**PROPOSAL TUGAS AKHIR**

**MANAJEMEN KEUANGAN MAHASISWA DENGAN ALGORITMA APRIORI**



**Oleh :**

**MUHAMMAD AQIILAH PALIGIE**

**2211083014**

**PROGRAM STUDI TEKNOLOGI REKAYASA PERANGKAT LUNAK JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI PADANG**

# ABSTRAK

Manajemen keuangan yang baik merupakan aspek penting bagi mahasiswa dalam menjaga kestabilan finansial selama masa studi. Namun, banyak mahasiswa menghadapi kesulitan dalam mengelola pengeluaran mereka, seperti kurangnya pencatatan yang sistematis dan tidak adanya analisis terhadap pola pengeluaran. Hal ini sering kali menyebabkan mahasiswa mengalami krisis keuangan di pertengahan atau akhir bulan. Untuk mengatasi permasalahan tersebut, penelitian ini bertujuan untuk mengembangkan aplikasi manajemen keuangan berbasis mobile yang mampu menganalisis pola pengeluaran mahasiswa menggunakan Algoritma Apriori. Aplikasi ini dikembangkan menggunakan Flutter sebagai teknologi frontend untuk memastikan pengalaman pengguna yang responsif dan optimal di platform Android maupun iOS. Sementara itu, backend dibangun menggunakan Flask sebagai framework server-side yang ringan dan cepat, dengan PostgreSQL sebagai basis data utama untuk menyimpan informasi transaksi keuangan mahasiswa. Manajemen database dilakukan menggunakan Prisma ORM, yang mempermudah manipulasi data dengan pendekatan yang aman dan efisien.

**Kata Kunci**

*Manajemen Keuangan, Mahasiswa, Algoritma Apriori, Flutter, Flask, PostgreSQL, Prisma ORM.*

DAFTAR ISI

[ABSTRAK 2](#_Toc203560077)

[BAB I PENDAHULUAN 4](#_Toc203560078)

[1.1 Latar Belakang 4](#_Toc203560079)

[1.2 Rumusan Masalah 5](#_Toc203560080)

[1.3 Tujuan 6](#_Toc203560081)

[1.4 Manfaat 6](#_Toc203560082)

[1.5 Batasan Masalah 7](#_Toc203560083)

[BAB II TINJAUAN PUSTAKA 8](#_Toc203560084)

[2.1 Penelitian Terkait 8](#_Toc203560085)

[2.2 Teori Penunjang 9](#_Toc203560086)

[BAB III METODOLOGI 10](#_Toc203560087)

[3.1 Analisis Sistem 10](#_Toc203560088)

[3.2 Perancangan Sistem 11](#_Toc203560089)

[3.3 Flowchart Aplikasi 12](#_Toc203560090)

[3.4 Flowchart Algoritma 14](#_Toc203560091)

[3.5 Use Case Diagram 15](#_Toc203560092)

[3.6 ERD (Entity Relationship Diagram) 15](#_Toc203560093)

[3.7 Jadwal Pelaksanaan 16](#_Toc203560094)

[BAB IV IMPLEMENTASI DAN ANALISIS 16](#_Toc203560095)

[4.1 Spesifikasi Sistem 17](#_Toc203560096)

[4.2 Implementasi Antarmuka 18](#_Toc203560097)

[4.3 Analisis Algoritma 24](#_Toc203560098)

[4.4 Pengujian 24](#_Toc203560099)

[BAB V PENUTUP 30](#_Toc203560100)

[5.1 Kesimpulan 30](#_Toc203560101)

[5.2 Saran 30](#_Toc203560102)

[DAFTAR PUSTAKA 31](#_Toc203560103)

# PENDAHULUAN

## Latar Belakang

Manajemen keuangan merupakan salah satu aspek penting dalam kehidupan Mahasiswa terutama dalam mengatur pemasukan dan pengeluaran agar tetap stabil selama masa studi. Banyak mahasiswa mengalami kesulitan dalam mengelola keuangan mereka, seperti pengeluaran yang tidak terkontrol, kurangnya alokasi anggaran yang tepat, serta kurangnya pemahaman tentang pola pengeluaran yang mereka lakukan. Hal ini sering kali menyebabkan mahasiswa mengalami kesulitan keuangan di pertengahan atau akhir bulan, sehingga berdampak pada kesejahteraan mereka secara keseluruhan.

mayoritas mahasiswa hanya mengandalkan pencatatan manual atau aplikasi keuangan sederhana tanpa adanya analisis yang mendalam terkait pola pengeluaran mereka. Selain itu, belum banyak sistem yang dapat memberikan rekomendasi berbasis data untuk membantu mahasiswa dalam mengatur anggaran secara lebih efektif. Tanpa adanya analisis pola pengeluaran yang baik, mahasiswa cenderung mengulang kebiasaan keuangan yang kurang sehat, sehingga meningkatkan risiko ketidakpastian finansial. Oleh karena itu, dibutuhkan suatu aplikasi yang dapat membantu mahasiswa dalam mengelola keuangan mereka dengan lebih baik, memberikan wawasan mengenai pola pengeluaran, serta memberikan rekomendasi pengelolaan keuangan yang lebih optimal.

Dalam pengembangan sistem ini, dipilihlah Flutter sebagai teknologi frontend untuk membangun aplikasi mobile yang cross-platform (Android dan iOS) dengan UI/UX yang responsif dan modern, Flutter menawarkan keunggulan dalam pengembangan aplikasi mobile dengan performa tinggi, serta fleksibilitas dalam desain antarmuka pengguna. Untuk backend, sistem ini dikembangkan menggunakan Express.js sebagai framework backend yang ringan dan cepat, serta PostgreSQL sebagai database utama yang memiliki dukungan kuat terhadap transaksi dan relasi data. Manajemen database akan dilakukan menggunakan Prisma ORM, yang memungkinkan pengelolaan data menjadi lebih efisien dengan pendekatan type-safe query serta kemudahan dalam integrasi dengan Express.js. Dengan ini, diharapkan sistem dapat memiliki performa yang optimal dan mudah untuk dikembangkan lebih lanjut.

Selain itu, aplikasi ini akan menerapkan Algoritma Apriori, salah satu teknik dalam data mining yang digunakan untuk menemukan pola hubungan antar transaksi atau pengeluaran mahasiswa. Algoritma ini memungkinkan aplikasi untuk menganalisis pola pengeluaran mahasiswa berdasarkan data transaksi sebelumnya, sehingga dapat memberikan rekomendasi anggaran, prediksi pengeluaran bulanan, serta saran finansial yang lebih terarah. Dengan pendekatan ini, mahasiswa dapat memahami kebiasaan keuangan mereka dan mengambil keputusan yang lebih bijak dalam mengelola keuangan mereka.

Dengan adanya aplikasi manajemen keuangan berbasis mobile ini, diharapkan mahasiswa dapat mengelola keuangan mereka dengan lebih cerdas, memahami pola pengeluaran, serta mendapatkan rekomendasi keuangan yang lebih efektif. Dengan pemilihan teknologi Flutter, Express.js, PostgreSQL, dan Prisma, serta pemanfaatan Algoritma Apriori, aplikasi ini bertujuan untuk menciptakan solusi yang efisien, akurat, dan mudah digunakan dalam membantu mahasiswa mengelola keuangan mereka secara lebih terencana.

## Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, terdapat beberapa rumusan masalah sebagai berikut:

1. Bagaimana menerapkan Algoritma Apriori untuk mengidentifikasi pola transaksi dan kebiasaan pengeluaran mahasiswa?
2. Bagaimana membangun aplikasi mobile yang dapat membantu mahasiswa dalam mencatat dan menganalisis pola pengeluaran mereka secara lebih efektif?
3. Bagaimana aplikasi dapat memberikan rekomendasi keuangan yang sesuai berdasarkan pola pengeluaran mahasiswa agar mereka dapat mengelola keuangan secara lebih optimal?
4. Bagaimana integrasi antara Flutter sebagai frontend dan Express.js dengan PostgreSQL sebagai backend dalam membangun aplikasi yang efisien dan mudah digunakan?

## Tujuan

Berdasarkan rumusan masalah diatas, tedapat tujuan pada sistem ini sebagai berikut:

1. Mengembangkan aplikasi mobile berbasis Flutter yang dapat membantu mahasiswa dalam mencatat, memantau, dan menganalisis pengeluaran mereka secara lebih efektif dan sistematis.
2. Menerapkan Algoritma Apriori untuk mengidentifikasi pola transaksi dan kebiasaan pengeluaran mahasiswa berdasarkan data historis, sehingga dapat memberikan wawasan yang lebih mendalam terkait pengelolaan keuangan mereka.
3. Menyediakan fitur rekomendasi keuangan berbasis data yang membantu mahasiswa dalam mengatur anggaran mereka dengan lebih optimal, sehingga dapat mengurangi risiko ketidakstabilan finansial.
4. Meningkatkan kesadaran mahasiswa dalam pengelolaan keuangan dengan menyediakan solusi berbasis teknologi yang dapat memberikan insight mengenai kebiasaan pengeluaran mereka serta membantu dalam pengambilan keputusan keuangan yang lebih bijak.

## Manfaat

Berdasarkan Tujuan diatas, tedapat tujuan pada sistem ini sebagai berikut:

1. Membantu Mahasiswa dalam mengelola keuangan pribadi secara lebih efektif dan terencana.
2. Memberikan wawasan mengenai pola pengeluaran sehingga dapat menghindari kebiasaan boros dan mengatur anggaran dengan lebih baik.
3. Menyediakan rekomendasi berbasis data untuk membantu dalam pengambilan keputusan keuangan yang lebih tepat.

## Batasan Masalah

Batasan masalah bertujuan untuk mempertegas ruang lingkup penyelesaian tugas akhir. Berikut adalah beberapa batasan masalah pada sistem ini:

1. Data yang digunakan dalam aplikasi ini terbatas pada transaksi pemasukan dan pengeluaran mahasiswa, seperti uang saku, biaya makan, transportasi, kebutuhan akademik, serta pengeluaran lainnya yang berkaitan dengan kehidupan sehari-hari mahasiswa.
2. Algoritma Apriori yang digunakan dalam penelitian ini hanya berfungsi untuk menemukan asosiasi antar kategori pengeluaran dan tidak mendukung prediksi jumlah pengeluaran di masa depan.
3. Sistem ini akan dibangun berbasis *mobile* menggunakan *Flutter* untuk frontend dan *Express.js* untuk backend. Sistem ini juga akan menggunakan PostgreSQL sebagai database dan Prisma sebagai *database server*.

# TINJAUAN PUSTAKA

## Penelitian Terkait

Pada penelitian yang dilakukan oleh Wahyuni & Setiawan dengan judul “Analisis Data Mining untuk Rekomendasi Keuangan Mahasiswa Menggunakan Algoritma Apriori”, dijelaskan bahwa mahasiswa sering mengalami kesulitan dalam mengelola keuangan mereka akibat kurangnya pemahaman terhadap pola pengeluaran. Penelitian ini bertujuan untuk menganalisis data transaksi keuangan mahasiswa menggunakan Algoritma Apriori guna menemukan pola asosiasi yang dapat digunakan sebagai rekomendasi keuangan. Metode yang digunakan dalam pengembangan sistem adalah *Software Development Life Cycle (SDLC)* dengan model **Waterfall**, yang mencakup tahapan analisis, perancangan, implementasi, dan pengujian. Hasil dari penelitian ini menunjukkan bahwa sistem yang dikembangkan dapat mengidentifikasi pola pengeluaran mahasiswa serta memberikan saran anggaran yang lebih efektif. Pengujian dilakukan menggunakan metode *Black Box Testing*, yang menunjukkan bahwa aplikasi berjalan sesuai dengan spesifikasi yang dibutuhkan.

Hasil dari penelitian di atas menunjukkan bahwa pemanfaatan Algoritma Apriori dalam sistem rekomendasi keuangan dapat memberikan wawasan yang lebih baik terhadap pola pengeluaran mahasiswa. Dengan pendekatan berbasis data, mahasiswa dapat lebih mudah memahami kebiasaan finansial mereka serta mengelola anggaran dengan lebih bijak. Oleh karena itu, penelitian ini menjadi landasan dalam pengembangan aplikasi Manajemen Keuangan Mahasiswa Berbasis Mobile dengan Algoritma Apriori, yang akan memanfaatkan Flutter untuk frontend serta Express.js, PostgreSQL, dan Prisma untuk backend guna menghasilkan sistem yang lebih interaktif dan optimal dalam membantu mahasiswa dalam mengelola keuangan mereka.

## Teori Penunjang

Dalam penelitian Manajemen Keuangan Mahasiswa dengan Algoritma Apriori, terdapat beberapa teori yang menjadi dasar dalam pengembangan aplikasi, baik dari segi keuangan, data mining, hingga teknologi yang digunakan. Berikut adalah beberapa teori penunjang yang relevan:

1. Manajemen Keuangan Pribadi

Manajemen keuangan pribadi adalah proses perencanaan dan pengelolaan pendapatan, pengeluaran, investasi, serta tabungan seseorang agar mencapai kondisi keuangan yang stabil. Menurut Gitman & Joehnk (2014), manajemen keuangan yang baik mencakup beberapa aspek utama, yaitu perencanaan anggaran, pengendalian pengeluaran, pencatatan keuangan, dan pengambilan keputusan finansial.

1. Algoritma Apriori dalam Data Mining

Algoritma Apriori adalah salah satu metode dalam data mining yang digunakan untuk menemukan pola hubungan antar transaksi dalam sebuah dataset. Algoritma ini dikembangkan oleh Agrawal dan Srikant (1994) dan sering digunakan dalam analisis market basket atau pencarian association rules.

1. Pengembangan Aplikasi Mobile dengan Flutter

Dalam pembuatan tugas akhir ini, Flutter dipilih sebagai teknologi Frontend agar aplikasi yang dikembangkan memiliki antarmuka pengguna yang menarik dan responsif

1. Backend Development dengan Express.js dan PostgreSQL
2. Express.js adalah kerangka kerja (*framework*) berbasis Node.js yang digunakan untuk membangun API yang ringan dan cepat. Express.js mendukung pengelolaan rute, middleware, serta koneksi ke basis data, yang membuatnya ideal untuk aplikasi berbasis RESTful API.
3. PostgreSQL adalah sistem manajemen basis data relasional (RDBMS) yang dikenal karena keandalan dan dukungannya terhadap transaksi kompleks. Dalam penelitian ini, PostgreSQL digunakan untuk menyimpan data transaksi keuangan mahasiswa, termasuk histori pengeluaran dan hasil analisis Algoritma Apriori.
4. Prisma ORM digunakan untuk menghubungkan aplikasi dengan PostgreSQL, mempermudah manajemen database dengan sintaks yang lebih sederhana dan mendukung fitur type-safe queries.

# METODOLOGI

## Analisis Sistem

Metode yang digunakan untuk merancang dan membangun sistem ini yaitu metode waterfall. Berikut adalah tahapan-tahapan dari metode waterfall yang akan dilakukan pada perancangan aplikasi manajemen keuangan mahasiswa dengan algoritma apriori:

1. Identifikasi Masalah dan Kebutuhan

Mahasiswa sering mengalami kesulitan dalam mengatur keuangan, termasuk pengeluaran yang tidak terkontrol, kurangnya perencanaan anggaran, dan minimnya kesadaran terhadap pola keuangan mereka. Sistem ini bertujuan untuk:

1. Memudahkan mahasiswa dalam mencatat pemasukan dan pengeluaran harian.
2. Menganalisis pola pengeluaran menggunakan Algoritma Apriori untuk memberikan rekomendasi pengelolaan keuangan yang lebih baik.
3. Menyajikan laporan keuangan yang interaktif dan mudah dipahami.
4. Alur Kerja Sistem
5. Pencatatan Transaksi
6. Analisis Pola Pengeluaran
7. Pemberian Rekomendasi
8. Visualisasi Data
9. Komponen Sistem
10. Menggunakan Flutter sebagai frontend untuk memastikan pengalaman pengguna yang responsif dan interaktif.
11. Menggunakan Express.js sebagai framework backend untuk menangani API dan pengolahan data.
12. Menggunakan PostgreSQL sebagai database utama untuk menyimpan transaksi pengguna.
13. Prisma ORM membantu dalam pengelolaan data dengan query dengan query yang lebih efisien.
14. Algoritma Apriori untuk membantu sistem dalam memberikan rekomendasi anggaran berdasarkan pola transaksi sebelumnya

## Perancangan Sistem

1. Arsitektur Sistem

Sistem ini menggunakan Client-Server Architecture dengan teknologi sebagai berikut :

1. Menggunakan Flutter sebagai Frontend
2. Menggunakan Express.js sebagai Backend
3. Menggunakan PostgreSQL sebagai database dan Prisma ORM sebagai database server
4. Menggunakan Algoritma Apriori sebagai analisis untuk menemukan pola pengeluaran mahasiswa
5. Desain Antarmuka Pengguna (UI/UX)

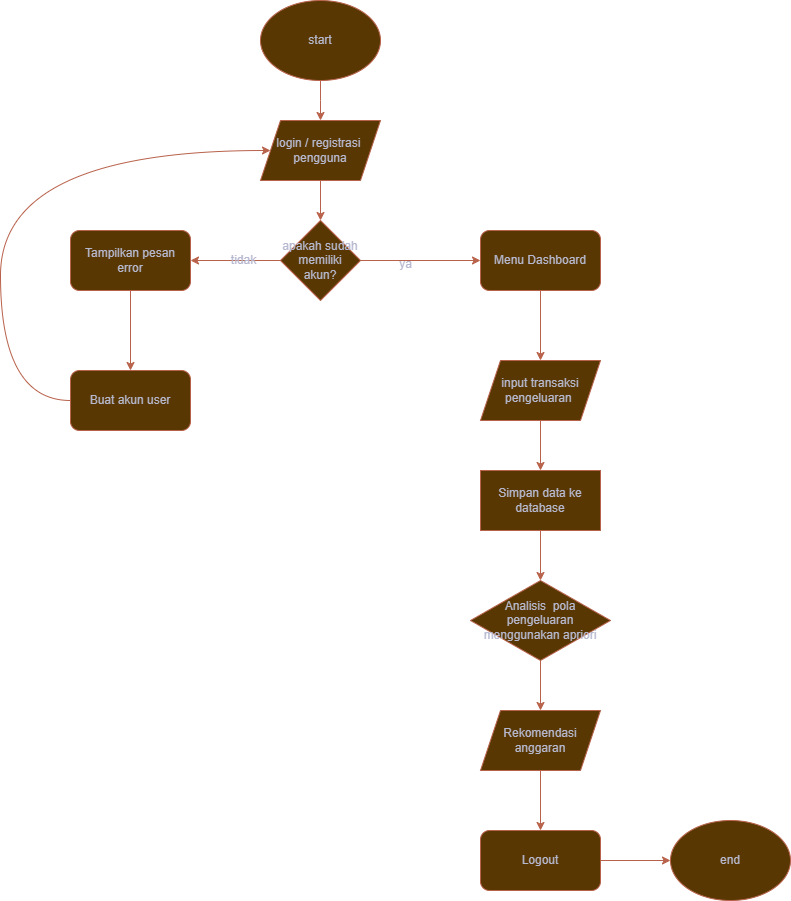
Desain UI/UX dibuat menggunakan Figma dan mencakup beberapa halaman-halaman sebagai berikut :

1. Halaman Login / Register
2. Dashboard Keuangan
3. Pencatatan Transaksi
4. Analisis Pengeluaran menggunakan algoritma apriori
5. Halaman Rekomendasi Keuangan
6. Pengaturan Profil
7. Algoritma Apriori untuk Analisis pola pengeluaran

Sistem menggunakan Algoritma Apriori untuk menganalisis pola transaksi mahasiswa berdasarkan data historis. Langkah-langkah penerapan algoritma ini adalah :

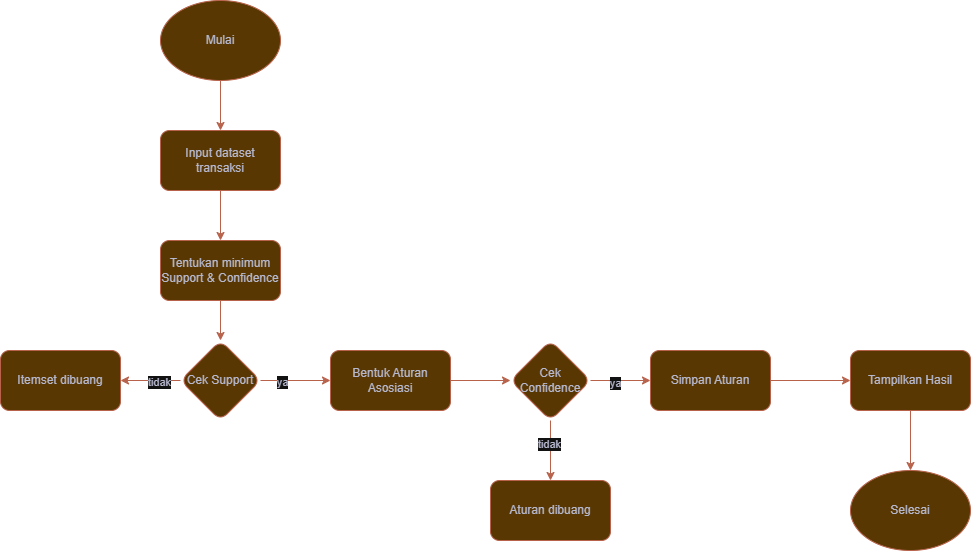
1. Mengelompokkan transaksi mahasiswa ke dalam kategori pengeluaran.
2. Menghitung frekuensi transaksi pada tiap kategori.
3. Menentukan aturan asosiasi untuk memberikan rekomendasi keuangan.

## Flowchart Aplikasi



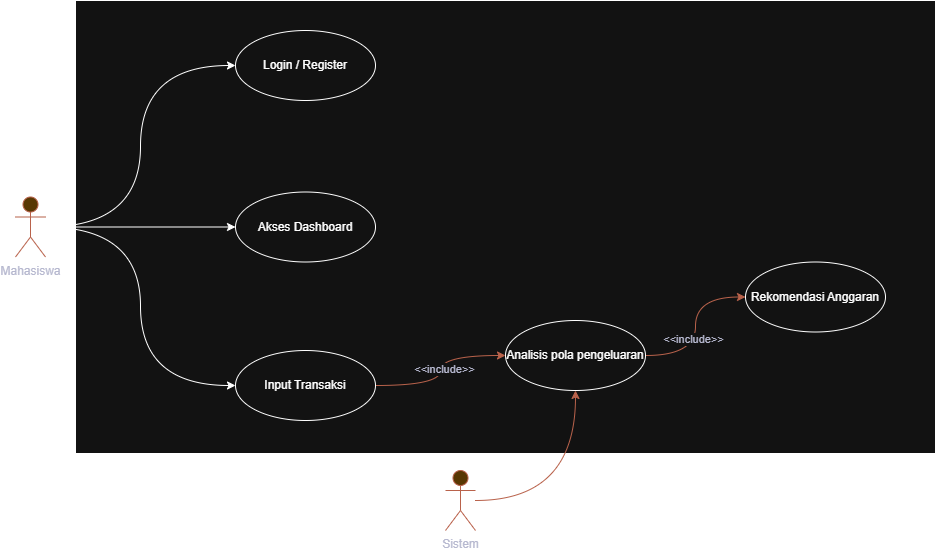
Flowchart diatas merupakan proses alur sistem manajemen keuangan mahasiswa menggunakan algoritma apriori yang diawali dengan proses pembuatan akun. Jika akun ada maka akan dilanjutkan ke halaman dashboard tetapi jika tidak ada maka menampilkan pesan error seperti ”akun anda belum terdaftar” maka harus buat atau registrasi akun terlebih dahulu. Di menu dashboard kita bisa menginputkan transaksi pengeluaran kita yang nantinya akan disimpan ke dalam database. Lalu sistem akan melakukan analisis pola pengeluaran menggunakan Algoritma Apriori untuk menganalisis pengeluran mahasiswa. Ketika analisis selesai maka sistem akan memunculkan rekomendasi anggaran sebagai saran atau pengingat berdasarkan pengeluaran pengguna. Contohnya yaitu saran untuk mengurangi pengeluaran di kategori tertentu.

## Flowchart Algoritma



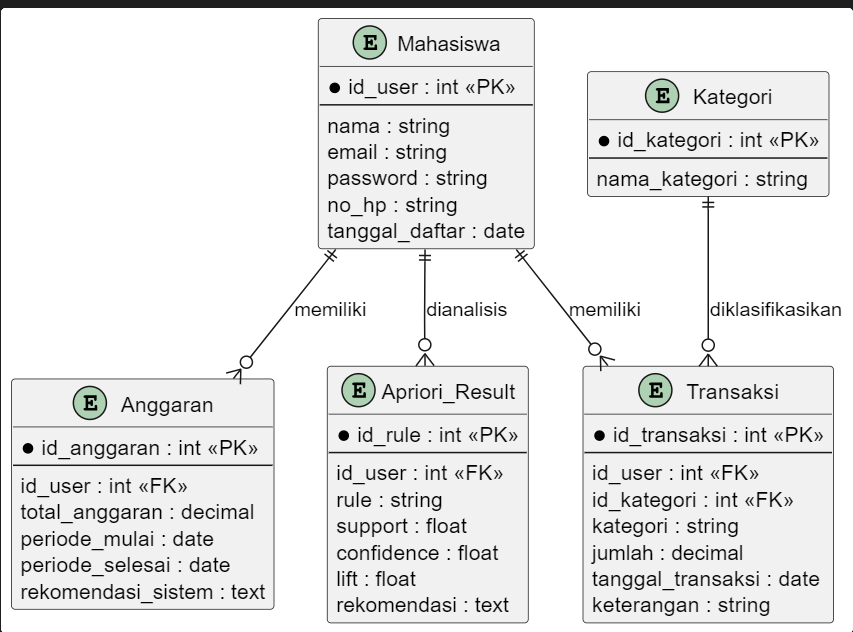
Flowchart algoritma yang digunakan dalam sistem ini menggambarkan proses penerapan algoritma Apriori untuk menemukan pola asosiasi dari data transaksi pengguna. Proses dimulai dari membaca data transaksi yang telah tercatat di dalam sistem. Selanjutnya, data tersebut diproses menjadi bentuk basket berdasarkan kategori pengeluaran bulanan setiap pengguna. Setelah itu, algoritma Apriori dijalankan untuk mencari frequent itemsets berdasarkan nilai support yang ditentukan. Itemset yang lolos akan dilanjutkan ke tahap pembentukan aturan asosiasi (association rules) berdasarkan nilai confidence dan lift. Jika ditemukan rule yang valid, maka hasilnya akan disimpan ke dalam database dan dikirim ke backend aplikasi untuk ditampilkan sebagai rekomendasi. Flowchart ini membantu dalam memahami alur kerja sistem secara logis dan sistematis.

## Use Case Diagram



Use Case diagram dalam aplikasi ini menggambarkan hubungan antara aktor (dalam hal ini adalah pengguna) dengan fungsionalitas utama sistem. Aktor utama yaitu pengguna dapat melakukan berbagai tindakan, seperti login ke dalam aplikasi, menambahkan transaksi keuangan, melihat riwayat transaksi, mengatur anggaran, dan mengakses hasil rekomendasi dari analisis algoritma Apriori. Selain itu, pengguna juga dapat mengelola data profil pribadi, termasuk memperbarui nama, nomor telepon, serta foto profil. Diagram Use Case berfungsi untuk memperjelas cakupan sistem dari sudut pandang pengguna, dan membantu dalam memastikan bahwa semua fitur penting telah diakomodasi dalam perancangan aplikasi.

## ERD (Entity Relationship Diagram)



Entity Relationship Diagram (ERD) menggambarkan hubungan antar tabel dalam basis data yang digunakan oleh sistem. Beberapa entitas utama meliputi tabel user, transaksi, kategori, anggaran, dan hasil\_apriori. Entitas user memiliki relasi satu ke banyak terhadap entitas transaksi, karena satu pengguna dapat memiliki banyak transaksi. Entitas transaksi juga berelasi dengan entitas kategori untuk mengelompokkan jenis pengeluaran atau pemasukan. Tabel anggaran menyimpan alokasi dana dan periode yang ditetapkan oleh pengguna. Sementara itu, hasil\_apriori menyimpan hasil analisis pola dari transaksi yang dianalisis menggunakan algoritma Apriori. ERD ini dirancang untuk memastikan integritas data dan efisiensi dalam penyimpanan serta pengambilan informasi yang dibutuhkan sistem.

## Jadwal Pelaksanaan

Berikut ini adalah jadwal perancangan yang menjadi target dalam merancang manajemen keuangan mahasiswa dengan algoritma apriori:



# IMPLEMENTASI DAN ANALISIS

## Spesifikasi Sistem

1. Perangkat Keras

Disini saya menggunakan Laptop Lenovo Ideapad Slim 3 dengan spesifikasi sebagai berikut :

|  |  |
| --- | --- |
| Laptop | Ideapad Slim 3 |
| Prossesor | Intel Core i3-1115G4 |
| RAM | 8GB DDR4 |
| Penyimpanan | 256 GB |
| Layar | 14 Inci |

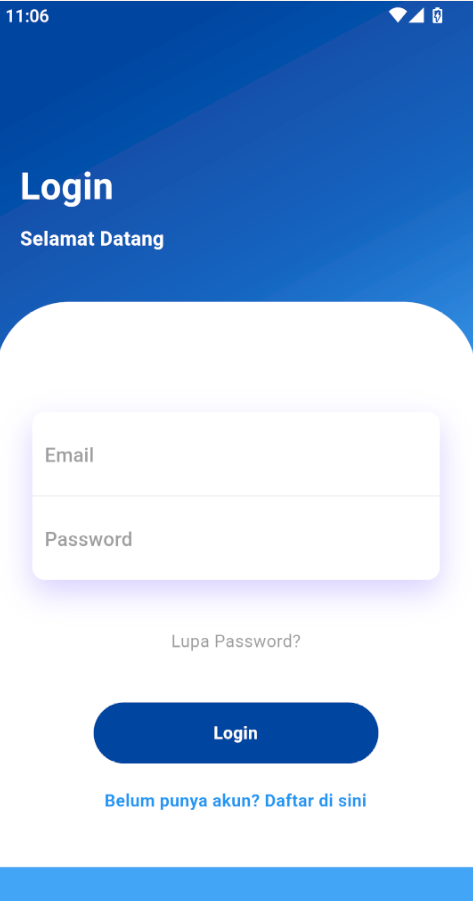
1. Perangkat Lunak

Berikut ini adalah spesifikasi sistem perangkat lunak yang saya gunakan :

|  |  |
| --- | --- |
| Sistem Operasi | Windows 11 64-bit |
| Bahasa Pemograman | Typescript, Python |
| Framework Frontend | Flutter |
| Framework Backend | Express.js |
| Database | PostgreSQL |
| Tools Pendukung | VSCode, Thunder Client, Genymotion |

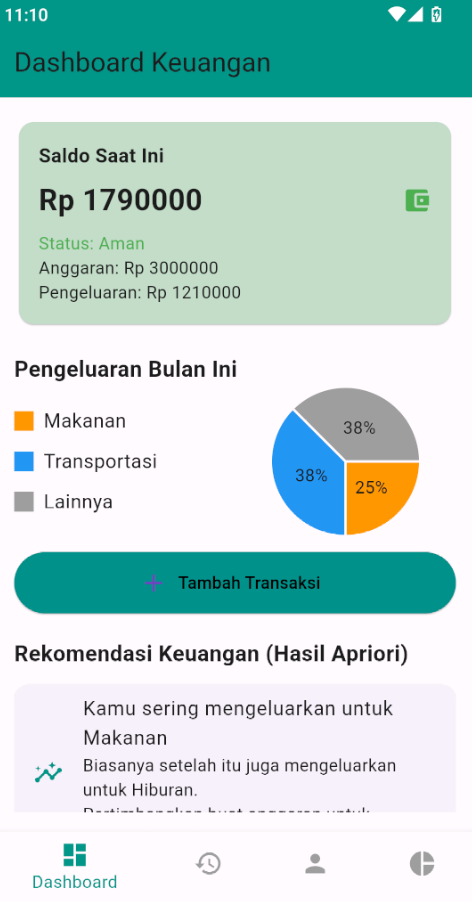
## Implementasi Antarmuka

1. Halaman Login



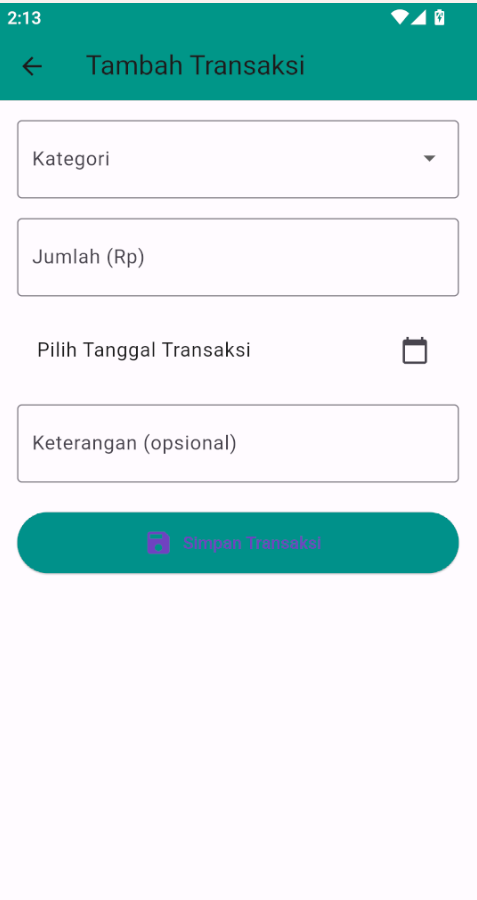
Halaman login merupakan titik awal bagi pengguna untuk mengakses aplikasi. Di dalam halaman ini, pengguna diminta untuk memasukkan alamat email dan kata sandi yang telah terdaftar sebelumnya. Setelah proses autentikasi berhasil dilakukan, sistem akan memberikan token JWT (JSON Web Token) yang kemudian disimpan di local storage sebagai bukti sesi aktif pengguna. Token ini digunakan untuk mengakses halaman-halaman lainnya yang membutuhkan otentikasi pengguna secara aman.

1. Dashboard



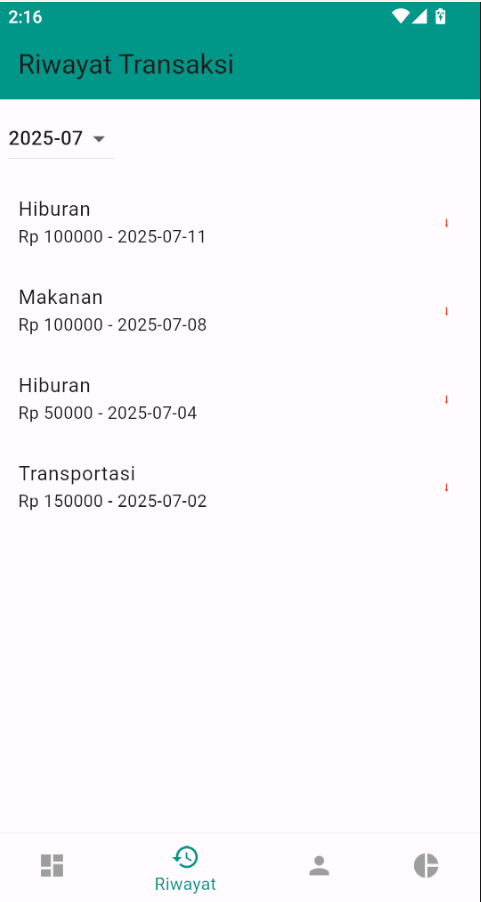
Dashboard berfungsi sebagai pusat informasi utama bagi pengguna setelah berhasil login. Pada halaman ini, ditampilkan ringkasan keuangan pengguna, seperti saldo saat ini, status anggaran, serta grafik pie yang menggambarkan proporsi pengeluaran berdasarkan kategori. Tidak hanya itu, hasil analisis Algoritma Apriori yang berisi rekomendasi keuangan berdasarkan pola pengeluaran pengguna juga disajikan secara ringkas dan mudah dipahami. Hal ini bertujuan memberikan insight secara langsung agar pengguna dapat mengambil keputusan keuangan yang lebih bijak.

1. Tambah Transaksi



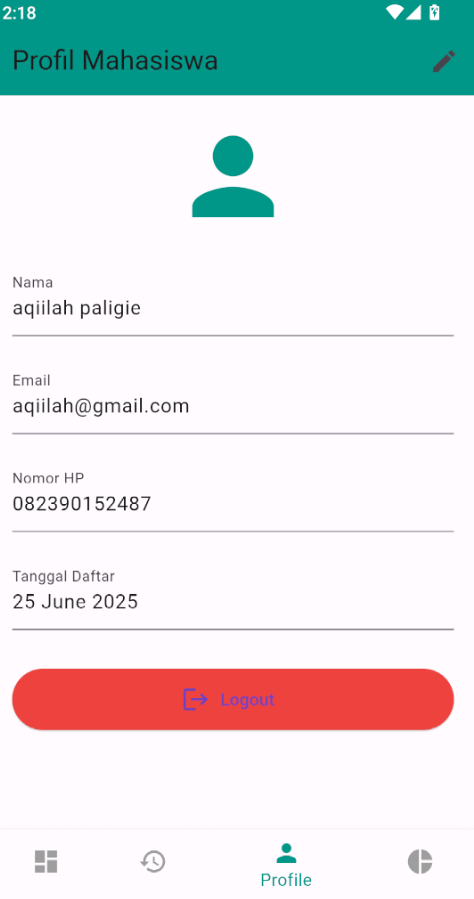
Halaman tambah transaksi memungkinkan pengguna untuk mencatat detail keuangan mereka secara manual. Pengguna dapat memilih kategori transaksi (misalnya: makanan, transportasi, dll.), mengisi jumlah nominal, memilih tanggal transaksi, serta memberikan keterangan tambahan. Setelah data disimpan, sistem akan memperbarui saldo serta memperhitungkan data tersebut dalam analisis keuangan yang akan datang. Fitur ini memastikan seluruh pengeluaran tercatat dengan baik dan menjadi bagian dari histori keuangan pengguna.

1. Riwayat Transaksi



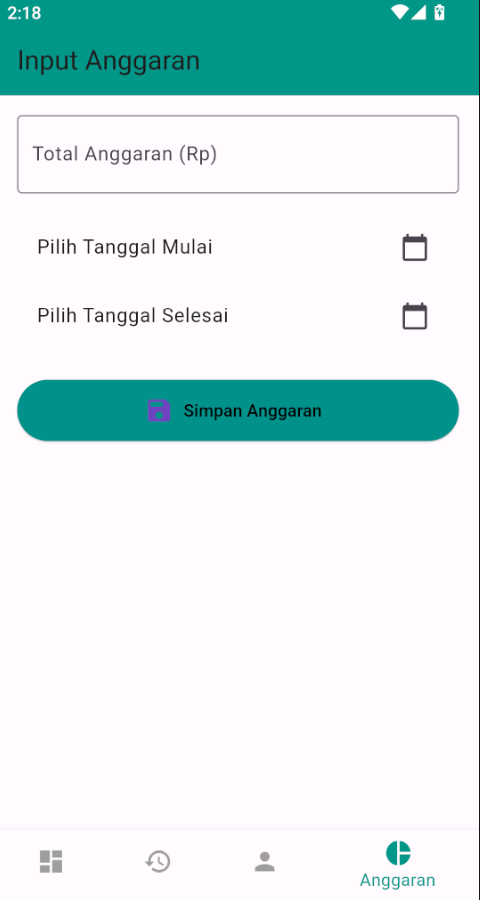
Halaman riwayat transaksi menampilkan daftar seluruh transaksi yang pernah dicatat oleh pengguna. Riwayat ini dapat difilter berdasarkan bulan dan jenis transaksi (pemasukan atau pengeluaran), sehingga pengguna dapat menelusuri pola pengeluaran tertentu dalam periode waktu tertentu. Tampilan disusun secara kronologis dan menyertakan informasi penting seperti kategori, tanggal, jumlah, dan keterangan, menjadikan halaman ini sangat berguna untuk mengevaluasi kebiasaan keuangan.

1. Profile



Halaman profil menampilkan data pribadi pengguna seperti nama lengkap, alamat email, nomor telepon, dan tanggal pendaftaran. Selain menampilkan informasi, pengguna juga diberi opsi untuk mengedit data mereka secara langsung melalui form yang telah disediakan. Selain itu, fitur unggah foto profil juga disediakan untuk memberikan pengalaman pengguna yang lebih personal dan interaktif. Halaman ini menjadi pusat kontrol identitas pengguna dalam sistem.

1. Form Anggaran



Pada halaman ini, pengguna dapat menetapkan anggaran bulanan sesuai kebutuhan dan kemampuan finansial mereka. Form ini mencakup input total anggaran yang ingin dialokasikan serta periode berlakunya. Informasi anggaran ini akan digunakan oleh sistem untuk menghitung sisa saldo dan menilai apakah pengeluaran pengguna telah melampaui batas yang telah ditentukan. Dengan adanya fitur ini, pengguna bisa lebih disiplin dan terarah dalam membelanjakan uangnya.

## Analisis Algoritma

Algoritma yang digunakan adalaha Apriori, dijalankan di bagian backend untuk menghasilkan rekomendasi keuangan berdasarkan pola transaksi sebelumnya.

Berikut ini adalah potongan kode dari algoritma apriori tersebut :

frequent\_itemsets = apriori(basket, min\_support=0.01, use\_colnames=True)

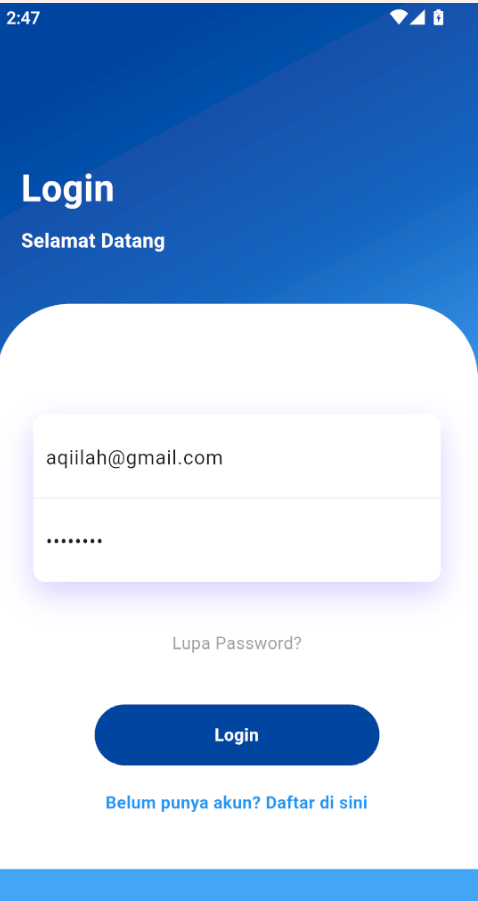
rules = association\_rules(frequent\_itemsets, metric="confidence", min\_threshold=0.1)

Dari codingan diatas python diatas menjelaskan setelah jumlah transaksi mencukupi (> 10), maka sistem akan otomatis mengeksekusi analisis asosiasi dan menyimpan hasilnya ke tabel hasil\_apriori

## Pengujian

Disini akan diuji beberapa function yang ada di project ini :

1. Login



Di halaman login kita memasukkan email dan password yang telah dimasukkan sebelum nya dan ketika email dan password benar maka akan diarahkan ke halaman dashboard.

Ini adalah codingan untuk menguji email dan password yang benar

export const loginUser = async (req: Request, res: Response) => {

    const { email, password } = req.body;

    try {

        const user = await prisma.user.findUnique({ where: { email } });

        if (!user) {

            return res.status(404).json({ message: 'User tidak ditemukan' });

        }

        if (!user.password) {

            return res.status(500).json({ message: 'User tidak memiliki password yang valid' });

        }

        const isPasswordValid = await verifyHash(password, user.password);

        if (!isPasswordValid) {

            return res.status(401).json({ message: 'Password salah' });

        }

        const token = jwt.sign({ id: user.id\_user, email: user.email }, SECRET, { expiresIn: '1d' });

        return res.status(200).json({

            message: 'Login berhasil',

            token,

            data: {

                id\_user: user.id\_user,

                nama: user.nama,

                email: user.email,

                no\_hp: user.no\_hp,

                tanggal\_daftar: user.tanggal\_daftar,

            },

        });

    } catch (error) {

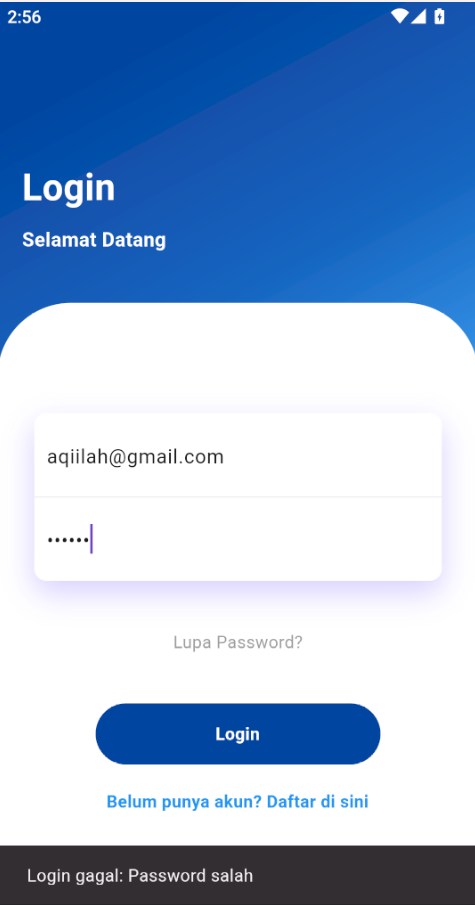
        console.error('Login error:', error);

        return res.status(500).json({ message: 'Terjadi kesalahan server' });

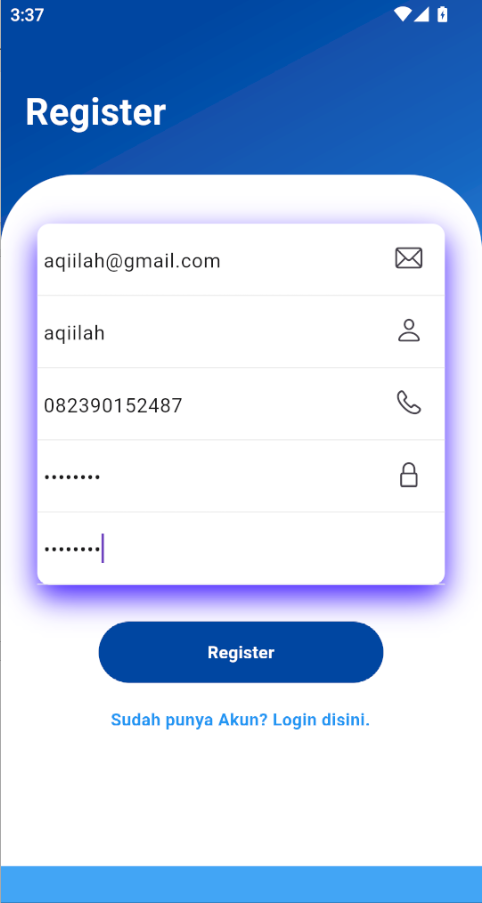
    }

};

Lalu ada juga pengujian ketika login gagal tampilan nya akan seperti ini



1. Register



Di halaman register, kita akan memasukkan email, nama, nomor telepon, password, dan confirm password dan ketika berhasil maka akan diarahkan ke halaman login. Ini adalah codingan untuk menguji pembuatan akun agar berhasil

export const register = async (req: Request, res: Response) => {

    const errors = validationResult(req);

    if (!errors.isEmpty()) {

        return res.status(400).json({ errors: errors.array() });

    }

    try {

        const user = await registerMahasiswa(req.body);

        const token = jwt.sign(

            { id: user.id\_user, email: user.email },

            JWT\_SECRET,

            { expiresIn: '1d' }

        );

        res.status(200).json({

            message: 'Pendaftaran berhasil',

            user,

            token,

        });

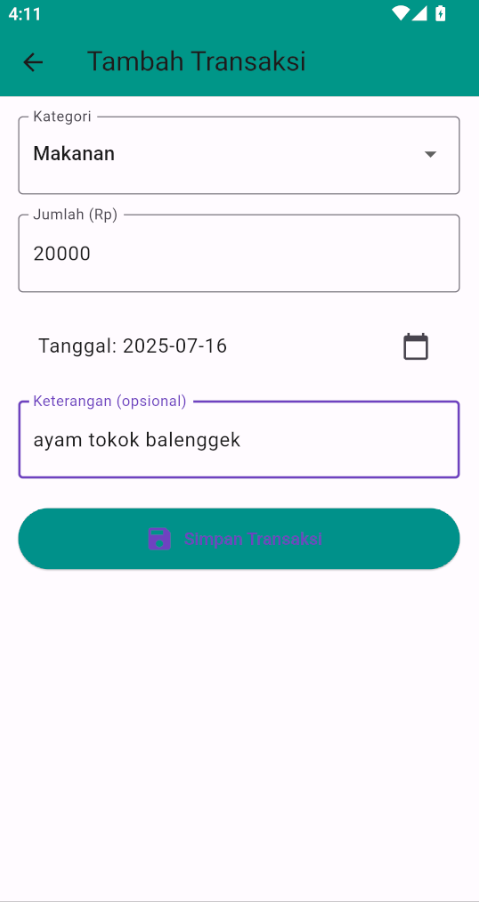
    } catch (error: any) {

        res.status(400).json({ error: error.message });

    }

};

1. Tambah Transaksi (POST)



Di halaman Tambah Transaksi ini kita mengisi form kategori, harga dari kategori, tanggal transaksi, dan keterangan. Ketika simpan transaksi berhasil maka akan diarahkan ke halaman dashboard dan akan menampilkan grafik dari transaksi tersebut. Ini adalah code untuk menangani tambah transaksi jika berhasil

export const tambahTransaksi = async (req: AuthRequest, res: Response) => {

    try {

        const { id\_kategori, jumlah, tanggal\_transaksi, keterangan } = req.body;

        const id\_user = req.user?.id;

        if (!id\_user) {

            return res.status(401).json({ message: 'User tidak terautentikasi' });

        }

        if (!id\_kategori || !jumlah || !tanggal\_transaksi) {

            return res.status(400).json({ message: 'Semua field wajib diisi' });

        }

        const kategori = await prisma.kategori.findUnique({

            where: { id\_kategori: Number(id\_kategori) },

        });

        if (!kategori) {

            return res.status(404).json({ message: 'Kategori tidak ditemukan' });

        }

        const transaksi = await prisma.transaksi.create({

            data: {

                id\_user: id\_user,

                id\_kategori: Number(id\_kategori),

                kategori\_nama: kategori.nama\_kategori,

                jumlah: parseFloat(jumlah),

                tanggal\_transaksi: new Date(tanggal\_transaksi),

                keterangan: keterangan || null,

            },

        });

        return res.status(201).json(transaksi);

    } catch (error) {

        console.error('Gagal tambah transaksi:', error);

        return res.status(500).json({ message: 'Gagal menambahkan transaksi' });

    }

};

1. Riwayat Transaksi (GET)

export const getTransaksi = async (req: AuthRequest, res: Response) => {

const id\_user = req.user?.id;

const { bulan, jenis } = req.query;

try {

const whereClause: any = { id\_user };

if (bulan && typeof bulan === 'string') {

const startDate = new Date(`${bulan}-01T00:00:00`);

const endDate = new Date(startDate);

endDate.setMonth(endDate.getMonth() + 1);

whereClause.tanggal\_transaksi = {

gte: startDate,

lt: endDate,

};

}

if (jenis && typeof jenis === 'string') {

whereClause.kategori\_relasi = { tipe: jenis };

}

const transaksi = await prisma.transaksi.findMany({

where: whereClause,

include: { kategori\_relasi: true },

orderBy: { tanggal\_transaksi: 'desc' },

});

const formatted = transaksi.map((item) => ({

...item,

tanggal\_transaksi: item.tanggal\_transaksi.toISOString().split('T')[0],

}));

res.json(formatted);

} catch (error) {

console.error(error);

res.status(500).json({ error: 'Gagal memuat transaksi' });

}

};

1. Tambah Anggaran (POST)

router.post('/anggaran', verifyToken, async (req: AuthRequest, res: Response) => {

const { total\_anggaran, periode\_mulai, periode\_selesai } = req.body;

if (!total\_anggaran || !periode\_mulai || !periode\_selesai) {

return res.status(400).json({ message: 'Semua field wajib diisi' });

}

try {

const anggaran = await prisma.anggaran.create({

data: {

id\_user: req.user!.id,

total\_anggaran: parseFloat(total\_anggaran),

periode\_mulai: new Date(periode\_mulai),

periode\_selesai: new Date(periode\_selesai),

},

});

res.status(201).json(anggaran);

} catch (error) {

console.error(error);

res.status(500).json({ error: 'Gagal menyimpan anggaran' });

}

});

1. Profile (GET)

export const getProfile = async (req: AuthRequest, res: Response) => {

try {

const id\_user = req.user?.id;

if (!id\_user) {

return res.status(401).json({ message: 'Tidak ada user dari token' });

}

const user = await prisma.user.findUnique({

where: { id\_user },

select: {

nama: true,

email: true,

no\_hp: true,

tanggal\_daftar: true,

},

});

if (!user) {

return res.status(404).json({ message: 'User tidak ditemukan' });

}

return res.json(user);

} catch (error) {

console.error('Gagal mengambil profil:', error);

return res.status(500).json({ message: 'Gagal mengambil profil' });

}

};

1. Profile (PUT)

export const updateProfile = async (req: AuthRequest, res: Response) => {

const id\_user = req.user?.id;

const { nama, email, no\_hp } = req.body;

if (!nama || !email || !no\_hp) {

return res.status(400).json({ message: 'Semua field wajib diisi' });

}

try {

const updatedUser = await prisma.user.update({

where: { id\_user },

data: {

nama,

email,

no\_hp,

},

});

res.json({

message: 'Profil berhasil diperbarui',

user: {

nama: updatedUser.nama,

email: updatedUser.email,

no\_hp: updatedUser.no\_hp,

},

});

} catch (err) {

console.error(err);

res.status(500).json({ message: 'Gagal memperbarui profil' });

}

};

# PENUTUP

## Kesimpulan

Berdasarkan tujuan yang telah dirumuskan pada Bab I, dapat disimpulkan bahwa pengembangan aplikasi manajemen keuangan mahasiswa berbasis mobile telah berhasil direalisasikan dengan baik. Aplikasi ini memberikan kemudahan bagi mahasiswa untuk mencatat pemasukan dan pengeluaran, memantau saldo dan status anggaran secara real-time, serta mendapatkan rekomendasi keuangan yang relevan berdasarkan riwayat transaksi mereka. Penerapan algoritma Apriori terbukti mampu menghasilkan pola asosiasi antar kategori pengeluaran yang kemudian disajikan dalam bentuk rekomendasi yang mudah dipahami. Teknologi yang digunakan, seperti Flutter untuk antarmuka pengguna dan Express.js dengan PostgreSQL untuk sisi backend, mendukung performa dan fleksibilitas aplikasi.

## Saran

Saran untuk pengembangan aplikasi ke depannya adalah menambahkan fitur notifikasi otomatis ketika saldo mendekati batas anggaran, fitur ekspor laporan keuangan dalam format PDF, serta integrasi teknologi seperti QRIS atau pemindai struk untuk pencatatan otomatis transaksi. Selain itu, penerapan opsi mode gelap (dark mode) serta kustomisasi antarmuka dapat meningkatkan kenyamanan pengguna dalam menggunakan aplikasi ini.

# DAFTAR PUSTAKA

Affan, S., & dkk. (2022). *Aplikasi Pencarian dan Pengumuman Barang Hilang “Carlik-Caripemilik” Berbasis Mobile*. Jurnal Teknologi Informasi, **14**(2), 112-124.

Asmoro, S. S. (2021). *Aplikasi Pencarian Barang Hilang di Kota Solo Berbasis Web*. Jurnal Sistem Informasi, **10**(1), 45-58.

Han, J., Pei, J., & Kamber, M. (2011). *Data Mining: Concepts and Techniques (3rd ed.)*. Morgan Kaufmann.

Heaton, J. (2016). *Introduction to Neural Networks with Java (2nd ed.)*. Heaton Research.

Liao, S.-H., Chu, P.-H., & Hsiao, P.-Y. (2012). Data Mining Techniques and Applications – A Decade Review from 2000 to 2011. *Expert Systems with Applications*, **39**(12), 11303-11311.

Rahardjo, B. (2017). *Keamanan Sistem Informasi*. Informatika Bandung.

Suyanto, E. (2018). *Machine Learning: Konsep Dasar dan Aplikasi*. Informatika Bandung.

Witten, I. H., Frank, E., & Hall, M. A. (2016). *Data Mining: Practical Machine Learning Tools and Techniques (4th ed.)*. Morgan Kaufmann.

Zaki, M. J., & Meira, W. (2014). *Data Mining and Machine Learning: Fundamental Concepts and Algorithms*. Cambridge University Press.