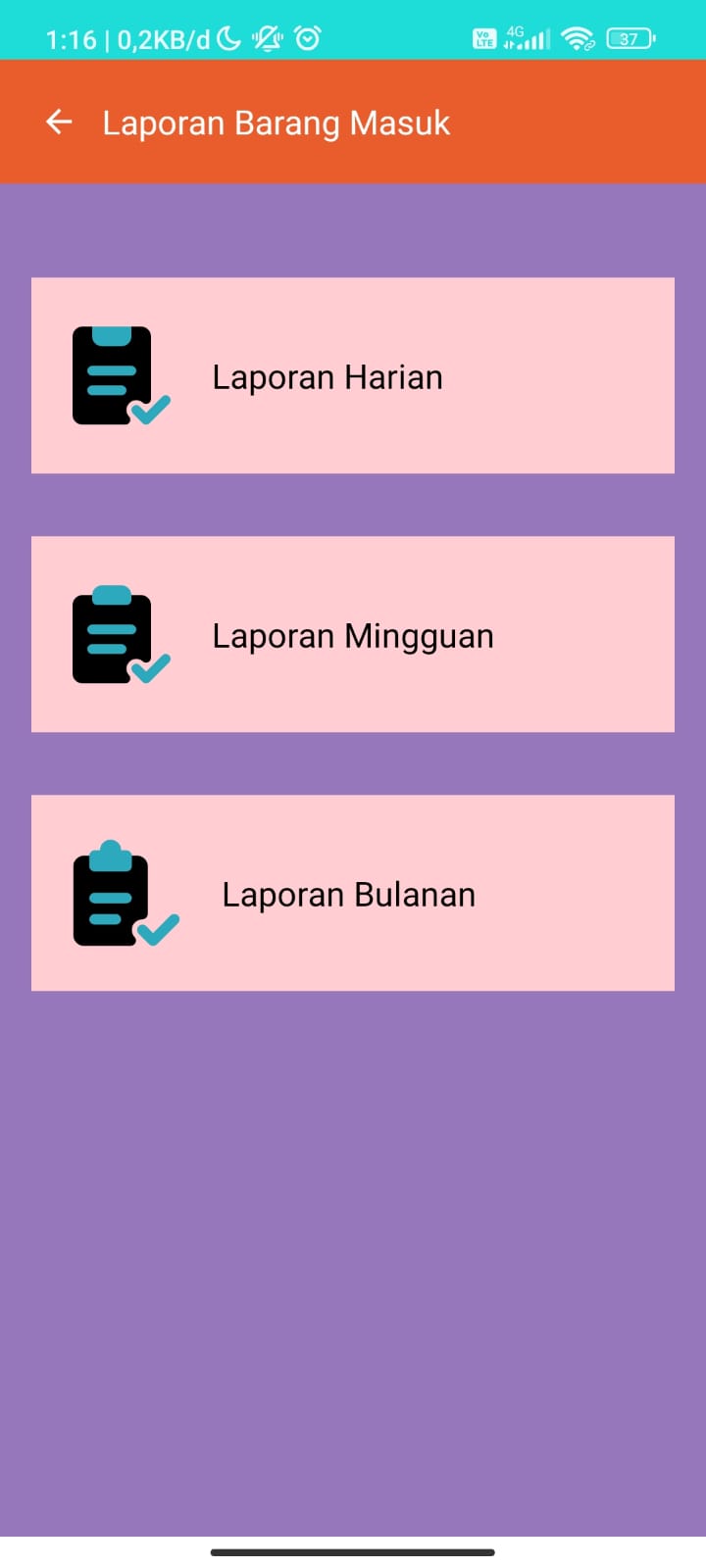
t

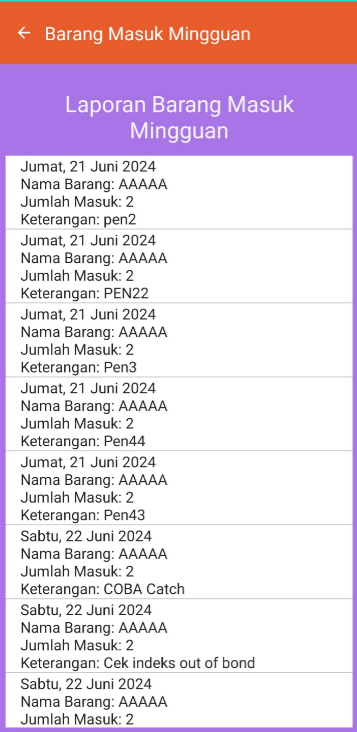
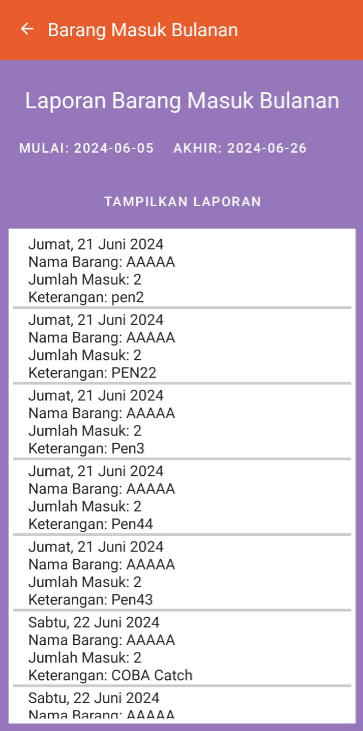
**Gambar 3.14 Tampilan Stock Barang**

Pada Tampilan Laporan terdapat 2 tombol yaitu tombol Laporan Barang Masuk dan Laporan barang keluar. Tombol Laporan Barang Masuk berfungsi untuk menampilkan laporan barang masuk yang terdapat laporan barang masuk harian, laporan barang masuk mingguan dan laporan barang masuk bulanan. laporan barang masuk harian akan menampilkan laporan barang-barang yang masuk pada hari itu, pada laporan barang masuk mingguan akan menampilkan barang-barang yang masuk pada minggu tersebut dan pada laporan bulanan dapat memilih rentang waktu untuk menampilkan laporan tersebut dan ketika menekan Tampilkan Laporan barulah laporan tersebut akn muncul sesuai dengan rentang waktu yang dipilih. Tombol Laporan Barang Keluar berfungsi untuk menampilkan laporan barang masuk yang terdapat laporan barang keluar harian, laporan barang keluar mingguan dan laporan barang keluar bulanan. laporan barang keluar harian akan menampilkan laporan barang-barang yang keluar pada hari itu, pada laporan barang keluar mingguan akan menampilkan barang-barang yang keluar pada minggu tersebut dan pada laporan bulanan dapat memilih rentang waktu untuk menampilkan laporan tersebut dan ketika menekan Tampilkan Laporan barulah laporan tersebut akn muncul sesuai dengan rentang waktu yang dipilih. Tampilan Laporan, Laporan Barang Masuk dan Tampilan Barang keluar dapat dilihat pada gambar 3.15, gambar 3.16 dan gambar 3.17.

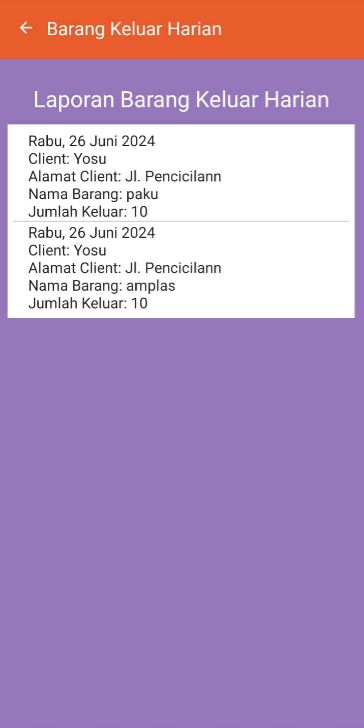


**Gambar 3.15 Tampilan Laporan**

** **

** **

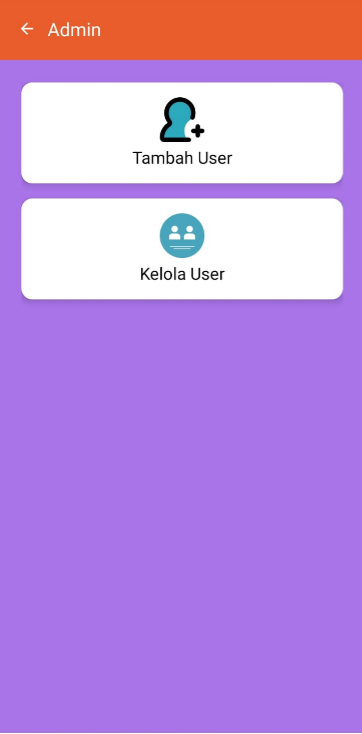
**Gambar 3.16 Tampilan Laporan Barang Masuk**

** **

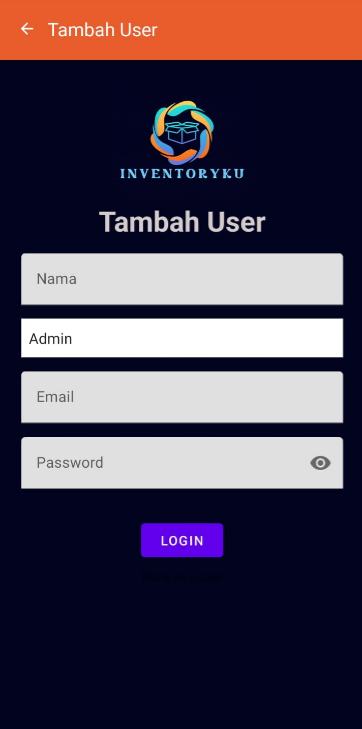
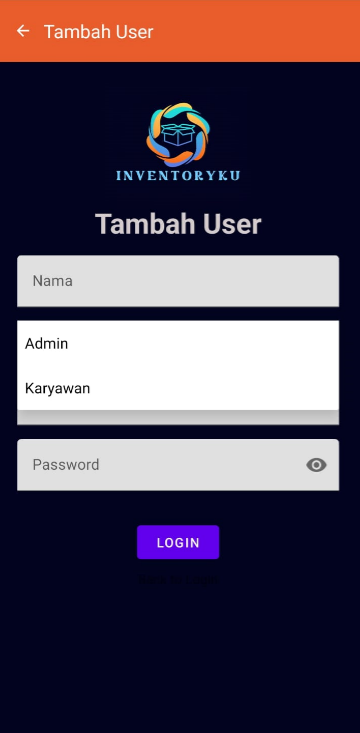
** **

**Gambar 3.16 Tampilan Laporan Barang Keluar**

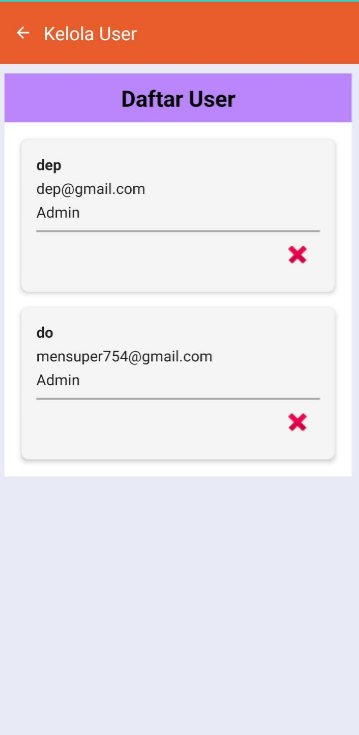
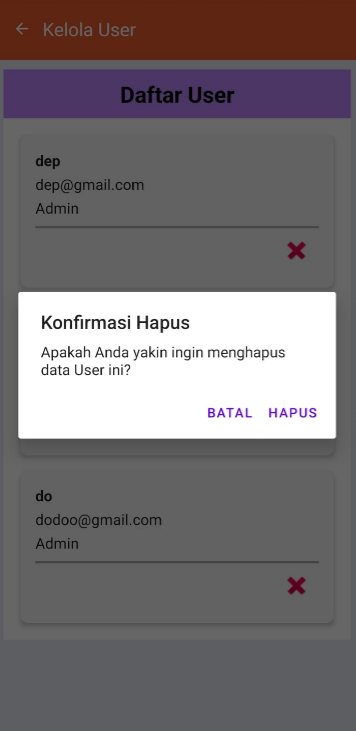
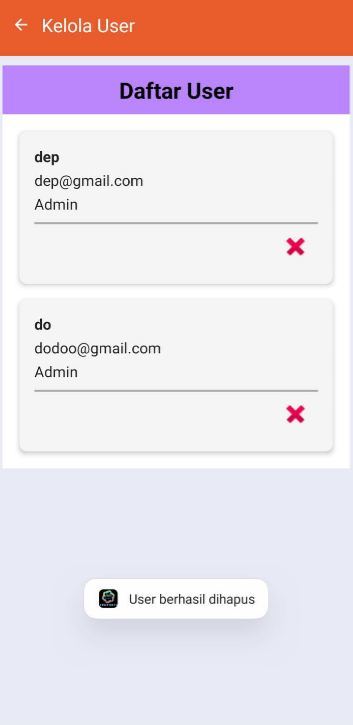
Pada tampilan Admin terdapat 2 menu yaitu menu Tambah user dan Kelola user. Saat Tambah user ditekan maka terdapat tampilan Tambah User yang berfungsi untuk menambahkan user. pada saat register harus mengisikan nama, role yang berupa karyawan atau admin, email dan password. Saat tampilan kelola user pada tampilan admin ditekan maka akan keluar daftar user dan tanda x yang berwarna merah untuk menghapus user saat user berhasil dihapus maka akan keluar toast yang bertuliskan user berhasil dihapus. Tampilan Admin, tampilan Tambah User, dan Kelola User dapat dilihat pada gambar 3.17, gambar 3.18, gambar 3.19.



**Gambar 3.17 Tampilan Admin**

** **

**Gambar 3.18 Tampilan Tambah User**

**  **

**Gambar 3.19 Tampilan Kelola User**

## Pengujian Program

Hasil pengujian aplikasi *InventoryKu* di CV Wijaya Lesmana Sejahtera akan ditunjukkan dalam bentuk tabel. Pengujian Login Activity meliputi pengisian data, *error* pada pengisian data, dan fungsi tombol yang dapat dilihat pada Tabel 3.1.

**Tabel 3.1 Uji Coba *Form Login***

| **No** | **Kasus** | **Hasil yang Diharapkan** | **Hasil Uji Coba** |
| --- | --- | --- | --- |
| 1. | Pengguna memasukkan *email* dan *password*  yang salah lalu menekan tombol login | Muncul *messagebox* yang memberitahukan bahwa Login Gagal | Berhasil |
| 2. | Pengguna tidak memasukkan email. | Aplikasi menampilkan error pada field email dengan pesan "Silakan Masukan Email". | Berhasil |
| 3. | Pengguna tidak memasukkan password | Aplikasi menampilkan error pada field password dengan pesan "Silakan Masukan Password". | Berhasil |
| 4. | Masukkan *username* dan *password* benar dan menekan tombol *login* | Muncul *messagebox* bahwa *login* berhasil dan masuk ke menu utama beserta *messagebox* status jabatan | Berhasil |
| 5. | Pengguna menekan  tombol Lupa *Password.* | Aplikasi menampilkan halaman Lupa *Password*. | Berhasil |
| 6. | Pengguna menginput  email untuk mengirimkan notifikasi email secara otomatis sesuai dengan email yang diinput. | Aplikasi menampilkan pesan “Reset Password aplikasi telah dikirimkan melalui email anda” | Berhasil |
| 7. | Pengguna sudah login dan membuka aplikasi kembali. | Aplikasi langsung mengarahkan pengguna ke halaman MainActivity tanpa perlu login ulang. | Berhasil |

Pengujian Lupa Password Activity meliputi pengisian email, error pengisian email, dan fungsi tombol yang dapat dilihat pada Tabel 3.2

**Tabel 3.2 Uji Coba Fitur Lupa Password**

| **No** | **Kasus** | **Hasil yang Diharapkan** | **Hasil Uji Coba** |
| --- | --- | --- | --- |
| 1. | Pengguna memasukkan email yang valid. | Aplikasi menampilkan pesan: Reset password sudah terkirim ke email anda | Berhasil |
| 2. | Pengguna memasukkan email yang tidak valid. | Error message "Alamat email tidak valid" muncul pada field email. | Berhasil |
| 3. | Pengguna tidak memasukkan email. | Aplikasi menampilkan Pesan Toast "Masukkan Email Anda" muncul. | Berhasil |
| 4. | Pengguna menekan tombol back (panah kembali). | Progress bar tampil. Sistem mengirimkan email reset password Pesan Toast "Reset password sudah terkirim ke email anda" muncul. | Berhasil |
| 5. | Pengguna memasukkan email yang tidak valid. | Progress bar tampil dan kemudian hilang. Pesan Toast "Error: (pesan error Firebase)". | Berhasil |
| 6. | Pengguna mulai mengetik di field email. | Validasi real-time memeriksa format email.c | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |

Pengujian Stock Barang Activity meliputi pengisian email, error pengisian email, dan fungsi tombol yang dapat dilihat pada Tabel 3.3.

**Tabel 3.3 Uji Coba Fitur Stock Barang**

| **No** | **Kasus** | **Hasil yang Diharapkan** | **Hasil Uji Coba** |
| --- | --- | --- | --- |
| 1. | Pengguna menambahkann barang baru. | Barang baru disimpan ke Firebase dan muncul di daftar stok. | Berhasil |
| 2. | Pengguna mencari barang melalui SearchView. | Daftar stok difilter sesuai query; tampilkan pesan jika tidak ada barang yang cocok. | Berhasil |
| 3. | Pengguna melihat detail barang dengan mengklik item di daftar stok. | Dialog detail barang muncul dengan informasi lengkap. | Berhasil |
| 4. | Pengguna mengubah status barang. | Status barang di Firebase diperbarui sesuai tombol yang ditekan. | Berhasil |
| 5. | Kesalahan saat menyimpan barang ke Firebase. | Menampilkan pesan error yang sesuai dan ProgressDialog ditutup setelah proses selesai. | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |

Pengujian Client Activity meliputi pengisian email, error pengisian email, dan fungsi tombol yang dapat dilihat pada Tabel 3.4.

**Tabel 3.4 Uji Coba Fitur Client**

| **No** | **Kasus** | **Hasil yang Diharapkan** | **Hasil Uji Coba** |
| --- | --- | --- | --- |
| 1. | Pengguna menambahkan klien baru. | Klien baru disimpan ke Firebase dengan ID unik dan ditampilkan di daftar klien. | Berhasil |
| 2. | Pengguna mencari klien melalui SearchView. | Daftar klien difilter sesuai query dan menampilkan pesan jika tidak ada klien yang cocok. | Berhasil |
| 3. | Pengguna mengklik item di daftar klien. | Dialog detail klien muncul dengan informasi lengkap klien. | Berhasil |
| 4. | Pengguna mengedit data klien. | Data klien di Firebase diperbarui dan daftar klien diperbarui dengan data yang telah diubah. | Berhasil |
| 5. | Pengguna menghapus klien dari daftar. | Klien dihapus dari Firebase dan daftar klien diperbarui. | Berhasil |
| 6. | Pengguna mengisi formulir penambahan atau perubahan klien. | Validasi dilakukan pada setiap field dan menampilkan pesan error jika ada field yang tidak valid. | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |

Pengujian Barang Masuk Activity meliputi pengisian email, error pengisian email, dan fungsi tombol yang dapat dilihat pada Tabel 3.5.

**Tabel 3.5 Uji Coba Fitur Barang Masuk**

| **No** | **Kasus** | **Hasil yang Diharapkan** | **Hasil Uji Coba** |
| --- | --- | --- | --- |
| 1. | Pengguna menginput form pengisian barang masuk. | Data barang masuk di simpan ke dalam database | Berhasil |
| 2. | Pengguna mengklik simpan tanpa mengisi nama barang yang masuk. | Error message "Nama barang harus diisi" muncul pada field Nama Barang. | Berhasil |
| 3. | Pengguna mengklik simpan tanpa mengisi jumlah barang yang masuk. | Error message "Jumlah barang harus diisi" muncul pada field jumlah barang masuk. | Berhasil |
| 4. | Pengguna memilih Barang yang tersedia. | Mengambil data dari table Stock dan menampilkan dalam form. | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |

Pengujian Barang Keluar Activity meliputi pengisian email, error pengisian email, dan fungsi tombol yang dapat dilihat pada Tabel 3.6

**Tabel 3.6 Uji Coba Fitur Barang Keluar**

| **No** | **Kasus** | **Hasil yang Diharapkan** | **Hasil Uji Coba** |
| --- | --- | --- | --- |
| 1. | Pengguna menginput form pengisian barang keluar. | Data Barang Keluar disimpan dalam database | Berhasil |
| 2. | Pengguna mengosongkan form barang keluar dan mengklik untuk menyimpan. | Pesan “pastikan semua data terisi dengan benar” | Berhasil |
| 3. | Pengguna memilih barang atau mengetikkan nama barang dan memilih barang. | Mengambil data dari database tabel stock barang dan menampikan data yang dipanggil | Berhasil |
| 4. | Pengguna memilih atau mengetikkan nama klien dan memilih klien. | Mengambil data dari database table client dan menampilkan data yang dipanggil | Berhasil |
| 5. | Pengguna menginputkan jumlah barang keluar melebihi stok barang yang tersedia | Menampilkan pesan “ jumlah barang keluar melebihi stok yang ada” | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |

Pengujian Pengajuan Activity meliputi pengisian email, error pengisian email, dan fungsi tombol yang dapat dilihat pada Tabel 3.7.

**Tabel 3.7 Uji Coba Fitur Pengajuan**

| **No** | **Kasus** | **Hasil yang Diharapkan** | **Hasil Uji Coba** |
| --- | --- | --- | --- |
| 1. | Admin menginputkan data Barang Masuk dan Barang Keluar | Data yang di input admin tidak mengirimkan pengajuan kehalaman pengajuan | Berhasil |
| 2. | Karyawan Menginputkan data barang masuk dan barang keluar | Pengajuan yang berasal dari karyawan berhasil ditampilkan dalam halaman pengajuan. | Berhasil |
| 3. | Admin Menyetujui Pengajuan barang masuk atau barang keluar yang diajukan karyawan | Update Data Stok barangmasuk atau barang keluar | Berhasil |
| 4. | Admin menolak Pengajuan barang masuk atau barang keluar yang diajukan karyawan. | Data Stok tidak di update | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |

Pengujian Riwayat Activity meliputi pengisian email, error pengisian email, dan fungsi tombol yang dapat dilihat pada Tabel 3.8.

**Tabel 3.8 Uji Coba Fitur Riwayat Pengajuan**

| **No** | **Kasus** | **Hasil yang Diharapkan** | **Hasil Uji Coba** |
| --- | --- | --- | --- |
| 1. | Admin Menyetujui Pengajuan Barang Masuk atau Barang Keluar yang diajukan oleh karyawan | Halaman Riwayat menampilkan data pengajuan yang diterima oleh admin dengan status “Diterima”. Dan update stok barang | Berhasil |
| 2. | Admin Menolak Pengajuan Barang Masuk atau Barang Keluar yang diajukan oleh karyawan | Halaman Riwayat menampilkan data pengajuan yang ditolak  eh admin dengan status “ditolak” | Berhasil |
| 3. |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |

Pengujian Laporan Activity meliputi pengisian email, error pengisian email, dan fungsi tombol yang dapat dilihat pada Tabel 3.9.

**Tabel 3.9 Uji Coba Fitur Laporan**

| **No** | **Kasus** | **Hasil yang Diharapkan** | **Hasil Uji Coba** |
| --- | --- | --- | --- |
| 1. |  |  | Berhasil |
| 2. |  |  | Berhasil |
| 3. |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |

Pengujian Admin Activity meliputi pengisian email, error pengisian email, dan fungsi tombol yang dapat dilihat pada Tabel 3.10.

**Tabel 3.10 Uji Coba Fitur Admin**

| **No** | **Kasus** | **Hasil yang Diharapkan** | **Hasil Uji Coba** |
| --- | --- | --- | --- |
| 1. |  |  | Berhasil |
| 2. |  |  | Berhasil |
| 3. |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |
|  |  |  | Berhasil |

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua:

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea.

## Hasil Kuesioner

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor Lorem ipsum dolor sit amet, consecte.

**Tabel 3.4 Dosen Pengajar Mata Kuliah Semester Gasal**

| **Kode** | **Matakuliah** | **NIK** | **Dosen** |
| --- | --- | --- | --- |
| I4110 | Administrasi Jaringan Komputer | 041035 | Abdul Rahman, S.Si., M.T.I |
| I2101 | Algoritma dan Struktur Data II | 151103 | Siska Devella, M.Kom |
| 021013 | Gasim, Dr., M.Si |
| 121079 | Fithri S. Jumeilah, S.Kom., M.T.I |
| I1105 | Aljabar Linear & Matriks | 151103 | Siska Devella, M.Kom |
| 081050 | Dien Novita, S.Si. M.T.I |
| I1102 | Bahasa Indonesia | 162262 | Dra. Mulyati, M.Pd |
| 162265 | Mursal Saputra, M.Pd |
| I1101 | Bahasa Inggris I | 172276 | Hariana Amalia, M.Pd. |
| 172275 | Despita, M.Pd |
| I3107 | Bahasa Mandarin | 172273 | Mega, B.A.Ed |
| I2104 | Basis Data Terapan | 071046 | Yoannita, M.Kom. |
| 071045 | Daniel Udjulawa, S.Kom., M.T.I |
| 101057 | Iis Pradesan, S.Kom., M.T.I |
| I1107 | Dasar Pemrograman | 151103 | Siska Devella, M.Kom |
| 111065 | Renni Angreni, M.Kom |
| 121082 | Triana Elizabeth, S.Kom, M.T.I |
| I3109 | E-Business | 121080 | Fransiska Prihatini S, S.SI, MTI |
| 121083 | Dicky Pratama, S.Kom., M.T.I |
| I2106 | Fisika Dasar | 131091 | Meiriyama, M.Kom |
| 111061 | Eka Puji W., S.T., M.Kom |
| I2102 | Jaringan Komputer | 101058 | Dedy Hermanto, S.Kom., M.T.I |
| 141102 | Anggoro Aryo P, S.Kom., M.T.I |
| I1104 | Kalkulus I | 051038 | Rizani Teguh, Ir., MT |
| 111069 | Derry A, S.Si., M.Kom.,M.Pd. |
| 162263 | Rahmat Nursalim, S.Pd., M.Si. |
| I3103 | Keamanan Informasi | 141101 | Dorie P. Kesuma, S.Kom., MTI |
| 121083 | Dicky Pratama, S.Kom., M.T.I |
| I4111 | Keamanan Jaringan | 151110 | Molavi Arman, M.Kom |
| I3106 | Kecerdasan Buatan II | 021013 | Gasim, Dr., M.Si |
| 131092 | M.Ezar Al Rivan, M.Kom |
| I4103 | Kewirausahaan | 021018 | Dr. Wijang Widhiarso, M.Kom |
| 151107 | Erwin Azhari W, M.Kom |
| I1108 | Logika Informatika | 151103 | Siska Devella, M.Kom |
| 021005 | Sudiadi, Ir, M.M.A.E |
| 131091 | Meiriyama, M.Kom |
| I4101 | Manajemen Proyek TI | 071045 | Daniel Udjulawa, S.Kom., M.T.I |
| 021014 | M.Rachmadi, S.T., M.T.I |
| I2105 | Metode Numerik | 021005 | Sudiadi, Ir, M.M.A.E |
| 051038 | Rizani Teguh, Ir., MT |

## Judul Subbab

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor Lorem ipsum dolor sit amet, consecte.

**Tabel 3.5 FGA 10 Kromosom & 2 Individu Setiap Kromosom**

| **COR** | **ILR** | **GLR** | **DR** | **Generasi Jadwal Terbaik** | | | **Pelanggaran HC Jadwal Terbaik** | | | **Pelanggaran SC Jadwal Terbaik** | | | **Fitness Jadwal Terbaik** | | | **Waktu Eksekusi (ms)** | | | **Success Rate** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Min** | **Rata-Rata** | **Max** | **Min** | **Rata-Rata** | **Max** | **Min** | **Rata-Rata** | **Max** | **Min** | **Rata-Rata** | **Max** | **Min** | **Rata-Rata** | **Max** |
| 0,5 | 0,05 | 0,05 | 0 | 7 | 60,7 | 165 | 12 | 16,95 | 20 | 15 | 20,4 | 25 | 0,00385 | 0,00435 | 0,00535 | 3944 | 4158,6 | 6025 | 0.00% |
| 0,5 | 0,05 | 0,05 | 0,05 | 35 | 356,2 | 955 | 9 | 13,3 | 17 | 15 | 21 | 26 | 0,00427 | 0,00514 | 0,00676 | 4269 | 4939 | 8142 | 0.00% |
| 0,5 | 0,05 | 0,05 | 0,1 | 79 | 479,15 | 987 | 9 | 13,75 | 16 | 17 | 20,8 | 26 | 0,00418 | 0,00503 | 0,00649 | 4236 | 4464,9 | 4776 | 0.00% |
| 0,5 | 0,05 | 0,1 | 0 | 3 | 59,1 | 148 | 10 | 16,25 | 19 | 14 | 20,05 | 25 | 0,00394 | 0,00454 | 0,00685 | 3909 | 4002,9 | 4191 | 0.00% |
| 0,5 | 0,05 | 0,1 | 0,05 | 21 | 544,1 | 977 | 10 | 13,85 | 19 | 16 | 21,05 | 26 | 0,00403 | 0,00499 | 0,00621 | 4151 | 4267,45 | 4470 | 0.00% |
| 0,5 | 0,05 | 0,1 | 0,1 | 10 | 489,4 | 991 | 10 | 14,2 | 17 | 17 | 21,5 | 27 | 0,00413 | 0,00487 | 0,00617 | 4210 | 4290,7 | 4449 | 0.00% |
| 0,5 | 0,05 | 0,15 | 0 | 4 | 55,4 | 112 | 10 | 16,1 | 20 | 16 | 21,95 | 26 | 0,00376 | 0,00445 | 0,00588 | 3914 | 4010,85 | 4314 | 0.00% |
| 0,5 | 0,05 | 0,15 | 0,05 | 11 | 473,1 | 957 | 9 | 14,1 | 19 | 14 | 19,8 | 26 | 0,00403 | 0,00504 | 0,00645 | 4186 | 4264,1 | 4472 | 0.00% |
| 0,5 | 0,05 | 0,15 | 0,1 | 55 | 482,9 | 860 | 10 | 14,4 | 17 | 12 | 19,55 | 24 | 0,00417 | 0,00495 | 0,0061 | 4192 | 4283,35 | 4537 | 0.00% |
| 0,5 | 0,1 | 0,05 | 0 | 1 | 35,3 | 64 | 10 | 15 | 20 | 14 | 21,9 | 29 | 0,00388 | 0,00466 | 0,00588 | 3906 | 4012,3 | 4207 | 0.00% |
| 0,5 | 0,1 | 0,05 | 0,05 | 120 | 712,7 | 996 | 8 | 10,55 | 14 | 17 | 21,05 | 28 | 0,00498 | 0,00594 | 0,00709 | 4192 | 4276,3 | 4514 | 0.00% |
| 0,5 | 0,1 | 0,05 | 0,1 | 24 | 559 | 970 | 7 | 12,35 | 16 | 13 | 19,9 | 26 | 0,00446 | 0,00549 | 0,00763 | 4198 | 4312,95 | 4463 | 0.00% |
| 0,5 | 0,1 | 0,1 | 0 | 8 | 35,7 | 60 | 13 | 16,35 | 21 | 14 | 20,5 | 25 | 0,00365 | 0,00448 | 0,00529 | 3880 | 4002,2 | 4177 | 0.00% |
| 0,5 | 0,1 | 0,1 | 0,05 | 448 | 798,15 | 987 | 9 | 11,3 | 14 | 13 | 20,35 | 26 | 0,00469 | 0,00579 | 0,00719 | 4189 | 4282,2 | 4456 | 0.00% |
| 0,5 | 0,1 | 0,1 | 0,1 | 169 | 604,7 | 990 | 11 | 12,95 | 15 | 12 | 18 | 23 | 0,00476 | 0,00545 | 0,00641 | 4206 | 4291,5 | 4476 | 0.00% |
| 0,5 | 0,1 | 0,15 | 0 | 1 | 41,75 | 89 | 9 | 15,55 | 21 | 14 | 20,9 | 29 | 0,00392 | 0,00463 | 0,00625 | 3906 | 4004,85 | 4162 | 0.00% |
| 0,5 | 0,1 | 0,15 | 0,05 | 159 | 665,15 | 977 | 8 | 11,45 | 17 | 13 | 20,95 | 25 | 0,00439 | 0,00568 | 0,00667 | 4142 | 4264,3 | 4469 | 0.00% |
| 0,5 | 0,1 | 0,15 | 0,1 | 89 | 605,6 | 973 | 11 | 14 | 18 | 13 | 17,95 | 22 | 0,00441 | 0,00517 | 0,00602 | 4220 | 4300 | 4534 | 0.00% |
| 0,5 | 0,15 | 0,05 | 0 | 1 | 25,3 | 44 | 12 | 16,9 | 21 | 15 | 20,35 | 26 | 0,00377 | 0,00437 | 0,00581 | 3881 | 3997,5 | 4171 | 0.00% |
| 0,5 | 0,15 | 0,05 | 0,05 | 330 | 833,6 | 985 | 4 | 7,55 | 10 | 11 | 17,85 | 23 | 0,00658 | 0,00776 | 0,00935 | 4201 | 4292,1 | 4452 | 0.00% |
| 0,5 | 0,15 | 0,05 | 0,1 | 95 | 657,95 | 972 | 9 | 11,8 | 14 | 11 | 17,55 | 22 | 0,00515 | 0,00587 | 0,00746 | 4245 | 4333,05 | 4557 | 0.00% |
| 0,5 | 0,15 | 0,1 | 0 | 1 | 24,95 | 48 | 9 | 15,6 | 20 | 16 | 20,8 | 26 | 0,00375 | 0,00464 | 0,00602 | 3868 | 3999,95 | 4245 | 0.00% |
| 0,5 | 0,15 | 0,1 | 0,05 | 577 | 835,2 | 997 | 4 | 7,6 | 10 | 12 | 18,85 | 24 | 0,00633 | 0,00758 | 0,00952 | 4197 | 4292,85 | 4525 | 0.00% |
| 0,5 | 0,15 | 0,1 | 0,1 | 171 | 626,15 | 980 | 9 | 11,5 | 15 | 11 | 17,05 | 25 | 0,00495 | 0,00603 | 0,00704 | 4242 | 4336,95 | 4560 | 0.00% |
| 0,5 | 0,15 | 0,15 | 0 | 3 | 33,6 | 79 | 9 | 15,4 | 19 | 13 | 20,75 | 29 | 0,00376 | 0,00467 | 0,00592 | 3892 | 3988,85 | 4229 | 0.00% |
| 0,5 | 0,15 | 0,15 | 0,05 | 505 | 824,75 | 999 | 6 | 7,85 | 10 | 13 | 17,3 | 24 | 0,00667 | 0,00767 | 0,00909 | 4223 | 4301,5 | 4469 | 0.00% |
| 0,5 | 0,15 | 0,15 | 0,1 | 294 | 648,05 | 973 | 10 | 11,75 | 13 | 14 | 17,85 | 22 | 0,00524 | 0,00583 | 0,00645 | 4193 | 4338,25 | 4518 | 0.00% |
| 0,7 | 0,05 | 0,05 | 0 | 9 | 33,9 | 73 | 12 | 15,95 | 20 | 13 | 20,95 | 28 | 0,0038 | 0,00454 | 0,00571 | 3924 | 4019,65 | 4230 | 0.00% |
| 0,7 | 0,05 | 0,05 | 0,05 | 45 | 536,95 | 995 | 10 | 14,75 | 20 | 10 | 20,95 | 27 | 0,0038 | 0,0048 | 0,0061 | 4168 | 4252,8 | 4471 | 0.00% |
| 0,7 | 0,05 | 0,05 | 0,1 | 148 | 593,6 | 1000 | 12 | 14,15 | 16 | 15 | 19,8 | 24 | 0,00452 | 0,00497 | 0,00549 | 4208 | 4318,55 | 4491 | 0.00% |
| 0,7 | 0,05 | 0,1 | 0 | 5 | 36,35 | 79 | 12 | 16,45 | 20 | 13 | 20,75 | 29 | 0,00379 | 0,00443 | 0,00552 | 3936 | 4025,6 | 4223 | 0.00% |
| 0,7 | 0,05 | 0,1 | 0,05 | 5 | 528,55 | 978 | 11 | 13,9 | 18 | 16 | 20,85 | 26 | 0,00412 | 0,00498 | 0,00592 | 4169 | 4271,2 | 4555 | 0.00% |
| 0,7 | 0,05 | 0,1 | 0,1 | 10 | 415,8 | 932 | 10 | 14,15 | 17 | 16 | 20,5 | 27 | 0,00441 | 0,00495 | 0,00621 | 4190 | 4325,95 | 4655 | 0.00% |
| 0,7 | 0,05 | 0,15 | 0 | 3 | 39,7 | 91 | 10 | 17,3 | 22 | 15 | 19,75 | 28 | 0,0036 | 0,00435 | 0,00588 | 3877 | 4026,9 | 4242 | 0.00% |
| 0,7 | 0,05 | 0,15 | 0,05 | 37 | 484,7 | 972 | 11 | 13,8 | 19 | 17 | 21,55 | 28 | 0,00413 | 0,00494 | 0,00575 | 4150 | 4255,05 | 4435 | 0.00% |
| 0,7 | 0,05 | 0,15 | 0,1 | 19 | 505,55 | 995 | 12 | 14,8 | 19 | 15 | 19,85 | 24 | 0,00424 | 0,00483 | 0,00535 | 4228 | 4333 | 4584 | 0.00% |
| 0,7 | 0,1 | 0,05 | 0 | 14 | 33,85 | 48 | 11 | 15,35 | 20 | 14 | 20,85 | 30 | 0,00383 | 0,00465 | 0,00556 | 3926 | 4017,25 | 4240 | 0.00% |
| 0,7 | 0,1 | 0,05 | 0,05 | 60 | 647,05 | 975 | 7 | 11,15 | 15 | 13 | 19,7 | 25 | 0,00461 | 0,00592 | 0,00746 | 4201 | 4290 | 4453 | 0.00% |
| 0,7 | 0,1 | 0,05 | 0,1 | 149 | 556,1 | 990 | 9 | 12,5 | 15 | 15 | 20,4 | 26 | 0,005 | 0,00537 | 0,00637 | 4237 | 4344,35 | 4567 | 0.00% |
| 0,7 | 0,1 | 0,1 | 0 | 6 | 31 | 51 | 13 | 16,15 | 20 | 18 | 21,7 | 27 | 0,00382 | 0,00443 | 0,00505 | 3882 | 4017,55 | 4176 | 0.00% |
| 0,7 | 0,1 | 0,1 | 0,05 | 201 | 683,45 | 969 | 7 | 12,2 | 16 | 14 | 19,45 | 24 | 0,00452 | 0,00559 | 0,008 | 4183 | 4312,2 | 4500 | 0.00% |
| 0,7 | 0,1 | 0,1 | 0,1 | 224 | 757,7 | 991 | 8 | 12,95 | 15 | 16 | 21,65 | 27 | 0,00448 | 0,00514 | 0,00617 | 4209 | 4348,95 | 4562 | 0.00% |
| 0,7 | 0,1 | 0,15 | 0 | 3 | 32,8 | 64 | 13 | 16,5 | 22 | 16 | 21 | 29 | 0,00364 | 0,00442 | 0,00532 | 3876 | 4007 | 4268 | 0.00% |
| 0,7 | 0,1 | 0,15 | 0,05 | 130 | 677,2 | 976 | 10 | 12,7 | 15 | 14 | 19,95 | 24 | 0,00476 | 0,00536 | 0,00621 | 4201 | 4297,95 | 4492 | 0.00% |
| 0,7 | 0,1 | 0,15 | 0,1 | 22 | 515,5 | 965 | 10 | 13,6 | 16 | 14 | 19,35 | 27 | 0,00452 | 0,00516 | 0,00633 | 4213 | 4323,15 | 4713 | 0.00% |
| 0,7 | 0,15 | 0,05 | 0 | 6 | 27,45 | 64 | 11 | 15,45 | 20 | 17 | 20,55 | 28 | 0,00383 | 0,00466 | 0,00562 | 3925 | 4008,5 | 4164 | 0.00% |
| 0,7 | 0,15 | 0,05 | 0,05 | 431 | 748,2 | 988 | 4 | 7,7 | 11 | 16 | 19,35 | 25 | 0,00629 | 0,00748 | 0,0098 | 4205 | 4317,1 | 4520 | 0.00% |
| 0,7 | 0,15 | 0,05 | 0,1 | 43 | 545,4 | 957 | 10 | 11,75 | 14 | 15 | 19,1 | 28 | 0,00526 | 0,00571 | 0,00671 | 4251 | 4366,3 | 4509 | 0.00% |
| 0,7 | 0,15 | 0,1 | 0 | 2 | 24,7 | 47 | 11 | 16,3 | 22 | 17 | 22,2 | 29 | 0,00368 | 0,0044 | 0,00617 | 3921 | 4002,7 | 4155 | 0.00% |
| 0,7 | 0,15 | 0,1 | 0,05 | 104 | 652 | 994 | 6 | 8,95 | 12 | 14 | 18,15 | 25 | 0,00562 | 0,00695 | 0,00794 | 4202 | 4302,95 | 4470 | 0.00% |
| 0,7 | 0,15 | 0,1 | 0,1 | 168 | 608,2 | 901 | 8 | 11,35 | 14 | 14 | 19,6 | 26 | 0,00521 | 0,0058 | 0,00709 | 4256 | 4362,25 | 4516 | 0.00% |
| 0,7 | 0,15 | 0,15 | 0 | 6 | 27,95 | 51 | 12 | 16,5 | 22 | 16 | 21,6 | 26 | 0,00352 | 0,00441 | 0,00571 | 3882 | 4005,1 | 4202 | 0.00% |
| 0,7 | 0,15 | 0,15 | 0,05 | 275 | 752,15 | 998 | 6 | 8,25 | 11 | 14 | 17,85 | 22 | 0,00617 | 0,00735 | 0,00862 | 4196 | 4297,2 | 4481 | 0.00% |
| 0,7 | 0,15 | 0,15 | 0,1 | 74 | 646,2 | 948 | 8 | 11,75 | 14 | 15 | 18,95 | 25 | 0,00505 | 0,00576 | 0,00758 | 4248 | 4320,1 | 4508 | 0.00% |
| 1 | 0,05 | 0,05 | 0 | 1 | 18,5 | 33 | 13 | 18,6 | 23 | 11 | 19,95 | 28 | 0,00365 | 0,00407 | 0,00472 | 3926 | 4028,4 | 4227 | 0.00% |
| 1 | 0,05 | 0,05 | 0,05 | 7 | 438,25 | 995 | 12 | 15,75 | 19 | 15 | 21,9 | 27 | 0,00386 | 0,0045 | 0,00559 | 4195 | 4279,4 | 4429 | 0.00% |
| 1 | 0,05 | 0,05 | 0,1 | 24 | 450,7 | 978 | 11 | 15 | 18 | 15 | 21,25 | 28 | 0,00408 | 0,00469 | 0,00556 | 4277 | 4336 | 4492 | 0.00% |
| 1 | 0,05 | 0,1 | 0 | 1 | 19 | 51 | 12 | 18,1 | 22 | 13 | 19,85 | 26 | 0,00365 | 0,00419 | 0,00571 | 3917 | 4025,3 | 4235 | 0.00% |
| 1 | 0,05 | 0,1 | 0,05 | 2 | 416 | 990 | 11 | 15,55 | 21 | 13 | 21,6 | 26 | 0,0035 | 0,00458 | 0,00556 | 4185 | 4280,15 | 4435 | 0.00% |
| 1 | 0,05 | 0,1 | 0,1 | 31 | 388,05 | 936 | 11 | 15 | 19 | 17 | 21,9 | 28 | 0,00398 | 0,00465 | 0,00546 | 4227 | 4343,4 | 4547 | 0.00% |
| 1 | 0,05 | 0,15 | 0 | 1 | 24,9 | 65 | 16 | 18,3 | 21 | 16 | 21,1 | 26 | 0,00372 | 0,00406 | 0,00465 | 3918 | 4031,2 | 4315 | 0.00% |
| 1 | 0,05 | 0,15 | 0,05 | 1 | 361,1 | 909 | 10 | 15,15 | 22 | 15 | 21,25 | 32 | 0,00372 | 0,00468 | 0,00588 | 4169 | 4312,8 | 4558 | 0.00% |
| 1 | 0,05 | 0,15 | 0,1 | 21 | 472,85 | 940 | 12 | 14,85 | 17 | 13 | 20,75 | 26 | 0,00417 | 0,00475 | 0,00535 | 4237 | 4351,2 | 4581 | 0.00% |
| 1 | 0,1 | 0,05 | 0 | 1 | 12,65 | 33 | 14 | 18,55 | 24 | 16 | 22,5 | 26 | 0,00346 | 0,00397 | 0,00461 | 3914 | 4012,55 | 4187 | 0.00% |
| 1 | 0,1 | 0,05 | 0,05 | 1 | 524,15 | 986 | 11 | 15,3 | 22 | 13 | 22,15 | 30 | 0,00352 | 0,00462 | 0,00588 | 4236 | 4309,8 | 4477 | 0.00% |
| 1 | 0,1 | 0,05 | 0,1 | 166 | 554,6 | 889 | 11 | 14,5 | 19 | 17 | 21,15 | 26 | 0,00398 | 0,00483 | 0,00585 | 4247 | 4366,35 | 4589 | 0.00% |
| 1 | 0,1 | 0,1 | 0 | 4 | 12,7 | 30 | 14 | 18,75 | 24 | 14 | 21,15 | 29 | 0,00341 | 0,004 | 0,00465 | 3911 | 4017,35 | 4215 | 0.00% |
| 1 | 0,1 | 0,1 | 0,05 | 7 | 501,35 | 990 | 11 | 15,3 | 18 | 15 | 20,6 | 27 | 0,00382 | 0,00468 | 0,00595 | 4257 | 4335,45 | 4507 | 0.00% |
| 1 | 0,1 | 0,1 | 0,1 | 44 | 479 | 942 | 12 | 14,65 | 17 | 18 | 21,05 | 25 | 0,00422 | 0,00477 | 0,00562 | 4275 | 4358,9 | 4531 | 0.00% |
| 1 | 0,1 | 0,15 | 0 | 1 | 15,85 | 34 | 16 | 18,1 | 21 | 18 | 22,75 | 28 | 0,00357 | 0,00401 | 0,00459 | 3928 | 4012,8 | 4191 | 0.00% |
| 1 | 0,1 | 0,15 | 0,05 | 4 | 576,85 | 982 | 11 | 14,95 | 17 | 15 | 22,1 | 28 | 0,00397 | 0,00466 | 0,00543 | 4231 | 4338,45 | 4478 | 0.00% |
| 1 | 0,1 | 0,15 | 0,1 | 96 | 492,8 | 981 | 12 | 15,1 | 19 | 13 | 21,25 | 27 | 0,00408 | 0,00466 | 0,00541 | 4248 | 4350,8 | 4562 | 0.00% |
| 1 | 0,15 | 0,05 | 0 | 1 | 11,9 | 44 | 15 | 19,45 | 24 | 17 | 21,95 | 26 | 0,00329 | 0,00385 | 0,00455 | 3863 | 4020,4 | 4230 | 0.00% |
| 1 | 0,15 | 0,05 | 0,05 | 133 | 487,55 | 956 | 9 | 12,95 | 16 | 16 | 20,75 | 26 | 0,00435 | 0,00522 | 0,00599 | 4244 | 4343,15 | 4553 | 0.00% |
| 1 | 0,15 | 0,05 | 0,1 | 152 | 667,3 | 1000 | 10 | 13,7 | 17 | 12 | 19,55 | 25 | 0,00463 | 0,00511 | 0,00588 | 4262 | 4391,65 | 4604 | 0.00% |
| 1 | 0,15 | 0,1 | 0 | 2 | 12,65 | 34 | 12 | 18,8 | 23 | 15 | 21,5 | 28 | 0,0034 | 0,00399 | 0,00552 | 3894 | 4027,3 | 4203 | 0.00% |
| 1 | 0,15 | 0,1 | 0,05 | 168 | 689,9 | 997 | 10 | 13,55 | 17 | 11 | 19,9 | 25 | 0,00442 | 0,00516 | 0,00645 | 4242 | 4338,6 | 4537 | 0.00% |
| 1 | 0,15 | 0,1 | 0,1 | 103 | 576,55 | 976 | 11 | 14,05 | 16 | 12 | 21,05 | 27 | 0,00448 | 0,00491 | 0,00552 | 4273 | 4378,6 | 4572 | 0.00% |
| 1 | 0,15 | 0,15 | 0 | 1 | 15,15 | 42 | 15 | 19,35 | 23 | 13 | 21,85 | 29 | 0,0034 | 0,00389 | 0,005 | 3915 | 4030,3 | 4286 | 0.00% |

## Judul Subbab

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua:

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi.

**Tabel 3.6 Hasil Pengujian FGA Skenario 20 Individu Semester Gasal**

| **No** | **Generasi Jadwal Terbaik** | **Pelanggaran HC Jadwal Terbaik** | **Pelanggaran SC Jadwal Terbaik** | **Fitness Jadwal Terbaik** | **Waktu Eksekusi** |
| --- | --- | --- | --- | --- | --- |
| **1** | 808 | 7 | 18 | 0,008 | 5673 |
| **2** | 941 | 6 | 12 | 0,01031 | 4676 |
| **3** | 994 | 6 | 20 | 0,00826 | 4550 |
| **4** | 982 | 8 | 20 | 0,00709 | 4540 |
| **5** | 581 | 7 | 15 | 0,00862 | 4556 |
| **6** | 557 | 6 | 14 | 0,00971 | 4681 |
| **7** | 941 | 3 | 18 | 0,01176 | 4708 |
| **8** | 773 | 5 | 18 | 0,00952 | 5394 |
| **9** | 922 | 7 | 19 | 0,00781 | 4749 |
| **10** | 968 | 4 | 16 | 0,01124 | 4742 |
| **11** | 983 | 6 | 12 | 0,01031 | 4752 |
| **12** | 748 | 7 | 14 | 0,00885 | 4768 |
| **13** | 977 | 7 | 15 | 0,00862 | 5064 |
| **14** | 511 | 7 | 11 | 0,00962 | 4534 |
| **15** | 846 | 5 | 21 | 0,00877 | 4544 |
| **16** | 847 | 5 | 14 | 0,01075 | 4677 |
| **17** | 878 | 8 | 15 | 0,00794 | 4688 |
| **18** | 949 | 6 | 16 | 0,00917 | 4761 |
| **19** | 767 | 8 | 17 | 0,00758 | 4670 |
| **20** | 972 | 4 | 17 | 0,01087 | 4687 |
| **21** | 841 | 5 | 17 | 0,0098 | 4579 |
| **22** | 161 | 7 | 22 | 0,0073 | 4540 |
| **23** | 897 | 6 | 20 | 0,00826 | 4685 |
| **24** | 970 | 6 | 16 | 0,00917 | 4674 |
| **25** | 464 | 7 | 17 | 0,0082 | 4683 |
| **26** | 720 | 5 | 21 | 0,00877 | 4698 |
| **27** | 881 | 5 | 16 | 0,0101 | 4675 |
| **28** | 803 | 5 | 13 | 0,01111 | 4556 |
| **29** | 733 | 4 | 17 | 0,01087 | 4682 |
| **30** | 989 | 8 | 21 | 0,00694 | 4680 |
| **31** | 981 | 5 | 16 | 0,0101 | 4701 |
| **32** | 879 | 6 | 20 | 0,00826 | 4758 |
| **33** | 977 | 3 | 19 | 0,01136 | 4579 |
| **34** | 977 | 8 | 13 | 0,00833 | 4581 |
| **35** | 729 | 5 | 16 | 0,0101 | 4669 |
| **36** | 619 | 4 | 22 | 0,00935 | 4675 |
| **37** | 940 | 7 | 19 | 0,00781 | 4677 |
| **38** | 978 | 8 | 13 | 0,00833 | 4701 |
| **39** | 942 | 5 | 19 | 0,00926 | 4701 |
| **40** | 990 | 4 | 21 | 0,00962 | 4579 |
| **41** | 799 | 7 | 15 | 0,00862 | 4555 |
| **42** | 996 | 3 | 19 | 0,01136 | 4691 |
| **43** | 879 | 6 | 16 | 0,00917 | 4679 |
| **44** | 574 | 6 | 16 | 0,00917 | 4692 |
| **45** | 940 | 5 | 20 | 0,00901 | 4580 |
| **46** | 522 | 5 | 17 | 0,0098 | 4551 |
| **47** | 439 | 5 | 20 | 0,00901 | 4677 |
| **48** | 843 | 5 | 17 | 0,0098 | 4673 |
| **49** | 900 | 9 | 14 | 0,00752 | 4673 |
| **50** | 895 | 4 | 20 | 0,0099 | 4688 |
| **Minimum** | **161** | **3** | **11** | **0,00694** | **4534** |
| **Rata-Rata** | **824,06** | **5,8** | **17,08** | **0,009224** | **4698,92** |
| **Maksimum** | **996** | **9** | **22** | **0,01176** | **5673** |

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi.

**Tabel 3.7 Hasil Pengujian GA Skenario 20 Individu Semester Gasal**

| **No** | **Generasi Jadwal Terbaik** | **Pelanggaran HC Jadwal Terbaik** | **Pelanggaran SC Jadwal Terbaik** | **Fitness Jadwal Terbaik** | **Waktu Eksekusi** |
| --- | --- | --- | --- | --- | --- |
| **1** | 864 | 0 | 7 | 0,04545 | 4215 |
| **2** | 716 | 0 | 8 | 0,04 | 3205 |
| **3** | 787 | 0 | 9 | 0,03571 | 3165 |
| **4** | 901 | 0 | 8 | 0,04 | 3163 |
| **5** | 877 | 0 | 6 | 0,05263 | 3228 |
| **6** | 750 | 0 | 5 | 0,0625 | 3211 |
| **7** | 910 | 0 | 6 | 0,05263 | 3181 |
| **8** | 981 | 0 | 7 | 0,04545 | 3174 |
| **9** | 892 | 1 | 9 | 0,02632 | 3166 |
| **10** | 844 | 0 | 9 | 0,03571 | 3174 |
| **11** | 696 | 0 | 9 | 0,03571 | 3173 |
| **12** | 785 | 0 | 5 | 0,0625 | 3169 |
| **13** | 836 | 0 | 7 | 0,04545 | 3169 |
| **14** | 801 | 0 | 7 | 0,04545 | 3193 |
| **15** | 884 | 0 | 7 | 0,04545 | 3254 |
| **16** | 925 | 0 | 9 | 0,03571 | 3191 |
| **17** | 745 | 1 | 8 | 0,02857 | 3172 |
| **18** | 698 | 1 | 5 | 0,03846 | 3174 |
| **19** | 798 | 0 | 7 | 0,04545 | 3159 |
| **20** | 790 | 0 | 8 | 0,04 | 3150 |
| **21** | 830 | 0 | 8 | 0,04 | 3160 |
| **22** | 803 | 0 | 12 | 0,02703 | 3155 |
| **23** | 876 | 0 | 7 | 0,04545 | 3199 |
| **24** | 938 | 0 | 7 | 0,04545 | 3246 |
| **25** | 754 | 0 | 5 | 0,0625 | 3215 |
| **26** | 720 | 0 | 6 | 0,05263 | 3176 |
| **27** | 830 | 1 | 3 | 0,05 | 3159 |
| **28** | 954 | 0 | 7 | 0,04545 | 3152 |
| **29** | 959 | 0 | 3 | 0,1 | 3173 |
| **30** | 362 | 0 | 4 | 0,07692 | 3176 |
| **31** | 702 | 0 | 7 | 0,04545 | 3213 |
| **32** | 646 | 0 | 7 | 0,04545 | 3249 |
| **33** | 780 | 0 | 8 | 0,04 | 3171 |
| **34** | 870 | 0 | 6 | 0,05263 | 3187 |
| **35** | 929 | 0 | 4 | 0,07692 | 3176 |
| **36** | 687 | 0 | 6 | 0,05263 | 3156 |
| **37** | 844 | 0 | 8 | 0,04 | 3152 |
| **38** | 987 | 0 | 6 | 0,05263 | 3232 |
| **39** | 874 | 0 | 9 | 0,03571 | 3218 |
| **40** | 750 | 0 | 9 | 0,03571 | 3323 |
| **41** | 982 | 0 | 4 | 0,07692 | 3187 |
| **42** | 826 | 0 | 6 | 0,05263 | 3164 |
| **43** | 985 | 0 | 5 | 0,0625 | 3163 |
| **44** | 890 | 0 | 3 | 0,1 | 3167 |
| **45** | 473 | 0 | 9 | 0,03571 | 3165 |
| **46** | 943 | 0 | 4 | 0,07692 | 3143 |
| **47** | 831 | 0 | 8 | 0,04 | 3184 |
| **48** | 925 | 0 | 7 | 0,04545 | 3244 |
| **49** | 997 | 0 | 7 | 0,04545 | 3184 |
| **50** | 954 | 0 | 6 | 0,05263 | 3182 |
| **Minimum** | **362** | **0** | **3** | **0,02632** | **3143** |
| **Rata-Rata** | **827,62** | **0,08** | **6,74** | **0,049398** | **3207,14** |
| **Maksimum** | **997** | **1** | **12** | **0,1** | **4215** |

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi

# PENUTUP

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud dolor Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

## Kesimpulan

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip:

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea.

## Saran

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis:

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor Lorem ipsum dolor sit amet, consectetur.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor Lorem ipsum dolor sit amet, consectetur adipiscing.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation.
4. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip.

**DAFTAR PUSTAKA**

Sede, D. W. E., Sinsuw, A. A. E., & Najoan, X. B. N. (2015). Rancang Bangun Aplikasi Pemesanan Tiket Online Kapal Laut Berbasis Android. *E-journal Teknik Informatika*, 6(1). ISSN: 2301-8364.

Novita, R., & Zahri, H. (2018). Rancang Bangun Aplikasi Katalog Kerja Praktek dan Tugas Akhir Berbasis Android. *Jurnal Ilmiah Rekayasa dan Manajemen Sistem Informasi*, 4(2), 209-215. e-ISSN 2502-8995, p-ISSN 2460-8181

Budiyanto, N. E., & Sultonuddin, R. (2021). Penerapan QR Code pada Presensi Seminar Kerja Praktek Teknik Informatika Unwahas Berbasis Android. *Jurnal Informatika dan Rekayasa Perangkat Lunak*, 3(2), 148-155. ISSN: 2656-2855. e-ISSN: 2685-5518.

Alda, M. (2021). Pemanfaatan Barcode Scanner Pada Aplikasi Manajemen Inventory Barang Berbasis Android. *SISFOKOM (Sistem Informasi dan Komputer)*, *10*(3), 368-375.

Payara, G. R., & Tanone, R. (2018). Penerapan Firebase Realtime Database pada prototype aplikasi pemesanan makanan berbasis Android. *Jurnal Teknik Informatika dan Sistem Informasi*, 4(3), 397.

Khawas, C. (2018). Application of Firebase in Android App Development-A Study. *International Journal of Computer Applications, 179*(46), 49.

Hermawati, F. A., & Primasetya, A. (2021). Pemanfaatan Microsoft Excel untuk aplikasi data pelanggan pada usaha kecil jasa pembersihan sepatu. INAMISIA: Jurnal Pengabdian Kepada Masyarakat, 5(4), 850-859.

Abdelhalim, E. A., & El Khayat, G. A. (2016). A utilization-based genetic algorithm for solving the university timetabling problem (UGA). *Alexandria Engineering Journal*, *55*(2), 1395–1409. https://doi.org/10.1016/j.aej.2016.02.017

Akkan, C., & Gülcü, A. (2018). A bi-criteria hybrid genetic algorithm with robustness objective for the course timetabling problem. *Computers and Operations Research*, *90*, 22–32. https://doi.org/10.1016/j.cor.2017.09.007

de Oliveira, L. L., Freitas, A. A., & Tinós, R. (2018). Multi-objective genetic algorithms in the study of the genetic code’s adaptability. *Information Sciences*, *425*, 48–61. https://doi.org/10.1016/j.ins.2017.10.022

Febrita, R. E., & Mahmudy, W. F. (2017). Modified genetic algorithm for high school time-table scheduling with fuzzy time window. *International Conference on Sustainable Information Engineering and Technology*, 88–92.

Gao, S., & Silva, C. W. De. (2016). A modified estimation distribution algorithm based on extreme elitism. *BioSystems*, *150*, 149–166. https://doi.org/10.1016/j.biosystems.2016.10.001

Ghasemi, E., Moradi, P., & Fathi, M. (2015). Integrating ABC with genetic grouping for university course timetabling problem. *2015 5th International Conference on Computer and Knowledge Engineering (lCCKE)*, 24–29.

Jafari-Marandi, R., & Smith, B. K. (2017). Fluid genetic algorithm (FGA). *Journal of Computational Design and Engineering*, *4*(2), 158–167. https://doi.org/10.1016/j.jcde.2017.03.001

Lei, Y., Shi, J., & Yan, Z. (2018). A memetic algorithm based on MOEA/D for the examination timetabling problem. *Soft Computing*, *22*(5), 1511–1523. https://doi.org/10.1007/s00500-017-2886-y

Lewis, R., & Thompson, J. (2014). Analysing the effects of solution space connectivity with an effective metaheuristic for the course timetabling problem. *European Journal of Operational Research*, *240*(3), 637–648. https://doi.org/10.1016/j.ejor.2014.07.041

Liu, W., Zhu, H., Wang, Y., Zhou, S., Bai, Y., & Zhao, C. (2013). Topology optimization of support structure of telescope skin based on bit-matrix representation NSGA-II. *Chinese Journal of Aeronautics*, *26*(6), 1422–1429. https://doi.org/10.1016/j.cja.2013.07.046

Mahiba, A. A., & Durai, C. A. D. (2012). Genetic algorithm with search bank strategies for university course timetabling problem. *Procedia Engineering*, *38*, 253–263. https://doi.org/10.1016/j.proeng.2012.06.033

Parera, S., Sukmana, H. T., & Wardhani, L. K. (2016). Application of genetic algorithm for class scheduling (case study: faculty of science and technology UIN Jakarta). *2016 4th International Conference on Cyber and IT Service Management*, 1–5. https://doi.org/10.1109/CITSM.2016.7577525

Yousef, A. H., Salama, C., Jad, M. Y., El-gafy, T., Matar, M., & Habashi, S. S. (2016). A GPU based genetic algorithm solution for the timetabling problem. *2016 11th International Conference on Computer Engineering & Systems (ICCES)*, 103–109. https://doi.org/10.1109/ICCES.2016.7821982

# LAMPIRAN

*Tuliskan lampiran-lampiran yang ada sesuai dengan tugas akhir yang dipilih*

Misalnya:

1. Surat pengantar KP dari kampus
2. Surat balasan KP dari perusahaan
3. Surat selesai KP dari perusahaan
4. Surat-surat resmi lainnya jika ada
5. Foto Ruangan Kerja Praktik
6. Foto dengan Penyelia dan/atau staf perusahaan (peserta KP **memakai jas almamater**)
7. KAKP (Kerangka Acuan Kerja Praktik)
8. Log Harian
9. Lembar Konsultasi dari SIMPONI
10. Lembar-lembar bukti wawancara
11. Foto bukti wawancara (jika ada)
12. Bukti pengambilan data / bukti kegiatan KP di perusahaan
13. Lampiran Kuesioner
14. notulen seminar kerja praktik
15. Kode program inti
16. dan lain lain

## Lembar Konsultasi

|  |  |
| --- | --- |
|  | **LEMBAR KONSULTASI**  **KERJA PRAKTIK**  **(CONTOH) (FILE ASLI HARUS DIUNDUH DI SIMPONI)** |
| XX-XXX-XXX-XX.XX |

TAHUN AKADEMIK : 202X/202X Hal x dari x

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nama / NPM | | | | Xxxxxxxxxxx | | xxxxxxx | | |
| Program Studi | | | | XXXXXXX | | | | |
| Topik | | | | Xxxxxxxxxxxxxxxxx | | | | |
| Judul | | | | Xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx | | | | |
| Pembimbing | | | | **(lembar diunduh pada menu di SIMPONI pada akhir bimbingan saat mendaftar sidang TA)** | | | | |
|  | | | |  | | | | |
| **No** | **Tanggal** | | **Uraian** | | | | **Paraf**  **Pembimbing** | |
| **I** | **II** |
|  |  | |  | | | |  |  |
|  |  | |  | | | |  |  |
| Persetujuan Pembimbing | | | | | JUDUL TUGAS AKHIR :  Xxxxxxxxxxxxxxxxxxx | | | |
| Pembimbing 1 | | Pembimbing 2 | | |

**Lampiran Foto**

A group of men standing together

Description automatically generated

Foto bersama dengan penyelia

A group of men sitting at a table

Description automatically generated

Foto kegiatan Kp di CV Wijaya Lesmana

A computer on a desk

Description automatically generated

Foto Ruangan Kerja Praktek

## Kode Program

XXXXXXXXXXXXXXXXXXXX

Jika kode program terlalu banyak, seperti aplikasi website, cukup ambil contoh/intinya saja, misal kode program terkait penjualan

XXXXXXXXXX

Kode Program Halaman SplashSreen.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import androidx.appcompat.app.AppCompatActivity;

4.

5. import android.content.Intent;

6. import android.os.Bundle;

7. import android.os.Handler;

8.

9. import com.example.barangku.R;

10.

11. public class SplashScreen extends AppCompatActivity {

12.

13. @Override

14. protected void onCreate(Bundle savedInstanceState) {

15. super.onCreate(savedInstanceState);

16. setContentView(R.layout.activity\_splash\_screen);

17. new Handler().postDelayed(new Runnable() {

18. @Override

19. public void run() {

20. Intent mainIntent = new Intent(SplashScreen.this, Login.class);

21. startActivity(mainIntent);

22. finish();

23. }

24. }, 3000); // 3000 milliseconds = 3 seconds

25. }

26. }

27.

Kode Program Halaman Login.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import androidx.annotation.NonNull;

4. import androidx.appcompat.app.AppCompatActivity;

5.

6. import android.content.Intent;

7. import android.os.Bundle;

8. import android.text.TextUtils;

9. import android.view.View;

10. import android.widget.Button;

11. import android.widget.TextView;

12. import android.widget.Toast;

13.

14. import com.example.barangku.R;

15.

16. import com.google.android.gms.tasks.OnCompleteListener;

17. import com.google.android.gms.tasks.Task;

18. import com.google.android.material.textfield.TextInputEditText;

19. import com.google.firebase.auth.AuthResult;

20. import com.google.firebase.auth.FirebaseAuth;

21. import com.google.firebase.auth.FirebaseUser;

22.

23. public class Login extends AppCompatActivity {

24. private TextInputEditText etEmail\_login,etPassword\_login;

25. private Button btn\_login;

26. private TextView tv\_register,tv\_lupapassword;

27. private FirebaseAuth mAuth;

28. private String email, password;

29. @Override

30. public void onStart() {

31. super.onStart();

32. FirebaseUser currentUser = mAuth.getCurrentUser();

33. if(currentUser != null ){

34. Intent intent = new Intent(getApplicationContext(), MainActivity.class);

35. startActivity(intent);

36.

37. }

38. }

39.

40. @Override

41. protected void onCreate(Bundle savedInstanceState) {

42. super.onCreate(savedInstanceState);

43. setContentView(R.layout.activity\_login);

44. tv\_register=findViewById(R.id.tv\_register);

45. mAuth=FirebaseAuth.getInstance();

46. etEmail\_login= findViewById(R.id.et\_email\_login);

47. etPassword\_login=findViewById(R.id.et\_password\_login);

48. btn\_login=findViewById(R.id.btn\_login\_button);

49. tv\_lupapassword=findViewById(R.id.tv\_lupaPassword);

50.

51. tv\_lupapassword.setOnClickListener(new View.OnClickListener() {

52. @Override

53. public void onClick(View v) {

54. Intent intent = new Intent(Login.this, LupaPassword.class);

55. startActivity(intent);

56. finish();

57. }

58. });

59.

60. btn\_login.setOnClickListener(new View.OnClickListener() {

61. @Override

62. public void onClick(View v) {

63. email = etEmail\_login.getText().toString();

64. password= etPassword\_login.getText().toString();

65.

66. if(email.trim().isEmpty()){

67. etEmail\_login.setError("Silakan Masukan Email ");

68. }

69. if(password.trim().isEmpty()){

70. etPassword\_login.setError("Silakan Masukan Password ");

71. }

72. String email = etEmail\_login.getText().toString().trim();

73. String password = etPassword\_login.getText().toString().trim();

74.

75. if (!TextUtils.isEmpty(email) && !TextUtils.isEmpty(password)) {

76.

77. mAuth.signInWithEmailAndPassword(email, password)

78. .addOnCompleteListener( new OnCompleteListener<AuthResult>() {

79. @Override

80. public void onComplete(@NonNull Task<AuthResult> task) {

81. if (task.isSuccessful()) {

82. Toast.makeText(Login.this, "Login berhasil", Toast.LENGTH\_SHORT).show();

83. FirebaseUser user = mAuth.getCurrentUser();

84. Intent intent = new Intent(Login.this, MainActivity.class);

85. startActivity(intent);

86. finish();

87. } else {

88. Toast.makeText(Login.this, "login Gagal.",

89. Toast.LENGTH\_SHORT).show();

90. }

91. }

92. });

93. }

94. }

95. });

96. tv\_register.setOnClickListener(new View.OnClickListener() {

97. @Override

98. public void onClick(View v) {

99. Intent intent = new Intent(Login.this, Register.class);

100. startActivity(intent);

101. finish();

102. }

103. });

104. }

105. }

106.

Kode Program Halaman LupaPassword.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import androidx.annotation.NonNull;

4. import androidx.appcompat.app.AppCompatActivity;

5.

6. import android.content.Intent;

7. import android.os.Bundle;

8. import android.text.Editable;

9. import android.text.TextUtils;

10. import android.text.TextWatcher;

11. import android.view.View;

12. import android.widget.ImageView;

13. import android.widget.ProgressBar;

14. import android.widget.TextView;

15. import android.widget.Toast;

16.

17. import com.example.barangku.R;

18. import com.google.android.gms.tasks.OnCompleteListener;

19. import com.google.android.gms.tasks.OnFailureListener;

20. import com.google.android.gms.tasks.OnSuccessListener;

21. import com.google.android.gms.tasks.Task;

22. import com.google.android.material.button.MaterialButton;

23. import com.google.android.material.textfield.TextInputEditText;

24. import com.google.android.material.textfield.TextInputLayout;

25. import com.google.firebase.auth.FirebaseAuth;

26. import com.google.firebase.auth.SignInMethodQueryResult;

27.

28. import java.util.regex.Matcher;

29. import java.util.regex.Pattern;

30.

31. public class LupaPassword extends AppCompatActivity {

32.

33. private MaterialButton btnResetPassword;

34. private TextInputEditText etEmail;

35. private TextInputLayout ilEmail;

36. private ProgressBar progressBar;

37. private FirebaseAuth mAuth;

38. private String email;

39. private TextView tv\_toolbar;

40. private ImageView ivback;

41.

42. @Override

43. protected void onCreate(Bundle savedInstanceState) {

44. super.onCreate(savedInstanceState);

45. setContentView(R.layout.activity\_lupa\_password);

46.

47. btnResetPassword = findViewById(R.id.btn\_resetpassword);

48. etEmail = findViewById(R.id.et\_email);

49. ilEmail = findViewById(R.id.il\_email);

50. ivback = findViewById(R.id.iv\_back);

51. progressBar = findViewById(R.id.progressBar);

52. mAuth = FirebaseAuth.getInstance();

53.

54. tv\_toolbar = findViewById(R.id.tv\_judul);

55. tv\_toolbar.setText("Lupa Password");

56.

57. ivback.setOnClickListener(new View.OnClickListener() {

58. @Override

59. public void onClick(View v) {

60. Intent intent = new Intent(LupaPassword.this, Login.class);

61. startActivity(intent);

62. }

63. });

64.

65. etEmail.addTextChangedListener(new TextWatcher() {

66. @Override

67. public void beforeTextChanged(CharSequence s, int start, int count, int after) {

68.

69. }

70.

71. @Override

72. public void onTextChanged(CharSequence s, int start, int before, int count) {

73. String email = s.toString();

74. Pattern pattern = Pattern.compile("^[\\w-\\.]+@([\\w-]+\\.)+[\\w-]{2,4}$");

75. Matcher matcher = pattern.matcher(email);

76. boolean isValid = matcher.matches();

77. if (isValid) {

78. ilEmail.setError("");

79. } else {

80. ilEmail.setError("Alamat email tidak valid");

81. }

82. }

83.

84. @Override

85. public void afterTextChanged(Editable s) {

86. String email = s.toString();

87. if (email.isEmpty()) {

88. ilEmail.setError("");

89. }

90. }

91. });

92.

93. btnResetPassword.setOnClickListener(new View.OnClickListener() {

94. @Override

95. public void onClick(View v) {

96. progressBar.setVisibility(View.VISIBLE);

97.

98. email = String.valueOf(etEmail.getText());

99. if (TextUtils.isEmpty(email)) {

100. Toast.makeText(LupaPassword.this, "Masukkan Email Anda", Toast.LENGTH\_SHORT).show();

101. progressBar.setVisibility(View.GONE);

102. return;

103. }

104.

105. mAuth.fetchSignInMethodsForEmail(email).addOnCompleteListener(new OnCompleteListener<SignInMethodQueryResult>() {

106. @Override

107. public void onComplete(@NonNull Task<SignInMethodQueryResult> task) {

108. if (task.isSuccessful()) {

109. boolean isRegistered = !task.getResult().getSignInMethods().isEmpty();

110. if (isRegistered) {

111. mAuth.sendPasswordResetEmail(email).addOnSuccessListener(new OnSuccessListener<Void>() {

112. @Override

113. public void onSuccess(Void unused) {

114. Toast.makeText(LupaPassword.this, "Reset password sudah terkirim ke email anda", Toast.LENGTH\_SHORT).show();

115. Intent intent = new Intent(getApplicationContext(), Login.class);

116. startActivity(intent);

117. finish();

118. }

119. }).addOnFailureListener(new OnFailureListener() {

120. @Override

121. public void onFailure(@NonNull Exception e) {

122. Toast.makeText(LupaPassword.this, "Error : " + e.getMessage(), Toast.LENGTH\_SHORT).show();

123. progressBar.setVisibility(View.GONE);

124. }

125. });

126. } else {

127. Toast.makeText(LupaPassword.this, "Email tidak terdaftar", Toast.LENGTH\_SHORT).show();

128. progressBar.setVisibility(View.GONE);

129. }

130. } else {

131. Toast.makeText(LupaPassword.this, "Error : " + task.getException().getMessage(), Toast.LENGTH\_SHORT).show();

132. progressBar.setVisibility(View.GONE);

133. }

134. }

135. });

136. }

137. });

138. }

139.

140. @Override

141. public void onBackPressed() {

142. super.onBackPressed();

143. Intent intent = new Intent(this, Login.class);

144. intent.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP | Intent.FLAG\_ACTIVITY\_NEW\_TASK);

145. startActivity(intent);

146. finish();

147. }

148. }

149.

Kode Program MainActivity.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import androidx.annotation.NonNull;

4. import androidx.appcompat.app.AlertDialog;

5. import androidx.appcompat.app.AppCompatActivity;

6.

7. import android.app.Activity;

8. import android.content.Intent;

9. import android.os.Bundle;

10. import android.view.View;

11. import android.widget.Button;

12. import android.widget.GridLayout;

13. import android.widget.Toast;

14.

15. import com.example.barangku.R;

16. import com.example.barangku.activity.user\_activity.laporan.Laporan;

17. import com.google.firebase.auth.FirebaseAuth;

18. import com.google.firebase.auth.FirebaseUser;

19. import com.google.firebase.database.DataSnapshot;

20. import com.google.firebase.database.DatabaseError;

21. import com.google.firebase.database.DatabaseReference;

22. import com.google.firebase.database.FirebaseDatabase;

23. import com.google.firebase.database.ValueEventListener;

24.

25. public class MainActivity extends AppCompatActivity {

26. GridLayout menuUser, menuAdmin;

27. FirebaseAuth mAuth = FirebaseAuth.getInstance();

28. FirebaseUser mUser = mAuth.getCurrentUser();

29. private boolean toastShown = false;

30. DatabaseReference reference = FirebaseDatabase.getInstance().getReference("User");

31.

32. private Button btnSignout, btnBarangMasukUser, btnBarangKeluarUser,btnAdmin, btnPengajuan, btnRiwayatPengajuan;

33. private Button btnBarangMasukAdmin, btnBarangKeluarAdmin, btnLaporanAdmin, btnClientAdmin, btnStockAdmin;

34.

35. @Override

36. protected void onCreate(Bundle savedInstanceState) {

37. super.onCreate(savedInstanceState);

38. setContentView(R.layout.activity\_main);

39.

40. menuUser = findViewById(R.id.menu\_user);

41. menuAdmin = findViewById(R.id.menu\_admin);

42.

43. btnPengajuan = findViewById(R.id.btn\_pengajuan);

44. btnRiwayatPengajuan = findViewById(R.id.btn\_riwayat\_pengajuan);

45.

46. btnBarangMasukUser = findViewById(R.id.btn\_Barang\_Masuk\_User);

47. btnBarangKeluarUser = findViewById(R.id.btn\_Barang\_Keluar\_User);

48.

49. btnBarangMasukAdmin = findViewById(R.id.btn\_barang\_masuk\_admin);

50. btnBarangKeluarAdmin = findViewById(R.id.btn\_barang\_keluar\_admin);

51. btnClientAdmin = findViewById(R.id.btn\_client\_admin);

52. btnLaporanAdmin = findViewById(R.id.btn\_laporan\_admin);

53. btnStockAdmin = findViewById(R.id.btn\_stok\_barang\_admin);

54. btnAdmin = findViewById(R.id.btn\_admin);

55.

56. btnSignout = findViewById(R.id.btn\_Signout);

57.

58. reference.addValueEventListener(new ValueEventListener() {

59. @Override

60. public void onDataChange(@NonNull DataSnapshot snapshot) {

61. if (toastShown) return;

62. for (DataSnapshot userSnapshot : snapshot.getChildren()) {

63. if (userSnapshot.getKey().equals(mAuth.getCurrentUser().getUid())) {

64. String jabatan = userSnapshot.child("jabatan").getValue(String.class);

65. if (jabatan.equals("Admin")) {

66. menuAdmin.setVisibility(View.VISIBLE);

67. menuUser.setVisibility(View.GONE);

68. Toast.makeText(MainActivity.this, "Anda Login Sebagai Admin", Toast.LENGTH\_SHORT).show();

69. btnSignout.setVisibility(View.VISIBLE);

70. } else {

71. menuUser.setVisibility(View.VISIBLE);

72. menuAdmin.setVisibility(View.GONE);

73. Toast.makeText(MainActivity.this, "Anda Login Sebagai Karyawan", Toast.LENGTH\_SHORT).show();

74. btnSignout.setVisibility(View.VISIBLE);

75. }

76. toastShown = true;

77. break; // Exit the loop once the user is found

78. }

79. }

80. }

81.

82. @Override

83. public void onCancelled(@NonNull DatabaseError error) {

84. // Handle database error

85. Toast.makeText(MainActivity.this, "Database error: " + error.getMessage(), Toast.LENGTH\_SHORT).show();

86. }

87. });

88.

89. setupButtonListeners();

90. }

91.

92. private void setupButtonListeners() {

93. btnPengajuan.setOnClickListener(v -> pindahHalaman(MainActivity.this, Pengajuan.class));

94. btnRiwayatPengajuan.setOnClickListener(v -> pindahHalaman(MainActivity.this, RiwayatPengajuan.class));

95. btnBarangMasukUser.setOnClickListener(v -> pindahHalaman(MainActivity.this, BarangMasuk.class));

96. btnBarangKeluarUser.setOnClickListener(v -> pindahHalaman(MainActivity.this, BarangKeluar.class));

97. btnBarangMasukAdmin.setOnClickListener(v -> pindahHalaman(MainActivity.this, BarangMasuk.class));

98. btnBarangKeluarAdmin.setOnClickListener(v -> pindahHalaman(MainActivity.this, BarangKeluar.class));

99. btnClientAdmin.setOnClickListener(v -> pindahHalaman(MainActivity.this, Client.class));

100. btnLaporanAdmin.setOnClickListener(v -> pindahHalaman(MainActivity.this, Laporan.class));

101. btnStockAdmin.setOnClickListener(v -> pindahHalaman(MainActivity.this, StockBarang.class));

102. btnAdmin.setOnClickListener(v -> pindahHalaman(MainActivity.this, Admin.class));

103. btnSignout.setOnClickListener(v -> {

104. FirebaseAuth.getInstance().signOut();

105. Intent intent = new Intent(MainActivity.this, Login.class);

106. startActivity(intent);

107. finish();

108. });

109. }

110.

111. private void pindahHalaman(Activity activityawal, Class activitytujuan) {

112. Intent intent = new Intent(activityawal, activitytujuan);

113. intent.setFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK);

114. startActivity(intent);

115. }

116.

117. @Override

118. public void onBackPressed() {

119. if (getSupportFragmentManager().getBackStackEntryCount() == 0) {

120. new AlertDialog.Builder(this)

121. .setMessage("Are you sure you want to exit the app?")

122. .setCancelable(false)

123. .setPositiveButton("Yes", (dialog, id) -> {

124. moveTaskToBack(true);

125. android.os.Process.killProcess(android.os.Process.myPid());

126. System.exit(1);

127. })

128. .setNegativeButton("No", (dialog, id) -> dialog.cancel())

129. .create()

130. .show();

131. } else {

132. super.onBackPressed();

133. }

134. }

135. }

136.

Kode Program Menu BarangMasuk.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import androidx.annotation.NonNull;

4. import androidx.annotation.Nullable;

5. import androidx.appcompat.app.AppCompatActivity;

6.

7. import android.content.Intent;

8. import android.net.Uri;

9. import android.os.Bundle;

10. import android.view.View;

11. import android.widget.AdapterView;

12. import android.widget.ArrayAdapter;

13. import android.widget.AutoCompleteTextView;

14. import android.widget.Button;

15. import android.widget.EditText;

16. import android.widget.ImageView;

17. import android.widget.TextView;

18. import android.widget.Toast;

19.

20. import com.bumptech.glide.Glide;

21. import com.example.barangku.R;

22. import com.example.barangku.activity.model.ModelBarangMasuk;

23. import com.example.barangku.activity.model.ModelStock;

24. import com.github.dhaval2404.imagepicker.ImagePicker;

25. import com.google.firebase.auth.FirebaseAuth;

26. import com.google.firebase.database.DataSnapshot;

27. import com.google.firebase.database.DatabaseError;

28. import com.google.firebase.database.DatabaseReference;

29. import com.google.firebase.database.FirebaseDatabase;

30. import com.google.firebase.database.ValueEventListener;

31.

32. import java.time.LocalDate;

33. import java.util.ArrayList;

34. import java.util.List;

35.

36. public class BarangMasuk extends AppCompatActivity {

37. private TextView tvToolbar, tvTanggalMasuk, tvNamaBarang, tvSatuan, tvJumlahItem;

38. private AutoCompleteTextView etNamaBarang;

39. private EditText etJumlah, etKeterangan;

40. private Button btnSimpan;

41. private ImageView ivback, ivKalender, ivGambar;

42. private Uri gambarBarang;

43. private String simpan, namaBarang, satuan, keterangan, tanggalMasuk;

44. private int jumlahMasuk = 0;

45. private String userRole;

46. private DatabaseReference reference = FirebaseDatabase.getInstance().getReference("StockBarang");

47. private DatabaseReference barangMasukref = FirebaseDatabase.getInstance().getReference("BarangMasuk");

48. private DatabaseReference userRef = FirebaseDatabase.getInstance().getReference("User");

49. private List<String> namaBarangList = new ArrayList<>();

50. private List<ModelStock> stockList = new ArrayList<>();

51.

52. private ModelStock currentStockItem;

53. private FirebaseAuth mAuth;

54.

55. @Override

56. protected void onCreate(Bundle savedInstanceState) {

57. super.onCreate(savedInstanceState);

58. setContentView(R.layout.activity\_barang\_masuk);

59.

60. tvToolbar = findViewById(R.id.tv\_judul);

61. tvToolbar.setText("Barang Masuk");

62.

63. ivback = findViewById(R.id.iv\_back);

64. etNamaBarang = findViewById(R.id.et\_nama\_barang);

65. etJumlah = findViewById(R.id.et\_jumlah\_barang);

66. etKeterangan = findViewById(R.id.et\_keterangan);

67.

68. tvNamaBarang = findViewById(R.id.tv\_nama\_barang);

69. tvJumlahItem = findViewById(R.id.tv\_jumlah\_item);

70. tvSatuan = findViewById(R.id.tv\_satuan);

71.

72. tvTanggalMasuk = findViewById(R.id.tv\_tanggal\_masuk);

73. ivKalender = findViewById(R.id.iv\_kalender);

74. ivGambar = findViewById(R.id.iv\_gambar\_barang);

75.

76. mAuth = FirebaseAuth.getInstance();

77. String currentUserId = mAuth.getCurrentUser().getUid();

78.

79. LocalDate localDate = LocalDate.now();

80. int yearToday = localDate.getYear();

81. int monthToday = localDate.getMonthValue();

82. int dayOfMonthToday = localDate.getDayOfMonth();

83.

84. tanggalMasuk = String.format("%04d-%02d-%02d", yearToday, monthToday, dayOfMonthToday);

85. tvTanggalMasuk.setText((tanggalMasuk));

86.

87. btnSimpan = findViewById(R.id.btn\_simpan);

88.

89. ivback.setOnClickListener(new View.OnClickListener() {

90. @Override

91. public void onClick(View v) {

92. Intent intent = new Intent(BarangMasuk.this, MainActivity.class);

93. startActivity(intent);

94. }

95. });

96.

97. btnSimpan.setOnClickListener(new View.OnClickListener() {

98. @Override

99. public void onClick(View view) {

100. if (validateFields()) {

101. try {

102. namaBarang = tvNamaBarang.getText().toString();

103. keterangan = etKeterangan.getText().toString();

104. jumlahMasuk = Integer.parseInt(etJumlah.getText().toString());

105. satuan = tvSatuan.getText().toString();

106. } catch (NumberFormatException e) {

107. Toast.makeText(BarangMasuk.this, "Jumlah harus berupa angka", Toast.LENGTH\_SHORT).show();

108. return;

109. }

110.

111. gambarBarang = ivGambar.getDrawable() != null ? gambarBarang : Uri.EMPTY;

112. if (namaBarang.trim().isEmpty() || jumlahMasuk == 0) {

113. Toast.makeText(BarangMasuk.this, "Pastikan semua data terisi dengan benar", Toast.LENGTH\_SHORT).show();

114. } else {

115. simpanBarangMasuk();

116. // Reset fields after saving

117. resetFields();

118. }

119. }

120. }

121. });

122.

123. etNamaBarang.setOnItemClickListener(new AdapterView.OnItemClickListener() {

124. @Override

125. public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

126. String searchNamaBarang = parent.getItemAtPosition(position).toString();

127. cariBarang(searchNamaBarang);

128. }

129. });

130.

131. retrieveStockData();

132. checkUserRole(currentUserId);

133. } // Akhir On Create

134.

135. private boolean validateFields() {

136. if (etNamaBarang.getText().toString().trim().isEmpty()) {

137. etNamaBarang.setError("Nama barang harus diisi");

138. return false;

139. }

140. if (etJumlah.getText().toString().trim().isEmpty()) {

141. etJumlah.setError("Jumlah barang harus diisi");

142. return false;

143. }

144. if (tvNamaBarang.getText().toString().trim().isEmpty()) {

145. Toast.makeText(this, "Nama barang tidak boleh kosong", Toast.LENGTH\_SHORT).show();

146. return false;

147. }

148. if (tvSatuan.getText().toString().trim().isEmpty()) {

149. Toast.makeText(this, "Satuan barang tidak boleh kosong", Toast.LENGTH\_SHORT).show();

150. return false;

151. }

152. return true;

153. }

154.

155. private void resetFields() {

156. etKeterangan.setText("");

157. etNamaBarang.setText("");

158. etJumlah.setText("");

159. tvSatuan.setText("");

160. tvJumlahItem.setText("");

161. ivGambar.setImageDrawable(getResources().getDrawable(R.drawable.insert\_image));

162. }

163.

164. private void checkUserRole(String userId) {

165. userRef.child(userId).addListenerForSingleValueEvent(new ValueEventListener() {

166. @Override

167. public void onDataChange(@NonNull DataSnapshot snapshot) {

168. if (snapshot.exists()) {

169. userRole = snapshot.child("jabatan").getValue(String.class);

170. updateUIBasedOnRole();

171. } else {

172. Toast.makeText(BarangMasuk.this, "Gagal memuat data pengguna", Toast.LENGTH\_SHORT).show();

173. }

174. }

175.

176. @Override

177. public void onCancelled(@NonNull DatabaseError error) {

178. Toast.makeText(BarangMasuk.this, "Gagal memuat data pengguna", Toast.LENGTH\_SHORT).show();

179. }

180. });

181. }

182.

183. private void updateUIBasedOnRole() {

184. if ("Admin".equals(userRole)) {

185. // Admin can see all functionalities

186. } else {

187. // User (non-admin) has limited functionalities

188. ivKalender.setVisibility(View.GONE); // Hide the calendar icon for users

189. }

190. }

191.

192. private void simpanBarangMasuk() {

193. String id = barangMasukref.push().getKey();

194. if (id == null) {

195. Toast.makeText(this, "Gagal mendapatkan ID unik", Toast.LENGTH\_SHORT).show();

196. return;

197. }

198.

199. ModelBarangMasuk bm = new ModelBarangMasuk(id, namaBarang, satuan, keterangan, tanggalMasuk, jumlahMasuk);

200.

201. if ("Admin".equals(userRole)) {

202. // Admin saves directly

203. barangMasukref.child(id).setValue(bm).addOnCompleteListener(task -> {

204. if (task.isSuccessful()) {

205. updateStockBarang(currentStockItem, jumlahMasuk);

206. Toast.makeText(BarangMasuk.this, "Data berhasil disimpan", Toast.LENGTH\_SHORT).show();

207. resetFields();

208. } else {

209. Toast.makeText(BarangMasuk.this, "Gagal menyimpan data", Toast.LENGTH\_SHORT).show();

210. }

211. });

212. } else {

213. // User sends a request for approval

214. kirimPengajuanKeAdmin(bm);

215. }

216. }

217.

218. private void kirimPengajuanKeAdmin(ModelBarangMasuk bm) {

219. DatabaseReference notifRef = FirebaseDatabase.getInstance().getReference("PengajuanBarangMasuk");

220. String notifId = notifRef.push().getKey();

221. if (notifId == null) {

222. Toast.makeText(this, "Gagal mengirim pengajuan", Toast.LENGTH\_SHORT).show();

223. return;

224. }

225. notifRef.child(notifId).setValue(bm).addOnCompleteListener(task -> {

226. if (task.isSuccessful()) {

227. Toast.makeText(BarangMasuk.this, "Pengajuan dikirim ke admin", Toast.LENGTH\_SHORT).show();

228. resetFields();

229. } else {

230. Toast.makeText(BarangMasuk.this, "Gagal mengirim pengajuan ke admin", Toast.LENGTH\_SHORT).show();

231. }

232. });

233. }

234.

235. private void updateStockBarang(ModelStock stockItem, int jumlahMasuk) {

236. int jumlahBaru = Integer.parseInt(String.valueOf(Integer.parseInt(stockItem.getJumlahBarang()) + jumlahMasuk));

237. stockItem.setJumlahBarang(String.valueOf(jumlahBaru));

238.

239. reference.child(stockItem.getId()).setValue(stockItem).addOnCompleteListener(task -> {

240. if (task.isSuccessful()) {

241. Toast.makeText(BarangMasuk.this, "Stok barang berhasil diperbarui", Toast.LENGTH\_SHORT).show();

242. } else {

243. Toast.makeText(BarangMasuk.this, "Gagal memperbarui stok barang", Toast.LENGTH\_SHORT).show();

244. }

245. });

246. }

247.

248. private void cariBarang(String searchNamaBarang) {

249. boolean found = false;

250. for (ModelStock ms : stockList) {

251. if (ms.getNamaBarang().equalsIgnoreCase(searchNamaBarang)) {

252. currentStockItem = ms;

253. tvNamaBarang.setText(ms.getNamaBarang());

254. tvSatuan.setText(ms.getSatuan());

255. tvJumlahItem.setText(String.valueOf(ms.getJumlahBarang()));

256. Glide.with(BarangMasuk.this)

257. .load(ms.getGambar())

258. .into(ivGambar);

259. found = true;

260. break;

261. }

262. }

263. if (!found) {

264. Toast.makeText(BarangMasuk.this, "Barang tidak ditemukan", Toast.LENGTH\_SHORT).show();

265. }

266. }

267.

268. private void retrieveStockData() {

269. reference.orderByChild("enable").equalTo(true).addValueEventListener(new ValueEventListener() {

270. public void onDataChange(@NonNull DataSnapshot snapshot) {

271. namaBarangList.clear();

272. stockList.clear();

273.

274. for (DataSnapshot dataSnapshot : snapshot.getChildren()) {

275. ModelStock ms = dataSnapshot.getValue(ModelStock.class);

276. if (ms != null) {

277. namaBarangList.add(ms.getNamaBarang());

278. stockList.add(ms);

279. }

280. }

281. setupAutoCompleteTextView();

282. }

283.

284. @Override

285. public void onCancelled(@NonNull DatabaseError error) {

286. Toast.makeText(BarangMasuk.this, "Failed to load data", Toast.LENGTH\_SHORT).show();

287. }

288. });

289. }

290.

291. @Override

292. protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {

293. super.onActivityResult(requestCode, resultCode, data);

294. if (data != null && data.getData() != null) {

295. gambarBarang = data.getData();

296. ivGambar.setImageURI(gambarBarang);

297. }

298. }

299.

300. private void setupAutoCompleteTextView() {

301. ArrayAdapter<String> adapterBarang = new ArrayAdapter<>(this, R.layout.list\_nama\_barang, namaBarangList);

302. etNamaBarang.setAdapter(adapterBarang);

303. etNamaBarang.setOnItemClickListener(new AdapterView.OnItemClickListener() {

304. @Override

305. public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

306. Toast.makeText(BarangMasuk.this, parent.getItemAtPosition(position).toString(), Toast.LENGTH\_SHORT).show();

307. cariBarang(parent.getItemAtPosition(position).toString());

308. }

309. });

310. }

311.

312. }

313.

Kode Program Menu BarangKeluar.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import androidx.annotation.NonNull;

4. import androidx.appcompat.app.AppCompatActivity;

5.

6. import android.content.Intent;

7. import android.net.Uri;

8. import android.os.Bundle;

9. import android.util.Log;

10. import android.view.View;

11. import android.widget.ArrayAdapter;

12. import android.widget.AutoCompleteTextView;

13. import android.widget.Button;

14. import android.widget.EditText;

15. import android.widget.ImageView;

16. import android.widget.TextView;

17. import android.widget.Toast;

18.

19. import com.bumptech.glide.Glide;

20. import com.example.barangku.R;

21. import com.example.barangku.activity.model.ModelBarangKeluar;

22. import com.example.barangku.activity.model.ModelClient;

23. import com.example.barangku.activity.model.ModelStock;

24. import com.google.firebase.auth.FirebaseAuth;

25. import com.google.firebase.database.DataSnapshot;

26. import com.google.firebase.database.DatabaseError;

27. import com.google.firebase.database.DatabaseReference;

28. import com.google.firebase.database.FirebaseDatabase;

29. import com.google.firebase.database.ValueEventListener;

30.

31. import java.time.LocalDate;

32. import java.util.ArrayList;

33. import java.util.List;

34.

35. public class BarangKeluar extends AppCompatActivity {

36. private TextView tv\_toolbar, tvTanggalKeluar, tvNamaBarang, tvSatuan, tvJumlahItem, tvAlamatKlien;

37. private AutoCompleteTextView etNamaBarang, etNamaKlien;

38. private EditText etJumlah, etKeterangan;

39. private Button btnSimpan, btnCari;

40. private ImageView ivback, ivKalender, ivGambar;

41. private Uri gambarBarang;

42. private String namaBarang, satuan, keterangan, tanggalKeluar, namaKlien, alamatKlien, userRole;

43. private long jumlahKeluar = 0;

44. private DatabaseReference stockReference, barangKeluarRef, clientReference, userRef;

45. private List<String> namaBarangList, namaClientList;

46. private List<ModelStock> stockList;

47. private List<ModelClient> clientList;

48. private ModelStock currentStockItem;

49.

50. private FirebaseAuth mAuth;

51.

52. @Override

53. protected void onCreate(Bundle savedInstanceState) {

54. super.onCreate(savedInstanceState);

55. setContentView(R.layout.activity\_barang\_keluar);

56.

57.

58. initViews();

59. initFirebaseReferences();

60. setInitialDate();

61. setAdapters();

62. retrieveStockData();

63. retrieveClientData();

64. mAuth = FirebaseAuth.getInstance();

65. String currentUserId = mAuth.getCurrentUser().getUid();

66. checkUserRole(currentUserId); // Ganti dengan user ID yang sesuai

67.

68. ivback.setOnClickListener(v -> navigateToMainActivity());

69. btnCari.setOnClickListener(v -> searchBarang());

70. btnSimpan.setOnClickListener(v -> saveBarangKeluar());

71. }

72.

73. private void initViews() {

74. tv\_toolbar = findViewById(R.id.tv\_judul);

75. tv\_toolbar.setText("Barang Keluar");

76.

77. ivback = findViewById(R.id.iv\_back);

78. etNamaBarang = findViewById(R.id.et\_nama\_barang);

79. etNamaKlien = findViewById(R.id.et\_nama\_klien);

80. etJumlah = findViewById(R.id.et\_jumlah\_barang);

81. etKeterangan = findViewById(R.id.et\_keterangan);

82.

83. tvNamaBarang = findViewById(R.id.tv\_nama\_barang);

84. tvJumlahItem = findViewById(R.id.tv\_jumlah\_item);

85. tvSatuan = findViewById(R.id.tv\_satuan);

86. tvAlamatKlien = findViewById(R.id.tv\_alamat\_client);

87.

88. tvTanggalKeluar = findViewById(R.id.tv\_tanggal\_keluar);

89. ivKalender = findViewById(R.id.iv\_kalender);

90. ivGambar = findViewById(R.id.iv\_gambar\_barang);

91.

92. btnCari = findViewById(R.id.btn\_cari\_barang);

93. btnSimpan = findViewById(R.id.btn\_simpan);

94.

95. namaBarangList = new ArrayList<>();

96. namaClientList = new ArrayList<>();

97. stockList = new ArrayList<>();

98. clientList = new ArrayList<>();

99. }

100.

101. private void initFirebaseReferences() {

102. stockReference = FirebaseDatabase.getInstance().getReference("StockBarang");

103. barangKeluarRef = FirebaseDatabase.getInstance().getReference("BarangKeluar");

104. clientReference = FirebaseDatabase.getInstance().getReference("Client");

105. userRef = FirebaseDatabase.getInstance().getReference("User");

106. }

107.

108. private void setInitialDate() {

109. LocalDate localDate = LocalDate.now();

110. tanggalKeluar = String.format("%04d-%02d-%02d", localDate.getYear(), localDate.getMonthValue(), localDate.getDayOfMonth());

111. tvTanggalKeluar.setText(tanggalKeluar);

112. }

113.

114. private void setAdapters() {

115. ArrayAdapter<String> adapterBarang = new ArrayAdapter<>(this, R.layout.list\_nama\_barang, namaBarangList);

116. etNamaBarang.setAdapter(adapterBarang);

117. etNamaBarang.setOnItemClickListener((parent, view, position, id) -> updateSelectedBarang(parent.getItemAtPosition(position).toString()));

118.

119. ArrayAdapter<String> adapterClient = new ArrayAdapter<>(this, R.layout.list\_nama\_klien, namaClientList);

120. etNamaKlien.setAdapter(adapterClient);

121. etNamaKlien.setOnItemClickListener((parent, view, position, id) -> updateSelectedClient(parent.getItemAtPosition(position).toString()));

122. }

123.

124. private void updateSelectedBarang(String selectedBarang) {

125. Toast.makeText(this, selectedBarang, Toast.LENGTH\_SHORT).show();

126. cariBarang(selectedBarang);

127. }

128.

129. private void updateSelectedClient(String selectedClient) {

130. for (ModelClient client : clientList) {

131. if (client.getNama().equals(selectedClient)) {

132. tvAlamatKlien.setText(client.getAlamat());

133. break;

134. }

135. }

136. }

137.

138. private void navigateToMainActivity() {

139. Intent intent = new Intent(this, MainActivity.class);

140. startActivity(intent);

141. }

142.

143. private void searchBarang() {

144. String searchNamaBarang = etNamaBarang.getText().toString();

145. String searchNamaClient = etNamaKlien.getText().toString();

146. if (!searchNamaBarang.isEmpty() && !searchNamaClient.isEmpty()) {

147. cariBarang(searchNamaBarang);

148. cariKlien(searchNamaClient);

149. } else {

150. Toast.makeText(this, "Masukkan nama barang dan klien", Toast.LENGTH\_SHORT).show();

151. }

152. }

153.

154. private void saveBarangKeluar() {

155. if (validateFields()) {

156. namaBarang = tvNamaBarang.getText().toString();

157. namaKlien = etNamaKlien.getText().toString();

158. keterangan = etKeterangan.getText().toString();

159. try {

160. jumlahKeluar = Long.parseLong(etJumlah.getText().toString());

161. } catch (NumberFormatException e) {

162. Toast.makeText(this, "Jumlah harus berupa angka", Toast.LENGTH\_SHORT).show();

163. return;

164. }

165. satuan = tvSatuan.getText().toString();

166. alamatKlien = tvAlamatKlien.getText().toString();

167. gambarBarang = ivGambar.getDrawable() != null ? gambarBarang : Uri.EMPTY;

168.

169. if (currentStockItem != null) {

170. long stokTersedia = Long.parseLong(currentStockItem.getJumlahBarang());

171. if (jumlahKeluar > stokTersedia) {

172. Toast.makeText(this, "Jumlah barang keluar melebihi stok yang tersedia!!", Toast.LENGTH\_SHORT).show();

173. } else {

174. simpanBarangKeluar();

175. }

176. } else {

177. Toast.makeText(this, "Barang tidak ditemukan!!", Toast.LENGTH\_SHORT).show();

178. }

179. } else {

180. Toast.makeText(this, "Pastikan semua data terisi dengan benar", Toast.LENGTH\_SHORT).show();

181. }

182. }

183.

184. private boolean validateFields() {

185. if (tvNamaBarang.getText().toString().trim().isEmpty() ||

186. etNamaKlien.getText().toString().trim().isEmpty() ||

187. etKeterangan.getText().toString().trim().isEmpty() ||

188. etJumlah.getText().toString().trim().isEmpty() ||

189. tvSatuan.getText().toString().trim().isEmpty() ||

190. tvAlamatKlien.getText().toString().trim().isEmpty()) {

191. return false;

192. }

193. return true;

194. }

195.

196. private void simpanBarangKeluar() {

197. String id = barangKeluarRef.push().getKey();

198. if (id == null) {

199. Toast.makeText(this, "Gagal mendapatkan ID unik", Toast.LENGTH\_SHORT).show();

200. return;

201. }

202.

203. ModelBarangKeluar bk = new ModelBarangKeluar(id, namaBarang, satuan, keterangan, tanggalKeluar, namaKlien, alamatKlien, jumlahKeluar);

204.

205. if ("Admin".equals(userRole)) {

206. barangKeluarRef.child(id).setValue(bk).addOnCompleteListener(task -> {

207. if (task.isSuccessful()) {

208. updateStockBarang(currentStockItem, jumlahKeluar);

209. Toast.makeText(BarangKeluar.this, "Data berhasil disimpan", Toast.LENGTH\_SHORT).show();

210. resetFields();

211. } else {

212. Toast.makeText(BarangKeluar.this, "Gagal menyimpan data", Toast.LENGTH\_SHORT).show();

213. }

214. });

215. } else {

216. // User sends a request for approval

217. kirimPengajuanKeAdmin(bk);

218. }

219. }

220.

221. private void kirimPengajuanKeAdmin(ModelBarangKeluar bk) {

222. DatabaseReference pengajuanRef = FirebaseDatabase.getInstance().getReference("PengajuanBarangKeluar");

223. String pengajuanId = pengajuanRef.push().getKey();

224. if (pengajuanId != null) {

225. pengajuanRef.child(pengajuanId).setValue(bk).addOnCompleteListener(task -> {

226. if (task.isSuccessful()) {

227. Toast.makeText(this, "Pengajuan barang keluar berhasil dikirim", Toast.LENGTH\_SHORT).show();

228. resetFields();

229. } else {

230. Toast.makeText(this, "Pengajuan barang keluar gagal dikirim", Toast.LENGTH\_SHORT).show();

231. }

232. });

233. } else {

234. Toast.makeText(this, "Gagal membuat pengajuan", Toast.LENGTH\_SHORT).show();

235. }

236. }

237.

238. private void updateStockBarang(ModelStock stock, long jumlahKeluar) {

239. long stokBaru = Long.parseLong(stock.getJumlahBarang()) - jumlahKeluar;

240. stockReference.child(stock.getId()).child("jumlahBarang").setValue(String.valueOf(stokBaru));

241. }

242.

243. private void resetFields() {

244. etNamaBarang.setText("");

245. etNamaKlien.setText("");

246. etJumlah.setText("");

247. etKeterangan.setText("");

248. tvNamaBarang.setText("");

249. tvSatuan.setText("");

250. tvJumlahItem.setText("");

251. tvAlamatKlien.setText("");

252. ivGambar.setImageResource(R.drawable.insert\_image);

253. currentStockItem = null;

254. }

255.

256. private void retrieveStockData() {

257. stockReference.orderByChild("enable").equalTo(true).addValueEventListener(new ValueEventListener() {

258. @Override

259. public void onDataChange(@NonNull DataSnapshot snapshot) {

260. namaBarangList.clear();

261. stockList.clear();

262. for (DataSnapshot dataSnapshot : snapshot.getChildren()) {

263. ModelStock stock = dataSnapshot.getValue(ModelStock.class);

264. if (stock != null) {

265. namaBarangList.add(stock.getNamaBarang());

266. stockList.add(stock);

267. }

268. }

269. setAdapters();

270. }

271.

272. @Override

273. public void onCancelled(@NonNull DatabaseError error) {

274. Log.e("BarangKeluar", "Gagal mengambil data stok: " + error.getMessage());

275. }

276. });

277. }

278.

279. private void retrieveClientData() {

280. clientReference.addValueEventListener(new ValueEventListener() {

281. @Override

282. public void onDataChange(@NonNull DataSnapshot snapshot) {

283. namaClientList.clear();

284. clientList.clear();

285. for (DataSnapshot dataSnapshot : snapshot.getChildren()) {

286. ModelClient client = dataSnapshot.getValue(ModelClient.class);

287. if (client != null) {

288. namaClientList.add(client.getNama());

289. clientList.add(client);

290. }

291. }

292. setAdapters();

293. }

294.

295. @Override

296. public void onCancelled(@NonNull DatabaseError error) {

297. Log.e("BarangKeluar", "Gagal mengambil data klien: " + error.getMessage());

298. }

299. });

300. }

301.

302. private void retrieveUserRole() {

303. // Assume you have a way to get the user role, e.g., from Firebase Authentication or Realtime Database

304. // Here is an example using Firebase Realtime Database

305. DatabaseReference userRoleRef = FirebaseDatabase.getInstance().getReference("UserRole");

306. String userId = "currentUserId"; // Replace with the actual current user ID

307. userRoleRef.child(userId).addListenerForSingleValueEvent(new ValueEventListener() {

308. @Override

309. public void onDataChange(@NonNull DataSnapshot snapshot) {

310. if (snapshot.exists()) {

311. userRole = snapshot.getValue(String.class);

312. }

313. }

314.

315. @Override

316. public void onCancelled(@NonNull DatabaseError error) {

317. Log.e("BarangKeluar", "Gagal mendapatkan peran pengguna: " + error.getMessage());

318. }

319. });

320. }

321.

322. private void cariBarang(String namaBarang) {

323. for (ModelStock stock : stockList) {

324. if (stock.getNamaBarang().equals(namaBarang)) {

325. tvNamaBarang.setText(stock.getNamaBarang());

326. tvJumlahItem.setText(stock.getJumlahBarang());

327. tvSatuan.setText(stock.getSatuan());

328. currentStockItem = stock;

329. loadImage(stock.getGambar());

330. break;

331. }

332. }

333. }

334.

335. private void cariKlien(String namaKlien) {

336. for (ModelClient client : clientList) {

337. if (client.getNama().equals(namaKlien)) {

338. tvAlamatKlien.setText(client.getAlamat());

339. break;

340. }

341. }

342. }

343.

344. private void loadImage(String imageUrl) {

345. if (!imageUrl.isEmpty()) {

346. Glide.with(this)

347. .load(imageUrl)

348. .into(ivGambar);

349. } else {

350. ivGambar.setImageResource(R.drawable.insert\_image);

351. }

352. }

353.

354. private void checkUserRole(String userId) {

355. userRef.child(userId).addListenerForSingleValueEvent(new ValueEventListener() {

356. @Override

357. public void onDataChange(@NonNull DataSnapshot snapshot) {

358. if (snapshot.exists()) {

359. userRole = snapshot.child("jabatan").getValue(String.class);

360. updateUIBasedOnRole();

361. } else {

362. Toast.makeText(BarangKeluar.this, "Gagal memuat data pengguna", Toast.LENGTH\_SHORT).show();

363. }

364. }

365. @Override

366. public void onCancelled(@NonNull DatabaseError error) {

367. Toast.makeText(BarangKeluar.this, "Gagal memuat data pengguna", Toast.LENGTH\_SHORT).show();

368. }

369. });

370. }

371.

372. private void updateUIBasedOnRole() {

373. if ("Admin".equals(userRole)) {

374. // Admin can see all functionalities

375. } else {

376. // User (non-admin) has limited functionalities

377. ivKalender.setVisibility(View.GONE); // Hide the calendar icon for users

378. }

379. }

380. }

381.

Kode Program Menu Client.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import androidx.annotation.NonNull;

4. import androidx.appcompat.app.AppCompatActivity;

5. import androidx.appcompat.widget.SearchView;

6. import androidx.recyclerview.widget.LinearLayoutManager;

7. import androidx.recyclerview.widget.RecyclerView;

8.

9. import android.app.Dialog;

10. import android.app.ProgressDialog;

11. import android.content.Intent;

12. import android.os.Bundle;

13. import android.view.View;

14. import android.view.ViewGroup;

15. import android.widget.EditText;

16. import android.widget.ImageView;

17. import android.widget.TextView;

18. import android.widget.Toast;

19.

20. import com.example.barangku.R;

21. import com.example.barangku.activity.adapter.AdapterClient;

22. import com.example.barangku.activity.model.ModelClient;

23. import com.example.barangku.activity.user\_activity.utils.ItemClickClient;

24. import com.google.android.material.floatingactionbutton.FloatingActionButton;

25. import com.google.firebase.database.DataSnapshot;

26. import com.google.firebase.database.DatabaseError;

27. import com.google.firebase.database.DatabaseReference;

28. import com.google.firebase.database.FirebaseDatabase;

29. import com.google.firebase.database.ValueEventListener;

30. import com.google.firebase.storage.FirebaseStorage;

31. import com.google.firebase.storage.StorageReference;

32.

33. import java.util.ArrayList;

34. import java.util.List;

35. import java.util.regex.Pattern;

36.

37. public class Client extends AppCompatActivity implements ItemClickClient {

38. private TextView tvToolbar;

39. private Dialog dialog, dialogClient, dialogEdit;

40. private FloatingActionButton fabTambah;

41. private String namaKlien, alamatKlien, noTelpKlien, emailKlien, catatanKlien;

42. private DatabaseReference reference = FirebaseDatabase.getInstance().getReference("Client");

43. DatabaseReference counterRef = FirebaseDatabase.getInstance().getReference().child("counter").child("clientId");

44. private StorageReference storageReference = FirebaseStorage.getInstance().getReference();

45. private SearchView searchView;

46. private ImageView ivback;

47. private ProgressDialog pd;

48. RecyclerView rvClient;

49. AdapterClient adapterClient;

50. private List<ModelClient> list\_client = new ArrayList<>();

51.

52.

53. @Override

54. protected void onCreate(Bundle savedInstanceState) {

55. super.onCreate(savedInstanceState);

56. setContentView(R.layout.activity\_client);

57.

58. tvToolbar = findViewById(R.id.tv\_judul);

59. tvToolbar.setText("Client");

60.

61. fabTambah = findViewById(R.id.fab\_tambah\_klien);

62. searchView = findViewById(R.id.search);

63. rvClient = findViewById(R.id.rv\_client);

64.

65. ivback = findViewById(R.id.iv\_back);

66. ivback.setOnClickListener(new View.OnClickListener() {

67. @Override

68. public void onClick(View v) {

69. Intent intent = new Intent(Client.this, MainActivity.class);

70. startActivity(intent);

71. }

72. });

73. pd = new ProgressDialog(this);

74. pd.setMessage("Loading");

75. pd.setCancelable(false);

76. pd.show();

77.

78.

79. searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {

80. @Override

81. public boolean onQueryTextSubmit(String query) {

82. filterList(query);

83. searchView.clearFocus();

84. return true;

85. }

86.

87. @Override

88. public boolean onQueryTextChange(String newText) {

89. filterList(newText);

90. return true;

91. }

92. });

93.

94.

95. adapterClient = new AdapterClient(Client.this, list\_client);

96. adapterClient.setItemClickListener(this);

97. LinearLayoutManager lm\_client = new LinearLayoutManager(this, LinearLayoutManager.VERTICAL, false);

98. rvClient.setLayoutManager(lm\_client);

99. rvClient.setAdapter(adapterClient);

100.

101. FirebaseDatabase database = FirebaseDatabase.getInstance();

102. DatabaseReference reference = database.getReference("Client");

103. reference.addValueEventListener(new ValueEventListener() {

104. @Override

105. public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

106. for (DataSnapshot snapshot : dataSnapshot.getChildren()) {

107. ModelClient mc = snapshot.getValue(ModelClient.class);

108. list\_client.add(mc);

109. }

110. adapterClient.notifyDataSetChanged();

111. pd.dismiss();

112. }

113.

114. @Override

115. public void onCancelled(@NonNull DatabaseError databaseError) {

116. }

117. });

118.

119.

120.

121.

122. fabTambah.setOnClickListener(new View.OnClickListener() {

123. @Override

124. public void onClick(View view) {

125. dialog = new Dialog(Client.this);

126. dialog.setContentView(R.layout.dialog\_tambah\_client);

127. dialog.getWindow().setLayout(ViewGroup.LayoutParams.WRAP\_CONTENT, ViewGroup.LayoutParams.WRAP\_CONTENT);

128. dialog.setCancelable(false);

129. dialog.show();

130.

131. EditText etNamaKlien = dialog.findViewById(R.id.et\_nama\_klien);

132. EditText etAlamatKlien = dialog.findViewById(R.id.et\_alamat\_klien);

133. EditText etNoTelpKlien = dialog.findViewById(R.id.et\_telp\_klien);

134. EditText etEmailKlien = dialog.findViewById(R.id.et\_email\_klien);

135. EditText etCatatanKlien = dialog.findViewById(R.id.et\_catatan\_klien);

136.

137.

138. dialog.findViewById(R.id.iv\_close).setOnClickListener(new View.OnClickListener() {

139. @Override

140. public void onClick(View view) {

141. dialog.dismiss();

142. }

143. });

144. dialog.findViewById(R.id.btn\_simpan).setOnClickListener(new View.OnClickListener() {

145. @Override

146. public void onClick(View view) {

147. namaKlien = etNamaKlien.getText().toString();

148. alamatKlien = etAlamatKlien.getText().toString();

149. noTelpKlien = etNoTelpKlien.getText().toString();

150. emailKlien = etEmailKlien.getText().toString();

151. catatanKlien = etCatatanKlien.getText().toString();

152.

153. if (namaKlien.trim().isEmpty()) {

154. etNamaKlien.setError("Required");

155. return;

156. } else if (alamatKlien.trim().isEmpty()) {

157. etAlamatKlien.setError("Required");

158. return;

159. } else if (noTelpKlien.trim().isEmpty()) {

160. etNoTelpKlien.setError("Required");

161. return;

162. } else if (noTelpKlien.length() > 13) {

163. etNoTelpKlien.setError("Nomor Telpon Tidak Boleh Melebihi 13 digit");

164. return;

165. } else if (emailKlien.trim().isEmpty()) {

166. etEmailKlien.setError("Required");

167. return;

168. }

169. else if (!isValidEmail(emailKlien)){

170. etEmailKlien.setError("Format Email Tidak Didukung");

171. return;

172. }

173. else if (catatanKlien.trim().isEmpty()) {

174. etCatatanKlien.setError("Required");

175. return;

176. } else {

177. TambahClient(namaKlien, alamatKlien,noTelpKlien, emailKlien, catatanKlien);

178. }

179.

180. }

181. });

182. }

183. });

184. }

185.

186. private void TambahClient(String namaKlien, String alamatKlien, String noTelpKlien, String emailKlien, String catatanKlien) {

187. counterRef.addListenerForSingleValueEvent(new ValueEventListener() {

188. @Override

189. public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

190. long currentId = dataSnapshot.exists() ? dataSnapshot.getValue(Long.class) : 0;

191. long newId = currentId + 1;

192.

193. String clientId = String.valueOf(newId);

194. ModelClient mc = new ModelClient(clientId, namaKlien, alamatKlien, noTelpKlien, emailKlien, catatanKlien);

195. reference.child(clientId).setValue(mc);

196. Toast.makeText(Client.this, "Nama Klien Berhasil Disimpan", Toast.LENGTH\_SHORT).show();

197. dialog.dismiss();

198.

199. counterRef.setValue(newId);

200. refreshClientList();

201. }

202.

203. @Override

204. public void onCancelled(@NonNull DatabaseError error) {

205.

206. }

207. });

208. }

209.

210. private void refreshClientList() {

211. reference.addListenerForSingleValueEvent(new ValueEventListener() {

212. @Override

213. public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

214. list\_client.clear();

215. for (DataSnapshot snapshot : dataSnapshot.getChildren()) {

216. ModelClient mc = snapshot.getValue(ModelClient.class);

217. list\_client.add(mc);

218. }

219. adapterClient.notifyDataSetChanged();

220. pd.dismiss();

221. }

222.

223. @Override

224. public void onCancelled(@NonNull DatabaseError databaseError) {

225. // Handle database errors (optional)

226. }

227. });

228. }

229.

230. private boolean isValidEmail(String emailKlien) {

231. String regex = "^[\\w-\_\\.+]+@[\\w-]+\\.[\\w-]{2,}$";

232. return Pattern.matches(regex, emailKlien);

233. }

234.

235.

236. private void filterList(String query) {

237. List<ModelClient> filteredList = new ArrayList<>();

238. String queryInLower = query.toLowerCase();

239. for (ModelClient mc : list\_client) {

240. if (mc.getNama().toLowerCase().contains(queryInLower)) {

241. filteredList.add(mc);

242. }

243. }

244. if (filteredList.isEmpty()) {

245. Toast.makeText(this, "No Clients Found", Toast.LENGTH\_SHORT).show();

246. } else {

247. adapterClient.setFilter(filteredList);

248. }

249. }

250.

251. @Override

252. protected void onStart() {

253. super.onStart();

254. }

255.

256. @Override

257. protected void onStop() {

258. super.onStop();

259. }

260.

261. @Override

262. public void onItemClickListener(ModelClient data, int position) {

263. dialogClient = new Dialog(Client.this);

264. dialogClient.setContentView(R.layout.dialog\_tampil\_client);

265. dialogClient.getWindow().setLayout(ViewGroup.LayoutParams.WRAP\_CONTENT, ViewGroup.LayoutParams.WRAP\_CONTENT);

266. dialogClient.setCancelable(false);

267. dialogClient.show();

268.

269. TextView tvNamaKlien = dialogClient.findViewById(R.id.tv\_nama\_klien);

270. tvNamaKlien.setText(data.getNama());

271. TextView tvAlamatKlien = dialogClient.findViewById(R.id.tv\_alamat\_klien);

272. tvAlamatKlien.setText(data.getAlamat());

273. TextView tvTelpKlien = dialogClient.findViewById(R.id.tv\_telp\_klien);

274. tvTelpKlien.setText(data.getNo\_telp());

275. TextView tvEmailKlien = dialogClient.findViewById(R.id.tv\_email\_klien);

276. tvEmailKlien.setText(data.getEmail());

277. TextView tvCatatanKlien = dialogClient.findViewById(R.id.tv\_catatan\_klien);

278. tvCatatanKlien.setText(data.getCatatan());

279. dialogClient.findViewById(R.id.iv\_close).setOnClickListener(view -> {dialogClient.dismiss();});

280.

281. dialogClient.findViewById(R.id.btn\_edit\_klien).setOnClickListener(new View.OnClickListener() {

282. @Override

283. public void onClick(View view) {

284. dialogEdit = new Dialog(Client.this);

285. dialogEdit.setContentView(R.layout.dialog\_edit\_client);

286. dialogEdit.getWindow().setLayout(ViewGroup.LayoutParams.WRAP\_CONTENT, ViewGroup.LayoutParams.WRAP\_CONTENT);

287. dialogEdit.setCancelable(false);

288. dialogEdit.show();

289.

290. EditText etNamaKlien = dialogEdit.findViewById(R.id.et\_nama\_klien);

291. etNamaKlien.setText(data.getNama());

292. EditText etAlamatKlien = dialogEdit.findViewById(R.id.et\_alamat\_klien);

293. etAlamatKlien.setText(data.getAlamat());

294. EditText etNoTelpKlien = dialogEdit.findViewById(R.id.et\_telp\_klien);

295. etNoTelpKlien.setText(data.getNo\_telp());

296. EditText etEmailKlien = dialogEdit.findViewById(R.id.et\_email\_klien);

297. etEmailKlien.setText(data.getEmail());

298. EditText etCatatanKlien = dialogEdit.findViewById(R.id.et\_catatan\_klien);

299. etCatatanKlien.setText(data.getCatatan());

300. dialogEdit.findViewById(R.id.iv\_close).setOnClickListener(new View.OnClickListener() {

301. @Override

302. public void onClick(View view) {

303. dialogEdit.dismiss();

304. }

305. });

306.

307. dialogEdit.findViewById(R.id.btn\_simpan\_edit).setOnClickListener(new View.OnClickListener() {

308. @Override

309. public void onClick(View view) {

310. namaKlien = etNamaKlien.getText().toString();

311. alamatKlien = etAlamatKlien.getText().toString();

312. noTelpKlien = etNoTelpKlien.getText().toString();

313. emailKlien = etEmailKlien.getText().toString();

314. catatanKlien = etCatatanKlien.getText().toString();

315.

316. if (namaKlien.trim().isEmpty()) {

317. etNamaKlien.setError("Required");

318. } else if (alamatKlien.trim().isEmpty()) {

319. etAlamatKlien.setError("Required");

320. } else if (noTelpKlien.trim().isEmpty()) {

321. etNoTelpKlien.setError("Required");

322. } else if (noTelpKlien.length() > 13) {

323. etNoTelpKlien.setError("Nomor Telpon Tidak Boleh Melebihi 13 digit");

324. } else if (emailKlien.trim().isEmpty()) {

325. etEmailKlien.setError("Required");

326. } else if (!isValidEmail(emailKlien)){

327. etEmailKlien.setError("Format Email Tidak Didukung");

328. } else if (catatanKlien.trim().isEmpty()) {

329. etCatatanKlien.setError("Required");

330. } else {

331. updateClient(data.getId(), namaKlien, alamatKlien, noTelpKlien, emailKlien, catatanKlien);

332. dialogEdit.dismiss();

333. dialogClient.dismiss();

334. }

335.

336. }

337. });

338.

339.

340. }

341. });

342.

343. dialogClient.findViewById(R.id.btn\_hapus\_klien).setOnClickListener(new View.OnClickListener() {

344. @Override

345. public void onClick(View view) {

346. String clientId = data.getId();

347. reference.child(clientId).removeValue();

348. Toast.makeText(Client.this, "Data Klien Telah Dihapus", Toast.LENGTH\_SHORT).show();

349. refreshClientList();

350. dialogClient.dismiss();

351. }

352. });

353. }

354. private void updateClient(String id, String nama, String alamat, String noTelp, String email, String catatan) {

355. ModelClient mc = new ModelClient(id, nama, alamat, noTelp, email, catatan);

356. reference.child(id).setValue(mc);

357. Toast.makeText(Client.this, "Data Klien Berhasil Diubah", Toast.LENGTH\_SHORT).show();

358. refreshClientList();

359. }

360. }

Kode Program Menu KelolaUser.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import android.content.DialogInterface;

4. import android.content.Intent;

5. import android.os.Bundle;

6. import android.view.View;

7. import android.widget.ImageView;

8. import android.widget.TextView;

9. import android.widget.Toast;

10.

11. import androidx.annotation.NonNull;

12. import androidx.appcompat.app.AlertDialog;

13. import androidx.appcompat.app.AppCompatActivity;

14. import androidx.recyclerview.widget.LinearLayoutManager;

15. import androidx.recyclerview.widget.RecyclerView;

16.

17. import com.example.barangku.R;

18.

19. import com.example.barangku.activity.adapter.AdapterKaryawan;

20. import com.example.barangku.activity.model.ModelUser;

21. import com.google.android.gms.tasks.OnCompleteListener;

22. import com.google.android.gms.tasks.Task;

23. import com.google.firebase.database.DataSnapshot;

24. import com.google.firebase.database.DatabaseError;

25. import com.google.firebase.database.DatabaseReference;

26. import com.google.firebase.database.FirebaseDatabase;

27. import com.google.firebase.database.ValueEventListener;

28.

29. import java.util.ArrayList;

30. import java.util.List;

31.

32. public class KelolaUser extends AppCompatActivity {

33. private TextView tvToolbar;

34. private ImageView ivBack;

35. private RecyclerView rvKaryawan;

36. private DatabaseReference karyawanRef = FirebaseDatabase.getInstance().getReference("User");

37. private List<ModelUser> listKaryawan;

38. AdapterKaryawan adapter;

39. @Override

40. protected void onCreate(Bundle savedInstanceState) {

41. super.onCreate(savedInstanceState);

42. setContentView(R.layout.activity\_karyawan);

43. tvToolbar = findViewById(R.id.tv\_judul);

44. tvToolbar.setText("Kelola User");

45.

46. ivBack =findViewById(R.id.iv\_back);

47. ivBack.setOnClickListener(new View.OnClickListener() {

48. @Override

49. public void onClick(View view) {

50. Intent intent = new Intent(KelolaUser.this, Admin.class);

51. startActivity(intent);

52. }

53. });

54.

55. rvKaryawan = findViewById(R.id.rv\_karyawan);

56. rvKaryawan.setHasFixedSize(true);

57. rvKaryawan.setLayoutManager(new LinearLayoutManager(this));

58.

59. listKaryawan = new ArrayList<>();

60. adapter = new AdapterKaryawan(this, listKaryawan);

61. rvKaryawan.setAdapter(adapter);

62.

63. karyawanRef.orderByChild("jabatan").addValueEventListener(new ValueEventListener() {

64. @Override

65. public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

66. listKaryawan.clear();

67. for (DataSnapshot postSnapshot : dataSnapshot.getChildren()) {

68. ModelUser user = postSnapshot.getValue(ModelUser.class);

69. listKaryawan.add(user);

70. }

71. adapter.notifyDataSetChanged();

72. }

73.

74. @Override

75. public void onCancelled(@NonNull DatabaseError databaseError) {

76. Toast.makeText(KelolaUser.this, "Failed to load employees", Toast.LENGTH\_SHORT).show();

77. }

78. });

79. adapter.setOnItemDeleteListener(new AdapterKaryawan.OnItemDeleteListener() {

80. @Override

81. public void onItemDelete(int position) {

82. showDeleteConfirmationDialog(position);

83. }

84. });

85. }

86.

87. private void showDeleteConfirmationDialog(int position) {

88. new AlertDialog.Builder(this)

89. .setTitle("Konfirmasi Hapus")

90. .setMessage("Apakah Anda yakin ingin menghapus data User ini?")

91. .setPositiveButton("Hapus", new DialogInterface.OnClickListener() {

92. @Override

93. public void onClick(DialogInterface dialogInterface, int i) {

94. deleteKaryawan(position);

95. }

96. })

97. .setNegativeButton("Batal", null)

98. .show();

99. }

100.

101. private void deleteKaryawan(int position) {

102. ModelUser user = listKaryawan.get(position);

103. DatabaseReference userRef = karyawanRef.child(user.getId());

104. userRef.removeValue().addOnCompleteListener(new OnCompleteListener<Void>() {

105. @Override

106. public void onComplete(@NonNull Task<Void> task) {

107. if (task.isSuccessful()) {

108. Toast.makeText(KelolaUser.this, "User berhasil dihapus", Toast.LENGTH\_SHORT).show();

109. } else {

110. Toast.makeText(KelolaUser.this, "Gagal menghapus data user", Toast.LENGTH\_SHORT).show();

111. }

112. }

113. });

114. }

115. }

116.

Kode Program Menu Pengajuan.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import androidx.appcompat.app.AppCompatActivity;

4. import androidx.fragment.app.Fragment;

5. import androidx.fragment.app.FragmentManager;

6. import androidx.fragment.app.FragmentTransaction;

7.

8. import android.os.Bundle;

9.

10. import com.example.barangku.R;

11. import com.google.android.material.bottomnavigation.BottomNavigationView;

12.

13. public class Pengajuan extends AppCompatActivity {

14. @Override

15. protected void onCreate(Bundle savedInstanceState) {

16. super.onCreate(savedInstanceState);

17. setContentView(R.layout.activity\_pengajuan);

18. gantiFragment(new PengajuanBarangMasukFragment());

19. BottomNavigationView navigationView = findViewById(R.id.navigation\_bar);

20. navigationView.setOnItemSelectedListener(item -> {

21. if (item.getItemId() == R.id.nav\_barang\_masuk){

22. gantiFragment(new PengajuanBarangMasukFragment());

23. }else if(item.getItemId() == R.id.nav\_barang\_keluar){

24. gantiFragment(new PengajuanBarangKeluarFragment());

25. }

26. return true;

27. });

28. }

29. private void gantiFragment(Fragment fragment){

30. FragmentManager fragmentManager = getSupportFragmentManager();

31. FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();

32. fragmentTransaction.replace(R.id.framelayout, fragment);

33. fragmentTransaction.commit();

34. }

35. }

36.

Kode Program Halaman PengajuanBarangKeluarFragment.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import android.app.AlertDialog;

4. import android.content.DialogInterface;

5. import android.content.Intent;

6. import android.os.Bundle;

7. import androidx.annotation.NonNull;

8. import androidx.annotation.Nullable;

9. import androidx.fragment.app.Fragment;

10. import androidx.recyclerview.widget.LinearLayoutManager;

11. import androidx.recyclerview.widget.RecyclerView;

12. import android.view.LayoutInflater;

13. import android.view.View;

14. import android.view.ViewGroup;

15. import android.widget.ImageView;

16. import android.widget.TextView;

17. import android.widget.Toast;

18. import com.example.barangku.R;

19. import com.example.barangku.activity.adapter.AdapterPengajuanBarangKeluar;

20. import com.example.barangku.activity.model.ModelPengajuanBarangKeluar;

21. import com.example.barangku.activity.model.ModelBarangKeluar;

22. import com.example.barangku.activity.model.ModelStock;

23. import com.example.barangku.activity.user\_activity.utils.ItemClickPengajuanBarangKeluar;

24. import com.google.firebase.database.DataSnapshot;

25. import com.google.firebase.database.DatabaseError;

26. import com.google.firebase.database.DatabaseReference;

27. import com.google.firebase.database.FirebaseDatabase;

28. import com.google.firebase.database.ValueEventListener;

29.

30. import java.text.ParseException;

31. import java.text.SimpleDateFormat;

32. import java.util.ArrayList;

33. import java.util.Date;

34. import java.util.List;

35. import java.util.Locale;

36.

37. public class PengajuanBarangKeluarFragment extends Fragment implements ItemClickPengajuanBarangKeluar {

38. private TextView tvToolbar;

39. private ImageView ivBack;

40.

41. private RecyclerView rvPengajuanBarangKeluar;

42. private AdapterPengajuanBarangKeluar adapter;

43. private List<ModelPengajuanBarangKeluar> listPengajuan = new ArrayList<>();

44.

45. private DatabaseReference pengajuanRef = FirebaseDatabase.getInstance().getReference("PengajuanBarangKeluar");

46. private DatabaseReference riwayatPengajuanRef = FirebaseDatabase.getInstance().getReference("RiwayatPengajuanBarangKeluar");

47. DatabaseReference barangKeluarRef = FirebaseDatabase.getInstance().getReference("BarangKeluar");

48. DatabaseReference stokBarangRef = FirebaseDatabase.getInstance().getReference("StockBarang");

49.

50. public PengajuanBarangKeluarFragment() {

51. // Required empty public constructor

52. }

53.

54. @Override

55. public void onCreate(Bundle savedInstanceState) {

56. super.onCreate(savedInstanceState);

57. }

58.

59. @Override

60. public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

61. // Inflate the layout for this fragment

62. return inflater.inflate(R.layout.fragment\_pengajuan\_barang\_keluar, container, false);

63. }

64.

65. @Override

66. public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {

67. super.onViewCreated(view, savedInstanceState);

68.

69. tvToolbar = view.findViewById(R.id.tv\_judul);

70. tvToolbar.setText("Pengajuan");

71.

72. ivBack = view.findViewById(R.id.iv\_back);

73. ivBack.setOnClickListener(new View.OnClickListener() {

74. @Override

75. public void onClick(View view) {

76. Intent intent = new Intent(getActivity(), MainActivity.class ) ;

77. startActivity(intent);

78. }

79. });

80. rvPengajuanBarangKeluar = view.findViewById(R.id.rv\_pengajuan);

81.

82. adapter = new AdapterPengajuanBarangKeluar(listPengajuan, getView().getContext());

83. adapter.setItemClickPengajuanBarangKeluar(this);

84. LinearLayoutManager lmPengajuanBk = new LinearLayoutManager(getView().getContext(), LinearLayoutManager.VERTICAL, false);

85. rvPengajuanBarangKeluar.setLayoutManager(lmPengajuanBk);

86. rvPengajuanBarangKeluar.setAdapter(adapter);

87.

88. loadPengajuanData();

89. }

90.

91. private void loadPengajuanData() {

92. pengajuanRef.addValueEventListener(new ValueEventListener() {

93. @Override

94. public void onDataChange(@NonNull DataSnapshot snapshot) {

95. listPengajuan.clear();

96.

97. for (DataSnapshot dataSnapshot : snapshot.getChildren()) {

98. String id = dataSnapshot.getKey();

99. String namaBarang = dataSnapshot.child("namaBarang").getValue(String.class);

100. String satuan = dataSnapshot.child("satuan").getValue(String.class);

101. String keterangan = dataSnapshot.child("keterangan").getValue(String.class);

102. String tanggalKeluar = dataSnapshot.child("tanggalKeluar").getValue(String.class);

103. String formattedTanggalKeluar = formatTanggal(tanggalKeluar);

104. String namaKlien = dataSnapshot.child("namaKlien").getValue(String.class);

105. String alamatKlien = dataSnapshot.child("alamatKlien").getValue(String.class);

106. String status = dataSnapshot.child("status").getValue(String.class);

107. Long jumlahBarang = dataSnapshot.child("jumlahBarang").getValue(Long.class);

108.

109. ModelPengajuanBarangKeluar mpbk = new ModelPengajuanBarangKeluar(id, namaBarang, satuan, keterangan, formattedTanggalKeluar, namaKlien, alamatKlien, status,jumlahBarang);

110.

111. if (mpbk != null) {

112. listPengajuan.add(mpbk);

113. }

114. }

115. adapter.notifyDataSetChanged();

116. }

117.

118. @Override

119. public void onCancelled(@NonNull DatabaseError error) {

120. Toast.makeText(requireActivity(), "Gagal memuat data pengajuan", Toast.LENGTH\_SHORT).show();

121. }

122. });

123. }

124. private String formatTanggal(String tanggal) {

125. SimpleDateFormat sdfSource = new SimpleDateFormat("yyyy-MM-dd", new Locale("id", "ID"));

126. SimpleDateFormat sdfDestination = new SimpleDateFormat("EEEE, dd MMMM yyyy",new Locale("id", "ID"));

127. try {

128. Date date = sdfSource.parse(tanggal);

129. return sdfDestination.format(date);

130. } catch (ParseException e) {

131. e.printStackTrace();

132. return tanggal;

133. }

134. }

135. private void approvePengajuan(ModelPengajuanBarangKeluar pengajuan, int position) {

136. String barangKeluarId = barangKeluarRef.push().getKey();

137. ModelBarangKeluar barangKeluarBaru = new ModelBarangKeluar(barangKeluarId, pengajuan.getNamaBarang(), pengajuan.getSatuan(), pengajuan.getKeterangan(), pengajuan.getTanggalKeluar() , pengajuan.getNamaKlien(), pengajuan.getAlamatKlien(), pengajuan.getJumlahBarang());

138. barangKeluarRef.child(barangKeluarId).setValue(barangKeluarBaru).addOnCompleteListener(addTask -> {

139. if (addTask.isSuccessful()) {

140. perbaruiStokBarang(pengajuan);

141. pengajuan.setStatus("Diterima");

142. simpanRiwayatPengajuan(pengajuan);

143. hapusPengajuanSetelahDisetujui(pengajuan, position);

144. } else {

145. Toast.makeText(requireActivity(), "Gagal menambahkan data baru ke BarangKeluar", Toast.LENGTH\_SHORT).show();

146. }

147. });

148. }

149.

150. private void rejectPengajuan(ModelPengajuanBarangKeluar pengajuan) {

151. pengajuan.setStatus("Ditolak");

152. simpanRiwayatPengajuan(pengajuan);

153. pengajuanRef.child(pengajuan.getId()).removeValue().addOnCompleteListener(removeTask -> {

154. if (removeTask.isSuccessful()) {

155. Toast.makeText(requireActivity(), "Pengajuan berhasil ditolak dan disimpan dalam riwayat", Toast.LENGTH\_SHORT).show();

156. } else {

157. Toast.makeText(requireActivity(), "Gagal menghapus pengajuan", Toast.LENGTH\_SHORT).show();

158. }

159. });

160. }

161.

162. private void perbaruiStokBarang(ModelPengajuanBarangKeluar pengajuan) {

163. stokBarangRef.orderByChild("namaBarang").equalTo(pengajuan.getNamaBarang()).addListenerForSingleValueEvent(new ValueEventListener() {

164. @Override

165. public void onDataChange(@NonNull DataSnapshot snapshot) {

166. boolean stokDitemukan = false;

167. for (DataSnapshot dataSnapshot : snapshot.getChildren()) {

168. ModelStock stockItem = dataSnapshot.getValue(ModelStock.class);

169. if (stockItem != null) {

170. stokDitemukan = true;

171. long currentStok = Long.parseLong(stockItem.getJumlahBarang());

172. long updatedStok = currentStok - pengajuan.getJumlahBarang();

173. stockItem.setJumlahBarang(String.valueOf(updatedStok));

174.

175. dataSnapshot.getRef().setValue(stockItem).addOnCompleteListener(updateTask -> {

176. if (updateTask.isSuccessful()) {

177. Toast.makeText(requireActivity(), "Pengajuan disetujui dan Jumlah stok diperbarui", Toast.LENGTH\_SHORT).show();

178. } else {

179. Toast.makeText(requireActivity(), "Gagal memperbarui stok barang", Toast.LENGTH\_SHORT).show();

180. }

181. });

182. break;

183. }

184. }

185. if (!stokDitemukan) {

186. Toast.makeText(requireActivity(), "Stok barang tidak ditemukan", Toast.LENGTH\_SHORT).show();

187. }

188. }

189.

190. @Override

191. public void onCancelled(@NonNull DatabaseError error) {

192. Toast.makeText(requireActivity(), "Gagal memperbarui stok barang", Toast.LENGTH\_SHORT).show();

193. }

194. });

195. }

196.

197. private void simpanRiwayatPengajuan(ModelPengajuanBarangKeluar pengajuan) {

198. String riwayatId = riwayatPengajuanRef.push().getKey();

199. riwayatPengajuanRef.child(riwayatId).setValue(pengajuan).addOnCompleteListener(task -> {

200. if (task.isSuccessful()) {

201. Toast.makeText(requireActivity(), "Pengajuan " + pengajuan.getStatus() + " dan disimpan dalam riwayat", Toast.LENGTH\_SHORT).show();

202. } else {

203. Toast.makeText(requireActivity(), "Gagal menyimpan pengajuan ke dalam riwayat", Toast.LENGTH\_SHORT).show();

204. }

205. });

206. }

207.

208. private void hapusPengajuanSetelahDisetujui(ModelPengajuanBarangKeluar pengajuan, int position) {

209. pengajuanRef.child(pengajuan.getId()).removeValue().addOnCompleteListener(removeTask -> {

210. if (!removeTask.isSuccessful()) {

211. Toast.makeText(requireActivity(), "Gagal menghapus pengajuan setelah disetujui", Toast.LENGTH\_SHORT).show();

212. } else {

213. if (position >= 0 && position < listPengajuan.size()) {

214. listPengajuan.remove(position);

215. adapter.notifyItemRemoved(position);

216. }

217. }

218. });

219. }

220.

221. @Override

222. public void onItemClickListener(ModelPengajuanBarangKeluar data, int position) {

223. if (position >= 0 && position < listPengajuan.size()) {

224. AlertDialog.Builder builder = new AlertDialog.Builder(requireActivity());

225. builder.setTitle("Pengajuan Barang Keluar")

226. .setMessage("Apakah Anda ingin menerima atau menolak pengajuan ini?")

227. .setPositiveButton("Terima", new DialogInterface.OnClickListener() {

228. @Override

229. public void onClick(DialogInterface dialog, int which) {

230. approvePengajuan(data, position);

231. }

232. })

233. .setNegativeButton("Tolak", new DialogInterface.OnClickListener() {

234. @Override

235. public void onClick(DialogInterface dialog, int which) {

236. rejectPengajuan(data);

237. }

238. })

239. .setNeutralButton("Batal", null)

240. .show();

241. } else {

242. Toast.makeText(requireActivity(), "Pengajuan tidak valid", Toast.LENGTH\_SHORT).show();

243. }

244. }

245. }

246.

Kode Program Halaman PengajuanBarangMasukFragment.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import android.app.AlertDialog;

4. import android.content.DialogInterface;

5. import android.content.Intent;

6. import android.os.Bundle;

7. import androidx.annotation.NonNull;

8. import androidx.annotation.Nullable;

9. import androidx.fragment.app.Fragment;

10. import androidx.recyclerview.widget.LinearLayoutManager;

11. import androidx.recyclerview.widget.RecyclerView;

12. import android.view.LayoutInflater;

13. import android.view.View;

14. import android.view.ViewGroup;

15. import android.widget.ImageView;

16. import android.widget.TextView;

17. import android.widget.Toast;

18. import com.example.barangku.R;

19. import com.example.barangku.activity.adapter.AdapterPengajuanBarangMasuk;

20. import com.example.barangku.activity.model.ModelBarangMasuk;

21. import com.example.barangku.activity.model.ModelPengajuanBarangMasuk;

22. import com.example.barangku.activity.model.ModelStock;

23. import com.example.barangku.activity.user\_activity.laporan.Laporan;

24. import com.example.barangku.activity.user\_activity.utils.ItemClickPengajuanBarangMasuk;

25. import com.google.firebase.database.DataSnapshot;

26. import com.google.firebase.database.DatabaseError;

27. import com.google.firebase.database.DatabaseReference;

28. import com.google.firebase.database.FirebaseDatabase;

29. import com.google.firebase.database.ValueEventListener;

30.

31. import java.text.ParseException;

32. import java.text.SimpleDateFormat;

33. import java.util.ArrayList;

34. import java.util.Date;

35. import java.util.List;

36. import java.util.Locale;

37.

38. public class PengajuanBarangMasukFragment extends Fragment implements ItemClickPengajuanBarangMasuk {

39. private TextView tvToolbar;

40. private ImageView ivBack;

41. private RecyclerView rvPengajuanBarangMasuk;

42. private AdapterPengajuanBarangMasuk adapter;

43. private List<ModelPengajuanBarangMasuk> pengajuanList = new ArrayList<>();

44. private DatabaseReference pengajuanRef = FirebaseDatabase.getInstance().getReference("PengajuanBarangMasuk");

45. private DatabaseReference riwayatPengajuanRef = FirebaseDatabase.getInstance().getReference("RiwayatPengajuanBarangMasuk");

46. DatabaseReference barangMasukRef = FirebaseDatabase.getInstance().getReference("BarangMasuk");

47. DatabaseReference stokBarangRef = FirebaseDatabase.getInstance().getReference("StockBarang");

48.

49. public PengajuanBarangMasukFragment() {

50. // Required empty public constructor

51. }

52.

53. @Override

54. public void onCreate(Bundle savedInstanceState) {

55. super.onCreate(savedInstanceState);

56. }

57.

58. @Override

59. public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

60. // Inflate the layout for this fragment

61. return inflater.inflate(R.layout.fragment\_pengajuan\_barang\_masuk, container, false);

62. }

63.

64. @Override

65. public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {

66. super.onViewCreated(view, savedInstanceState);

67.

68. tvToolbar = view.findViewById(R.id.tv\_judul);

69. tvToolbar.setText("Pengajuan");

70.

71. ivBack = view.findViewById(R.id.iv\_back);

72. ivBack.setOnClickListener(new View.OnClickListener() {

73. @Override

74. public void onClick(View view) {

75. Intent intent = new Intent(getActivity(), MainActivity.class ) ;

76. startActivity(intent);

77. }

78. });

79.

80. rvPengajuanBarangMasuk = view.findViewById(R.id.rv\_pengajuan);

81.

82. adapter = new AdapterPengajuanBarangMasuk(pengajuanList, getView().getContext());

83. adapter.setItemClickPengajuanBarangMasuk(this);

84. LinearLayoutManager lmPengajuanBm = new LinearLayoutManager(getView().getContext(), LinearLayoutManager.VERTICAL, false);

85. rvPengajuanBarangMasuk.setLayoutManager(lmPengajuanBm);

86. rvPengajuanBarangMasuk.setAdapter(adapter);

87.

88. loadPengajuanData();

89. }

90.

91. private void loadPengajuanData() {

92. pengajuanRef.addValueEventListener(new ValueEventListener() {

93. @Override

94. public void onDataChange(@NonNull DataSnapshot snapshot) {

95. pengajuanList.clear();

96.

97. for (DataSnapshot dataSnapshot : snapshot.getChildren()) {

98. String id = dataSnapshot.getKey();

99. String namaBarang = dataSnapshot.child("namaBarang").getValue(String.class);

100. String satuan = dataSnapshot.child("satuan").getValue(String.class);

101. String keterangan = dataSnapshot.child("keterangan").getValue(String.class);

102. String tanggalMasuk = dataSnapshot.child("tanggalMasuk").getValue(String.class);

103. String formattedTanggalMasuk = formatTanggal(tanggalMasuk);

104. String status = dataSnapshot.child("status").getValue(String.class);

105. int jumlahBarang = dataSnapshot.child("jumlahBarang").getValue(Integer.class);

106.

107. ModelPengajuanBarangMasuk mpbm = new ModelPengajuanBarangMasuk(id, namaBarang, satuan, keterangan, formattedTanggalMasuk, status, jumlahBarang);

108.

109. if (mpbm != null) {

110. pengajuanList.add(mpbm);

111. }

112. }

113. adapter.notifyDataSetChanged();

114. }

115.

116. @Override

117. public void onCancelled(@NonNull DatabaseError error) {

118. Toast.makeText(requireActivity(), "Gagal memuat data pengajuan", Toast.LENGTH\_SHORT).show();

119. }

120. });

121. }

122. private String formatTanggal(String tanggal) {

123. SimpleDateFormat sdfSource = new SimpleDateFormat("yyyy-MM-dd", new Locale("id", "ID"));

124. SimpleDateFormat sdfDestination = new SimpleDateFormat("EEEE, dd MMMM yyyy",new Locale("id", "ID"));

125. try {

126. Date date = sdfSource.parse(tanggal);

127. return sdfDestination.format(date);

128. } catch (ParseException e) {

129. e.printStackTrace();

130. return tanggal;

131. }

132. }

133. private void approvePengajuan(ModelPengajuanBarangMasuk pengajuan, int position) {

134. String barangMasukId = barangMasukRef.push().getKey();

135. ModelBarangMasuk barangMasukBaru = new ModelBarangMasuk(barangMasukId, pengajuan.getNamaBarang(), pengajuan.getSatuan(), pengajuan.getKeterangan(), pengajuan.getTanggalMasuk(), pengajuan.getJumlahBarang());

136. barangMasukRef.child(barangMasukId).setValue(barangMasukBaru).addOnCompleteListener(addTask -> {

137. if (addTask.isSuccessful()) {

138. perbaruiStokBarang(pengajuan);

139. pengajuan.setStatus("Diterima");

140. simpanRiwayatPengajuan(pengajuan);

141. hapusPengajuanSetelahDisetujui(pengajuan, position);

142. } else {

143. Toast.makeText(requireActivity(), "Gagal menambahkan data baru ke BarangMasuk", Toast.LENGTH\_SHORT).show();

144. }

145. });

146. }

147. private void rejectPengajuan(ModelPengajuanBarangMasuk pengajuan) {

148. pengajuan.setStatus("Ditolak");

149. simpanRiwayatPengajuan(pengajuan);

150. pengajuanRef.child(pengajuan.getId()).removeValue().addOnCompleteListener(removeTask -> {

151. if (removeTask.isSuccessful()) {

152. Toast.makeText(requireActivity(), "Pengajuan berhasil ditolak dan disimpan dalam riwayat", Toast.LENGTH\_SHORT).show();

153. } else {

154. Toast.makeText(requireActivity(), "Gagal menghapus pengajuan", Toast.LENGTH\_SHORT).show();

155. }

156. });

157. }

158.

159. private void perbaruiStokBarang(ModelPengajuanBarangMasuk pengajuan) {

160. stokBarangRef.orderByChild("namaBarang").equalTo(pengajuan.getNamaBarang()).addListenerForSingleValueEvent(new ValueEventListener() {

161. @Override

162. public void onDataChange(@NonNull DataSnapshot snapshot) {

163. boolean stokDitemukan = false;

164. for (DataSnapshot dataSnapshot : snapshot.getChildren()) {

165. ModelStock stockItem = dataSnapshot.getValue(ModelStock.class);

166. if (stockItem != null) {

167. stokDitemukan = true;

168. int currentStok = Integer.parseInt(stockItem.getJumlahBarang());

169. int updatedStok = currentStok + pengajuan.getJumlahBarang();

170. stockItem.setJumlahBarang(String.valueOf(updatedStok));

171.

172. dataSnapshot.getRef().setValue(stockItem).addOnCompleteListener(updateTask -> {

173. if (updateTask.isSuccessful()) {

174. Toast.makeText(requireActivity(), "Pengajuan disetujui dan Jumlah stok diperbarui", Toast.LENGTH\_SHORT).show();

175. } else {

176. Toast.makeText(requireActivity(), "Gagal memperbarui stok barang", Toast.LENGTH\_SHORT).show();

177. }

178. });

179. break;

180. }

181. }

182. if (!stokDitemukan) {

183. Toast.makeText(requireActivity(), "Stok barang tidak ditemukan", Toast.LENGTH\_SHORT).show();

184. }

185. }

186.

187. @Override

188. public void onCancelled(@NonNull DatabaseError error) {

189. Toast.makeText(requireActivity(), "Gagal memperbarui stok barang", Toast.LENGTH\_SHORT).show();

190. }

191. });

192. }

193.

194. private void simpanRiwayatPengajuan(ModelPengajuanBarangMasuk pengajuan) {

195. String riwayatId = riwayatPengajuanRef.push().getKey();

196. riwayatPengajuanRef.child(riwayatId).setValue(pengajuan).addOnCompleteListener(task -> {

197. if (task.isSuccessful()) {

198. Toast.makeText(requireActivity(), "Pengajuan " + pengajuan.getStatus() + " dan disimpan dalam riwayat", Toast.LENGTH\_SHORT).show();

199. } else {

200. Toast.makeText(requireActivity(), "Gagal menyimpan pengajuan ke dalam riwayat", Toast.LENGTH\_SHORT).show();

201. }

202. });

203. }

204.

205. private void hapusPengajuanSetelahDisetujui(ModelPengajuanBarangMasuk pengajuan, int position) {

206. pengajuanRef.child(pengajuan.getId()).removeValue().addOnCompleteListener(removeTask -> {

207. if (!removeTask.isSuccessful()) {

208. Toast.makeText(requireActivity(), "Gagal menghapus pengajuan setelah disetujui", Toast.LENGTH\_SHORT).show();

209. } else {

210. if (position >= 0 && position < pengajuanList.size()) {

211. pengajuanList.remove(position);

212. adapter.notifyItemRemoved(position);

213. }

214. }

215. });

216. }

217.

218. @Override

219. public void onItemClickListener(ModelPengajuanBarangMasuk data, int position) {

220. if (position >= 0 && position < pengajuanList.size()) {

221. AlertDialog.Builder builder = new AlertDialog.Builder(requireActivity());

222. builder.setTitle("Pengajuan Barang Masuk")

223. .setMessage("Apakah Anda ingin menerima atau menolak pengajuan ini?")

224. .setPositiveButton("Terima", new DialogInterface.OnClickListener() {

225. @Override

226. public void onClick(DialogInterface dialog, int which) {

227. approvePengajuan(data, position);

228. }

229. })

230. .setNegativeButton("Tolak", new DialogInterface.OnClickListener() {

231. @Override

232. public void onClick(DialogInterface dialog, int which) {

233. rejectPengajuan(data);

234. }

235. })

236. .setNeutralButton("Batal", null)

237. .show();

238. } else {

239. Toast.makeText(requireActivity(), "Pengajuan tidak valid", Toast.LENGTH\_SHORT).show();

240. }

241. }

242. }

243.

Kode Program Halaman Register.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import static android.content.ContentValues.TAG;

4. import androidx.annotation.NonNull;

5. import androidx.appcompat.app.AppCompatActivity;

6.

7. import android.content.Intent;

8. import android.os.Bundle;

9. import android.text.TextUtils;

10. import android.util.Log;

11. import android.view.View;

12. import android.widget.AdapterView;

13. import android.widget.ArrayAdapter;

14. import android.widget.Button;

15. import android.widget.ImageView;

16. import android.widget.Spinner;

17. import android.widget.TextView;

18. import android.widget.Toast;

19.

20. import com.example.barangku.R;

21. import com.example.barangku.activity.model.ModelUser;

22. import com.google.android.gms.tasks.OnCompleteListener;

23. import com.google.android.gms.tasks.OnFailureListener;

24. import com.google.android.gms.tasks.Task;

25. import com.google.android.material.textfield.TextInputEditText;

26. import com.google.firebase.auth.AuthResult;

27. import com.google.firebase.auth.FirebaseAuth;

28. import com.google.firebase.auth.FirebaseUser;

29. import com.google.firebase.database.DatabaseReference;

30. import com.google.firebase.database.FirebaseDatabase;

31. import com.google.firebase.messaging.FirebaseMessaging;

32.

33. public class Register extends AppCompatActivity {

34. private TextInputEditText etregister\_nama,etregister\_email,etregister\_password;

35. private Button btn\_register;

36. private TextView tv\_login,tv\_toolbar;

37. private Spinner spJabatan;

38. private String email, password;

39. private String nama,jabatan, token;

40. private FirebaseAuth mAuth;

41. private ImageView ivback;

42. DatabaseReference reference = FirebaseDatabase.getInstance().getReference("User");

43.

44. @Override

45. public void onStart() {

46. super.onStart();

47. FirebaseUser currentUser = mAuth.getCurrentUser();

48. if(currentUser != null){

49. }

50. }

51.

52. @Override

53. protected void onCreate(Bundle savedInstanceState) {

54. super.onCreate(savedInstanceState);

55. setContentView(R.layout.activity\_register);

56.

57. mAuth = FirebaseAuth.getInstance();

58.

59. etregister\_nama = findViewById(R.id.et\_nama\_register);

60. etregister\_email = findViewById(R.id.et\_email\_register);

61. etregister\_password = findViewById(R.id.et\_password\_register);

62.

63. tv\_login = findViewById(R.id.tv\_login);

64. btn\_register = findViewById(R.id.btn\_register\_button);

65. ivback = findViewById(R.id.iv\_back);

66.

67. spJabatan = findViewById(R.id.sp\_jabatan);

68.

69. ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(

70. Register.this,

71. R.array.Jabatan,

72. android.R.layout.simple\_spinner\_item

73. );

74.

75. adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

76. spJabatan.setAdapter(adapter);

77.

78. spJabatan.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {

79. @Override

80. public void onItemSelected(AdapterView<?> adapterView, View view, int pos, long l) {

81. jabatan = adapterView.getItemAtPosition(pos).toString();

82. }

83.

84. @Override

85. public void onNothingSelected(AdapterView<?> adapterView) {

86.

87. }

88. });

89.

90. tv\_toolbar = findViewById(R.id.tv\_judul);

91. tv\_toolbar.setText("Tambah User");

92.

93. // Set up login button click listener

94. tv\_login.setOnClickListener(new View.OnClickListener() {

95. @Override

96. public void onClick(View v) {

97. Intent intent = new Intent(Register.this, Login.class);

98. startActivity(intent);

99. // Close Register activity when login button is clicked

100. finish();

101. }

102. });

103.

104. ivback.setOnClickListener(new View.OnClickListener() {

105. @Override

106. public void onClick(View v) {

107. Intent intent = new Intent(Register.this, Login.class);

108. startActivity(intent);

109. }

110. });

111.

112. // Set up register button click listener

113. btn\_register.setOnClickListener(new View.OnClickListener() {

114. @Override

115. public void onClick(View v) {

116. nama = etregister\_nama.getText().toString();

117. email = etregister\_email.getText().toString();

118. password = etregister\_password.getText().toString();

119.

120. if (nama.trim().isEmpty()) {

121. etregister\_nama.setError("Silakan Masukan Nama");

122. return;

123. }

124. if (email.trim().isEmpty()) {

125. etregister\_email.setError("Silakan Masukan Email");

126. return;

127. }

128. if (password.trim().isEmpty()) {

129. etregister\_password.setError("Silakan Masukan Password");

130. return;

131. }

132.

133. // Check if email and password fields are not empty

134. if (!TextUtils.isEmpty(email) && !TextUtils.isEmpty(password)) {

135. // Create a new user account with email and password

136. mAuth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {

137. @Override

138. public void onComplete(@NonNull Task<AuthResult> task) {

139. if (task.isSuccessful()) {

140. FirebaseUser user = mAuth.getCurrentUser();

141. FirebaseMessaging.getInstance().getToken().addOnCompleteListener(new OnCompleteListener<String>() {

142. @Override

143. public void onComplete(@NonNull Task<String> task) {

144. token = task.getResult();

145. String userId = user.getUid();

146. ModelUser mu = new ModelUser(userId, email, nama, jabatan, token);

147. reference.child(userId).setValue(mu);

148.

149. Toast.makeText(Register.this, "Berhasil Daftar Akun", Toast.LENGTH\_SHORT).show();

150. //

151. }

152. }).addOnFailureListener(new OnFailureListener() {

153. @Override

154. public void onFailure(@NonNull Exception e) {

155. Toast.makeText(Register.this, "Gagal Mendapatkan Token FCM: " + e.getMessage(), Toast.LENGTH\_SHORT).show();

156. }

157. });

158. } else {

159. // Tambahkan lebih banyak detail tentang kesalahan yang terjadi

160. String errorMessage = task.getException() != null ? task.getException().getMessage() : "Unknown error";

161. Toast.makeText(Register.this, "Gagal Membuat Akun: " + errorMessage, Toast.LENGTH\_SHORT).show();

162. }

163. }

164. });

165. } else {

166. Toast.makeText(Register.this, "Masukan Email dan Password", Toast.LENGTH\_SHORT).show();

167. }

168. }

169. });

170.

171.

172.

173. }

174.

175. }

176.

Kode Program Halaman RiwayatPengajuan.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import androidx.appcompat.app.AppCompatActivity;

4. import androidx.fragment.app.Fragment;

5. import androidx.fragment.app.FragmentManager;

6. import androidx.fragment.app.FragmentTransaction;

7.

8. import android.os.Bundle;

9.

10. import com.example.barangku.R;

11. import com.google.android.material.bottomnavigation.BottomNavigationView;

12.

13. public class RiwayatPengajuan extends AppCompatActivity {

14.

15. @Override

16. protected void onCreate(Bundle savedInstanceState) {

17. super.onCreate(savedInstanceState);

18. setContentView(R.layout.activity\_riwayat\_pengajuan);

19. gantiFragment(new RiwayatPengajuanBarangMasukFragment());

20. BottomNavigationView navigationView = findViewById(R.id.navigations\_bar);

21. navigationView.setOnItemSelectedListener(item -> {

22. if (item.getItemId() == R.id.nav\_barang\_masuk){

23. gantiFragment(new RiwayatPengajuanBarangMasukFragment());

24. }else if(item.getItemId() == R.id.nav\_barang\_keluar){

25. gantiFragment(new RiwayatPengajuanBarangKeluarFragment());

26. }

27. return true;

28. });

29. }

30. private void gantiFragment(Fragment fragment){

31. FragmentManager fragmentManager = getSupportFragmentManager();

32. FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();

33. fragmentTransaction.replace(R.id.framelayout, fragment);

34. fragmentTransaction.commit();

35. }

36. }

37.

Kode Program Halaman RiwayatPengajuanBarangKeluarFragment.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import android.content.Intent;

4. import android.os.Bundle;

5.

6. import androidx.annotation.NonNull;

7. import androidx.annotation.Nullable;

8. import androidx.fragment.app.Fragment;

9. import androidx.recyclerview.widget.LinearLayoutManager;

10. import androidx.recyclerview.widget.RecyclerView;

11.

12. import android.view.LayoutInflater;

13. import android.view.View;

14. import android.view.ViewGroup;

15. import android.widget.ImageView;

16. import android.widget.TextView;

17. import android.widget.Toast;

18.

19. import com.example.barangku.R;

20. import com.example.barangku.activity.adapter.AdapterRiwayatPengajuanBarangKeluar;

21. import com.example.barangku.activity.adapter.AdapterRiwayatPengajuanBarangMasuk;

22. import com.example.barangku.activity.model.ModelRiwayatPengajuanBarangKeluar;

23. import com.example.barangku.activity.model.ModelRiwayatPengajuanBarangMasuk;

24. import com.google.firebase.database.DataSnapshot;

25. import com.google.firebase.database.DatabaseError;

26. import com.google.firebase.database.DatabaseReference;

27. import com.google.firebase.database.FirebaseDatabase;

28. import com.google.firebase.database.ValueEventListener;

29.

30. import java.text.ParseException;

31. import java.text.SimpleDateFormat;

32. import java.util.ArrayList;

33. import java.util.Date;

34. import java.util.List;

35. import java.util.Locale;

36.

37.

38. public class RiwayatPengajuanBarangKeluarFragment extends Fragment {

39. private TextView tvToolbar;

40. private ImageView ivBack;

41. private RecyclerView rvRiwayatPengajuan;

42. private AdapterRiwayatPengajuanBarangKeluar adapter;

43. private List<ModelRiwayatPengajuanBarangKeluar> riwayatList = new ArrayList<>();

44. private DatabaseReference riwayatPengajuanRef = FirebaseDatabase.getInstance().getReference("RiwayatPengajuanBarangKeluar");

45.

46. public RiwayatPengajuanBarangKeluarFragment() {

47. // Required empty public constructor

48. }

49.

50.

51. @Override

52. public void onCreate(Bundle savedInstanceState) {

53. super.onCreate(savedInstanceState);

54. }

55.

56. @Override

57. public View onCreateView(LayoutInflater inflater, ViewGroup container,

58. Bundle savedInstanceState) {

59. // Inflate the layout for this fragment

60. return inflater.inflate(R.layout.fragment\_riwayat\_pengajuan\_barang\_keluar, container, false);

61. }

62.

63. @Override

64. public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {

65. super.onViewCreated(view, savedInstanceState);

66.

67. tvToolbar = view.findViewById(R.id.tv\_judul);

68. tvToolbar.setText("Riwayat Pengajuan");

69.

70. ivBack = view.findViewById(R.id.iv\_back);

71. ivBack.setOnClickListener(new View.OnClickListener() {

72. @Override

73. public void onClick(View view) {

74. Intent intent = new Intent(getActivity(), MainActivity.class ) ;

75. startActivity(intent);

76. }

77. });

78.

79. rvRiwayatPengajuan = view.findViewById(R.id.rv\_riwayat\_pengajuan);

80.

81. adapter = new AdapterRiwayatPengajuanBarangKeluar(riwayatList, getView().getContext());

82. LinearLayoutManager lmRiwayat = new LinearLayoutManager(getView().getContext(), LinearLayoutManager.VERTICAL, false);

83. rvRiwayatPengajuan.setLayoutManager(lmRiwayat);

84. rvRiwayatPengajuan.setAdapter(adapter);

85.

86. loadRiwayatData();

87. }

88.

89. private void loadRiwayatData() {

90. riwayatPengajuanRef.addValueEventListener(new ValueEventListener() {

91. @Override

92. public void onDataChange(@NonNull DataSnapshot snapshot) {

93. riwayatList.clear();

94. for (DataSnapshot dataSnapshot : snapshot.getChildren()) {

95. ModelRiwayatPengajuanBarangKeluar riwayat = dataSnapshot.getValue(ModelRiwayatPengajuanBarangKeluar.class);

96. if (riwayat != null) {

97. riwayat.setTanggalKeluar(formatTanggal(riwayat.getTanggalKeluar()));

98. riwayatList.add(riwayat);

99. }

100. }

101. adapter.notifyDataSetChanged();

102. }

103.

104. @Override

105. public void onCancelled(@NonNull DatabaseError error) {

106. Toast.makeText(requireActivity(), "Gagal memuat data riwayat pengajuan", Toast.LENGTH\_SHORT).show();

107. }

108. });

109. }

110. private String formatTanggal(String tanggal) {

111. SimpleDateFormat sdfSource = new SimpleDateFormat("yyyy-MM-dd", new Locale("id", "ID"));

112. SimpleDateFormat sdfDestination = new SimpleDateFormat("EEEE, dd MMMM yyyy",new Locale("id", "ID"));

113. try {

114. Date date = sdfSource.parse(tanggal);

115. return sdfDestination.format(date);

116. } catch (ParseException e) {

117. e.printStackTrace();

118. return tanggal;

119. }

120. }

121. }

122.

Kode Program Halaman RiwayatPengajuanBarangMasukFragment.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import android.content.Intent;

4. import android.os.Bundle;

5. import androidx.annotation.NonNull;

6. import androidx.annotation.Nullable;

7. import androidx.fragment.app.Fragment;

8. import androidx.recyclerview.widget.LinearLayoutManager;

9. import androidx.recyclerview.widget.RecyclerView;

10. import android.view.LayoutInflater;

11. import android.view.View;

12. import android.view.ViewGroup;

13. import android.widget.ImageView;

14. import android.widget.TextView;

15. import android.widget.Toast;

16. import com.example.barangku.R;

17. import com.example.barangku.activity.adapter.AdapterRiwayatPengajuanBarangMasuk;

18. import com.example.barangku.activity.model.ModelRiwayatPengajuanBarangMasuk;

19. import com.google.firebase.database.DataSnapshot;

20. import com.google.firebase.database.DatabaseError;

21. import com.google.firebase.database.DatabaseReference;

22. import com.google.firebase.database.FirebaseDatabase;

23. import com.google.firebase.database.ValueEventListener;

24.

25. import java.text.ParseException;

26. import java.text.SimpleDateFormat;

27. import java.util.ArrayList;

28. import java.util.Date;

29. import java.util.List;

30. import java.util.Locale;

31.

32. public class RiwayatPengajuanBarangMasukFragment extends Fragment {

33.

34. private TextView tvToolbar;

35. private ImageView ivBack;

36. private RecyclerView rvRiwayatPengajuan;

37. private AdapterRiwayatPengajuanBarangMasuk adapter;

38. private List<ModelRiwayatPengajuanBarangMasuk> riwayatList = new ArrayList<>();

39. private DatabaseReference riwayatPengajuanRef = FirebaseDatabase.getInstance().getReference("RiwayatPengajuanBarangMasuk");

40.

41. public RiwayatPengajuanBarangMasukFragment() {

42. // Required empty public constructor

43. }

44.

45. @Override

46. public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

47. // Inflate the layout for this fragment

48. return inflater.inflate(R.layout.fragment\_riwayat\_pengajuan\_barang\_masuk, container, false);

49. }

50.

51. @Override

52. public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {

53. super.onViewCreated(view, savedInstanceState);

54. tvToolbar = view.findViewById(R.id.tv\_judul);

55. tvToolbar.setText("Riwayat Pengajuan");

56.

57. ivBack = view.findViewById(R.id.iv\_back);

58. ivBack.setOnClickListener(new View.OnClickListener() {

59. @Override

60. public void onClick(View view) {

61. Intent intent = new Intent(getActivity(), MainActivity.class ) ;

62. startActivity(intent);

63. }

64. });

65.

66. rvRiwayatPengajuan = view.findViewById(R.id.rv\_riwayat\_pengajuan);

67.

68. adapter = new AdapterRiwayatPengajuanBarangMasuk(riwayatList, getView().getContext());

69. LinearLayoutManager lmRiwayat = new LinearLayoutManager(getView().getContext(), LinearLayoutManager.VERTICAL, false);

70. rvRiwayatPengajuan.setLayoutManager(lmRiwayat);

71. rvRiwayatPengajuan.setAdapter(adapter);

72.

73. loadRiwayatData();

74. }

75.

76. private void loadRiwayatData() {

77. riwayatPengajuanRef.addValueEventListener(new ValueEventListener() {

78. @Override

79. public void onDataChange(@NonNull DataSnapshot snapshot) {

80. riwayatList.clear();

81. for (DataSnapshot dataSnapshot : snapshot.getChildren()) {

82. ModelRiwayatPengajuanBarangMasuk riwayat = dataSnapshot.getValue(ModelRiwayatPengajuanBarangMasuk.class);

83. if (riwayat != null) {

84. riwayat.setTanggalMasuk(formatTanggal(riwayat.getTanggalMasuk()));

85. riwayatList.add(riwayat);

86. }

87. }

88. adapter.notifyDataSetChanged();

89. }

90.

91. @Override

92. public void onCancelled(@NonNull DatabaseError error) {

93. Toast.makeText(requireActivity(), "Gagal memuat data riwayat pengajuan", Toast.LENGTH\_SHORT).show();

94. }

95. });

96. }

97. private String formatTanggal(String tanggal) {

98. SimpleDateFormat sdfSource = new SimpleDateFormat("yyyy-MM-dd", new Locale("id", "ID"));

99. SimpleDateFormat sdfDestination = new SimpleDateFormat("EEEE, dd MMMM yyyy",new Locale("id", "ID"));

100. try {

101. Date date = sdfSource.parse(tanggal);

102. return sdfDestination.format(date);

103. } catch (ParseException e) {

104. e.printStackTrace();

105. return tanggal;

106. }

107. }

108. }

109.

Kode Program Halaman StockBarang.java

1. package com.example.barangku.activity.user\_activity;

2.

3. import androidx.annotation.NonNull;

4. import androidx.annotation.Nullable;

5. import androidx.appcompat.app.AppCompatActivity;

6. import androidx.appcompat.widget.SearchView;

7. import androidx.recyclerview.widget.LinearLayoutManager;

8. import androidx.recyclerview.widget.RecyclerView;

9.

10. import android.app.Dialog;

11. import android.app.ProgressDialog;

12. import android.content.Intent;

13. import android.net.Uri;

14. import android.os.Bundle;

15. import android.view.View;

16. import android.view.ViewGroup;

17. import android.widget.AdapterView;

18. import android.widget.ArrayAdapter;

19. import android.widget.Button;

20. import android.widget.EditText;

21. import android.widget.ImageView;

22. import android.widget.Spinner;

23. import android.widget.TextView;

24. import android.widget.Toast;

25.

26. import com.bumptech.glide.Glide;

27. import com.example.barangku.R;

28. import com.example.barangku.activity.adapter.AdapterStock;

29. import com.example.barangku.activity.model.ModelStock;

30. import com.example.barangku.activity.user\_activity.utils.ItemClickStock;

31. import com.github.dhaval2404.imagepicker.ImagePicker;

32. import com.google.android.gms.tasks.OnCompleteListener;

33. import com.google.android.gms.tasks.OnFailureListener;

34. import com.google.android.gms.tasks.OnSuccessListener;

35. import com.google.android.gms.tasks.Task;

36. import com.google.android.material.floatingactionbutton.FloatingActionButton;

37. import com.google.firebase.database.DataSnapshot;

38. import com.google.firebase.database.DatabaseError;

39. import com.google.firebase.database.DatabaseReference;

40. import com.google.firebase.database.FirebaseDatabase;

41. import com.google.firebase.database.ValueEventListener;

42. import com.google.firebase.storage.FirebaseStorage;

43. import com.google.firebase.storage.StorageReference;

44. import com.google.firebase.storage.UploadTask;

45.

46. import java.text.SimpleDateFormat;

47. import java.util.ArrayList;

48. import java.util.Date;

49. import java.util.HashMap;

50. import java.util.List;

51. import java.util.Locale;

52.

53. public class StockBarang extends AppCompatActivity implements ItemClickStock {

54. private TextView tvToolbar, tvNamaBarang, tvJumlahItem, tvSatuan;

55. private Dialog dialog, dialogStock, dialogEdit;

56. private ImageView ivStockBarang;

57. private String namabarang, jumlahbarang, satuan;

58. private Uri gambar ;

59. private DatabaseReference reference = FirebaseDatabase.getInstance().getReference("StockBarang");

60. private StorageReference storageReference = FirebaseStorage.getInstance().getReference();

61. private FloatingActionButton fabTambah;

62. private ProgressDialog pd;

63. private SearchView searchView;

64. RecyclerView rvStock;

65. private ImageView ivback;

66. private List<ModelStock> list\_stock = new ArrayList<>();

67. private AdapterStock adapterStock;

68. @Override

69. protected void onCreate(Bundle savedInstanceState) {

70. super.onCreate(savedInstanceState);

71. setContentView(R.layout.activity\_stock\_barang);

72.

73. tvToolbar =findViewById(R.id.tv\_judul);

74. tvToolbar.setText("Stock Barang");

75. searchView = findViewById(R.id.search);

76.

77. tvNamaBarang = findViewById(R.id.tv\_nama\_barang);

78. tvJumlahItem = findViewById(R.id.tv\_jumlah\_item);

79. tvSatuan = findViewById(R.id.tv\_satuan);

80.

81. ivback=findViewById(R.id.iv\_back);

82. ivback.setOnClickListener(new View.OnClickListener() {

83. @Override

84. public void onClick(View v) {

85. Intent intent = new Intent(StockBarang.this, MainActivity.class);

86. startActivity(intent);

87. }

88. });

89. pd = new ProgressDialog(this);

90. pd.setMessage("Loading");

91. pd.setCancelable(false);

92. pd.show();

93.

94. rvStock = findViewById(R.id.rv\_stock);

95. searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {

96. @Override

97. public boolean onQueryTextSubmit(String query) {

98. filterList(query);

99. searchView.clearFocus();

100. return true;

101. }

102.

103. @Override

104. public boolean onQueryTextChange(String newText) {

105. filterList(newText);

106. return true;

107. }

108. });

109.

110. adapterStock = new AdapterStock(StockBarang.this, list\_stock);

111. adapterStock.setItemCLickStock(this);

112. LinearLayoutManager lmStock = new LinearLayoutManager(this, LinearLayoutManager.VERTICAL, false);

113. rvStock.setLayoutManager(lmStock);

114. rvStock.setAdapter(adapterStock);//

115.

116. reference.addValueEventListener(new ValueEventListener() {

117. @Override

118. public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

119. list\_stock.clear();

120. for (DataSnapshot snapshot: dataSnapshot.getChildren()){

121. ModelStock ms = snapshot.getValue(ModelStock.class);

122. list\_stock.add(ms);

123. }

124. pd.dismiss();

125. adapterStock.notifyDataSetChanged();//

126. }

127. @Override

128. public void onCancelled(@NonNull DatabaseError databaseError) {

129. Toast.makeText(StockBarang.this, "Data Gagal Dimuat", Toast.LENGTH\_SHORT).show();

130. }

131. });

132. fabTambah = findViewById(R.id.fab\_tambah\_barang);

133. fabTambah.setOnClickListener(new View.OnClickListener() {

134. @Override

135. public void onClick(View view) {

136. dialog = new Dialog(StockBarang.this);

137. dialog.setContentView(R.layout.dialog\_tambah\_stock);

138. dialog.getWindow().setLayout(ViewGroup.LayoutParams.WRAP\_CONTENT, ViewGroup.LayoutParams.WRAP\_CONTENT);

139. dialog.setCancelable(false);

140. dialog.show();

141. EditText etNamaBarang = dialog.findViewById(R.id.et\_nama\_barang);

142. EditText etJumlahBarang = dialog.findViewById(R.id.et\_jumlah\_barang);

143.

144. ivStockBarang = dialog.findViewById(R.id.iv\_stock\_barang);

145.

146. Spinner sp = dialog.findViewById(R.id.sp\_satuan);

147. ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(

148. StockBarang.this,

149. R.array.Satuan,

150. android.R.layout.simple\_spinner\_item

151. );

152.

153. adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

154. sp.setAdapter(adapter);

155.

156. sp.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {

157. @Override

158. public void onItemSelected(AdapterView<?> adapterView, View view, int pos, long l) {

159. satuan = adapterView.getItemAtPosition(pos).toString();

160. }

161.

162. @Override

163. public void onNothingSelected(AdapterView<?> adapterView) {

164.

165. }

166. });

167.

168. ivStockBarang.setOnClickListener(new View.OnClickListener() {

169. @Override

170. public void onClick(View view) {

171. ImagePicker.with(StockBarang.this)

172. .galleryMimeTypes(new String[]{"image/png", "image/jpg", "image/jpeg"})

173. .crop(16f, 9f)

174. .compress(256)

175. .maxResultSize(1080, 1080)

176. .start();

177. }

178. });

179.

180. dialog.findViewById(R.id.iv\_close).setOnClickListener(new View.OnClickListener() {

181. @Override

182. public void onClick(View view) {

183. dialog.dismiss();

184. }});

185.

186. dialog.findViewById(R.id.btn\_simpan).setOnClickListener(new View.OnClickListener() {

187. @Override

188. public void onClick(View view) {

189. namabarang = etNamaBarang.getText().toString();

190. jumlahbarang = etJumlahBarang.getText().toString();

191.

192. if (namabarang.trim().isEmpty()) {

193. etNamaBarang.setError("Required");

194. return;

195. }

196. if (jumlahbarang.trim().isEmpty()) {

197. etJumlahBarang.setError("Required");

198. return;

199. }

200. if (gambar == null) {

201. Toast.makeText(StockBarang.this, "Pilih Gambar", Toast.LENGTH\_SHORT).show();

202. return;

203. }//

204. else {

205. saveFirebase();

206. }

207. }

208. });

209.

210. }

211. });

212. }

213.

214. private void saveFirebase() {

215. StorageReference imagereference = storageReference.child("StockBarang").child(String.valueOf(System.currentTimeMillis()));

216. imagereference.putFile(gambar).addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {

217. @Override

218. public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {

219. imagereference.getDownloadUrl().addOnSuccessListener(new OnSuccessListener<Uri>() {

220. @Override

221. public void onSuccess(Uri uri) {

222. // String uniqueId = reference.getKey();

223. // ModelStock ms = new ModelStock(uniqueId,namabarang, jumlahbarang, satuan, uri.toString());

224. // reference.child(uniqueId).setValue(ms);

225.

226. // Generate ID Barang secara otomatis di Firebaase

227. // DatabaseReference newReference = reference.push();

228. // String id = newReference.getKey();

229. // ModelStock ms = new ModelStock(id, namabarang, jumlahbarang, satuan, uri.toString());

230. // newReference.setValue(ms);

231.

232. String idBarang = generateIdBarang(namabarang);

233.

234. ModelStock ms = new ModelStock(idBarang, namabarang, jumlahbarang, satuan, uri.toString(), true);

235. reference.child(idBarang).setValue(ms);

236.

237. Toast.makeText(StockBarang.this, "Stock Barang Berhasil Disimpan", Toast.LENGTH\_SHORT).show();

238. dialog.dismiss();

239. refreshStockList();

240. }

241.

242. }).addOnFailureListener(new OnFailureListener() {

243. @Override

244. public void onFailure(@NonNull Exception e) {

245. Toast.makeText(StockBarang.this, "Error! " + e.getMessage(), Toast.LENGTH\_SHORT).show();

246. }

247. });

248. }

249. });

250.

251. }

252.

253. private String generateIdBarang(String namabarang) {

254. int maxLength = 3;

255. String prefix = namabarang.length() >= maxLength ? namabarang.substring(0, maxLength).toUpperCase() : namabarang.toUpperCase();

256.

257. String timestamp = new SimpleDateFormat("yyyyMMddHHmmss", Locale.getDefault()).format(new Date());

258. String id = prefix.concat(timestamp.substring(timestamp.length() - (7 - maxLength)));

259. return id;

260. }

261.

262. private void refreshStockList() {

263. reference.addListenerForSingleValueEvent(new ValueEventListener() {

264. @Override

265. public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

266. // Clear the list again to ensure only new data is fetched

267. list\_stock.clear();

268.

269. for (DataSnapshot snapshot : dataSnapshot.getChildren()) {

270. ModelStock ms = snapshot.getValue(ModelStock.class);

271. list\_stock.add(ms);

272.

273. }

274. adapterStock.notifyDataSetChanged();

275. pd.dismiss();

276. }

277.

278. @Override

279. public void onCancelled(@NonNull DatabaseError databaseError) {

280. // Handle database errors (optional)

281. Toast.makeText(StockBarang.this, "Data Gagal Di Simpan", Toast.LENGTH\_SHORT).show();

282. }

283. });

284. }

285.

286. private void filterList(String query) {

287. List<ModelStock> filteredList = new ArrayList<>();

288. String queryInLower = query.toLowerCase();

289. for (ModelStock ms : list\_stock) {

290. if (ms.getNamaBarang().toLowerCase().contains(queryInLower)) {

291. filteredList.add(ms);

292. } else if (ms.getJumlahBarang().toLowerCase().contains(queryInLower)) {

293. filteredList.add(ms);

294. }

295. }

296. if (filteredList.isEmpty()) {

297. Toast.makeText(this, "Tidak Ada Stock Yang Tersedia", Toast.LENGTH\_SHORT).show();

298. } else {

299. adapterStock.setFilter(filteredList);

300. }

301. }

302.

303. @Override

304. protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {

305. super.onActivityResult(requestCode, resultCode, data);

306. gambar = data.getData();

307. ivStockBarang.setImageURI(gambar);

308. }

309.

310. @Override

311. public void onItemClickListener(ModelStock data, int position) {

312. dialogStock = new Dialog(StockBarang.this);

313. dialogStock.setContentView(R.layout.dialog\_tampil\_stock);

314. dialogStock.getWindow().setLayout(ViewGroup.LayoutParams.WRAP\_CONTENT, ViewGroup.LayoutParams.WRAP\_CONTENT);

315. dialogStock.setCancelable(false);

316.

317. dialogStock.show();

318.

319. ImageView ivStockBarang = dialogStock.findViewById(R.id.iv\_foto\_barang);

320. Glide.with(StockBarang.this)

321. .load(data.getGambar())

322. .into(ivStockBarang);

323.

324. TextView tvNamaBarang = dialogStock.findViewById(R.id.tv\_nama\_barang);

325. tvNamaBarang.setText(data.getNamaBarang());

326. TextView tvJumlahItem = dialogStock.findViewById(R.id.tv\_jumlah\_item);

327. tvJumlahItem.setText(data.getJumlahBarang());

328. TextView tvSatuan = dialogStock.findViewById(R.id.tv\_satuan);

329. tvSatuan.setText(data.getSatuan());

330.

331. Button btnEnable = dialogStock.findViewById(R.id.btn\_enable);

332. Button btnDisable = dialogStock.findViewById(R.id.btn\_disable);

333.

334. btnEnable.setOnClickListener(new View.OnClickListener() {

335. @Override

336. public void onClick(View view) {

337. HashMap<String,Object> updateData = new HashMap<>();

338. updateData.put("enable", true);

339. reference.child(data.getId()).updateChildren(updateData)

340. .addOnCompleteListener(new OnCompleteListener<Void>() {

341. @Override

342. public void onComplete(@NonNull Task<Void> task) {

343. if(task.isSuccessful()){

344. Toast.makeText(StockBarang.this, "Status Berhasil Diubah Ke Enable", Toast.LENGTH\_SHORT).show();

345. }

346. else {

347. Toast.makeText(StockBarang.this, "Gagal Mengubah Status", Toast.LENGTH\_SHORT).show();

348. }

349. }

350. });

351.

352. }

353. });

354. btnDisable.setOnClickListener(new View.OnClickListener() {

355. @Override

356. public void onClick(View view) {

357. HashMap<String,Object> updateData = new HashMap<>();

358. updateData.put("enable", false);

359. reference.child(data.getId()).updateChildren(updateData)

360. .addOnCompleteListener(new OnCompleteListener<Void>() {

361. @Override

362. public void onComplete(@NonNull Task<Void> task) {

363. if(task.isSuccessful()){

364. Toast.makeText(StockBarang.this, "Status Berhasil Diubah Ke Disable", Toast.LENGTH\_SHORT).show();

365. }

366. else {

367. Toast.makeText(StockBarang.this, "Gagal Mengubah Status", Toast.LENGTH\_SHORT).show();si

368. }

369. }

370. });

371. }

372. });

373. dialogStock.findViewById(R.id.iv\_close).setOnClickListener(view -> {dialogStock.dismiss();});

374.

375.

376. }

377.

378.

379. }

380.

381.

## <judul lampiran>

## <judul lampiran>