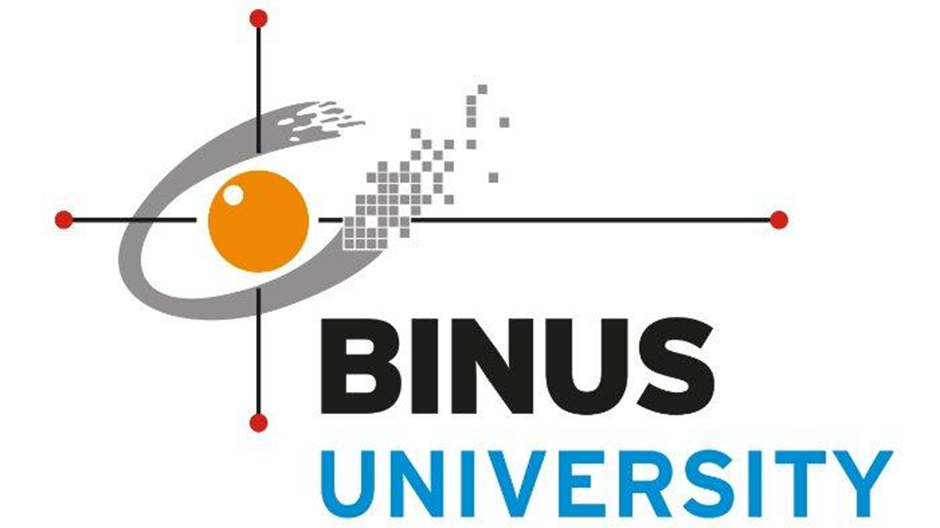
**LAPORAN AKHIR PERANCANGAN APLIKASI PENGELOLAAN FINANSIAL BERBASIS MOBILE BERNAMA CASHCOACH**



**DISUSUN OLEH:**

**NICHOLAS DECLAN MAROJAHAN HUTAPEA - 2502016001**

**MUHAMAD INDRA FAIZAL - 2502030100**

**MATTHIEAU VIC KUSTANTO - 2502017824**

**MUHAMMAD AKMAL KHARRAZI - 2502015983**

**MUHAMMAD FATHAN KASYFILLAH HERMAWAN - 2502023461**

**PROGRAM STUDI : COMPUTER SCIENCE**

**FAKULTAS COMPUTER SCIENCE**

**UNIVERSITAS BINA NUSANTARA**

**ALAM SUTERA**

**BAB 1**

**PENDAHULUAN**

**1.2. Rumusan Masalah**

Literasi finansial merupakan kemampuan seseorang untuk mengelola keadaan finansial yang mempengaruhi kesejahteraan kehidupannya. Literasi finansial masyarakat juga menjadi salah satu faktor kesejahteraan suatu negara. Oleh karena itu, meningkatkan literasi finansial masyarakat menjadi salah satu upaya penting untuk meningkatkan ekonomi negara.

Meskipun dengan pentingnya literasi finansial, Indonesia masih memiliki tingkat literasi finansial yang cukup rendah, terutama pada kaum generasi z. Menurut Otoritas Jasa Keuangan (OJK), persentase generasi z yang memiliki literasi finansial yang cukup baru mencapai sekitar 27%. Meskipun angka yang rendah, generasi z memiliki potensi untuk berkembang lebih daripada generasi lain menurut dewan OJK, Yudha Nugraha. Beliau percaya bahwa akses *smartphone* yang dimiliki oleh kaum gen z dapat membantu mereka dalam meningkatkan literasi finansial.

Berdasarkan latar belakang masalah sebelumnya, maka masalah yang perlu diuraikan dan dipecahkan menurut penulis adalah:

1. Bagaimana cara meningkatkan tingkat literasi finansial di kalangan gen z?
2. Fitur apa saja yang perlu digunakan suatu aplikasi untuk meningkatkan literasi finansial
3. Bagaimana agar pengguna aplikasi terus menggunakan aplikasinya agar meningkatkan literasi finansial mereka?

**1.2. Solusi Masalah**

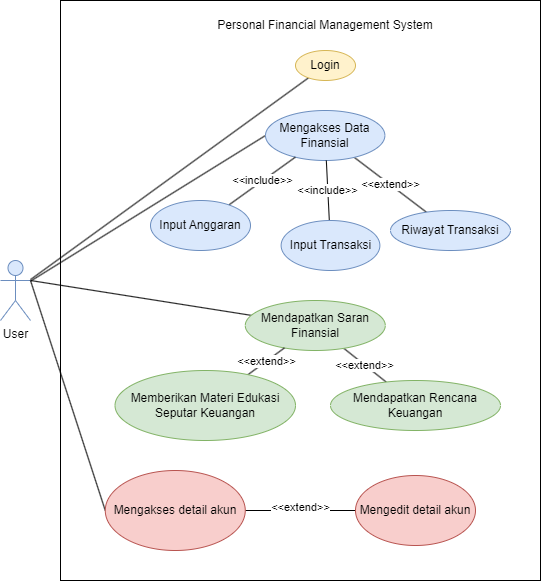
1. Meningkatkan literasi finansial, khususnya pada kaum mahasiswa yang memiliki kesulitan mengelola kondisi finansial mereka secara konsisten.
2. Memberikan platform untuk mengatur finansial pengguna dengan UI yang interaktif dan mudah dimengerti.
3. Memberikan materi pengelolaan finansial yang sesuai dengan tujuan pengguna.

**BAB 2**

**PENJELASAN PROSES MODEL**

**2.1. Use Case Diagram**

Berikut adalah *use case diagram* yang kami telah rancang untuk aplikasi kami yang mempunyai 3 fitur utama, yaitu *login*, akses data finansial, dan saran finansial:

****

**2.2. Use Case Narrative**

Kami telah membuat beberapa use case narrative untuk fitur-fitur utama kami, yang tertera di bawah.

**2.2.1. Login**

Berikut adalah *Use Case Narrative* yang telah kami rancang untuk fitur *Login* pada aplikasi kami:

| **Use Case** | Login |
| --- | --- |
| **Primary Actor** | pengguna |
| **Secondary Actor(s)** | System |
| **Goal in context** | pengguna ingin mengakses layanan aplikasi. |
| **Pre-conditions** | 1. Sistem memiliki database yang dapat menyimpan informasi akun pengguna. 2. pengguna telah sebelumnya melakukan registrasi akun. |
| **Trigger** | pengguna membuka aplikasi untuk pertama kalinya, atau setelah *logout* dari akun yang dimiliki. |
| **Includes/Extends/Inherits** | - |
| **Scenario** | 1. pengguna membuka aplikasi. 2. pengguna melakukan *login* dengan akun yang sudah diregistrasi. 3. Apabila email dan *password* yang diinput pengguna ada di dalam database, maka pengguna akan dapat masuk ke aplikasi. |
| **Exceptions** | 1. Akun pengguna tidak sama dengan database (*email*, *password*) 2. pengguna ingin mengakses page layanan dengan tanpa perlu login sebelumnya. |
| **Priority** | Sangat penting, harus diimplementasikan. |
| **When Available** | Interaksi pertama. |
| **Frequency of Use** | Sekali, saat pertama kali masuk ke dalam aplikasi. |
| **Channel to Actor** | Via antarmuka *log in*. |
| **Channels to Secondary Actor(s)** | System: tombol bantuan |
| **Open Issues** | 1. Apakah password harus memiliki syarat huruf, simbol, atau panjang tertentu? 2. Apakah *password* yang diinput dalam *password field* dapat dilihat? 3. Apakah *password* dan *email* dapat digantikan? 4. Bisakah *login* menggunakan akun dari platform lain (Google, LINE, Gojek)? |
| **Non-functional Requirements** | 1. Keamanan password dan informasi yang digunakan untuk akun. 2. Respon server yang cepat saat *login*. |

**2.2.2. Mengakses Data Finansial**

Berikut adalah *Use Case Narrative* yang telah kami rancang untuk fitur Mengakses Data Finansial pada aplikasi kami:

| **Use Case** | Mengakses Data Finansial |
| --- | --- |
| **Primary Actor** | pengguna |
| **Secondary Actor(s)** | System |
| **Goal in context** | pengguna ingin melihat, melacak, dan menginput data finansial yang dimiliki. |
| **Pre-conditions** | 1. Sistem memiliki database yang dapat menyimpan informasi finansial pengguna (anggaran, transaksi) 2. pengguna telah sebelumnya menginput data finansial (anggaran, transaksi) 3. Aplikasi diprogram untuk dapat menampilkan informasi finansial pengguna dalam bentuk grafis. 4. Aplikasi diprogram untuk dapat memprediksi total anggaran dalam rentang waktu tertentu berdasarkan jumlah transaksi. |
| **Trigger** | pengguna membuka aplikasi (setelah tahap *login*). |
| **Includes/Extends/Inherits** | Includes: Input Anggaran, Input Transaksi, Riwayat Transaksi |
| **Scenario** | 1. pengguna membuka aplikasi. 2. Aplikasi secara *default* akan memperlihatkan *home screen* yang menunjukkan *overview* data finansial pengguna. 3. pengguna memencet tombol untuk melihat detail lebih lanjut. 4. Aplikasi akan menampilkan semua data finansial pengguna (anggaran, dan riwayat transaksi). 5. pengguna dapat menginput data. |
| **Exceptions** | 1. pengguna belum *login* menggunakan akun. 2. pengguna tidak terkoneksi dengan jaringan internet. |
| **Priority** | Sangat penting, harus diimplementasikan. |
| **When Available** | Setelah pengguna membuka aplikasi (setelah tahap *login*). |
| **Frequency of Use** | Setiap kali menggunakan aplikasi |
| **Channel to Actor** | Via *navigation bar* dan antarmuka *data finansial*. |
| **Channels to Secondary Actor(s)** | - |
| **Open Issues** | 1. Apakah data yang ditunjukkan sesuai dengan data yang dimasukkan? 2. Apakah anggaran dapat dikoneksikan dengan saldo uang di platform lain? (GoPay, ShopeePay, dll) |
| **Non-functional Requirements** | 1. Keamanan informasi data finansial yang disimpan. 2. Keakuratan data dan prediksi finansial. 3. Respons server yang cepat saat menginput data. |

**2.2.3. Memberikan Saran Finansial**

Berikut adalah *Use Case Narrative* yang telah kami rancang untuk fitur Memberikan Saran Finansial pada aplikasi kami:

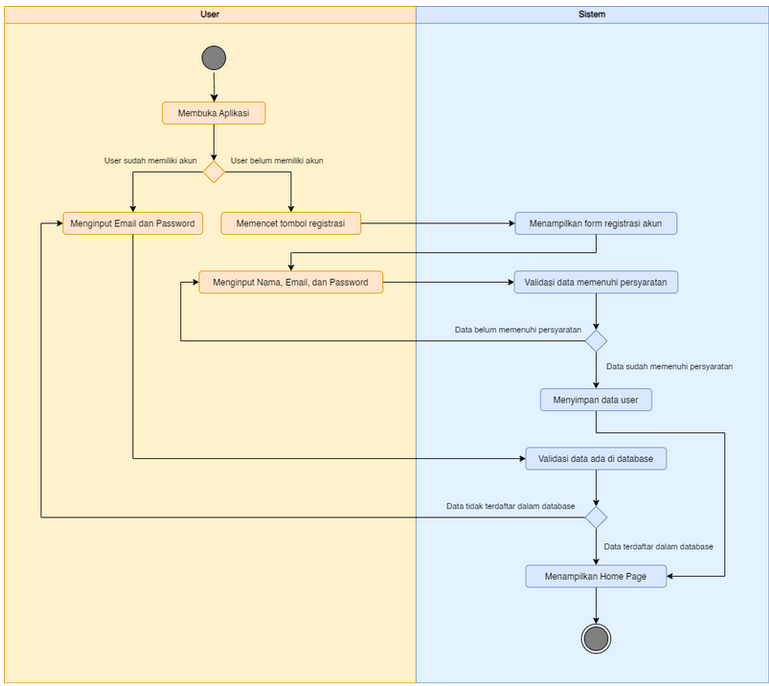
| **Use Case** | Memberikan Saran Finansial |
| --- | --- |
| **Primary Actor** | pengguna |
| **Secondary Actor(s)** | System |
| **Goal in context** | pengguna ingin mengakses saran finansial yang diberikan oleh sistem |
| **Pre-conditions** | 1. pengguna sudah login 2. pengguna telah memasukkan anggaran |
| **Trigger** | pengguna mengakses menu “login” |
| **Includes/Extends/Inherits** | Extends: Mendapatkan informasi identitas profile pengguna layanan. |
| **Scenario** | 1. pengguna telah menginput informasi privasi. 2. pengguna ingin mendapatkan informasi account untuk mengatur fiturnya. 3. pengguna mengakses menu login 4. pengguna menentukan menu fitur fiturnya. 5. Sistem mengeluarkan beberapa menu untuk pengguna. 6. pengguna memilih fitur layanan di Page utama. |
| **Exceptions** | 1. pengguna belum login 2. pengguna tidak mempunyai anggaran |
| **Priority** | Sedang |
| **When Available** | Sebelum login |
| **Frequency of Use** | Beberapa kali |
| **Channel to Actor** | Via navigation bar |
| **Channels to Secondary Actor(s)** | System: meminta ide finansial |
| **Open Issues** | 1. Apakah ide yang ditunjukkan sesuai dengan tujuan pengguna? 2. Apakah tujuan yang diinput pengguna realistis? |
| **Non-functional Requirements** | 1. Kecepatan sistem mengeluarkan ide finansial |

**2.3. Activity Diagram**

Kami telah merancang beberapa *activity diagram* untuk menjelaskan alur kerja setiap fitur-fitur dalam aplikasi kami.

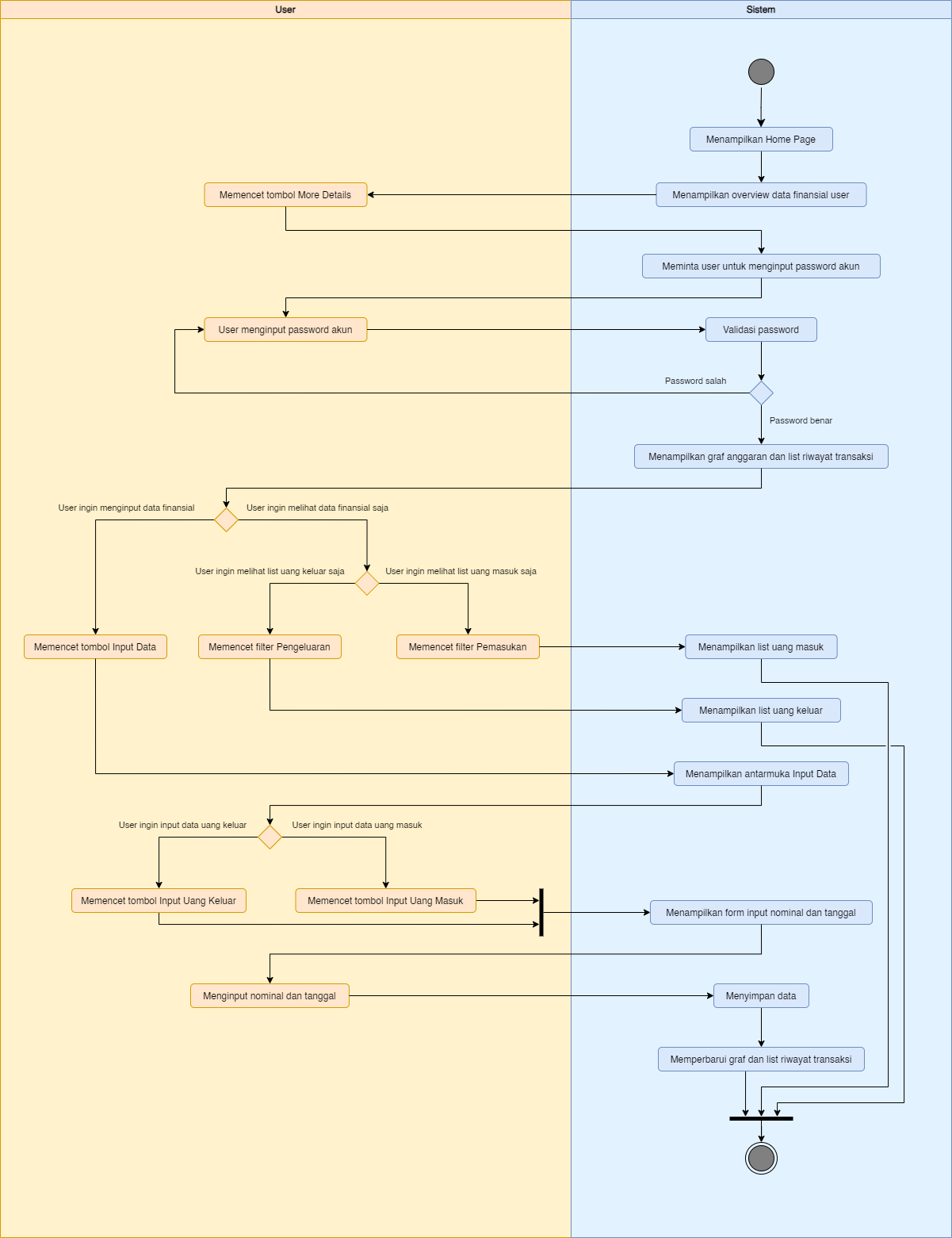
**2.3.1. Login**

Berikut adalah *activity diagram* untuk fitur *Login*:

****

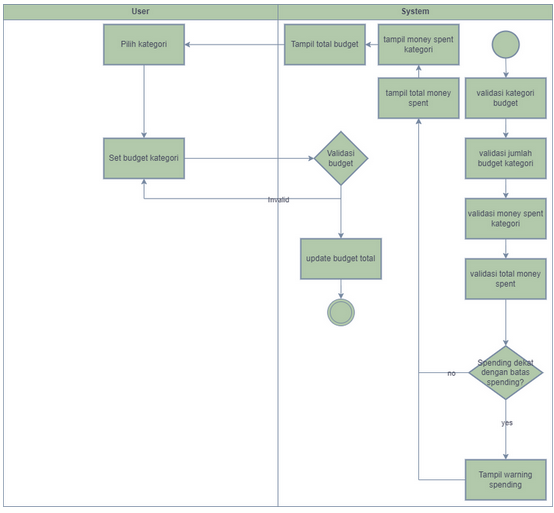
**2.3.2. Financial Details**

Berikut adalah *activity diagram* untuk fitur *Financial Details*:

****

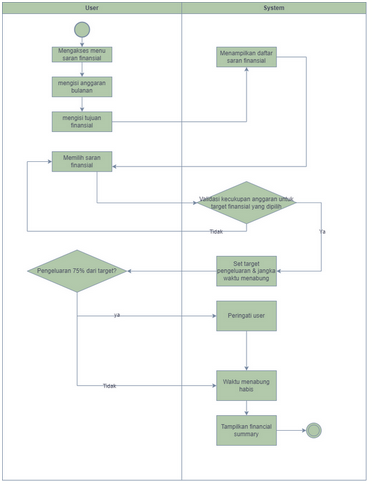
**2.3.3. Budget Setting**

Berikut adalah *activity diagram* untuk fitur *budget setting*:

****

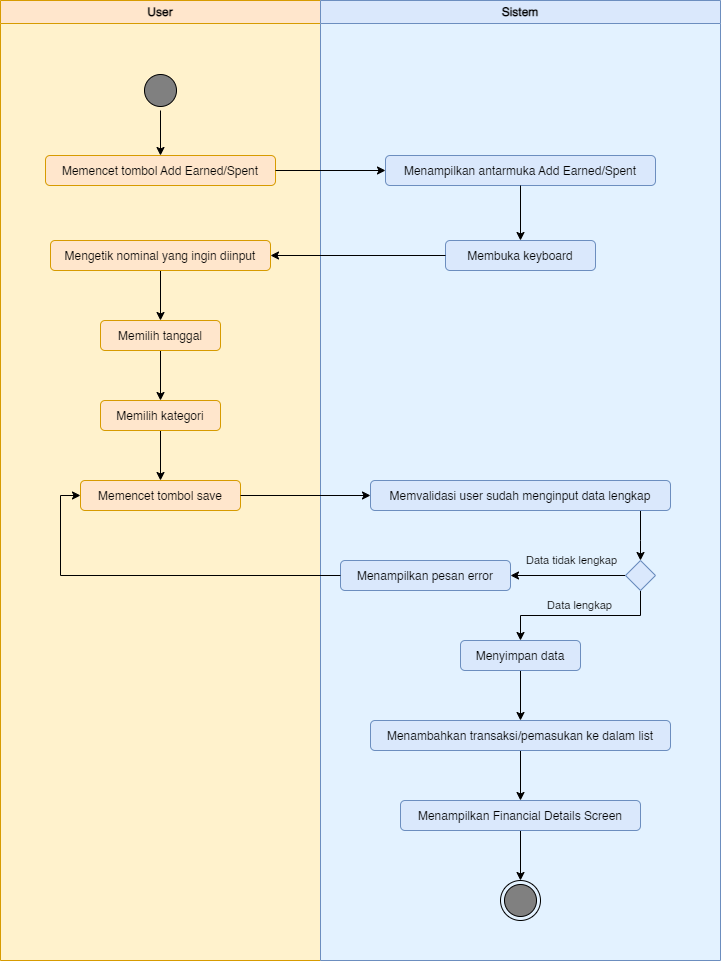
**2.3.4. Learning Financial Management**

Berikut adalah *activity diagram* untuk fitur *learning financial management*:

****

**2.3.5. Add Earned/Spent**

Berikut adalah *activity diagram* untuk fitur *add earned/spent*:

****

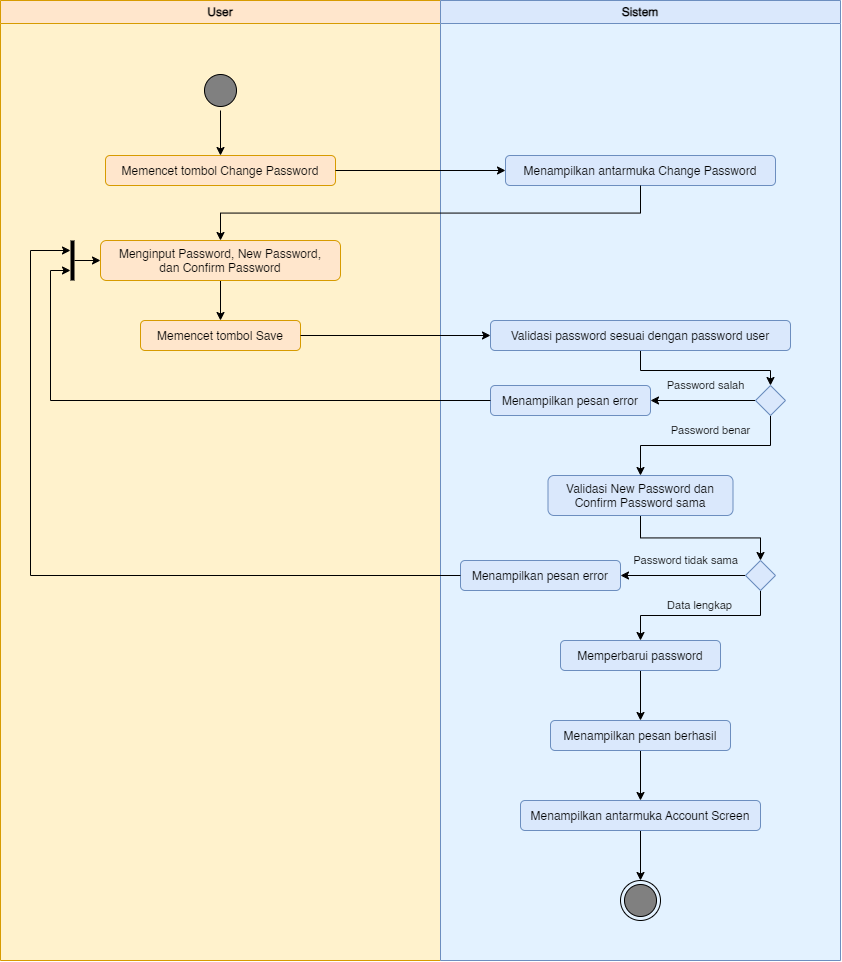
**2.3.6. Edit Profile**

Berikut adalah *activity diagram* untuk fitur *edit profile*:

****

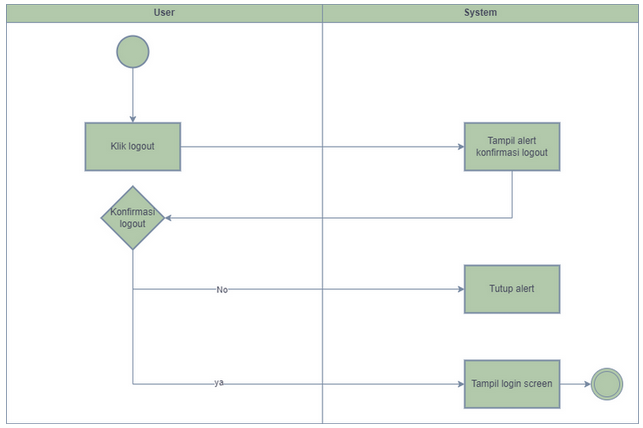
**2.3.7. Change Password**

Berikut adalah *activity diagram* untuk fitur *change password*:

****

**2.3.8. Logout**

Berikut adalah *activity diagram* untuk fitur *logout*:

****

**2.4. Class Diagram**

Di bawah adalah lampiran *class diagram* yang telah kami rancang untuk aplikasi kami, yang mempunyai beberapa *class* seperti pengguna atau *User*, *Account*, *Admin*. *Article*, *Budget*, *TransactionGroup*, dan *Transaction*.

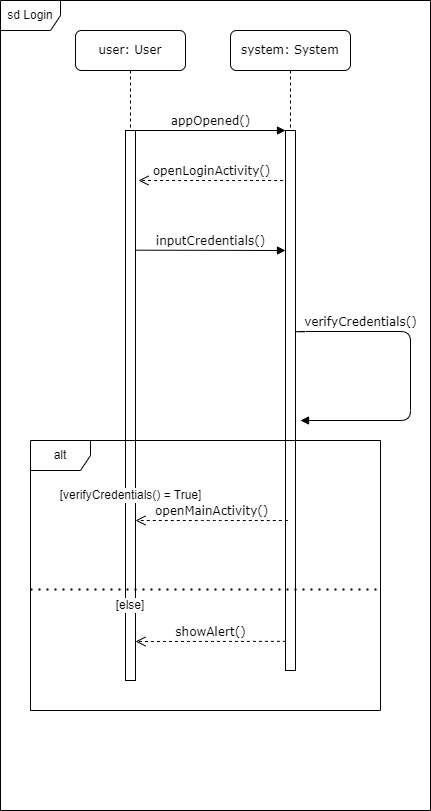
****

**2.5. Sequence Diagram**

Kami membuat beberapa *sequence diagram* untuk fitur-fitur utama dalam aplikasi kami, yang tertera di bawah.

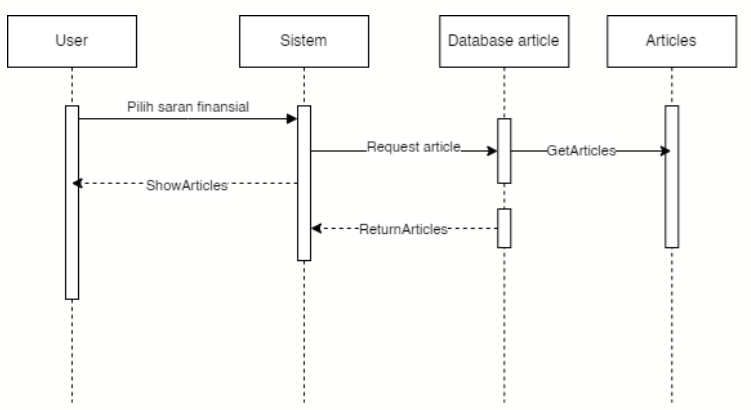
**2.5.1. Login**

Berikut adalah *sequence diagram* untuk fitur *login*:

****

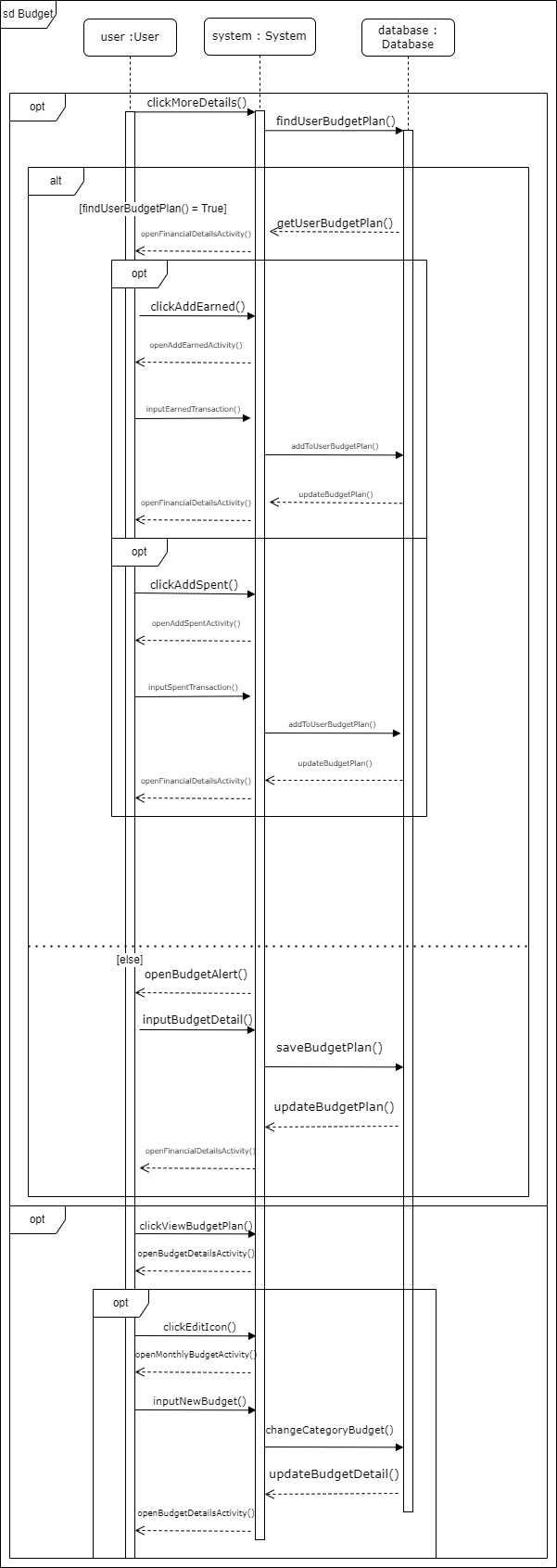
**2.5.2. Financial Education**

Berikut adalah *sequence diagram* untuk fitur *financial education*:

****

**2.5.3. Financial Tracking**

Berikut adalah *sequence diagram* untuk fitur *financial tracking*:

****

**2.6. Testing Scenario**

Di bawah ini adalah lampiran *testing scenario* untuk fitur-fitur utama dalam aplikasi kami dengan kepentingannya:

| **Personal Financial Management Mobile App Test Scenario** | | | |
| --- | --- | --- | --- |
| **Module** | **Importance** | **TestScenarioId** | **Test Scenario Description** |
| Login | Mandatory | TS\_01 | Verify that a pengguna can successfully log in with valid credentials. |
| Mandatory | TS\_02 | Verify that an error message is displayed when invalid credentials are entered. |
| Mandatory | TS\_03 | Verify that the "Forgot Password" functionality allows users to reset their password. |
| Important | TS\_04 | Verify that the system enforces password complexity rules during the login process. |
| Register | Mandatory | TS\_05 | Verify that a new user can successfully register with valid credentials. |
| Mandatory | TS\_06 | Verify that an error message is displayed when registering with invalid or incomplete information. |
| Mandatory | TS\_07 | Verify that the system checks for unique usernames during the registration process. |
| Mandatory | TS\_08 | Verify that an error message is displayed when attempting to register with an existing username. |
| Important | TS\_09 | Verify that the system enforces password complexity rules during the registration process. |
| Financial Tracker | Mandatory | TS\_10 | Verify that user can add a new income transaction and it reflects in the tracker. |
| Mandatory | TS\_11 | Verify that user can add a new expense transaction and it reflects in the tracker. |
| Mandatory | TS\_12 | Verify that user can categorize transactions and view income/expense breakdown. |
| Important | TS\_13 | Verify that the app calculates and displays total income and total expense for the month. |
| Important | TS\_14 | Verify that the app displays a graphical representation of income and expense trends over time. |
| Mandatory | TS\_15 | Verify that user can edit or delete a transaction and it updates the tracker accordingly. |
| Important | TS\_16 | Verify that user can search and filter transactions based on various criteria. |
| Important | TS\_17 | Verify that the app provides a summary of user's financial health based on income and expenses. |
| Budget Planner | Mandatory | TS\_18 | Verify that user can set a monthly budget and it is reflected in the planner. |
| Important | TS\_19 | Verify that user can view a summary of planned expenses and compare it with actual expenses. |
| Important | TS\_20 | Verify that the app provides notifications or alerts when planned expenses exceed budget limits. |
| Mandatory | TS\_21 | Verify that user can adjust the budget and planned expenses based on changing financial goals. |
| Important | TS\_22 | Verify that the app provides recommendations or suggestions for optimizing budget and expenses. |
| Important | TS\_23 | Verify that the app provides visual representations of budget utilization and goal progress. |
| Financial Articles | Mandatory | TS\_24 | Verify system behaviour when user selects a category |
| Mandatory | TS\_25 | Verify that articles are retrieved and displayed correctly |
| Mandatory | TS\_26 | Verify system behaviour when user selects an article |
| Mandatory | TS\_27 | Verify the article displayed is the same as the article that the user clicked |
| Important | TS\_28 | Verify system behaviour when article is not available |
| Important | TS\_29 | Verify system behaviour when user close the article |
| Account | Mandatory | TS\_30 | Verify successful creation of a new account with valid user details |
| Mandatory | TS\_31 | Validate error messages when invalid or incomplete user details are provided |
| Slightly Important | TS\_32 | Confirm account activation process through email verification |
| Slightly Important | TS\_33 | Verify error handling when activation link is expired or invalid |
| Important | TS\_34 | Test password strength requirements and validation |
| Slightly Important | TS\_35 | Validate successful password reset process through email |
| Important | TS\_36 | Verify error handling when email for password reset is not found or invalid |
| Mandatory | TS\_37 | Validate successful update of account information, such as email or phone number |
| Mandatory | TS\_38 | Verify error handling when updating account information with invalid data |
| Important | TS\_39 | Test account deletion process and confirm removal from the system |
| Mandatory | TS\_40 | Verify privacy and data protection measures, such as secure storage and encryption |
| Slightly Important | TS\_41 | Test concurrent account creation and data integrity |
| Mandatory | TS\_42 | Validate user interface responsiveness and usability of the account creation page |

**BAB 3. DESAIN DAN PERSYARATAN**

**3.1. Desain Konseptual**

Dalam fase ini, kami telah melakukan analisis kebutuhan dan merancang desain konseptual untuk aplikasi *financial tracker* kami. Tujuan utama kami adalah membuat desain yang dan *interface* yang interaktif dan responsif agar membuat pengguna merasa nyaman ketika menggunakanya. Dalam beberapa bagian berikutnya, kami telah membuat kebutuhan fungsional dan non-fungsional yang diperlukan aplikasi kami

**3.2.** **Persyaratan Fungsional**

* 1. Login/register
  + Pengguna akan diminta untuk memasukkan nama pengguna dan kata sandi yang valid untuk mengakses aplikasi.
  + Keamanan data dan perlindungan privasi akan diperhatikan dalam proses autentikasi ini.
  + Apabila pengguna belum memiliki akun, maka pengguna dapat membuat akun terlebih dahulu
  + kredensial pengguna akan disimpan dalam suatu database yang aman.
* 2. Mengakses data finansial
  + Pengguna dapat menambahkan dan mengatur anggaran untuk berbagai kategori pengeluaran, seperti makanan, transportasi, hiburan, dan lainnya.
  + Pengguna akan dapat mengganti anggaran yang telah ditentukan oleh pengguna kapan saja.
  + Fitur ini akan memungkinkan pengguna untuk mengelola alokasi keuangan mereka dengan lebih efektif.
  + Aplikasi akan menyediakan fitur untuk memasukkan pengeluaran atau pemasukan secara manual.
  + Aplikasi akan menyediakan tampilan ringkas tentang seberapa banyak uang yang telah dihabiskan pengguna dalam berbagai kategori pengeluaran.
  + Grafik dan diagram yang intuitif akan disediakan untuk membantu pengguna memahami pola pengeluaran mereka.
* 3. Mengakses artikel edukasi
  + Pengguna akan dapat mengakses artikel-artikel pendidikan yang berkaitan dengan manajemen keuangan pribadi sesuai kebutuhan mereka.
  + Artikel-artikel ini akan memberikan saran, tips, dan panduan untuk membantu pengguna meningkatkan pemahaman mereka tentang keuangan dan meningkatkan literasi keuangan pengguna.

**3.3. Persyaratan Non-Fungsional**

Selain persyaratan fungsional, kami juga telah mengidentifikasi persyaratan non-fungsional berikut untuk aplikasi *financial tracker* kami:

1. Kinerja

* Aplikasi harus responsif dan memberikan waktu respons yang cepat saat pengguna melakukan tindakan atau mengakses informasi.
* Keterlambatan dalam menampilkan data atau berpindah antarmuka harus diminimalkan.

2. Keandalan

* Aplikasi harus stabil dan dapat diandalkan, menghindari terjadinya crash, kelambatan, atau kesalahan yang sering.
* Perlu dilakukan pengujian yang memadai untuk memastikan keandalan aplikasi di berbagai kondisi penggunaan.

3. Keamanan

* Data pengguna harus diamankan dan dilindungi dari akses yang tidak sah atau kebocoran informasi.
* Aplikasi harus mematuhi praktik keamanan terbaik, termasuk enkripsi data sensitif dan perlindungan terhadap serangan seperti peretasan atau serangan *malware*.

4. Kemudahan penggunaan

* Antarmuka pengguna harus intuitif, mudah dipahami, dan ramah pengguna.
* Pengguna harus dapat dengan mudah mengakses dan menggunakan fitur-fitur yang tersedia tanpa menghadapi hambatan yang tidak perlu.

5. Skalabilitas

* Aplikasi harus dirancang dengan mempertimbangkan kemampuan untuk mengakomodasi pertumbuhan pengguna dan data.
* Ini berarti aplikasi harus mampu menangani peningkatan jumlah pengguna, volume data, dan beban lalu lintas yang meningkat seiring waktu.

6. Kompatibilitas

* Aplikasi harus kompatibel dengan berbagai model perangkat Android yang umum digunakan.
* Selain itu, kompatibilitas dengan berbagai versi sistem operasi Android dan ukuran layar yang berbeda juga harus diperhatikan.

**BAB 4. PROSES PENGEMBANGAN**

**4.1. Detail Pengembangan**

Proses pengembangan dari aplikasi CashCoach akan kami lakukan dengan beberapa detail berikut ini:

**4.1.1. Aplikasi Perancangan Antarmuka**

Sebelum memulai pengembangan aplikasi, kami terlebih dahulu harus merancang tampilan *pengguna interface* (UI) atau antarmuka dari aplikasi untuk dapat membayangkan tampilan dari aplikasi yang akan kami buat, beserta dengan informasi fitur-fitur apa saja yang akan dibutuhkan di masing-masing halamannya. Maka itu, kami menggunakan aplikasi desain grafis berbasis *cloud* bernama Figma untuk merancang *wireframe* dasar dari aplikasi kami.

Tidak hanya untuk perancangan UI, kami juga menggunakan *add-on* yang terdapat dalam aplikasi Figma, yakni *Iconify*, untuk mengambil beberapa *icon-icon* yang kami gunakan dalam aplikasi kami.

**4.1.2. Aplikasi Pengembangan**

Kami menggunakan Integrated Development Environment (IDE) Android Studio yang disediakan oleh Google untuk mengembangkan aplikasi mobile kami.

**4.1.3. *Framework***

IDE Android Studio memungkinkan kami untuk menggunakan berbagai jenis *framework* dan *library* yang tersedia yang dapat mendukung pengembangan aplikasi android, seperti Android Framework, Android Jetpack, Kotlin Standard Library, Firebase, Retrofit, dan lain-lain. Dalam pengembangan aplikasi kami, *framework* yang kami gunakan adalah Android Framework. Komponen dan API dari Android Framework yang akan kami gunakan dalam pengembangan aplikasi kami di antaranya adalah Activity dan Fragment, yang akan kami gunakan untuk mengembangkan UI dari aplikasi kami.

**4.1.4. Bahasa Pemrograman**

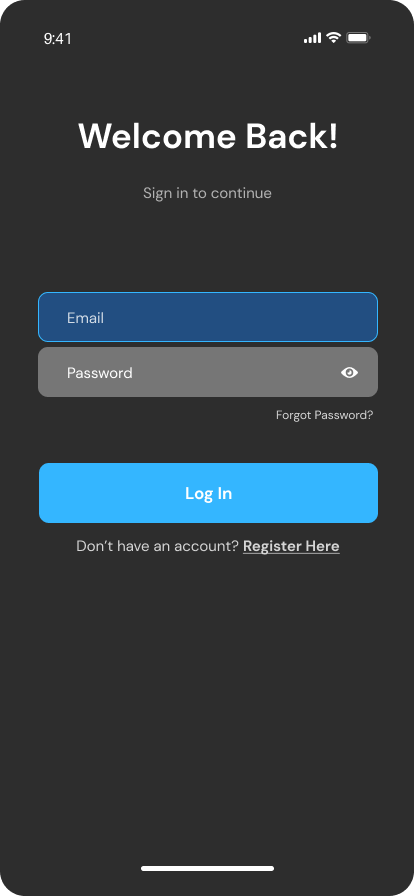
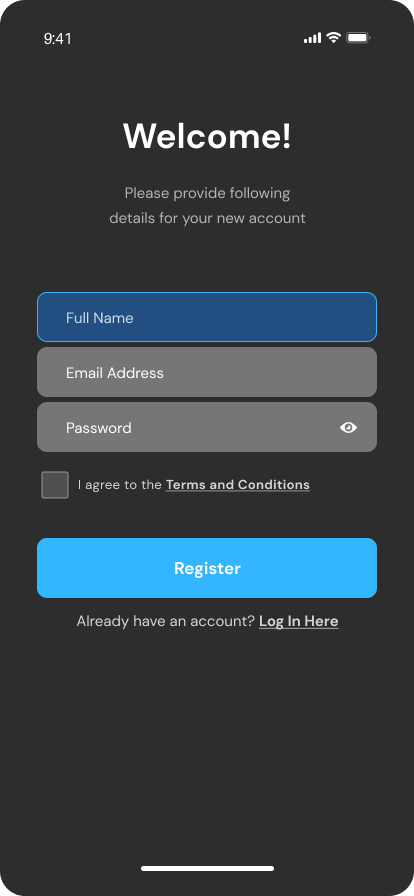
Pengembangan aplikasi kami menggunakan bahasa markup XML (eXtensible Markup Language) yang kami gunakan untuk merancang struktur elemen pada tampilan UI dari aplikasi kami, dan bahasa pemrograman Java yang kami gunakan untuk membangun logika sederhana yang dibutuhkan aplikasi kami seperti validasi *login* dan *passing* data pengguna yang baru membuat akun.

**4.1.5. Platform Version Control**

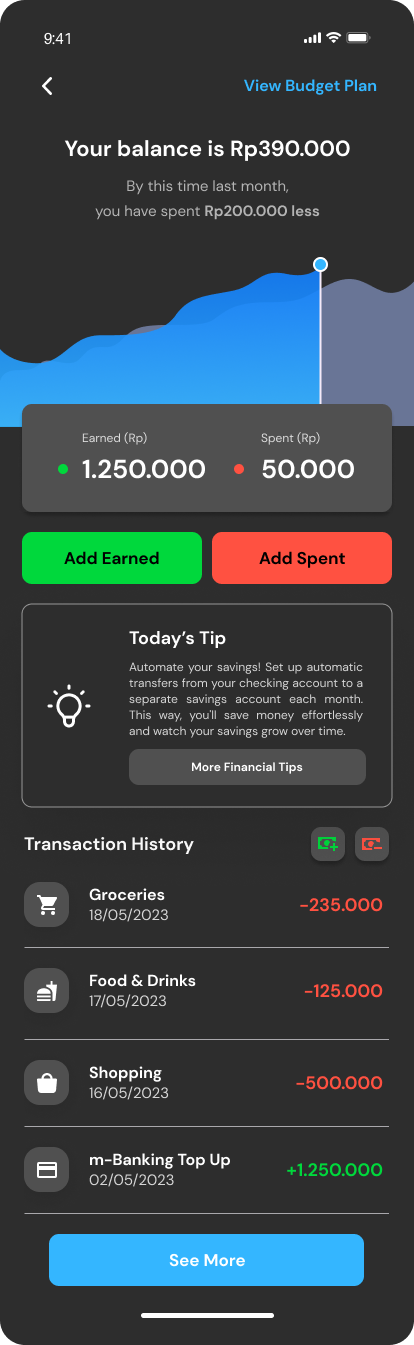
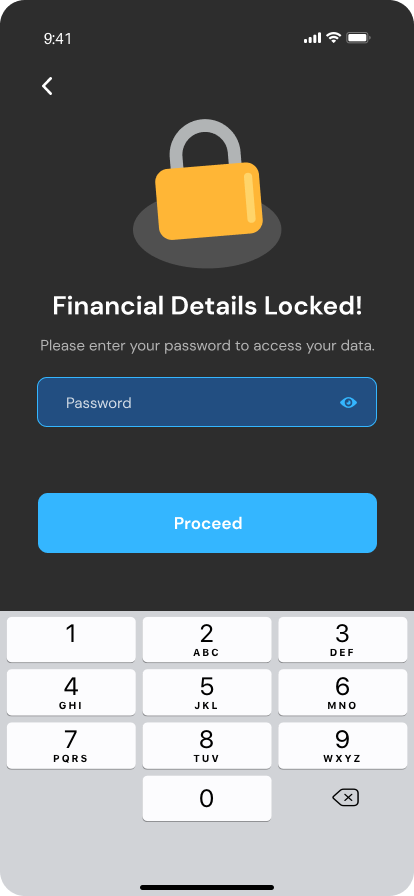
Dalam pengembangan aplikasi kami, *version control* yang kami gunakan sebagai sarana mengembangkan aplikasi kami adalah GitHub. GitHub merupakan *platform version control* yang terkenal dan terpercaya yang memungkinkan kolaborasi tim secara efisien dalam mengelola kode pengembangan aplikasi.

**4.2. Rancangan Antarmuka**

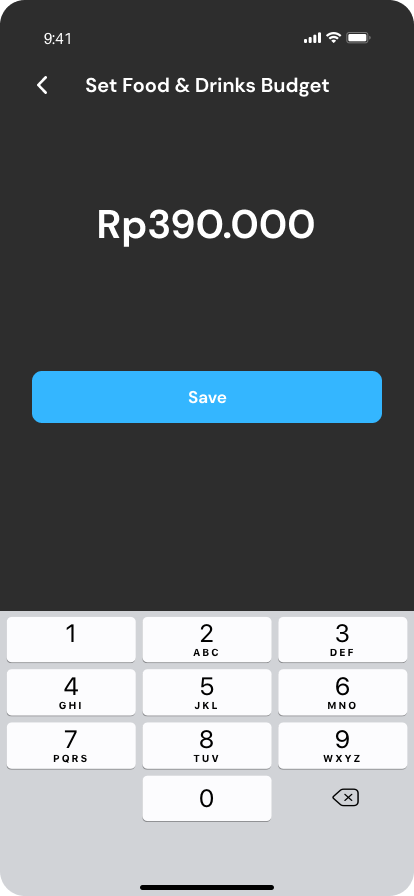
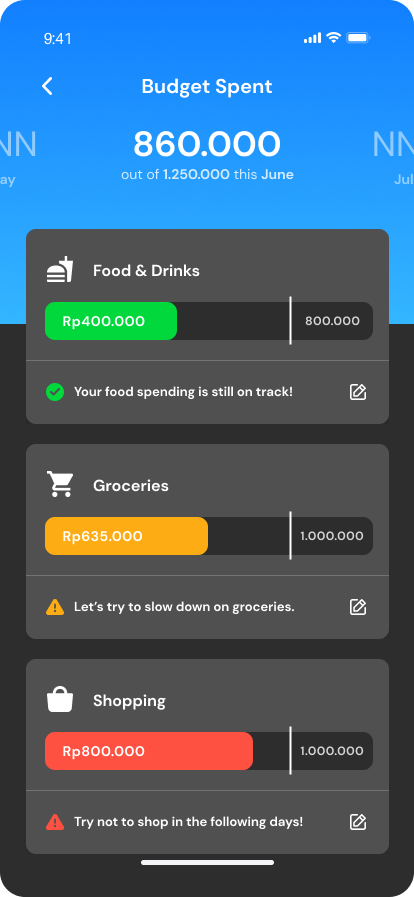
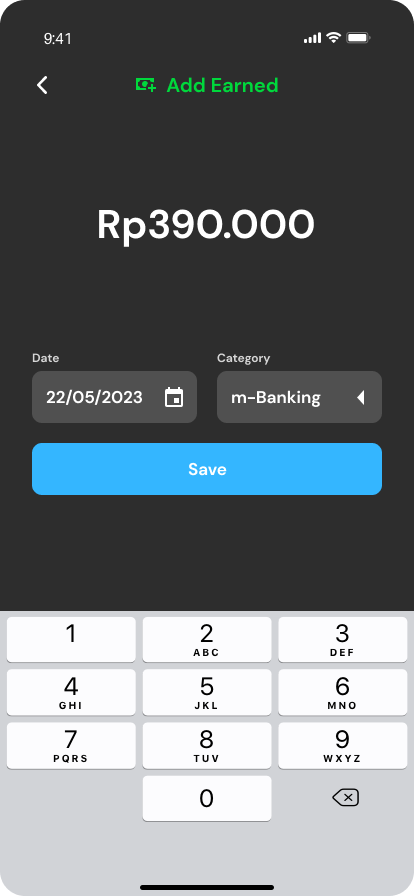
4.2.1. Splash, Register, and Login Screen



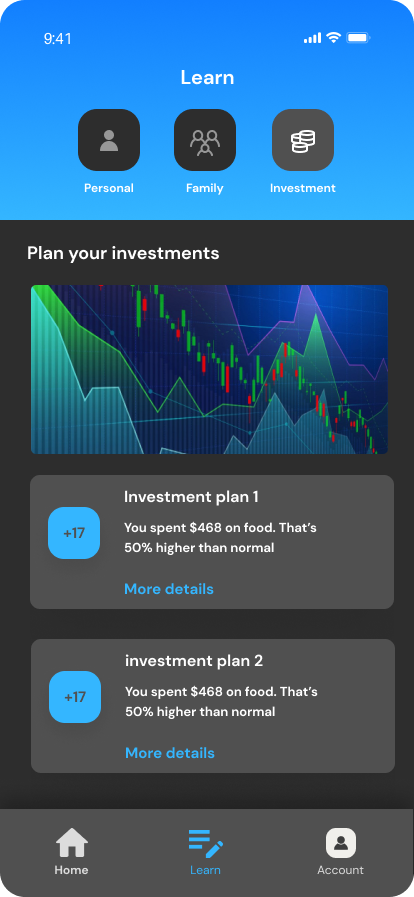
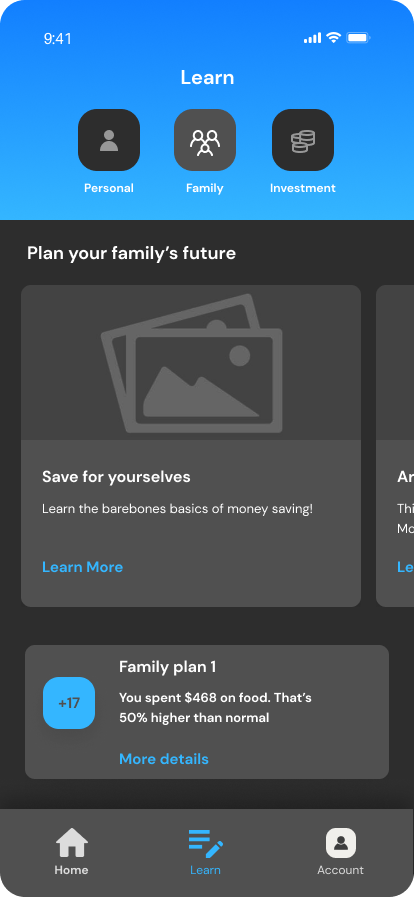
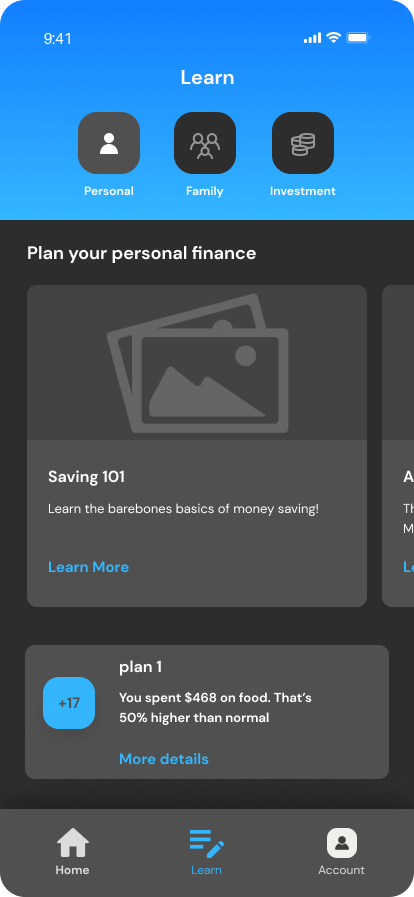
4.2.2. Home, Password Lock, and Financial Details Screen



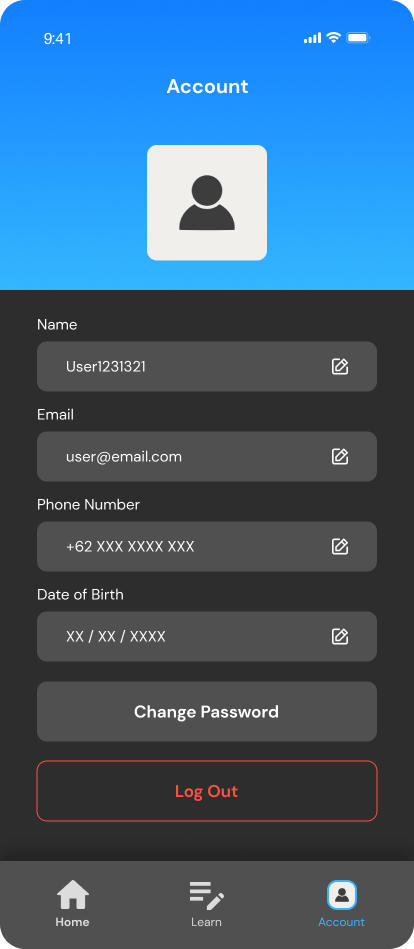
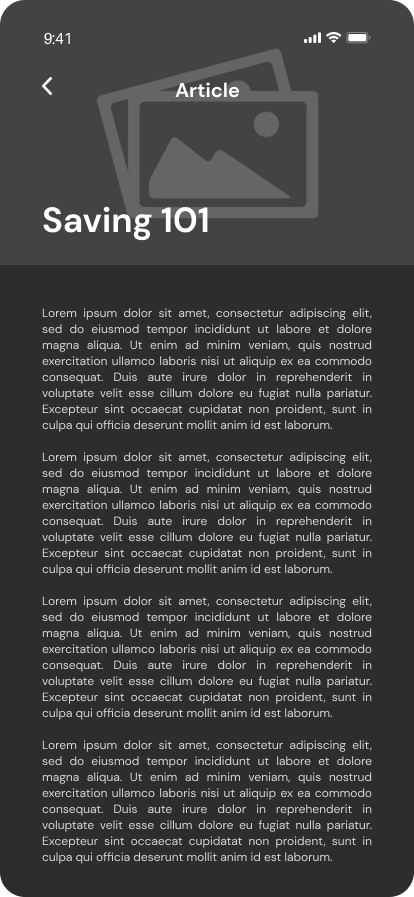
4.2.3. Add Earned, Budget Details, and Monthly Budget Screen



4.2.4. Learn: Personal, Family, and Investment Screen

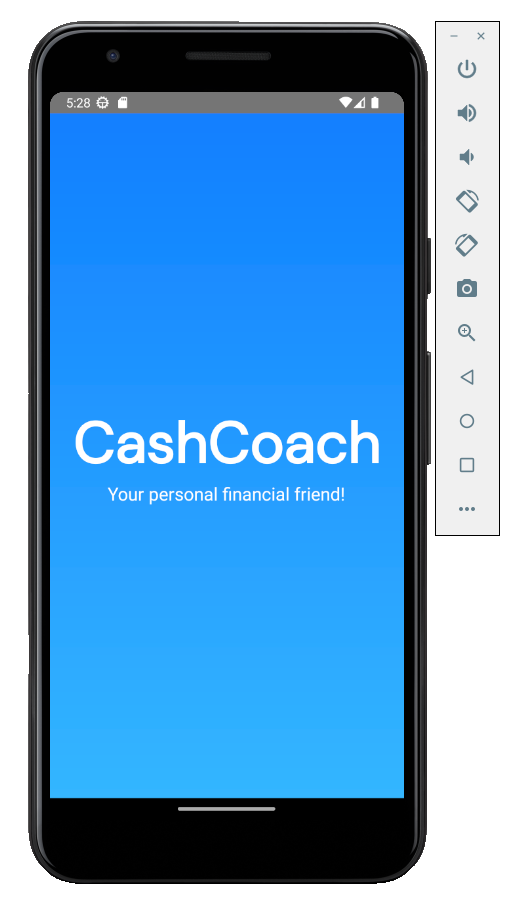


4.2.5. Article and Account Screen



**4.3. Hasil Pengembangan**

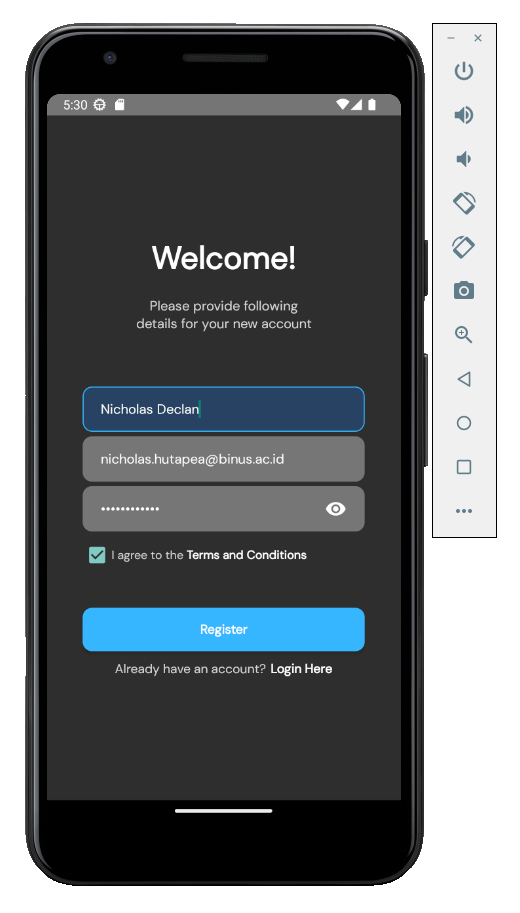
**4.3.1. Splash Screen**



Gambar 4.1. Antarmuka Splash Screen

Antarmuka ini akan ditampilkan selama 2 detik ketika pengguna membuka aplikasi.

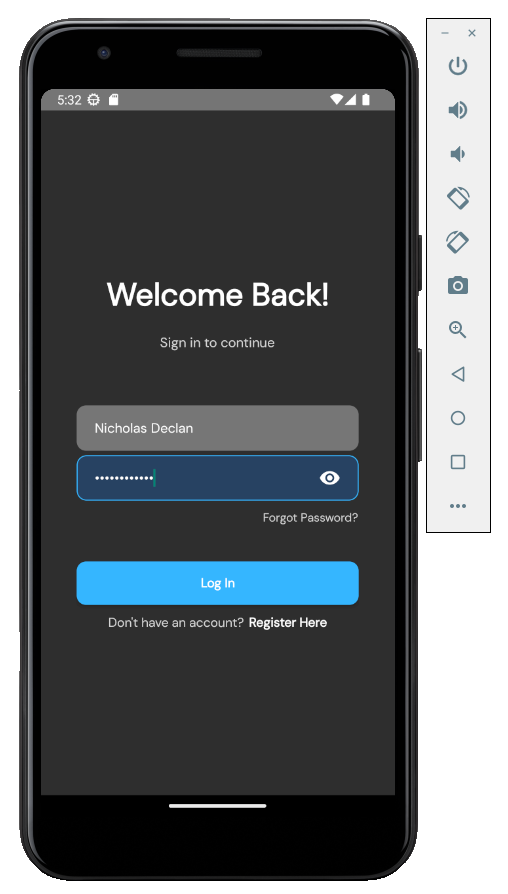
**4.3.2. Register Screen**



Gambar 4.2. Antarmuka Register Screen

Pengguna akan diredireksi dari *login screen* ke *register screen* ketika memencet tombol *Register Here* atau ketika pertama kali membuka aplikasi dari *splash screen.* Di *register screen* ini, pengguna dapat mengisi nama penuh, email, dan password untuk akunnya.

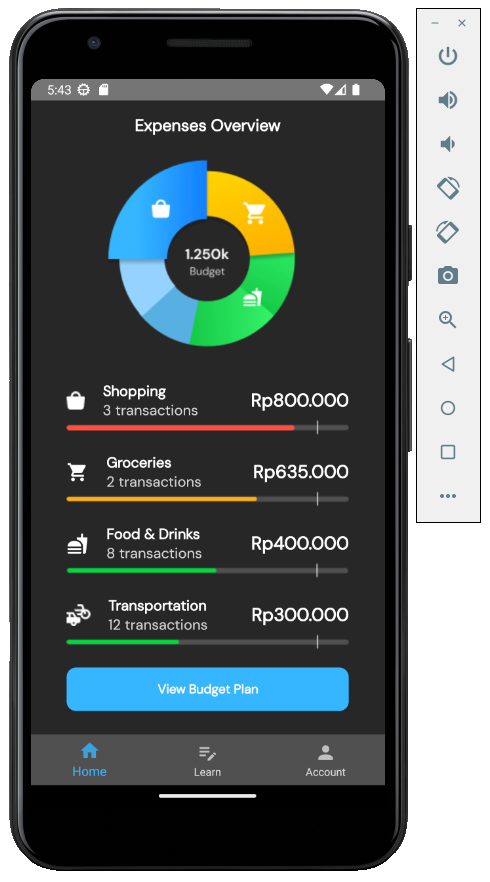
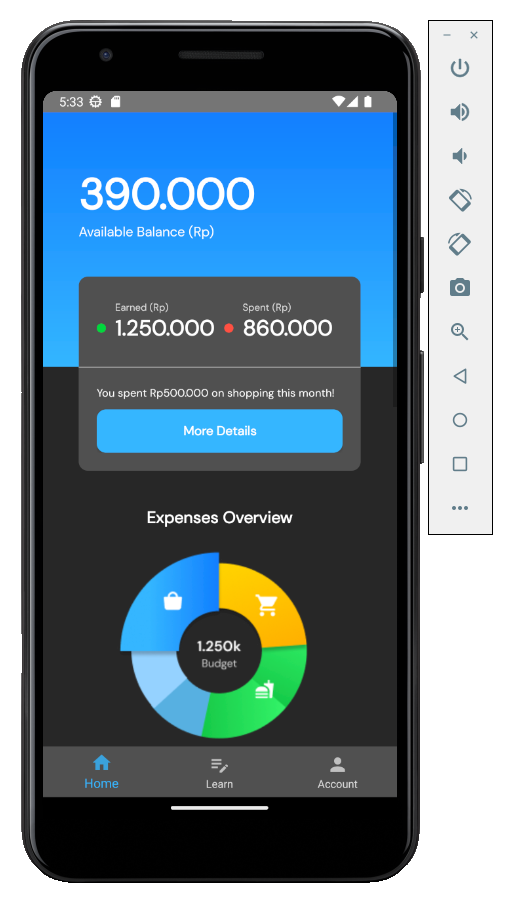
**4.3.3. Login Screen**

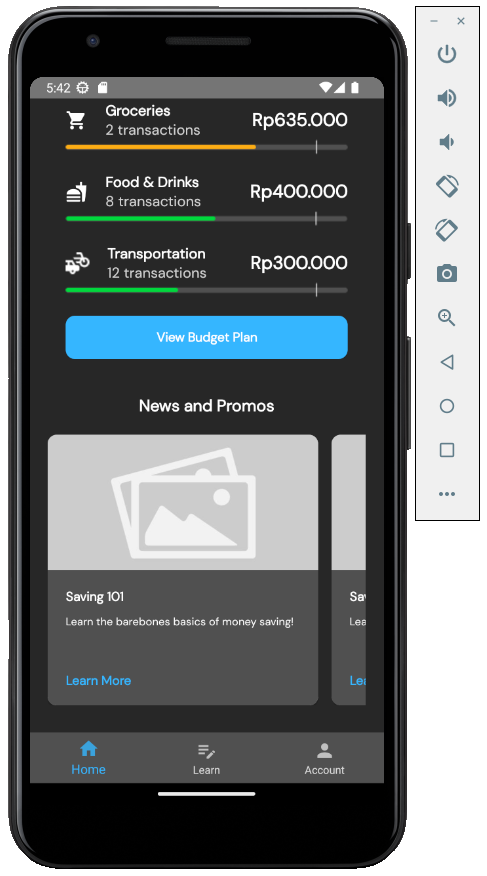


Gambar 4.3. Antarmuka Login Screen

Pengguna akan diredireksi dari *Splash Screen* ke login screen jika pernah mempunyai akun yang teregistrasi. Pengguna dapat masuk ke dalam akun yang telah terdaftar dengan memasukkan email dan password akun tersebut.

**4.3.4. Home Screen**



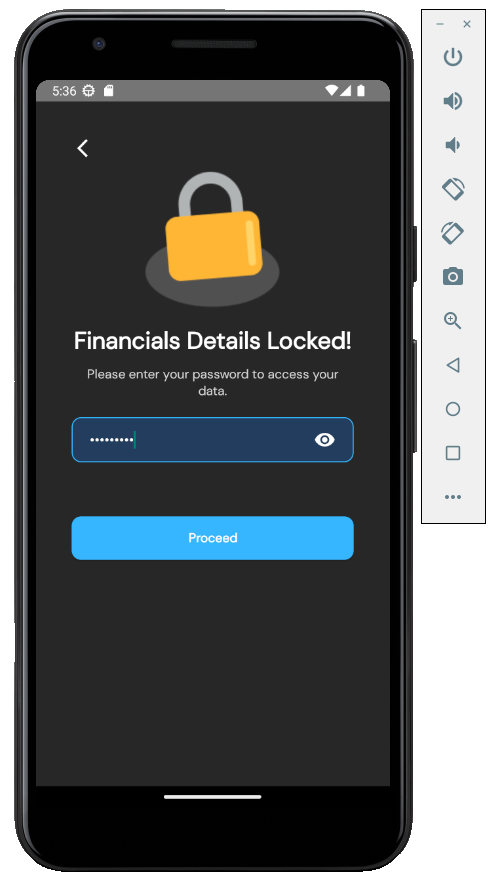


Gambar 4.4. Antarmuka Home Screen

*Home Screen* adalah antarmuka utama dari aplikasi kami yang akan tampil saat pengguna berhasil melakukan *login* atau *register*. Berikut adalah penjelasan untuk masing-masing elemen yang terdapat dalam *Home Screen*:

* Pada bagian atas *screen*, terdapat nominal *balance* tersisa yang dimiliki oleh pengguna selama satu bulan.
* Terdapat kotak yang menunjukkan details nominal *earned* dan *spent* yang menunjukkan nominal uang yang masuk dan keluar dari *wallet* pengguna.
* Tombol *More Details* yang bila dipencet akan membawa pengguna ke *Financial Details Screen*. Di *screen* ini, akan dijelaskan dengan lebih rinci mengenai detail pengeluaran dan pemasukan pengguna yang terjadi selama satu bulan.
* Diagram *Expenses Overview* beserta detail pengeluaran per kategori yang menjelaskan dengan sekilas 4 kategori pengguna yang mengalami paling banyak pengeluaran selama satu bulan.
* Tombol *View Budget Plan* yang bila dipencet akan membawa pengguna ke *Budget Details Screen*. Di *screen* ini, pengguna akan dapat melihat dan mengatur *budget plan* yang dimilikinya untuk bulan tersebut.
* Slider *News and Promos* yang dapat di-*swipe* secara horizontal untuk melihat semua *card* berita dan promosi yang kami tawarkan berdasarkan kategori-kategori pengeluaran pengguna.
* *Navigation Bar* yang dapat digunakan pengguna untuk menavigasi antara *Home Screen*, *Learn Screen*, dan *Account Screen*.

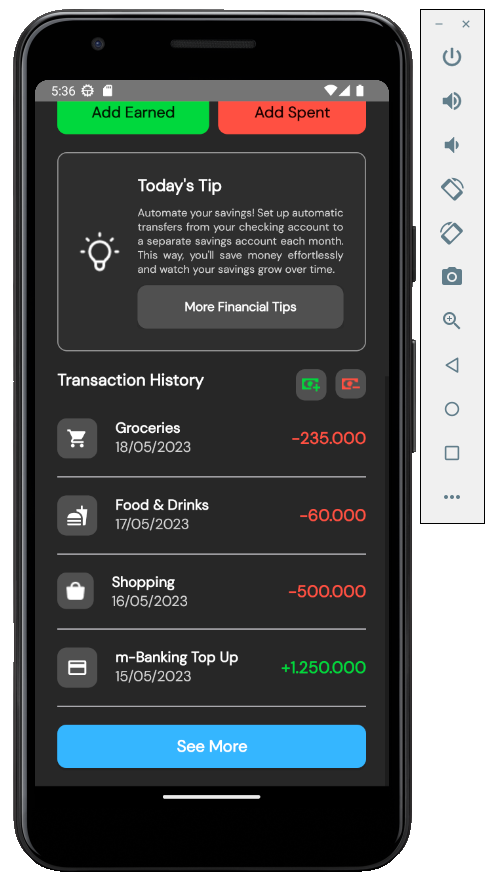
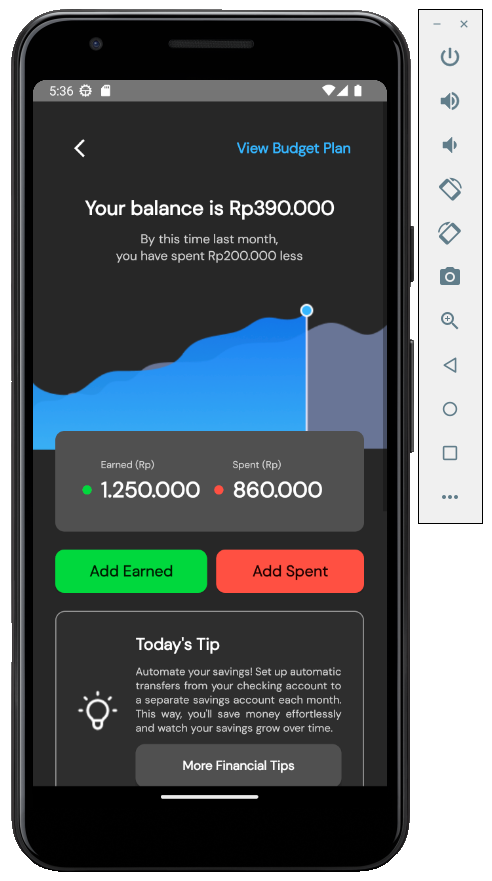
**4.3.5. Password Lock Screen**



Gambar 4.5. Antarmuka Password Lock Screen

*Password Lock Screen* ini adalah *screen* yang akan tampil saat pengguna memencet tombol *More Details* pada Home Screen. Sebelum dapat mengakses *Financial Details Screen*, dikarenakan informasi yang terkandung di dalamnya merupakan privasi pengguna yang harus dijaga, maka perlu dipastikan terlebih dahulu apakah yang sedang menggunakan aplikasi kami adalah pemilik akun yang sebenarnya. Pengguna diminta untuk memasukkan kembali *password* akun mereka, dan apabila sesuai, maka *Financial Details Screen* akan ditampilkan.

**4.3.6. Financial Details Screen**

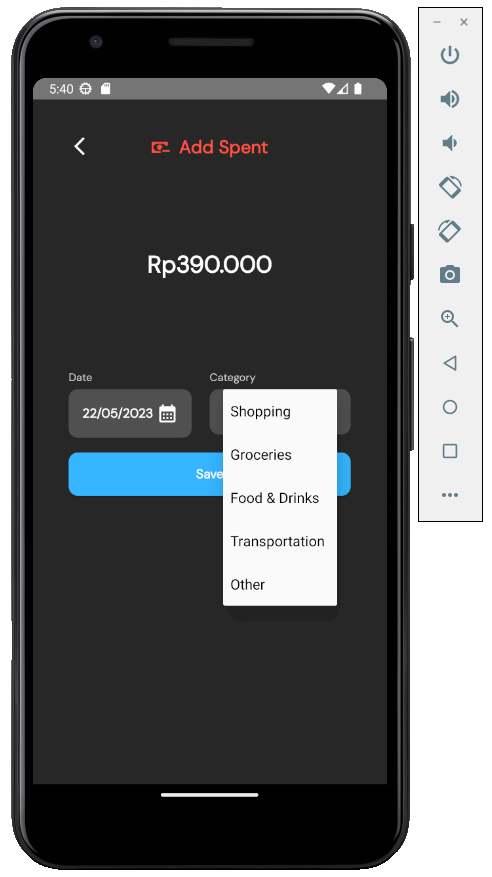
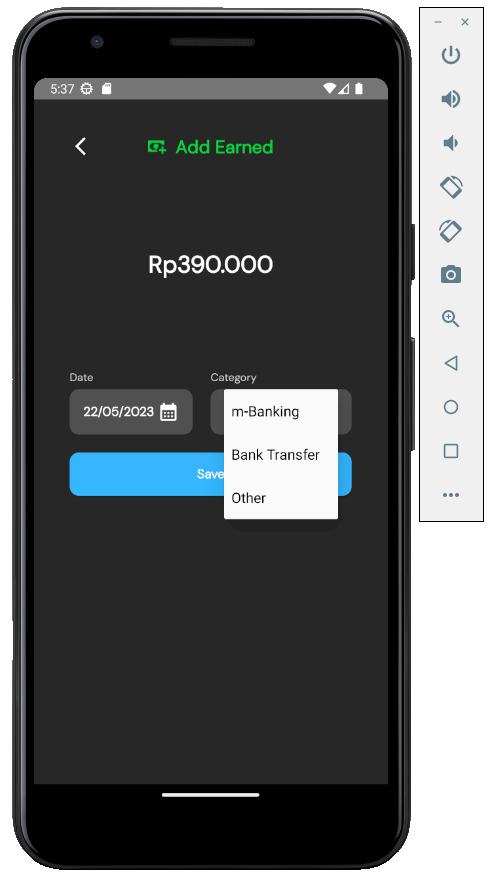


Gambar 4.6. Antarmuka Financial Details Screen

*Financial Details Screen* akan menampilkan semua detail mengenai pengeluaran dan pemasukan yang terjadi selama satu bulan. Berikut adalah penjelasan untuk masing-masing elemen yang terdapat dalam *Financial Details Screen*:

* Tombol *View Budget Plan* pada kanan atas *screen*, berfungsi sama dengan tombol *View Budget Plan* pada *Home Screen*. Ditambahkan pada *screen* ini untuk mempermudah pengguna untuk menavigasi dari *Financial Details Screen* ke *Budget Details Screen*.
* Nominal *balance* dan graf kondisi *balance* selama satu bulan. Graf menunjukkan progress yang terjadi pada *balance* pengguna selama satu bulan. Terdapat graf dengan warna yang lebih gelap tertutup di belakang graf berwarna biru terang. Graf tersebut menunjukkan kondisi *balance* pengguna bulan yang lalu agar pengguna dapat melakukan komparasi secara langsung.
* Tombol *Add Earned* dan *Spent* yang bila dipencet akan menampilkan *Add Earned/Spent Screen* di mana nantinya *user* akan dapat menambahkan data riwayat transaksi atau pemasukan.
* Kotak *Today’s Tip* yang menampilkan *tips-tips* sederhana mengenai pengelolaan keuangan yang akan disesuaikan dengan kondisi *balance* pengguna pada hari itu. Dalam kotak ini terdapat juga tombol *More Financial Tips* yang bila dipencet akan membawa pengguna ke *Learn* *Screen* untuk membaca artikel-artikel seputar pengelolaan uang.
* *Transaction History* yang menunjukkan semua riwayat pemasukan dan pengeluaran yang terjadi pada *wallet* pengguna selama satu bulan. Masing-masing pemasukan atau pengeluaran ini akan menunjukkan data nominal, tanggal, dan jenis kategori dari pengeluaran atau pemasukan yang terjadi. Tidak hanya itu, tampilan riwayat transaksi juga dapat di-*filter* sehingga hanya menampilkan pemasukan (*earned*) atau pengeluaran (*spent*) saja. Bila tombol *See More* dipencet oleh pengguna, maka riwayat transaksi yang ditampilkan akan menunjukkan seluruh riwayat transaksi dengan lebih lengkap.

**4.3.7. Add Earned/ Spent Screen**



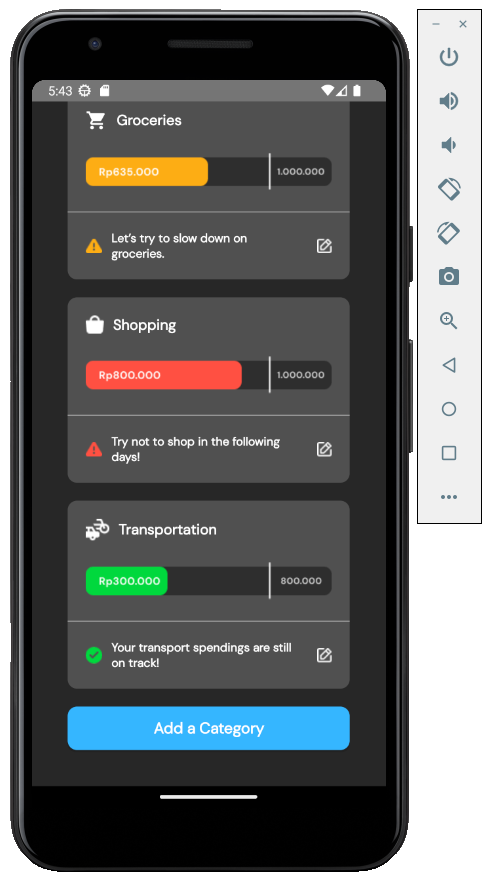
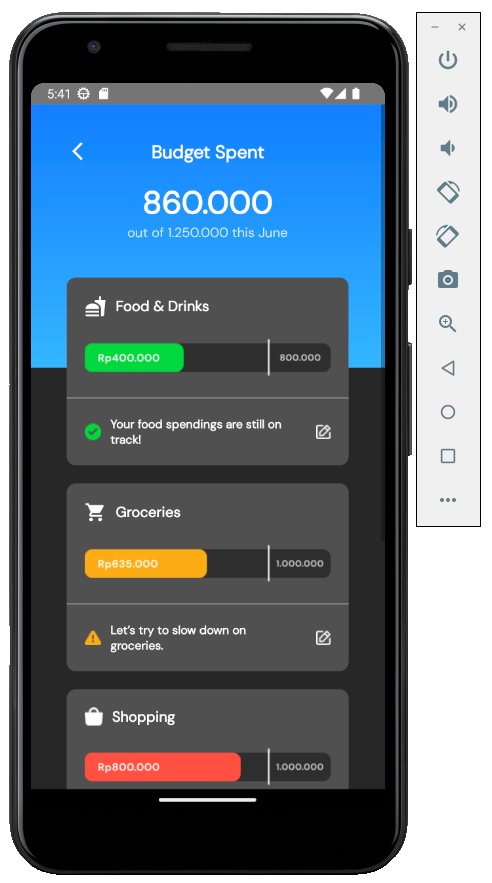
Gambar 4.7. Antarmuka Add Earned dan Add Spent Screen

*Add Earned/Spent Screen* tampil saat pengguna memencet tombol *Add Earned* ataupun *Add Spent* pada *Financial Details Screen*. Di *screen* ini, pengguna akan dapat memasukkan nominal pemasukan/pengeluaran yang ingin ditambahkan ke dalam riwayat transaksinya. Kemudian, pengguna juga harus memasukkan data tanggal dan kategori dari transaksi yang terjadi. Saat ini, kami menyediakan beberapa opsi kategori untuk transaksi pemasukan dan pengeluaran:

* Kategori Pemasukan (*Add Earned*)
  + *m-Banking*
  + *Bank Transfer*
  + *Other*
* Kategori Pengeluaran (*Add Spent*)
  + *Groceries*
  + *Food & Drinks*
  + *Shopping*
  + *Transportation*
  + *Other*

Setelah mengisi data yang dibutuhkan, pengguna dapat memencet tombol *Save* untuk memperbarui riwayat transaksi, dan detail transaksi yang baru saja ditambahkan akan tampil di *Financial Details Screen*.

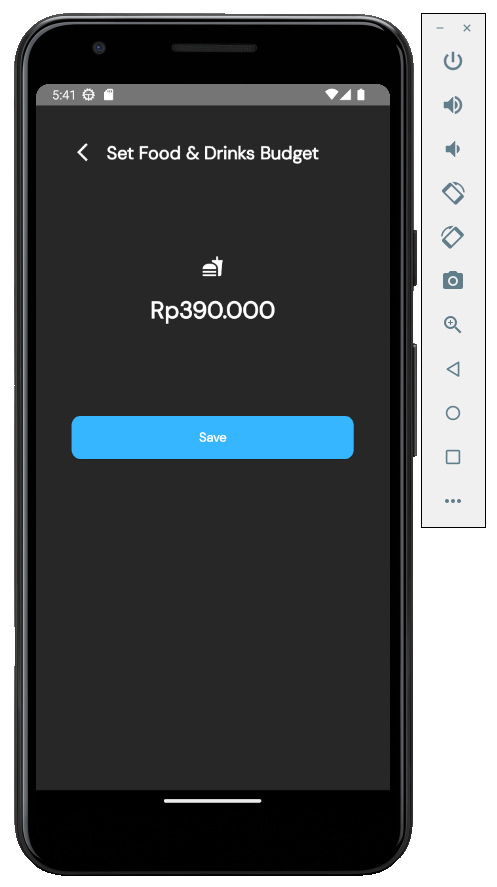
**4.3.8. Budget Details Screen**



Gambar 4.8. Antarmuka Budget Details Screen

*Budget Details Screen* menampilkan detail-detail mengenai perencanaan pengeluaran pengguna selama satu bulan. Di *screen* ini, pengguna dapat mengatur batas pengeluaran untuk masing-masing kategori secara manual. Akan terdapat kotak detail untuk setiap kategori yang dimiliki oleh pengguna yang menunjukkan *progress bar* pengeluaran pengguna untuk kategori tersebut. Pengguna dapat memencet tombol *icon* *Edit* yang terdapat di pojok kanan bawah kotak kategori untuk membuka *Monthly Budget Screen* di mana pengguna akan dapat mengatur batas maksimal pengeluaran untuk kategori yang dipencet. Di bagian bawah *progress bar* untuk setiap kategori, terdapat sebuah warning yang akan mengingatkan pengguna apabila pengeluarannya selama satu bulan telah mendekati batas maksimal pengeluaran yang ditentukan.

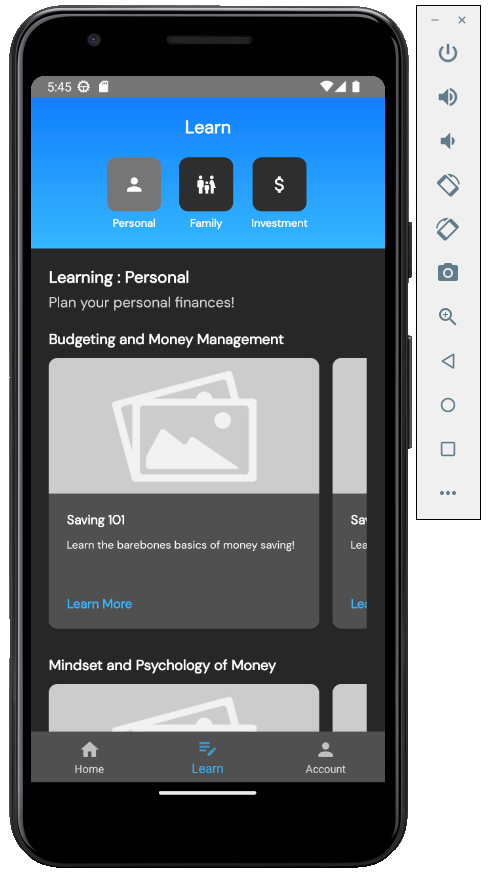
**4.3.9. Monthly Budget Screen**



Gambar 4.9. Antarmuka Monthly Budget Screen

Pada *Monthly Budget Screen*, pengguna akan dapat menginput nominal batas maksimal pengeluaran yang mereka ingin tentukan untuk salah satu kategorinya.

**4.3.10. Learn - Personal Finance Screen**

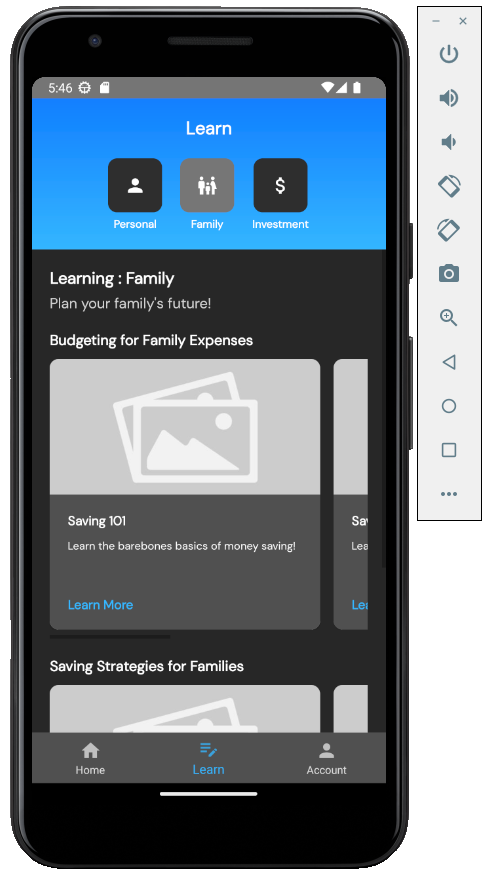


Gambar 4.10. Antarmuka Learn - Personal Finance Screen

*Learn menu* adalah menu di mana pengguna dapat mengakses artikel-artikel edukasi yang dapat membantu pengguna dalam mengatur finansial mereka. Ada 3 sub kategori dalam menu ini: *personal finance, family finance,* dan *investment*.

Ketika pengguna pertama kali masuk ke *learn screen* melewati *navbar*, pengguna akan dibawa ke sub kategori *personal finance*. Halaman ini terdiri dari *navbar* sub kategori untuk mengganti kategori menu *learn* dan *navbar* untuk menavigasikan aplikasi. Pengguna akan direkomendasikan artikel-artikel yang sesuai dengan target yang telah ditentukan oleh pengguna. Rekomendasi artikel ini disajikan dalam bentuk *image* *carousel* yang berinteraksi dengan pengguna. Ketika layar ini aktif, *navbar* bagian *personal* akan berbeda dari komponen *navbar* lainnya.

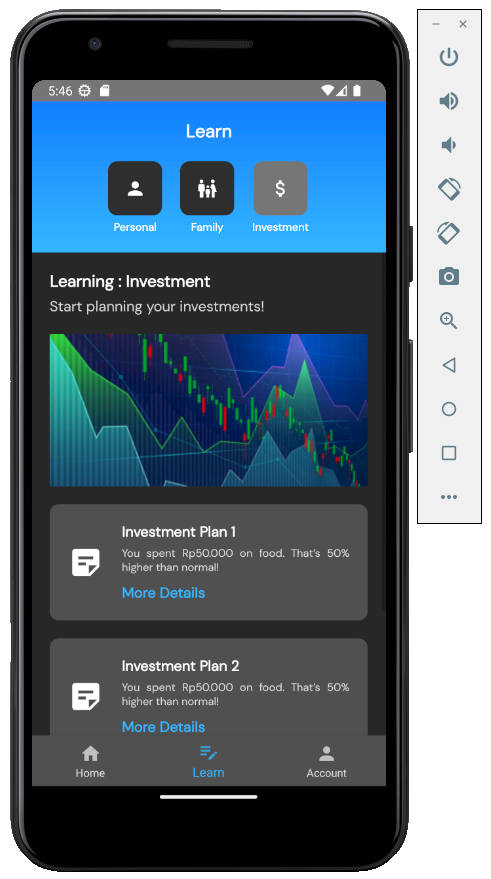
**4.3.11. Learn - Family Finance Screen**



Gambar 4.11. Antarmuka Learn - Family Finance Screen

*Screen* ini merupakan sub kategori dari *Learn Screen*. *Screen* ini terbuka ketika user memilih menu *Family* pada *navbar* yang berada di bagian atas *Learn Screen.* *Screen* ini memiliki antarmuka yang sama dengan *Personal Finance Screen*, tetapi konten yang direkomendasikan disesuaikan dengan kebutuhan finansial keluarga, bukan hanya individu saja. Ketika *screen* ini aktif, tombol *family* pada *navbar* *Learn Screen* akan berubah warnanya, menunjukkan keaktifannya.

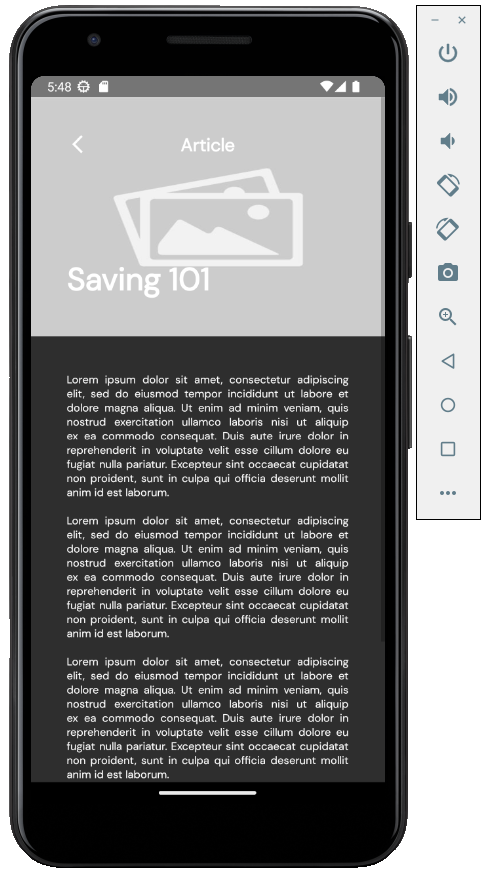
**4.3.12. Learn - Investment Screen**



Gambar 4.12. Antarmuka Investment Screen

Layar ini akan aktif ketika pengguna memilih menu *investment* pada *navbar* Layar Learn. Ketika terbuka, layar ini akan menampilkan *jumbotron* yang berupa grafik investasi yang dapat diambil dari API lain. Di bawahnya, sama seperti subkategori *Learn Screen* lainnya terdapat artikel-artikel edukasi, tetapi dengan tema investasi yang bisa diakses oleh pengguna.

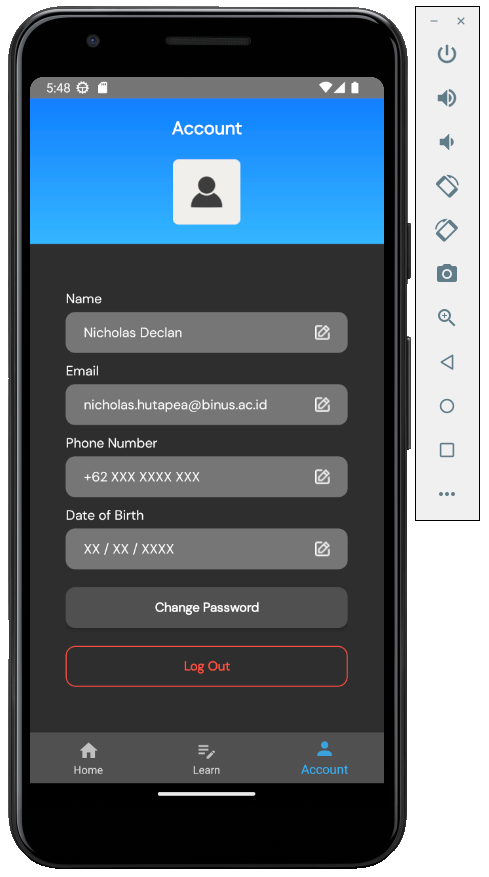
**4.3.13. Learn - Article Screen**



Gambar 4.13. Antarmuka Article Screen

Layar ini aktif ketika pengguna mengakses salah satu artikel yang terdapat pada Layar Learn. Layar ini memiliki *thumbnail* dari artikelnya di paling atas dan judul artikelnya agar pengguna dapat dengan mudah mengidentifikasikan artikel yang sedang dibaca. Dibawah *thumbnail* dan judul, pengguna bisa langsung melihat isi artikel. Ketika pengguna *scroll* layar ini ke bawah, *thumbnail* dan judul akan tertutup agar pengguna dapat membaca artikel lebih mudah.

**4.3.12. Account Screen**



Gambar 4.14. Antarmuka Account Screen

Account screen akan ditampilkan jika pengguna menekan tombol akun dalam *navigation bar*. Di sini pengguna dapat melihat dan mengganti gambar profil, nama, email, nomor telepon, tanggal lahir, dan password yang terdaftar dalam aplikasi. Pengguna juga dapat melakukan *log out* dari akunnya.

**4.4. Kendala yang Dihadapi**

Dengan jangka waktu satu bulan yang diberikan untuk merealisasikan proyek ini, serta sumber daya yang sangat terbatas, kami hanya dapat mengembangkan aplikasi CashCoach kami hingga tahap *user interface* saja, kami belum dapat mengintegrasikan *database* dan merealisasikan fungsionalitas lainnya yang kami rencanakan di awal untuk aplikasi kami. Kami sendiri sebagai mahasiswa program studi multimedia belum memiliki pemahaman yang begitu mendalam mengenai *mobile programming* dan berharap kedepannya kami akan mengasah kemampuan kami lebih lagi untuk dapat mengembangkan sebuah aplikasi yang berfungsi secara menyeluruh.

**4.5. Tautan Version Control Hasil Akhir Proyek**

Seperti yang telah kami sebutkan sebelumnya, kami menggunakan platform version control GitHub untuk mengelola kode pengembangan aplikasi kami. Berikut ini adalah tautan repositori GitHub dari hasil akhir aplikasi kami: <https://github.com/Maplestrike/CashCoach>. Pada tautan tersebut, dapat ditemukan versi akhir aplikasi kami yang telah kami kembangkan dengan menggunakan aplikasi Android Studio, dengan bahasa pemrograman Java.

**BAB 5. PENUTUP**

Proyek ini telah kami laksanakan dari awal hingga akhir, mulai dari tahap prototyping hingga pengembangan kode dalam Android Studio. Namun, karena adanya kendala waktu yang menjadi rintangan dalam penyelesaian pengembangan kode, kami hanya dapat merealisasikan User Interface aplikasi tersebut.

Kami memulai proyek ini dengan melakukan tahap prototyping yang melibatkan perencanaan desain dan fungsionalitas aplikasi. Setelah itu, kami melanjutkan ke tahap pengembangan kode dalam Android Studio. Pada tahap ini, kami menggunakan berbagai bahasa pemrograman dan kerangka kerja yang relevan untuk membangun aplikasi.

Namun, karena kendala yang tidak dapat dihindari, kami harus memprioritaskan penyelesaian User Interface aplikasi. Kami fokus pada desain tampilan yang menarik dan intuitif, sehingga pengguna dapat dengan mudah berinteraksi dengan aplikasi. Meskipun kami tidak dapat melanjutkan pengembangan kode secara menyeluruh, kami berusaha memastikan bahwa pengguna dapat mengalami aplikasi dengan tampilan yang siap digunakan.

Meskipun tidak semua fitur yang direncanakan dapat diimplementasikan, kami tetap berkomitmen untuk memberikan hasil terbaik dalam keterbatasan waktu yang ada. Kami menyadari bahwa masih ada ruang untuk perbaikan dan pengembangan lebih lanjut pada aplikasi ini di masa depan.

Kami berharap bahwa proyek ini dapat memberikan manfaat kepada pengguna, meskipun belum sepenuhnya lengkap. Terima kasih atas kesempatan yang diberikan kepada kami untuk mengerjakan proyek ini.