|  |
| --- |
| Personal Expense Tracker |
| Coin Track |
| Major Assignment Fundamental of Accounting |

|  |
| --- |
| Taimoor Ul Islam (14031)  12-9-2023 |

# iNTRODUCTION

In this project, I have developed a practical Expense Tracker app using Flutter for the frontend and Firebase as the backend. The app enables users to easily manage their finances by providing features such as user authentication, transaction recording with categories and descriptions, a transaction list view with delete functionality, and a comprehensive dashboard. The dashboard includes informative visualizations like pie charts for expense categories, bar charts for incomes, and a dynamic line chart depicting the variation of expenses and incomes over time. Additionally, the app offers a user-friendly filter option for viewing summaries based on specific time periods such as last week, last month, or last year

# Features

## Authentication:

The authentication feature ensures a secure environment for users to interact with the Expense Tracker app. With an easy-to-use Sign In and Sign Up process, users can create accounts, safeguarding their financial data. Email verification not only enhances the overall user experience but also establishes a trustworthy foundation for managing personal finances.

## Transaction Recording and deletion:

The Transaction Recording feature allows users to effortlessly log their financial activities within the app. Users can specify whether a transaction is an income or expense, assign relevant categories, and provide descriptions for detailed tracking. This intuitive process ensures that users maintain an accurate and organized record of their financial transactions, promoting effective money management.

## Transaction Summary:

The Transaction Summary feature encompasses a comprehensive view of users' financial data through the dashboard. Users can access a dynamic summary that includes total income, total expenses, total transactions, and the remaining balance. Visual aids, such as pie charts for expense categories and bar charts for incomes, provide users with a quick and visually appealing overview. The inclusion of a dynamic line chart illustrating the variation of expenses and incomes over time offers deeper insights, making the Transaction Summary feature a powerful tool for financial analysis. Additionally, a user-friendly filter option allows users to tailor the summary based on specific time periods, adding flexibility to their financial tracking experience.

# Tech Stack

## Frontend:

The app's frontend, the part you see and interact with, is powered by Flutter and Dart. Flutter is like the artist creating a beautiful painting – it allows us to craft a visually appealing and user-friendly interface. Dart, on the other hand, acts like the language that communicates with Flutter, giving the app its functionality. Together, Flutter and Dart work seamlessly to bring our app to life, making it not just functional but also aesthetically pleasing.

## Backend:

Behind the scenes, our app's backend, responsible for handling data and making everything work together, relies on Firebase services. Firebase Authentication ensures that user logins are secure and protected. It's like having a digital bouncer at the entrance of our app, allowing only authorized users inside. Firebase Firestore Database is where we store and organize all the information about transactions and user details. Think of it as a highly organized virtual filing cabinet – easy to access and update. And when it's time to share our app with the world, Firebase Hosting services make it accessible online, ensuring a smooth and reliable user experience. So, while Flutter and Dart make our app visually appealing and interactive, Firebase takes care of the behind-the-scenes magic, keeping everything secure, organized, and online.

# Architecture

## Data Layer:

In the Data Layer, we manage the raw data flow between our app and the Firebase backend. At the core of this layer is the repository, which acts as the intermediary. The repository retrieves raw data from the Firebase database using dedicated plugins and APIs. It then transforms this data into Dart custom models for transactions, providing a structured format for our app to work with. Conversely, when the app needs to add or delete a transaction, the repository takes the transaction model and communicates with the database accordingly. Essentially, the repository serves as a bridge, converting raw data to models and vice versa, ensuring seamless communication with the Firebase backend.

## Domain Layer:

The Domain Layer focuses on the business logic of our application, containing crucial use cases that dictate how our app functions. These use cases act as functional units that communicate with the repository to obtain or manipulate data. Each use case encapsulates specific functionalities accessible from the Presentation Layer. The use cases include:

**GetTransactions:** Retrieve transaction data.

**AddTransaction:** Add a new transaction.

**DeleteTransaction:** Remove a transaction.

**GetReportData:** Obtain comprehensive report data.

**GetIncomeSummaryData:** Retrieve income summary data.

**GetExpenseSummaryData:** Obtain expense summary data.

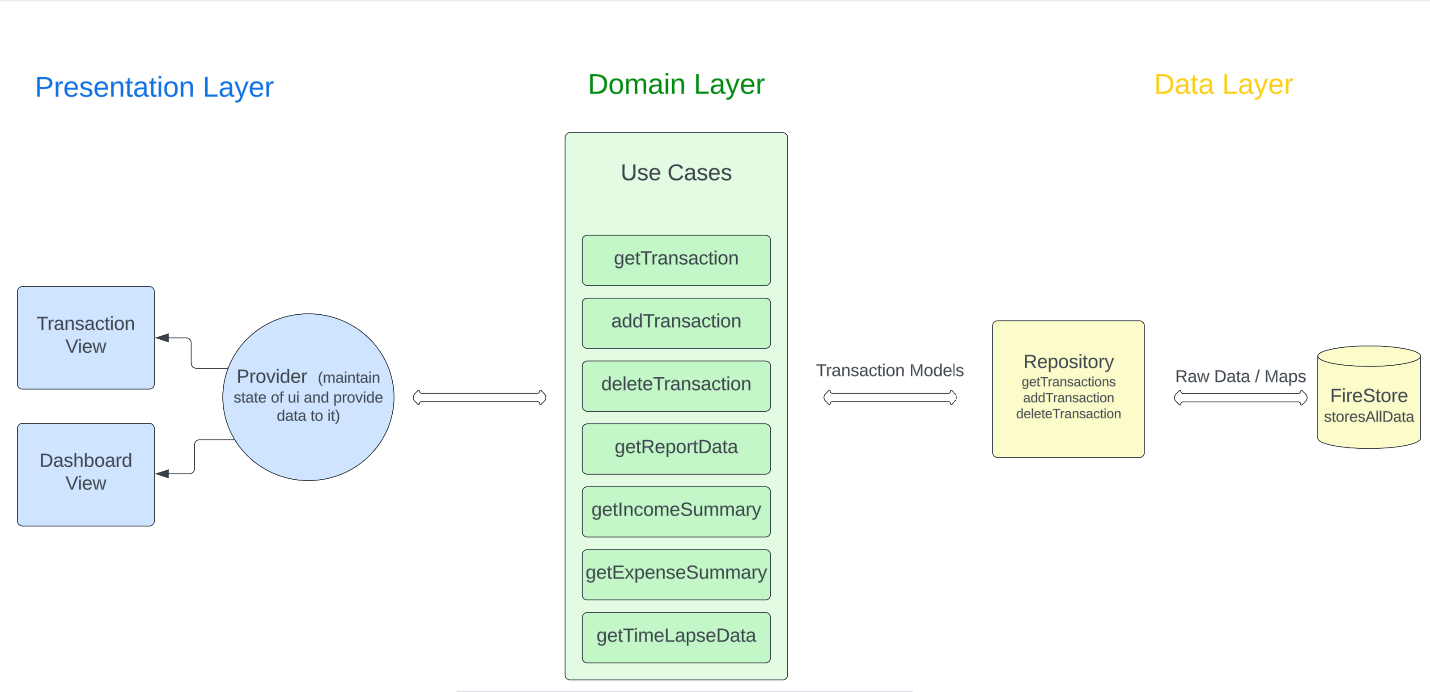
**GetTimeLapseData:** Access data based on a selected time lapse.

These use cases leverage the repository to fetch data, perform their respective logic, and hand over the processed data to the Presentation Layer's provider.

## Presentation Layer:

The Presentation Layer encompasses two primary components: the state management solution (Provider) and the user interface (UI). Provider acts as the orchestrator, managing the app's state and data flow. The UI, in turn, consists of two distinct views – the Transaction List View and the Dashboard View. The UI components retrieve data from Provider, ensuring that the app's visual elements accurately represent the underlying data.

Any user interaction or event on the UI triggers a corresponding use case from the Domain Layer. This ensures that every action aligns with the predefined business logic, maintaining a clear separation between the presentation and business layers. The seamless integration of Provider and UI creates a responsive and dynamic user experience while adhering to the defined functionalities established in the Domain Layer.



# Backend

## Firebase Authentication Service:

The backbone of our backend relies on Firebase Authentication Service, ensuring a secure and user-friendly experience. Users can confidently Sign In and Sign Up with their credentials, backed by robust authentication protocols. An additional layer of security is added through email verification, enhancing the trustworthiness of user accounts. This service acts as the digital gatekeeper, allowing only authenticated users to access and interact with our app, thus safeguarding sensitive financial information.

## Firebase Firestore Database:

Our app's data finds its home in Firebase Firestore Database, a NoSQL cloud database. This dynamic database accommodates our evolving needs, providing a flexible structure for storing and retrieving data efficiently. While it's not an SQL database, Firestore offers seamless integration with our Flutter app, ensuring real-time updates and quick access to transaction details. The database schema for transactions is thoughtfully designed, capturing essential elements such as transaction type, category, description, and timestamp, facilitating efficient data management and retrieval.

## Firebase Hosting Service:

Firebase Hosting Service brings our app to the online realm, making it accessible to users worldwide. This service ensures that our app is not confined to local devices but can be accessed seamlessly over the internet. Firebase Hosting provides a scalable and reliable platform for deploying our app, assuring a smooth user experience. With this service, our app becomes a virtual hub for financial tracking, allowing users to manage their transactions and access valuable insights from anywhere with an internet connection.

# Responsiveness

Our Expense Tracker app has been meticulously crafted to ensure a seamless user experience across various devices. Whether you're using it on a mobile device, tablet, or desktop, the app dynamically adjusts its layout and functionality to suit the screen size. This responsiveness guarantees that users can effortlessly manage their finances, view transaction details, and explore insightful summaries with equal ease on any device. The intuitive design adapts to different screen dimensions, making financial tracking accessible and user-friendly regardless of the platform, providing a consistent and responsive experience across the entire spectrum of devices.

# Dependencies

Our Expense Tracker app leverages several key dependencies to enhance functionality and provide a smooth user experience. The foundation of our frontend development is the Flutter SDK, a robust framework for building natively compiled applications for mobile, web, and desktop from a single codebase.

Firebase plugins are seamlessly integrated into our backend to manage user authentication, database interactions, and hosting services, ensuring secure and reliable cloud-based functionality.

For state management, we rely on the Provider package, a powerful solution that efficiently manages the app's state, facilitating dynamic updates and responsive user interfaces.

To simplify and streamline package installation, we utilize the Get It package, which serves as a convenient dependency injection solution.

In the realm of data visualization, we employ the FL Chart package to create compelling and informative charts on our app's dashboard.

Additionally, the Intl package enhances our app's internationalization capabilities, ensuring a user-friendly experience for individuals worldwide.

# Conclusion

In crafting our Expense Tracker app, we've seamlessly blended the power of Flutter and Firebase to deliver a comprehensive and user-friendly financial management solution. The app's intuitive design, backed by Flutter's dynamic SDK, ensures a consistent and responsive experience across mobile, tablet, and desktop devices. Firebase plays a pivotal role in the backend, offering secure authentication, robust database services, and reliable hosting. The dependencies such as Provider, Get It, FL Chart, and Intl contribute to the app's functionality, state management, and internationalization capabilities. With a carefully structured architecture, thoughtful data layer management, and a user-centric presentation layer, our Expense Tracker app not only simplifies financial tracking but also provides insightful summaries through intuitive charts and reports. Whether recording transactions, analyzing financial trends, or managing user authentication, our app seamlessly integrates technologies to create a holistic and efficient financial management experience.