8-2023

FINSERV ANDROID APPLICATION

Harsh Piyushkumar Shah

FINSERV ANDROID APPLICATION

A Project

Presented to the

Faculty of

California State University,

San Bernardino

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

---

by

Harsh Piyushkumar Shah

August 2023

FINSERV ANDROID APPLICATION

A Project

Presented to the

Faculty of

California State University,

San Bernardino

---

by

Harsh Piyushkumar Shah

August 2023

Approved by:

Dr. Fadi Muheidat, Advisor, Computer Science and Engineering

Dr. Jennifer Jin, Committee Member

Dr. Khalil Dajani, Committee Member

ABSTRACT

The FINSERV Android application is a mobile tool designed for individuals to manage and track their finances. In financially complex world, many people struggle to maintain a clear overview of their income, expenses, and financial goals. This application aims to bridge that gap by providing users with a powerful and user-friendly platform to efficiently monitor and optimize their personal finances.

With the Personal Finance Tracking Android Application, users can effortlessly track their income and expenses, categorize transactions, and gain valuable insights into their spending patterns. The application offers features such as expense categorization and real-time expense tracking.

To enhance usability and understanding, the application employs data visualization techniques. Financial information is presented in clear and concise graphs, charts, and reports, making it easy for users to comprehend their financial status at a glance. This visual representation enhances financial literacy and enables users to make informed decisions.

ACKNOWLEDGEMENTS

There are a lot of people who have helped us during the course of this project. Without their assistance this project would have been incomplete.

I, student at California Student University – San Bernardino; feel obliged to Department of Natural Sciences, for teaching us the basics of Software development and guiding me in the right direction.

I would like to extend our gratitude to our Dr. Fadi Muheidat, my advisor, and committee members Dr. Jennifer Jin and Dr. Khalil Dajani who mentored me throughout the project development process and provided their valuable assistance throughout different stages of software development.

DEDICATION

To Mom, Dad, and my friends: Your love, support, and friendship have shaped me. This project is dedicated to you with heartfelt gratitude and appreciation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE:

INTRODUCTION

Background

We developed an Android application for managing personal finance. It consists of budgeting tools, recording income and expenses, and finance lessons.

There are existing systems like Mint, YNAB, Mvelops, etc. These systems have functionalities to track your expenses and place them in budget categories. We can personalize these categories, pay down debt, save more money and track goals. The app also shows users their credit score and net worth. These applications are lacking to guide users with financial terminologies & learning.

We developed an application which helps in financial management as well as learning about budgeting, investment options, taxes, etc. which results in financial literacy.

Significance

This application will help users keep track of daily expenses by categorizing all expenses. Based on this application will recommend changes to users' spending habits.

Users can create a budget for all categories which will help users keep expenses in line with the budget.

We will notify users for credit card, utilities & other bill payment with help of data entered by the user. Paying credit card balances each month will help to prevent interest charges and repay and avoid debt.

There will be modules to learn financial terminologies in layman language, this will enable users to make investing, budgeting & financially independent.

Purpose

This application will help users keep track of financial activities and get financial literacy. It will assist users with financial planning and learning. Users can track income and expenses and get suggestions for managing finance based on activities. Users get complete financial literacy from basic modules to advance according to financial goals.

With data visualization information will be easier to grasp and handle when it is presented in charts and infographics. It will be appealing to users with infographics, charts, and dashboards to attract attention and impress.

CHAPTER TWO:

PROJECT PROFILE

Requirement Gathering

- Finserv will have Admin and User as primary actors in the system.
- The admin side functionality will include creating and managing learning lesson in the learning module as per user requirements. The user side will only be able to view lessons added by admin.
- Updating and modifying information has to be made smooth on admin side.
- The user side functionality will include creating and managing day-to-day transactions.
- To facilitate adding transaction user will have to manage correspondent accounts.
- User will be able to generate report for added transaction for specific period.

Non-Functional Requirement

- Scalability: System should be able to handle several users. For e.g., handling around thousand users at the same time.
- Usability: Simple user interfaces that a layman can understand.
- Speed: Speed of the system should be responsive i.e., response to a particular action should be available in short period of time.

Hardware Requirement

- Operating System: Windows 10/8/7 (64-bit), macOS 10.10 (Yosemite) or later, or a Linux distribution (such as Ubuntu)
- RAM: 8 GB RAM (16 GB or more recommended)
- Disk Space: Minimum 4 GB of available disk space, but at least 2 GB of additional space for Android SDK and emulator system images
- CPU: Intel i5 processor or equivalent, with support for Intel VT-x, Intel EM64T (Intel 64), and Execute Disable (XD) Bit functionality
- Screen Resolution: 1280x800 minimum screen resolution

Software Requirement

Java (Programming Language)

Java is a widely used programming language renowned for its versatility, platform independence, and extensive libraries. Developers use it to create robust applications in diverse domains - web development, mobile apps development, enterprise systems and more - using its toolbox of libraries and frameworks; including those that facilitate feature-rich mobile app creation on Android platforms.

Android SDK

The Android Software Development Kit (SDK) provides developers with a comprehensive collection of tools, libraries, and resources for creating Android

applications. The Android SDK contains an API library with documentation and sample code; as well as debugging tools to aid efficient development. In addition, emulators enable developers to test apps on virtual devices as part of the process of app creation; plus, there's access to platform features like camera functionality, location services and push notifications allowing for powerful yet engaging Android applications for any platform.

Gradle

Gradle is a powerful build automation tool used in Android development. It enables developers to efficiently define and manage project dependencies, compile source code, run tests, package applications and perform other build tasks efficiently. Gradle's declarative Groovy or Kotlin DSL makes the build configuration process straightforward while its plugin ecosystem integrates with Android Studio providing additional functionalities like code analysis, coverage analysis and deployment automation, making Gradle an indispensable part of Android app development.

Material Design

Material Design was developed by Google as a design language that provides guidelines and principles for crafting visually appealing and user-friendly interfaces. When developing for Android devices and versions, material Design components and styles can be integrated using Android development using its

Material Design Dependency component. By doing so, developers gain access to a host of pre-built UI components - buttons, cards, navigation drawers - that adhere to material Design guidelines, creating an enjoyable and cohesive user experience across devices while increasing its visual aesthetics and usability.

PHP API

An API (Application Programming Interface) in PHP refers to a collection of functions, classes and protocols that allows developers to interact with an PHP-based application or service. PHP APIs are often utilized when building web apps as they facilitate communication among various systems or components. APIs allow PHP applications to provide data retrieval or transmission functions as well as access functionality offered by their applications. They may take the form of RESTful APIs or SOAP APIs and support different data formats like JSON or XML. Developers can build PHP APIs using frameworks like Laravel or Symfony, or by customizing API endpoints through PHP scripting. These APIs play an essential role in providing integration, data exchange, and interoperability between systems in web development projects.

MySQL

MySQL is an extremely popular relational database management system (RDBMS) that facilitates efficient storage, management, and retrieval of structured data. With features like data integrity, scalability, and high

performance it enables efficient data management as well as creation of databases containing tables with relationships established among them. MySQL allows querying of data using SQL (Structured Query Language). With extensive community support it serves various applications ranging from small websites to enterprise systems - making it a reliable option for database management.

CHAPTER THREE:

APPLICATION FLOW

Application flow refers to the sequence of screens, actions and interactions experienced when using software applications by users. It defines how users navigate between various screens within an application as they complete various tasks within it.

An app's flow should be designed in an intuitive and logical fashion, leading users through its functions while minimizing confusion or frustration. When designing this experience for users, one must keep user goals, priority features and smooth transitions between screens in mind to create an enjoyable and seamless user journey.

Splash Screen

- A splash screen is a graphical element that briefly appears upon opening an application or software program, providing a visual representation of its logo as part of an initial introduction of its contents.
- The splash screen provides an efficient transition from launch phase to main interface and helps create a consistent user experience.

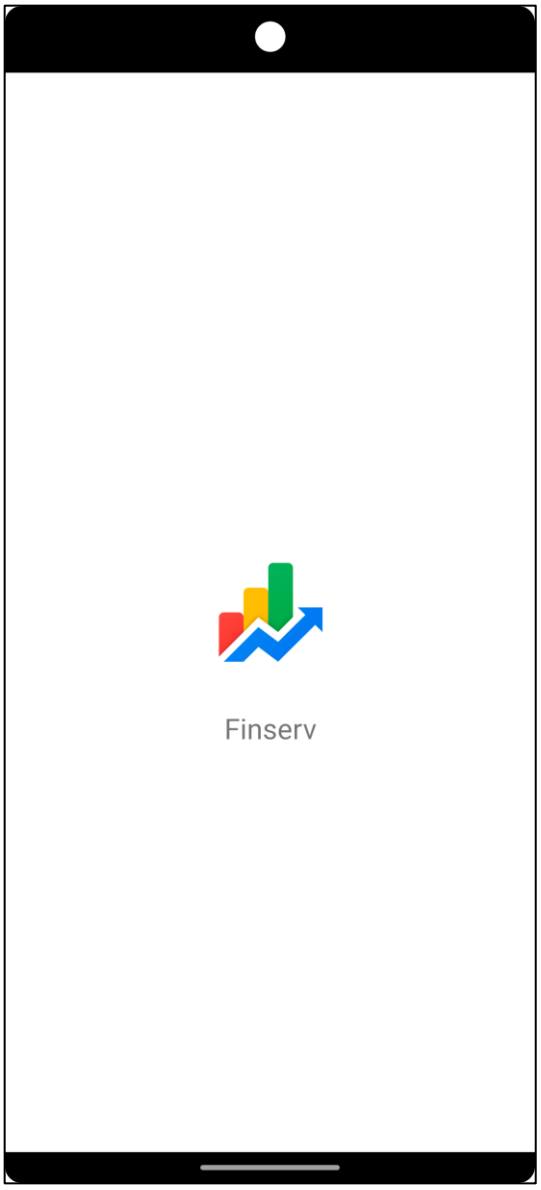

Figure 1. Splash Screen

Signup Screen

- The signup screen is a user interface component that allows individuals to create new accounts or register for a service or application.

- This includes fields for entering personal information such as name, email and password, along with a signup button to complete the registration process.

Figure 2. Signup Screen

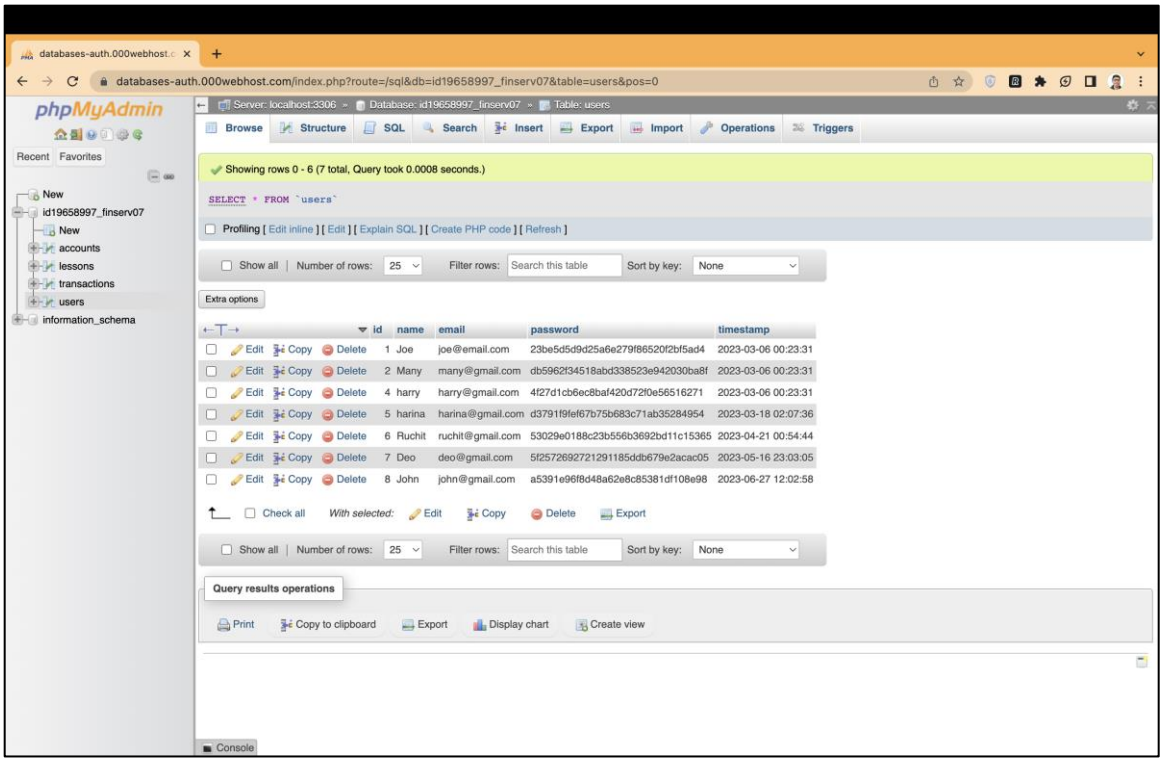

Figure 3. Database Snippet (User Registration)

Login Screen

- A login screen is a user interface component that prompts users to authenticate themselves before accessing a system, application, or online service. It typically contains fields for entering email and password credentials as part of this authentication process.
- Users can enter their credentials and click the login button to validate their identity and gain access to protected content or features. The login screen provides additional protection by masking passwords or using encryption algorithms such as MD5 for password masking and encryption of content using MD5.

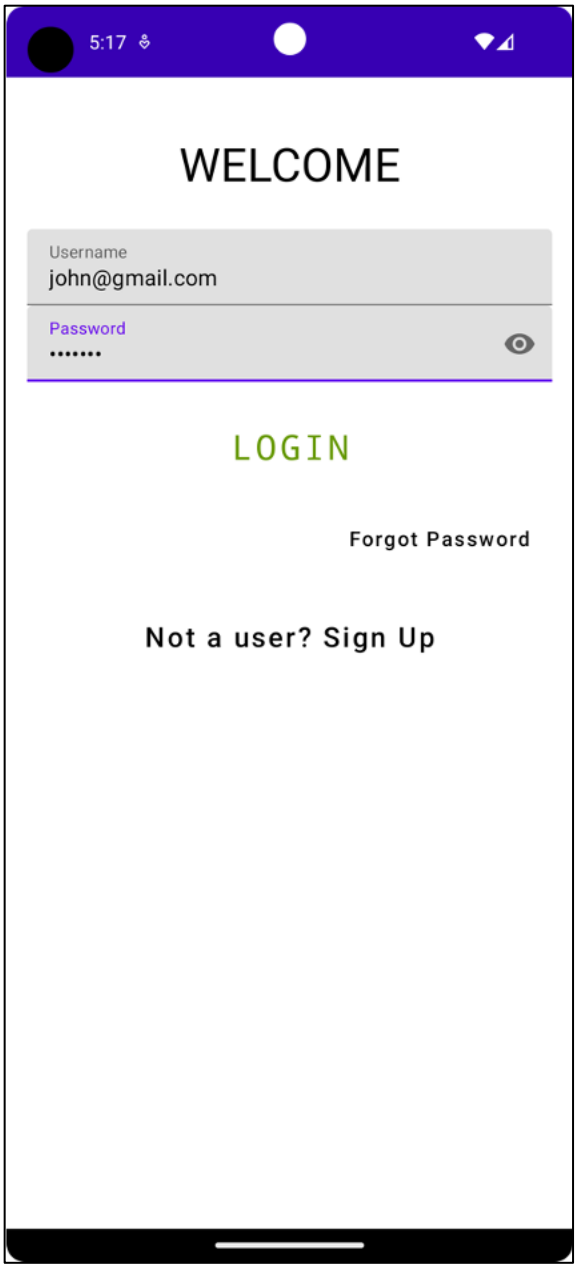

Figure 4. Login Screen

Accounts Fragment

- An Android fragment is a modular component used in development that represents part of an activity's user interface or behavior, reusable across screen sizes and orientations to create flexible layouts with responsive designs for different screen sizes and orientations. By promoting code reuse and managing complex user interfaces more easily in Android applications.
- In Accounts Fragment, user can create new accounts by providing information such as name of account, initial balance & type.

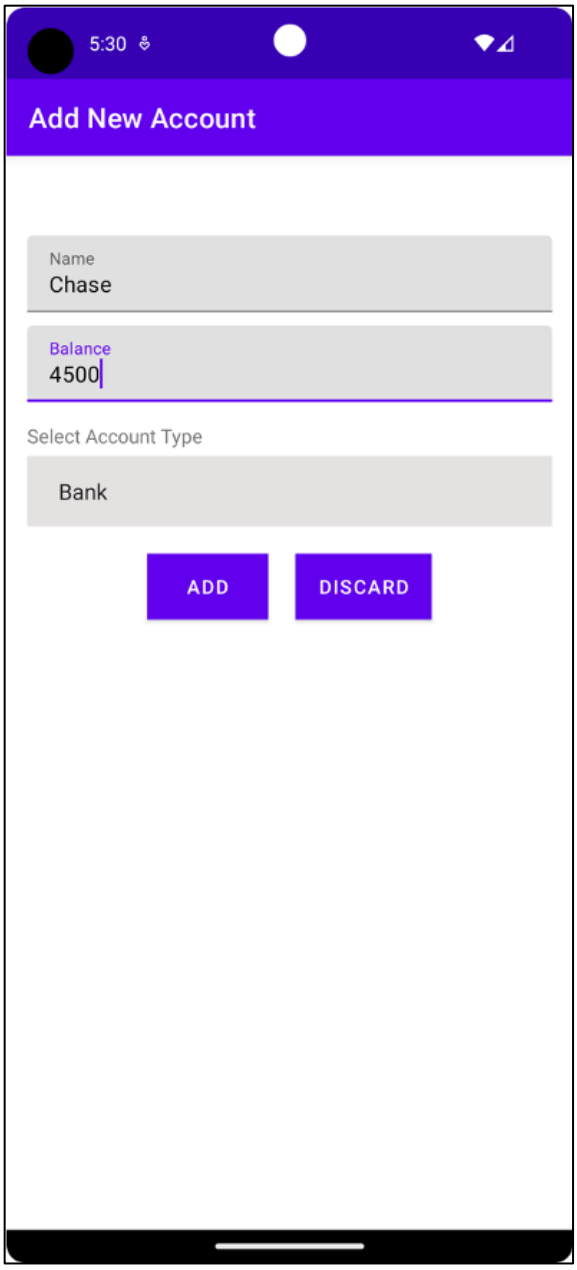

Figure 5. Add New Account Screen

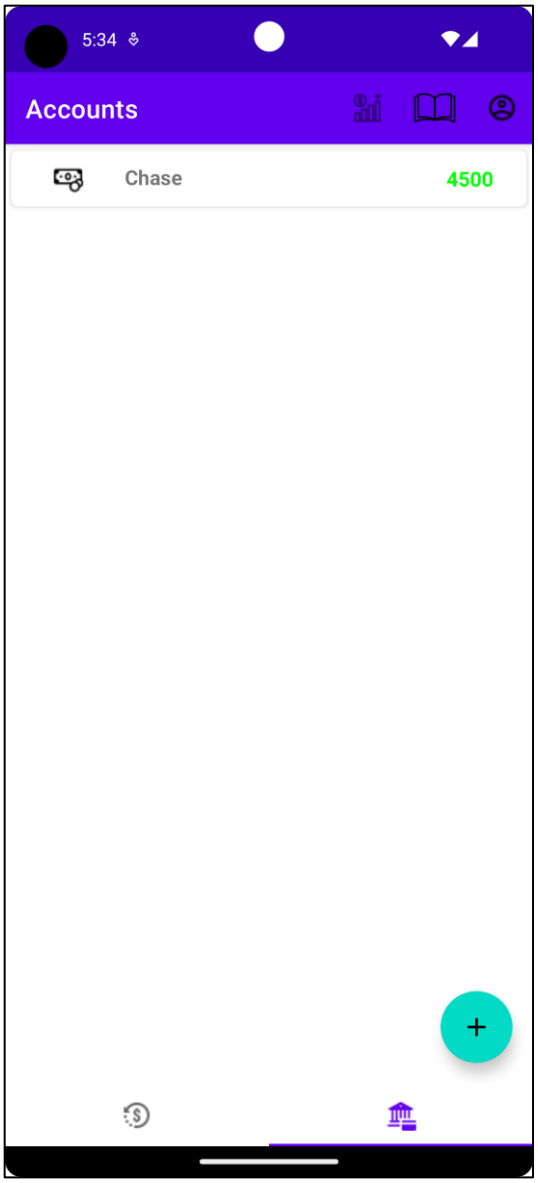

Figure 6. Accounts Fragment

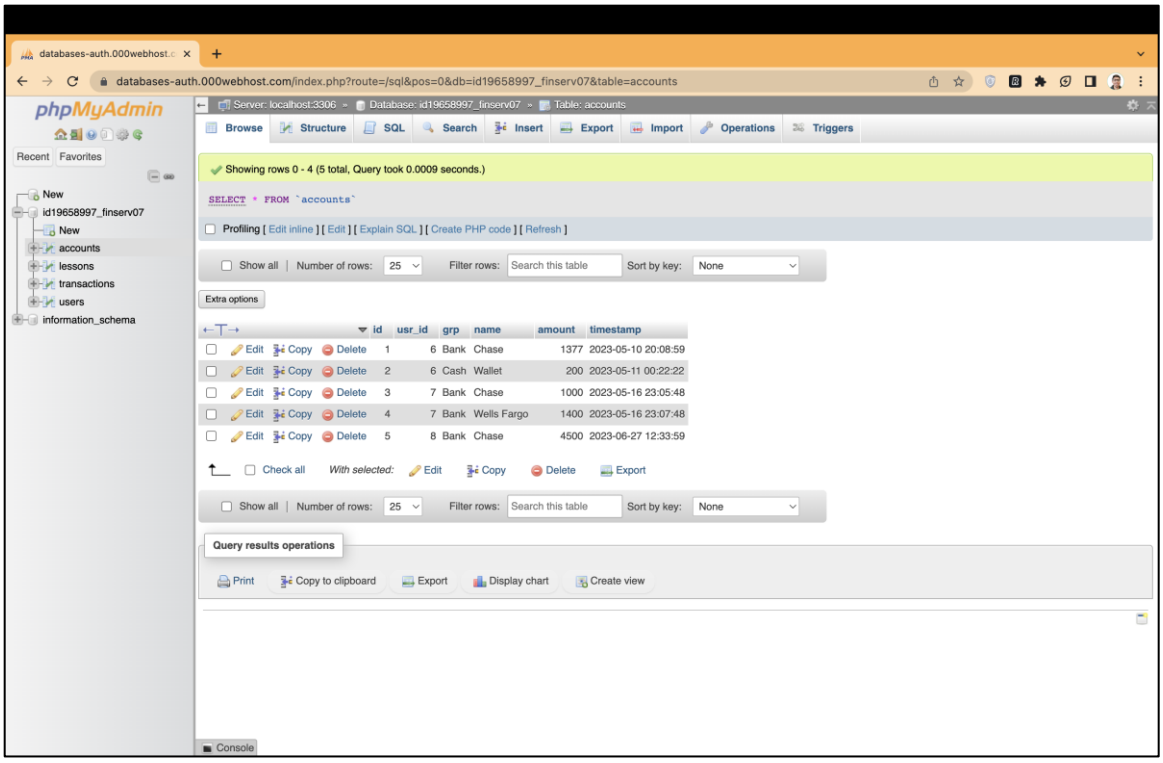

Figure 7. Database Snippet (Account Added)

Transactions Fragment

- A transactions screen is a user interface component that displays a list or overview of transactions within an application or system, often including details like transaction dates, descriptions, amounts and any relevant metadata.
- Users can use this screen to monitor financial or activity-related transactions chronologically as they happen - enabling them to filter or search specific transactions as desired while additional functions such as categorization and editing may also be available depending on its purpose.

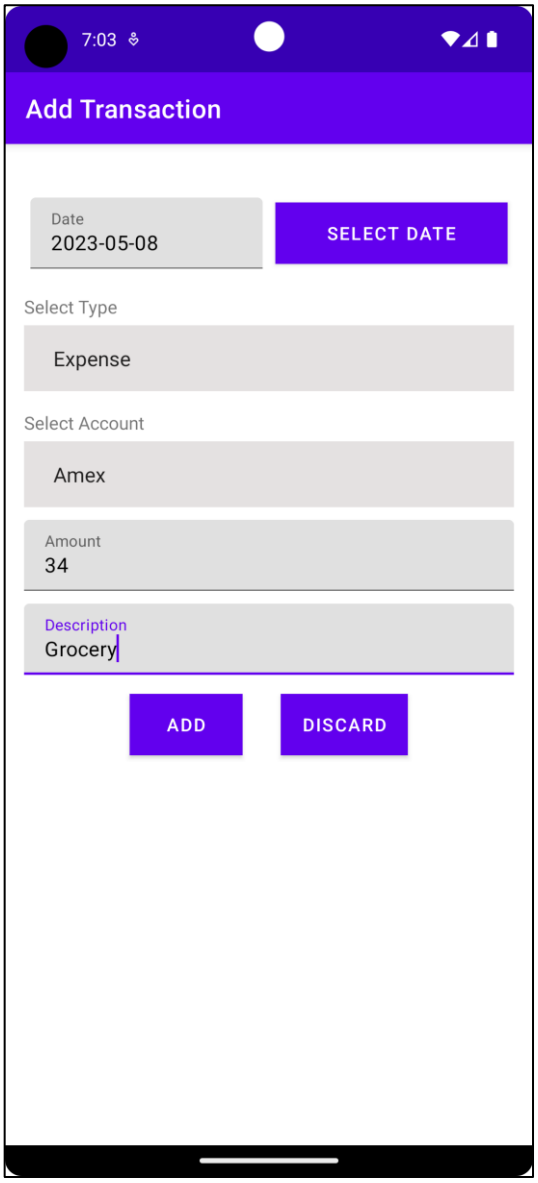

Figure 8. Add Transaction Screen

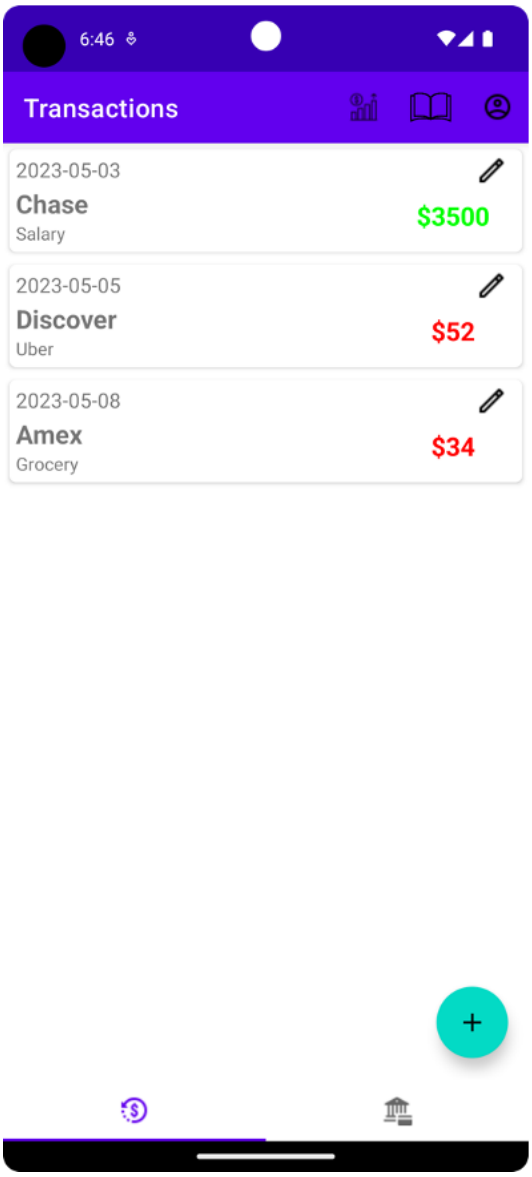

Figure 9. Transactions Fragment

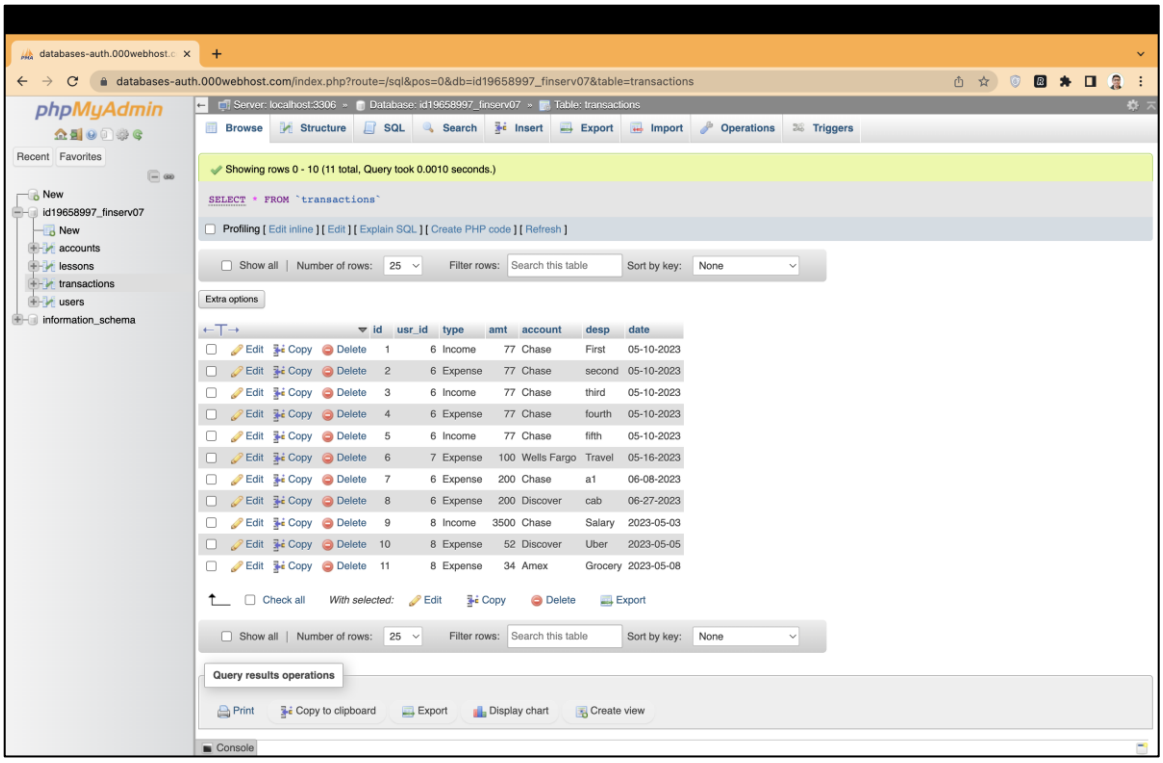

Figure 10. Database Snippet (Transaction Added)

Learning Module

- In text-based applications, learning modules refer to sections or components which offer educational content or lessons in written form, such as articles, tutorials, guides or interactive text-based exercises. They enable users to gain knowledge or develop skills via written material such as articles, tutorials or guides that help build knowledge or develop skills - these may include articles, tutorials or guides with interactive text exercises as well as written exercises which offer users the chance to gain knowledge or develop abilities through text materials such as articles. Users can

navigate their way around these modules with different chapters and topics being presented and engage with this text-based material in order to maximize understanding and learning outcomes and increase understanding and outcomes while improving results and outcomes.

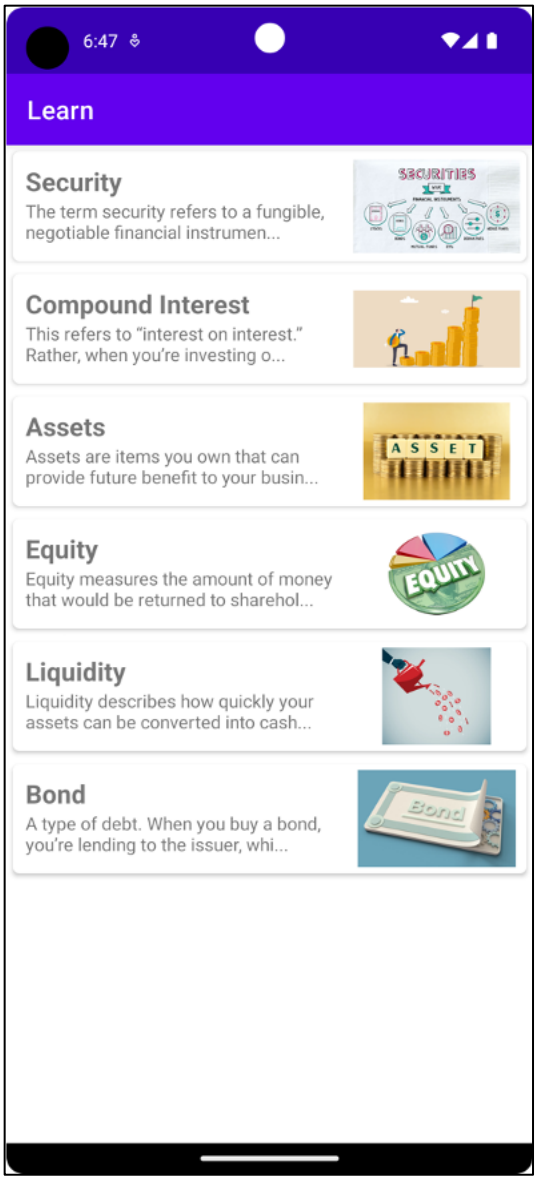

Figure 11. Learn Screen

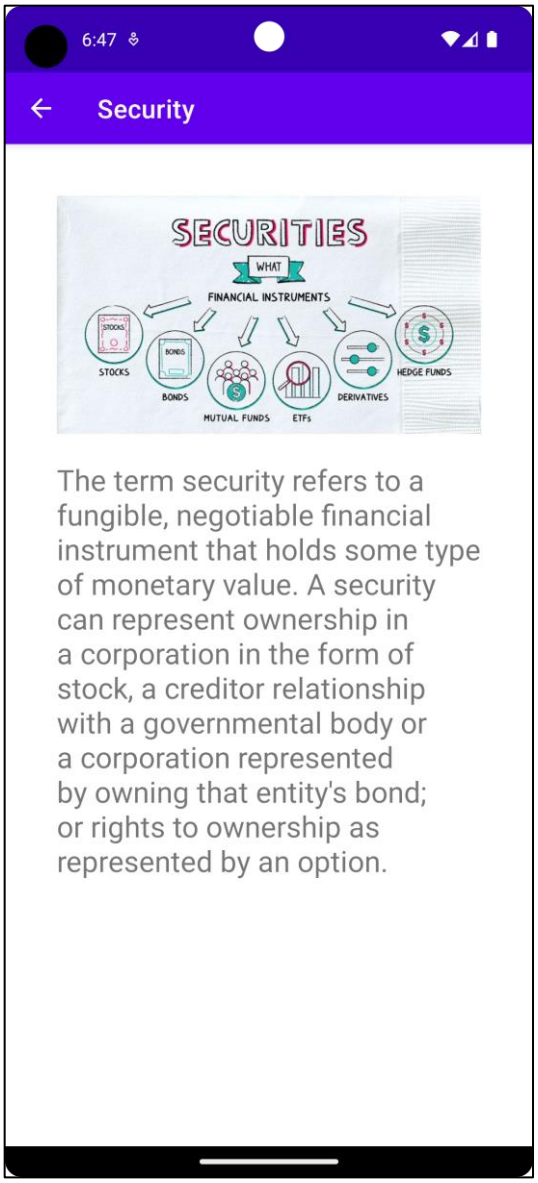

Figure 12. Description Screen

Manage Profile

- Managing a profile typically involves making changes, updates, or adjustments to a user's personal information, preferences, or settings within a system, application, or online platform. The

specific actions and options for managing a profile may vary depending on the context and the features provided by the platform.

- Editing personal information: Users can modify details such as their name, email address, and password.

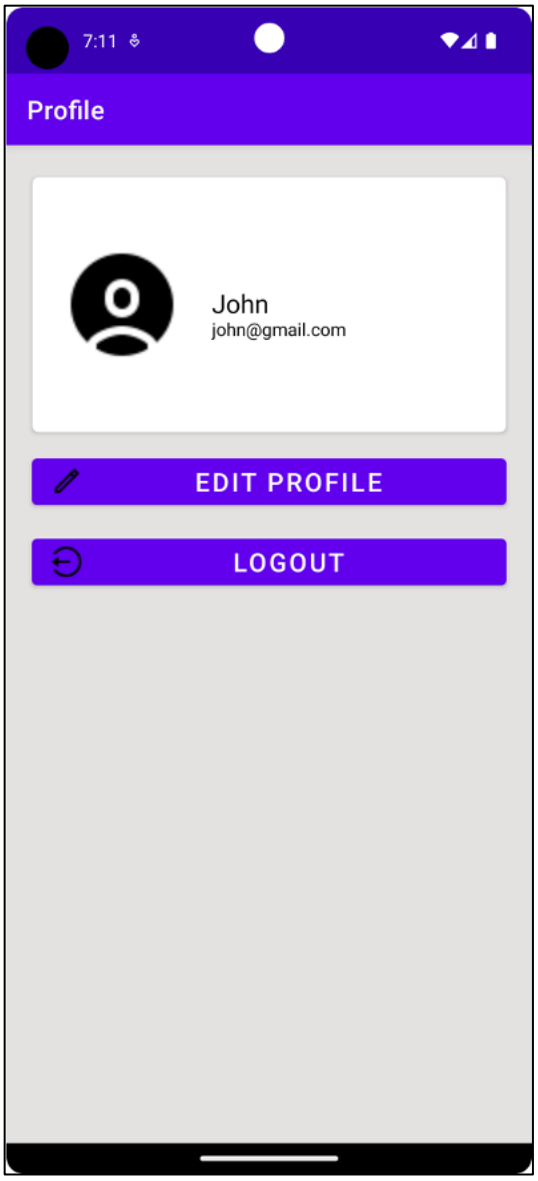

Figure 13. Profile Screen

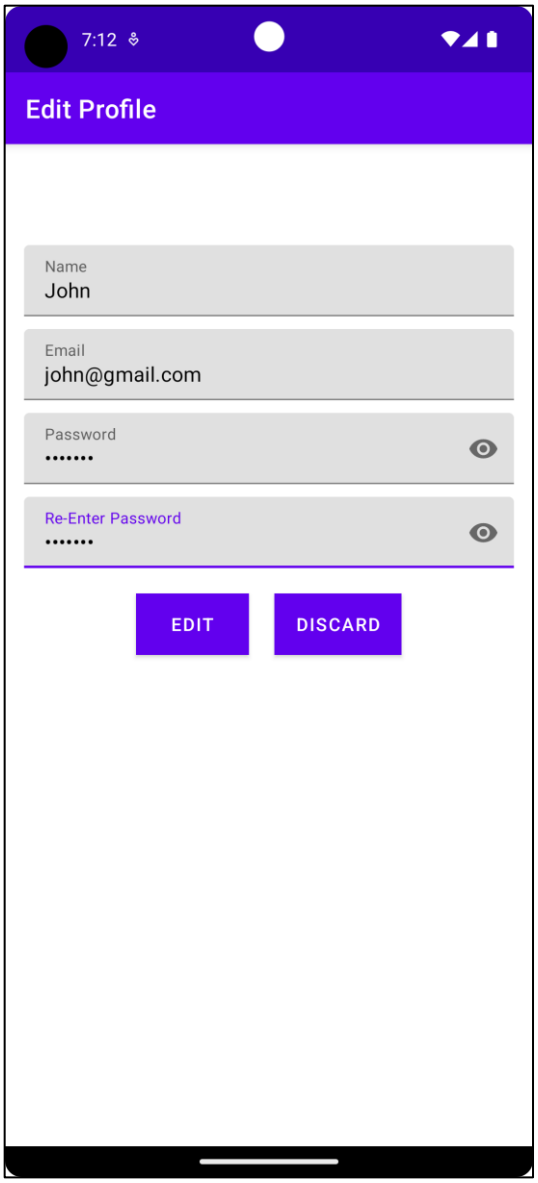

Figure 14. Edit Profile Screen

Transaction Statistics

- Transaction statistics refer to numerical data and metrics that provide insights into various aspects of transactions. These

statistics offer valuable information about the volume, patterns, trends, and performance of transactions.

- Total Transactions: The total number of transactions processed within a specific period.
- Total Amount: The total monetary value or amount of all transactions conducted.
- Total Income: Total value of income transactions.
- Total Expense: Total value of expense transactions.
- Net Total: Total expense subtracted for total income represents net total.

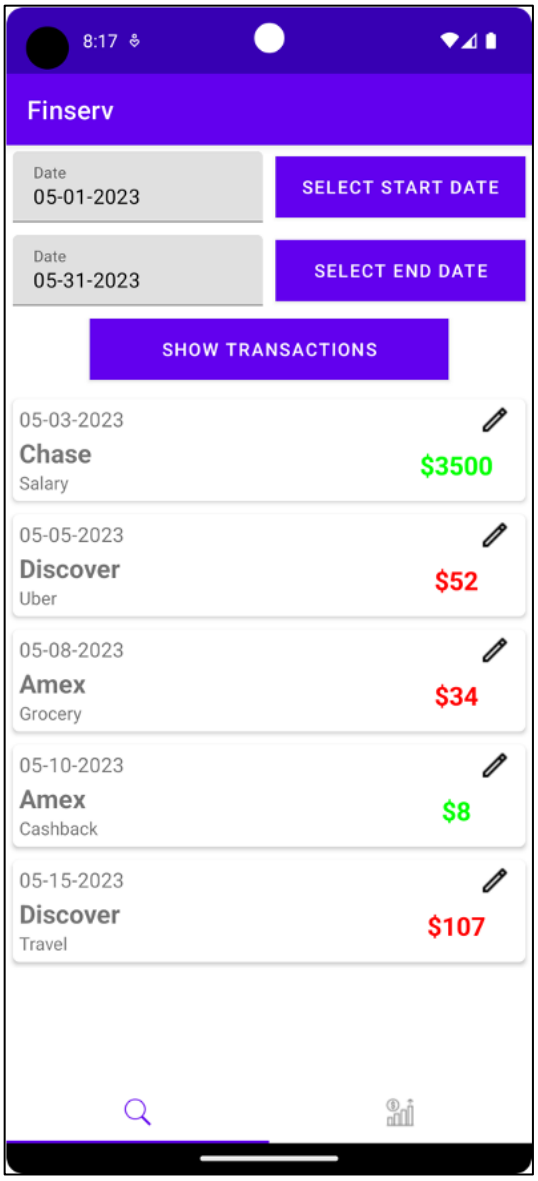

Figure 15. Search Transaction Fragment

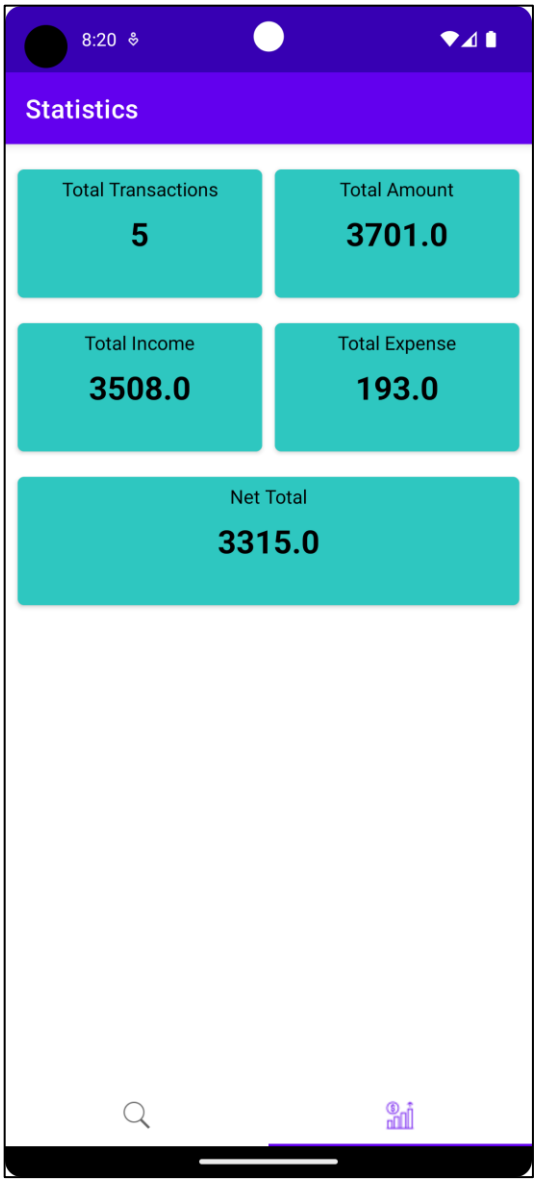

Figure 16. Statistics Screen

CHAPTER FOUR:

DATA DICTIONARY

Data Dictionaries are documentation resources designed to provide an in-depth description and definition of data elements used within an application, serving as an invaluable guide for understanding their structure, meaning, and relationships.

Attributes of Data Dictionary

Field Name

In a data dictionary, a field name refers to the unique identifier or label assigned to a specific data element or attribute within a system or database. It serves as a descriptive and concise identifier that helps users and developers understand and reference the data. Field names should be chosen carefully to reflect the nature and purpose of the data they represent, enabling effective data management and communication within the system.

Data Type

Data types in a data dictionary refer to the format or category of information stored within an attribute or field, such as text, numeric, date or boolean values. They provide guidance for how the data should be stored, validated and

processed - therefore understanding data types across each field is integral for proper handling and manipulation within systems or databases.

Size

Size in a data dictionary refers to the maximum size or length allowed for any particular field or attribute in terms of characters, bytes or both; typically this represents maximum character storage capacity or storage efficiency; size restrictions are often defined to maintain data integrity, optimize storage efficiency and facilitate processing; by specifying sizes within the dictionary developers and users are aware of their limitations in designing validating and processing their data to avoid truncation or loss of information.

Description

Descriptions in data dictionaries refer to textual explanations or summaries that provide more details about specific data elements or attributes, elaborating their purpose, meaning, context and significance for users and developers alike. A clear and informative description promotes data management within systems or databases by helping people better comprehend data usage, expected values or relationships between elements as well as any relevant business rules or considerations.

Tables in Database

Table 1. User Table

users				
Field Name	**Data Type**	**Size**	**Constraint**	**Description**
id	int	11	Primary Key	Auto generated ID
name	varchar	255	Not Null	name of the user
email	varchar	255	Not Null	email of the user
password	varchar	255	Not Null	encrypted password
timestamp	timestamp		Not Null	Auto generated UTD time

Table 2. Lessons Table

lessons				
Field Name	**Data Type**	**Size**	**Constraint**	**Description**
id	int	11	Primary Key	Auto generated ID
title	varchar	255	Not Null	title of the lesson
desp	varchar	1500	Not Null	brief description
file	varchar	255	Not Null	image file name
timestamp	timestamp	13	Not Null	Auto generated UTD time

Table 3. Accounts Table

accounts				
Field Name	**Data Type**	**Size**	**Constraint**	**Description**
id	int	11	Primary Key	Auto generated ID
usr_id	int	11	Foreign Key	Associated user ID
grp	varchar	255	Not Null	group of the account
name	varchar	255	Not Null	account name
amount	decimal	17	Not Null	account balance
timestamp	timestamp	13	Not Null	Auto generated UTD time

Table 4. Transactions Table

transactions				
Field Name	**Data Type**	**Size**	**Constraint**	**Description**
id	int	11	Primary Key	Auto generated ID
usr_id	int	11	Foreign Key	Associated user ID
type	varchar	255	Not Null	type of the transaction
amt	decimal	17	Not Null	account balance
account	varchar	255	Not Null	account name
desp	varchar	255	Not Null	Description
date	varchar	255	Not Null	date of transaction

APPENDIX A:

USE-CASE DIAGRAM

Use Case Diagrams are visual depictions of interactions among actors such as users, systems or external entities and an application or system. They help demonstrate its functionality and behaviors from its users' point of view. Here are important components and concepts found within a use case diagram:

1. Use Cases: Use cases are the specific actions, functionalities and behaviors provided by the system to its users. Each use case represents a goal or task the users wish to complete within it; use cases are depicted by ovals within the system boundary that connect directly with actors through lines.

2. Actors: Actors represent all the different users, systems, or external entities who interact with a system and can be represented as stick figures or icons placed outside its boundary. Actors can either be primary users such as customers or administrators or secondary users such as external systems or devices.

3. Relationships: Relationships represent the associations and dependencies among actors and use cases, and are defined in use case diagrams by various types of relationships such as:

Association and Generalization relationships establish general connections between actors and use cases. Generalization represents an "is-a"

relationship, signifying that one actor/use case represents a more specific version of another actor or use case.

Include: When one use case includes another use case's functionality, it becomes essential for its completion and should therefore be included as part of its base use case.

Extend: An extend use case represents optional or alternative functionality that extends a base use case, which may or may not be executed depending on certain conditions or scenarios.

4. System Boundary: The system boundary is a rectangular box enclosing all use cases, actors, and their relationships within its confines; this represents the scope or boundary of any system being evaluated.

5. Annotations: Use case diagrams may include additional textual information to provide additional clarification or description for certain elements, while annotations can be added to use cases, actors or relationships for added context or details.

Use case diagrams are an invaluable communication tool, helping stakeholders visualize and comprehend a system's functionality as well as user interactions with it. They offer a high-level overview of its behavior that can serve

as the basis for requirements gathering, system design, and documentation purposes.

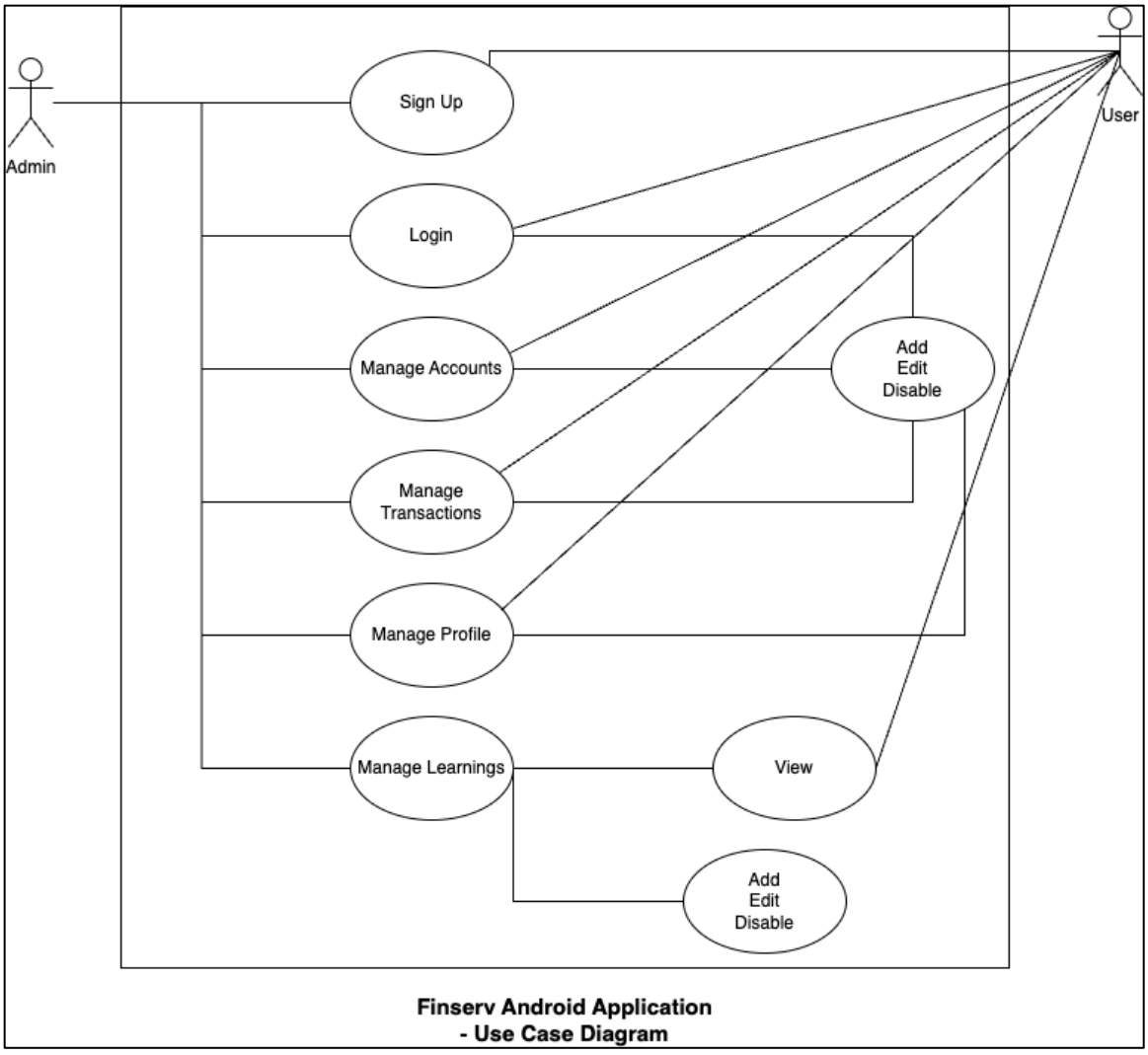

Figure 17. Use Case Diagram

REFERENCES

[1] GeekforGeeks, accessed 7 February 2023, <https://www.geeksforgeeks.org/>

[2] Jeff Friesen, (2014). *Learn Java for Android Development*. Apress

[3] Material Design, accessed 2 January 2023, <https://m3.material.io/>

[4] Official Android Documentation, accessed 23 September 2022,

<https://developer.android.com/docs>

[5] Stackoverflow, accessed 12 September 2022, <https://stackoverflow.com/>