Software Requirements Specification (SRS) and Design Document Template

1. Introduction

a) Purpose

The purpose of this Software Requirements Specification (SRS) is to describe in detail the functional and non-functional requirements of the project titled AI Financial Advisory and Planning Mobile Application. The document serves as a guide for developers, designers, testers, and evaluators involved in the project's lifecycle. It defines the purpose, scope, and system behavior in a clear and comprehensive manner to ensure the development of a robust and user-friendly application.

The AI Financial Advisory and Planning application aims to help users manage their finances efficiently by providing personalized financial advice using artificial intelligence and machine learning. The system is designed to analyze a user's income, expenses, age, financial goals, and risk profile to generate recommendations on savings, investments, SIPs, and emergency fund planning. This SRS ensures that every component of the system is well-defined, documented, and aligned with user requirements and project objectives.

b) Scope

The *AI Financial Advisory and Planning Mobile Application* is designed to act as a virtual financial assistant for users seeking guidance in personal finance management. The application's scope covers the automation of financial planning through intelligent algorithms that provide customized suggestions to users based on their financial inputs and future goals.

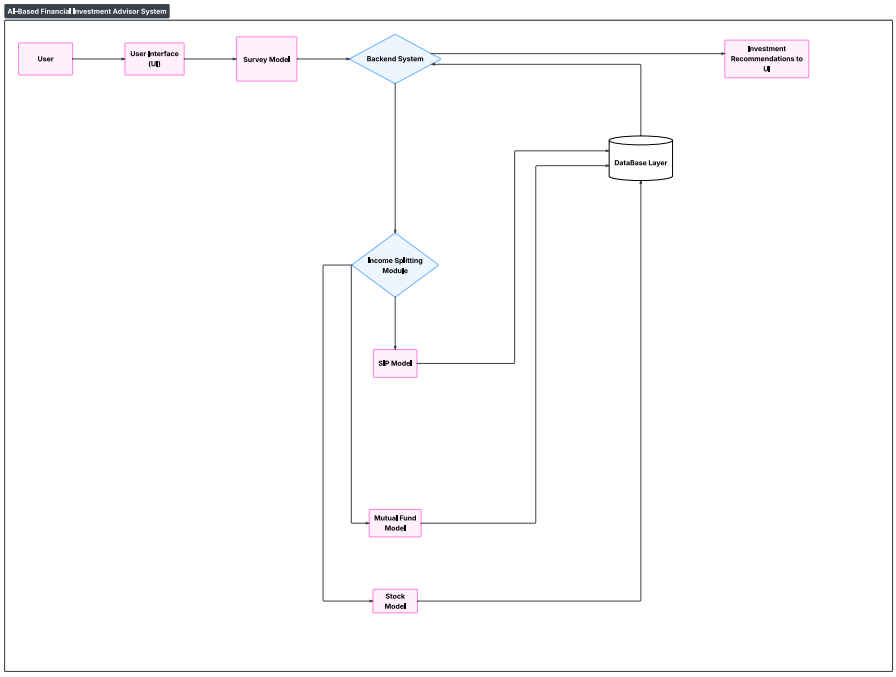
The mobile application will be developed using **Java and XML** for the Android frontend and **TensorFlow Lite** for machine learning-based financial prediction. The backend will handle user data processing, model inference, and storage operations. The app is capable of operating in both **online and offline modes**, allowing users to access predictions without an active internet connection once the model is downloaded.

The system will not perform real-time stock trading or live financial transactions. Its core purpose is to provide analytical recommendations, goal-based savings plans, and investment suggestions. The main benefits of this system include simplified financial management, real-time analytics, increased financial literacy, and data-driven decision-making.

2. Overall description

2.1 Product perspective

2.1.1. Block Diagram



2.1.2. Hardware Requirements

The hardware requirements for the system are minimal as the application is designed to run on standard Android smartphones.

- **Processor:** Minimum Octa-core 1.8 GHz or above
- RAM: Minimum 4 GB
- **Storage:** At least 10 GB of free internal memory
- **Internet Connection:** Required for model updates and Firebase synchronization (optional for offline prediction)

2.1.3. Software Requirements

- **Operating System:** Android 9.0 (Pie) or later
- **Development Environment:** Android Studio IDE
- **Programming Language:** Java and XML
- **Machine Learning Framework:** TensorFlow Lite
- **Database:** Firebase Realtime Database / SQLite (for local storage)
- **Build Tool:** Gradle
- **Version Control:** Git

The combination of Java, XML, and TensorFlow Lite ensures flexibility, high performance, and compatibility across a wide range of Android devices.

2.2 Product functions

The application performs several major functions to achieve its objectives of financial planning and recommendation generation:

1. **User Registration and Authentication:**
   Users create an account or log in using credentials. Firebase Authentication ensures secure access control and user management.
2. **Financial Data Input:**
   The user enters their income, monthly expenses, savings goals, and age. This data is stored securely in the system.
3. **AI-Based Prediction:**
   The system processes the input data and executes the TensorFlow Lite model to predict optimal financial advice tailored to the user's profile.
4. **Recommendation Generation:**
   The AI module produces detailed recommendations, including ideal savings percentages, suggested SIP amounts, investment avenues, and emergency fund targets.
5. **Data Visualization:**
   The results are displayed through intuitive charts and textual descriptions to help users better understand their financial standing.
6. **Profile Management:**
   Users can modify financial details, update goals, or adjust preferences at any time. All changes trigger new AI-based calculations.
7. **Offline Support:**
   The TensorFlow Lite model allows offline prediction without internet connectivity, improving usability and accessibility.
8. **Data Storage:**
   The application saves all user data securely using local or cloud-based databases, ensuring persistence across sessions.

3. Specific requirements

3.1 External interfaces

The application's external interface consists of user input forms and output displays. Users interact with the app through a mobile-friendly interface designed in XML. The input interface collects data such as income, expenses, age, and risk profile. The output interface presents the processed results from the AI model, which include investment recommendations, savings distribution, and other financial insights. The app interface is intuitive, simple to navigate, and responsive to different screen sizes.

3.2 Functional Requirements

The software's functionality is defined by a series of core features that dictate how it responds to user actions. The system shall allow users to register and log in using secure credentials authenticated through Firebase. It shall enable users to enter their financial data, including monthly income, expenses, and savings targets. Once this information is submitted, the system shall preprocess the data and pass it to the TensorFlow Lite model for analysis.

After processing, the system shall generate personalized recommendations that outline financial advice specific to each user. The application shall also provide an option to visualize this data through interactive graphs or progress indicators. Furthermore, the system shall allow users to modify or delete their previously entered data and save updated information. The application shall ensure that all user data remains confidential and secure at all times.

3.3 Performance requirements

The performance of the application must ensure both responsiveness and reliability. The system should be able to process a user's financial data and generate results within three seconds. The application shall support up to 1,000 simultaneous active users without performance degradation. Under normal operating conditions, the system should process 20–25 transactions or calculations per user session.

During peak load conditions, the system should maintain an acceptable response time without exceeding a delay of five seconds. Memory usage should not exceed 100 MB at runtime to ensure compatibility with standard Android devices. Furthermore, the TensorFlow Lite model should be optimized for low-latency inference, ensuring smooth and real-time prediction.

3.4 Design constraints

The design of the AI Financial Advisory application is subject to several constraints. It must comply with Android development standards and security policies. Hardware limitations, such as memory and processing power of mobile devices, restrict the complexity of the AI model that can be deployed. The AI model must be converted into a TensorFlow Lite format to ensure compatibility with mobile environments. Additionally, the system requires internet connectivity for downloading model updates and syncing user data. All development tools must be open-source or free for educational purposes.

3.5 Logical database requirements

The application uses either Firebase Cloud Firestore or SQLite for data storage, depending on connectivity. The database stores user profiles, income details, expense records, and generated financial recommendations. Each user is identified by a unique user ID to maintain data integrity and avoid duplication. Relationships exist between users and their multiple financial goals, allowing the system to store and retrieve recommendations efficiently.

The logical database design includes entities such as **User**, **FinancialInput**, **Goal**, and **Recommendation**.
Each user has a unique ID linked to multiple goals and corresponding financial recommendations.

- **Entities:**
  - User (User_ID, Name, Email, Age, RiskProfile)
  - FinancialInput (User_ID, Income, Expenses, SavingsGoal)
  - Recommendation (User_ID, AdviceText, SIPAmount, EmergencyFund)
- **Relationships:**
  - One-to-Many relationship between User and Recommendation.
  - One-to-One relationship between User and FinancialInput.
- **Integrity:**
  - All financial inputs must be validated before processing.
  - Each recommendation is linked to a single valid user profile.

3.6 Software system attributes

3.6.1 Reliability

The system is expected to deliver consistent and accurate predictions. The AI model should maintain a high reliability rate (above 95%) in providing correct financial guidance under the same input conditions. Error handling and exception management mechanisms ensure smooth execution even in case of invalid input or interrupted connectivity.

3.6.2 Availability

The application must remain available for use at all times. With its offline AI processing capability through TensorFlow Lite, users can access essential features without internet access. Firebase synchronization ensures that all user data is updated whenever connectivity is restored.

3.6.3 Security

Security is one of the most critical aspects of the system. All user data, including financial details and personal information, is encrypted before storage. Firebase Authentication manages login security, ensuring only verified users can access the system. Sensitive data is never shared externally, and all communications between the client and database are protected using HTTPS protocols.

3.6.4 Maintainability

The system is designed with modularity in mind, allowing easy maintenance and updates. The codebase follows the Model-View-Controller (MVC) architecture to separate concerns. New AI models or UI features can be integrated with minimal changes to the existing code. Regular updates and bug fixes can be rolled out without affecting the overall functionality.

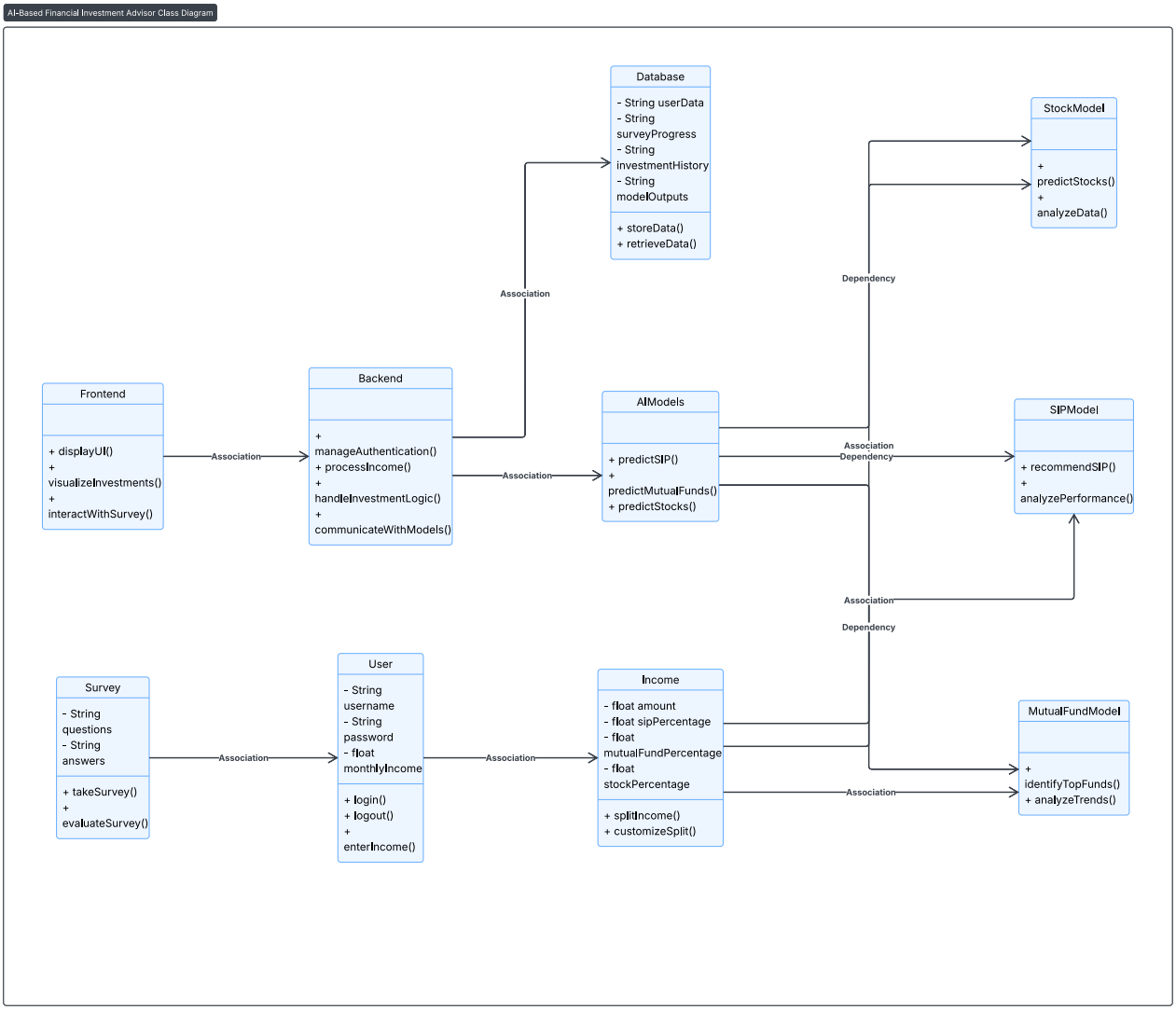
3.6.5 Portability

Although initially developed for the Android platform, the system is designed to be platform-independent at the logic level. The AI model and backend components can be easily ported to iOS or web applications in future versions using frameworks like Flutter or React Native. This ensures scalability and adaptability across multiple platforms.

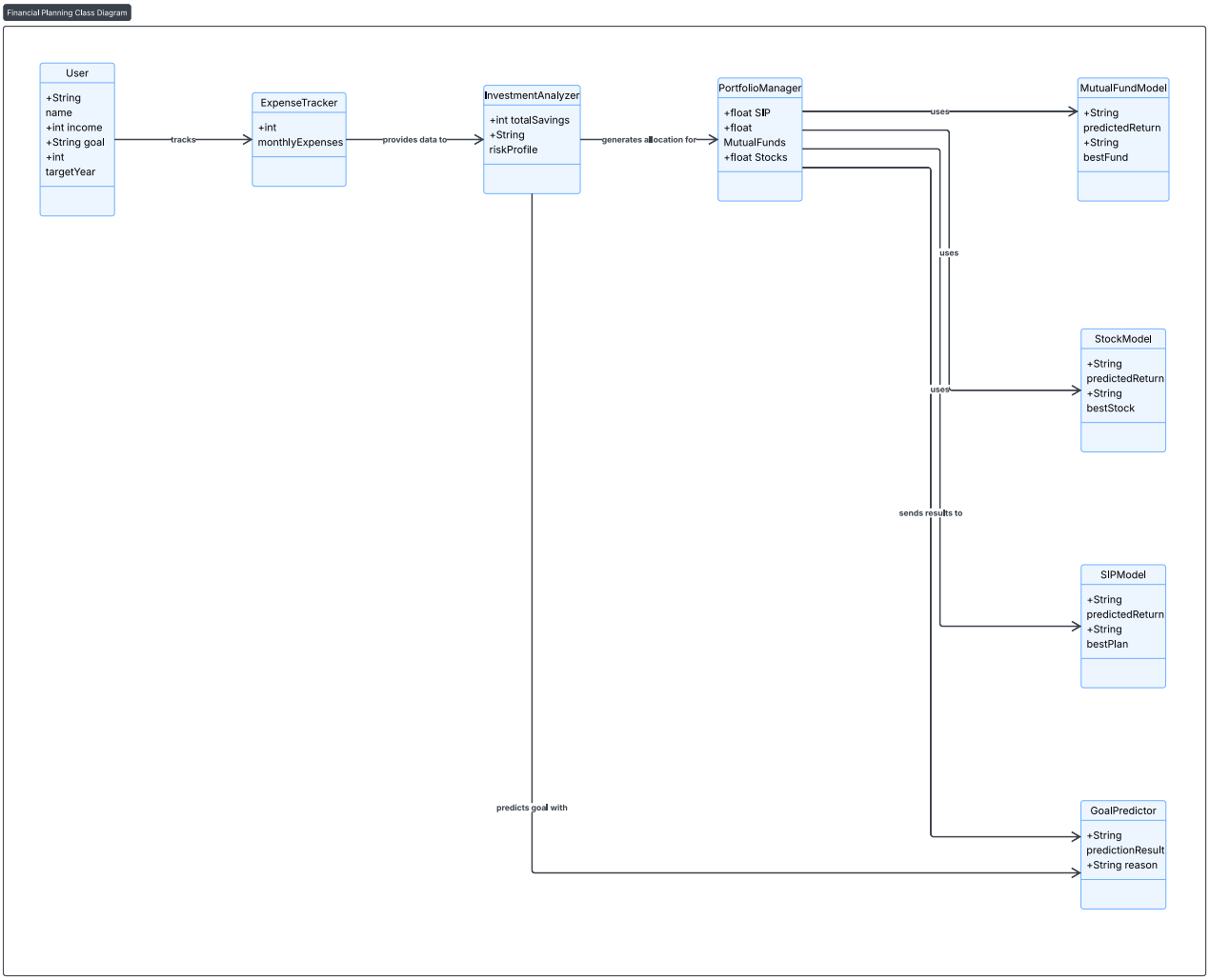
4. Software Design Document

4.1. **Structural Design**

a. Class Diagrams



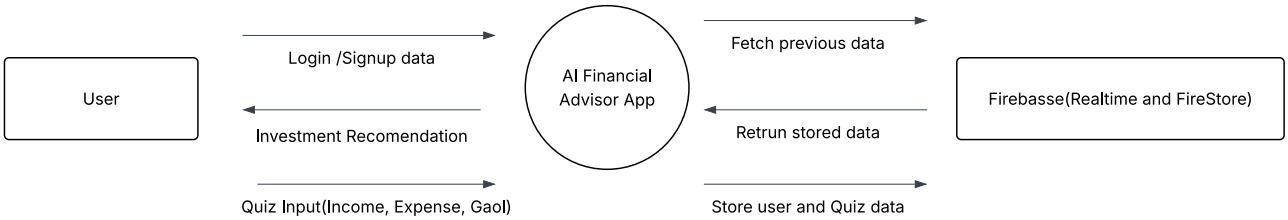
b. Object Diagrams



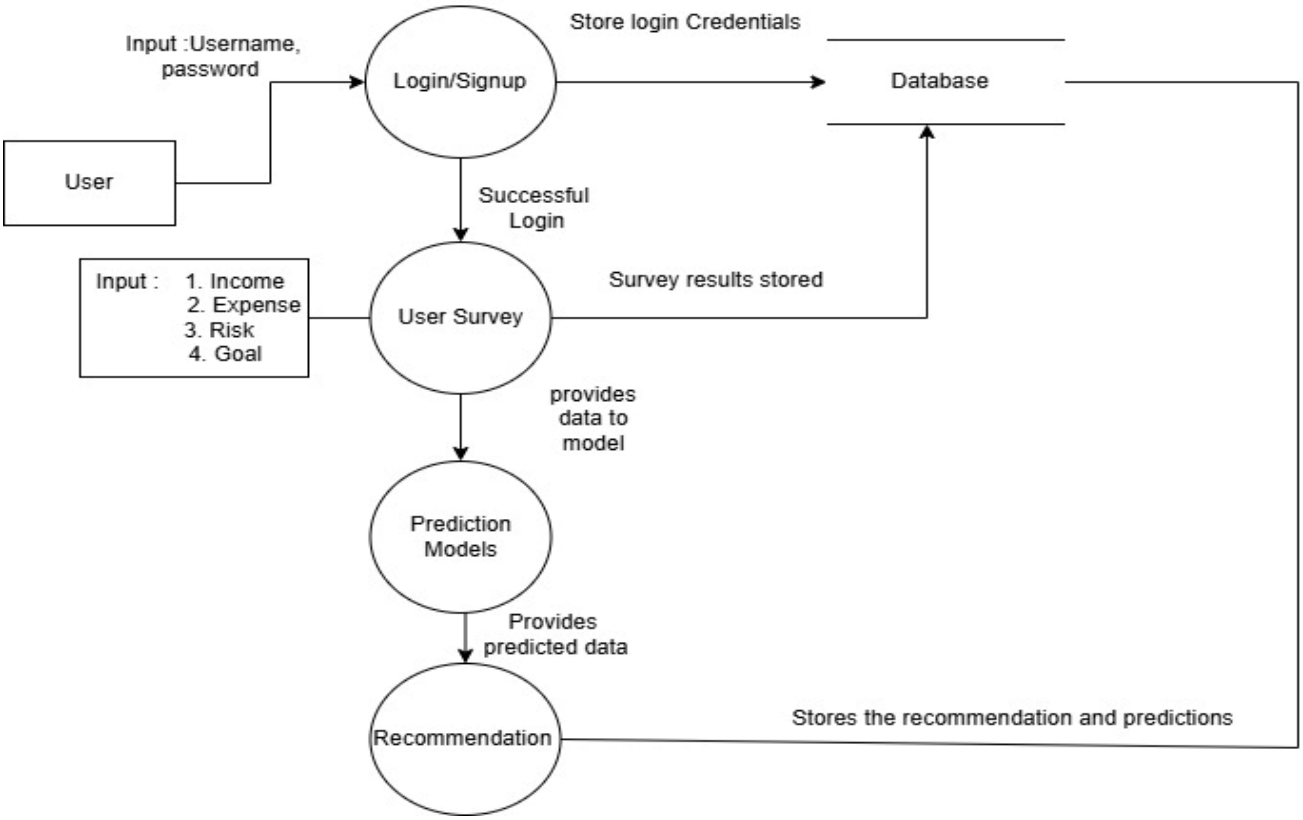
4.2. **Data Design**

a. Data-flow Diagram

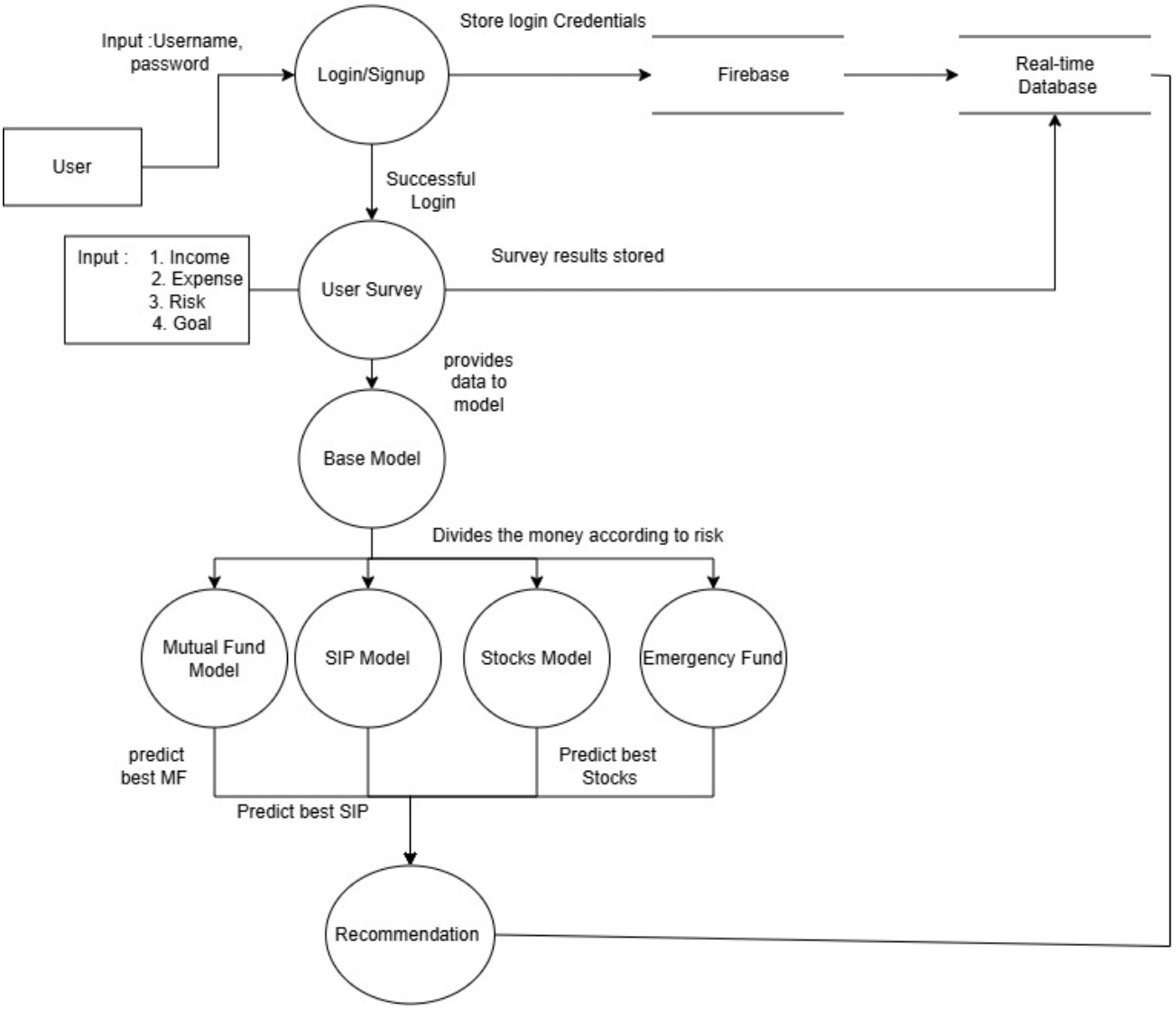
DFD Level 0



DFD Level 1

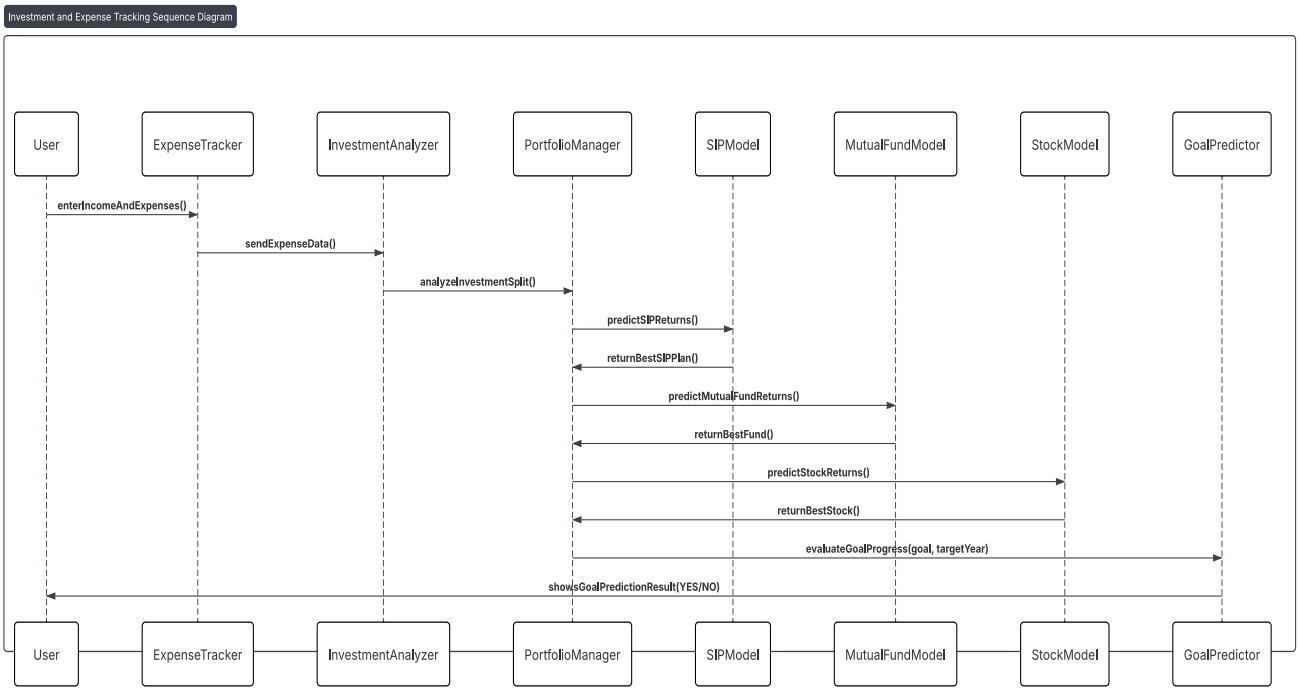


DFD Level 2

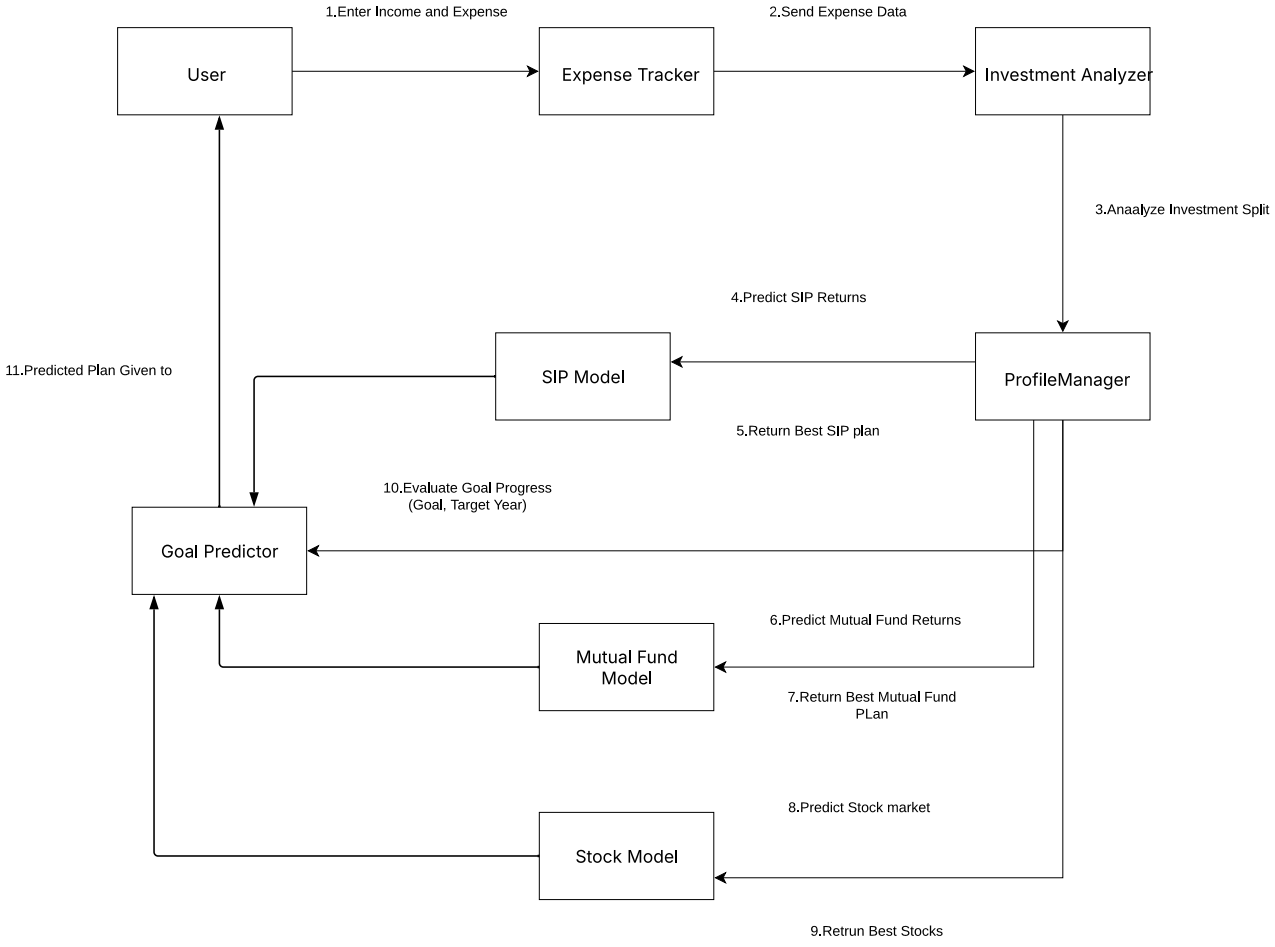

Input :Username, password
Login/Signup
Store login Credentials
Firebase
Real-time Database
User
Successful Login
Input : 1. Income 2. Expense 3. Risk 4. Goal
User Survey
Survey results stored
provides data to model
Base Model
Divides the money according to risk
Mutual Fund Model
SIP Model
Stocks Model
Emergency Fund
predict best MF
Predict best SIP
Predict best Stocks
Recommendation

4.3. **Behavioural Design**

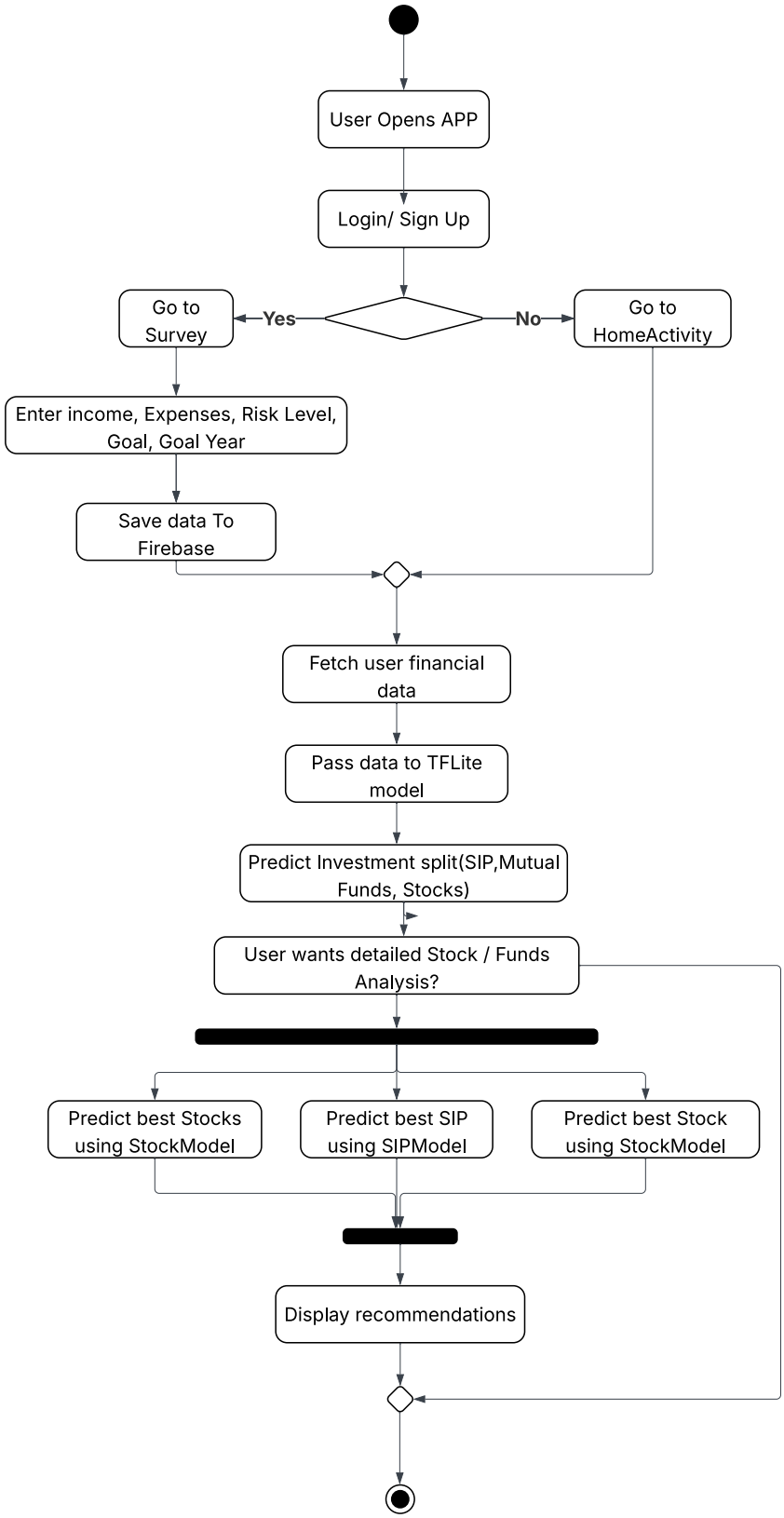
a. Sequence Diagram



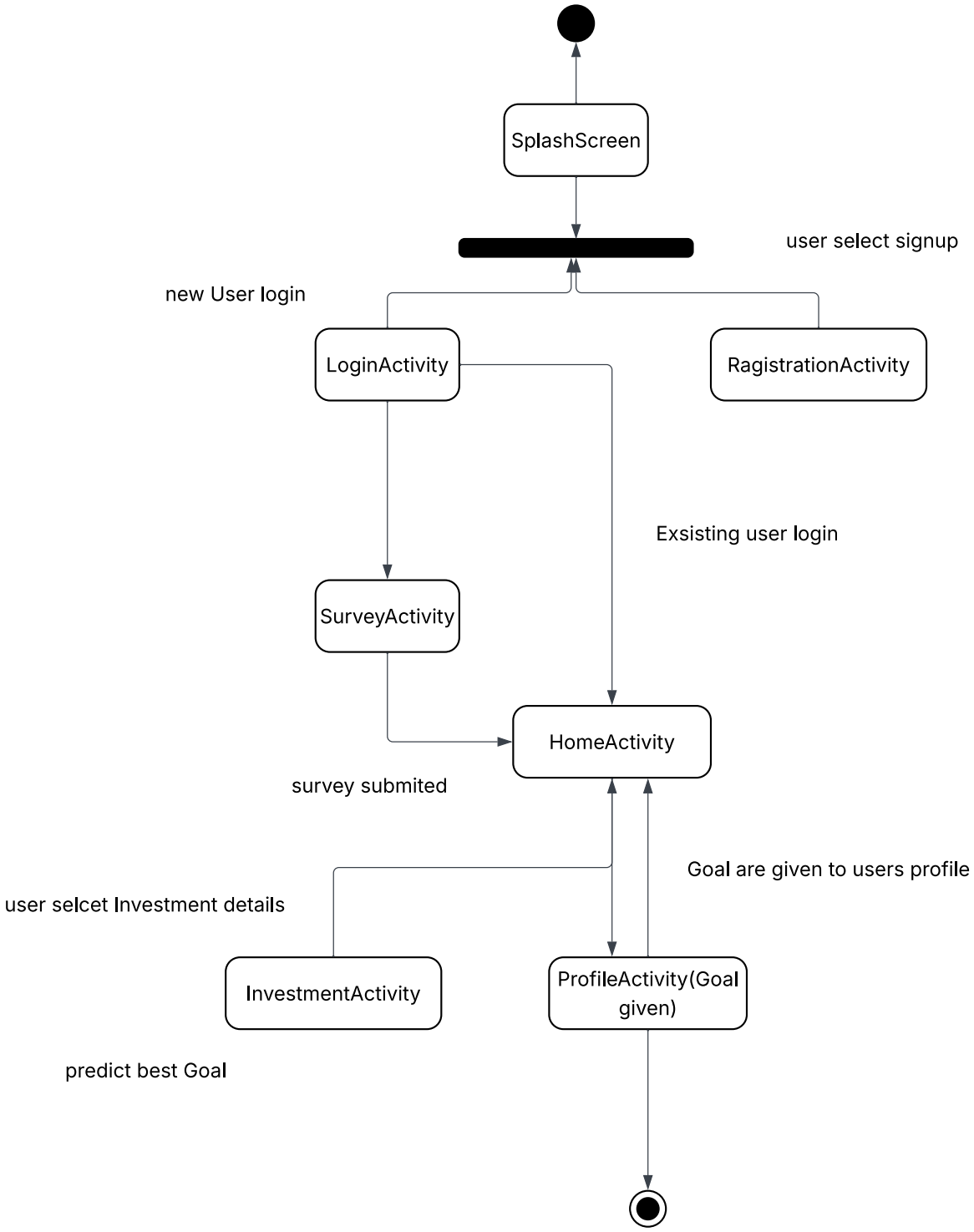
b. Collaboration diagram



c. Activity Diagram

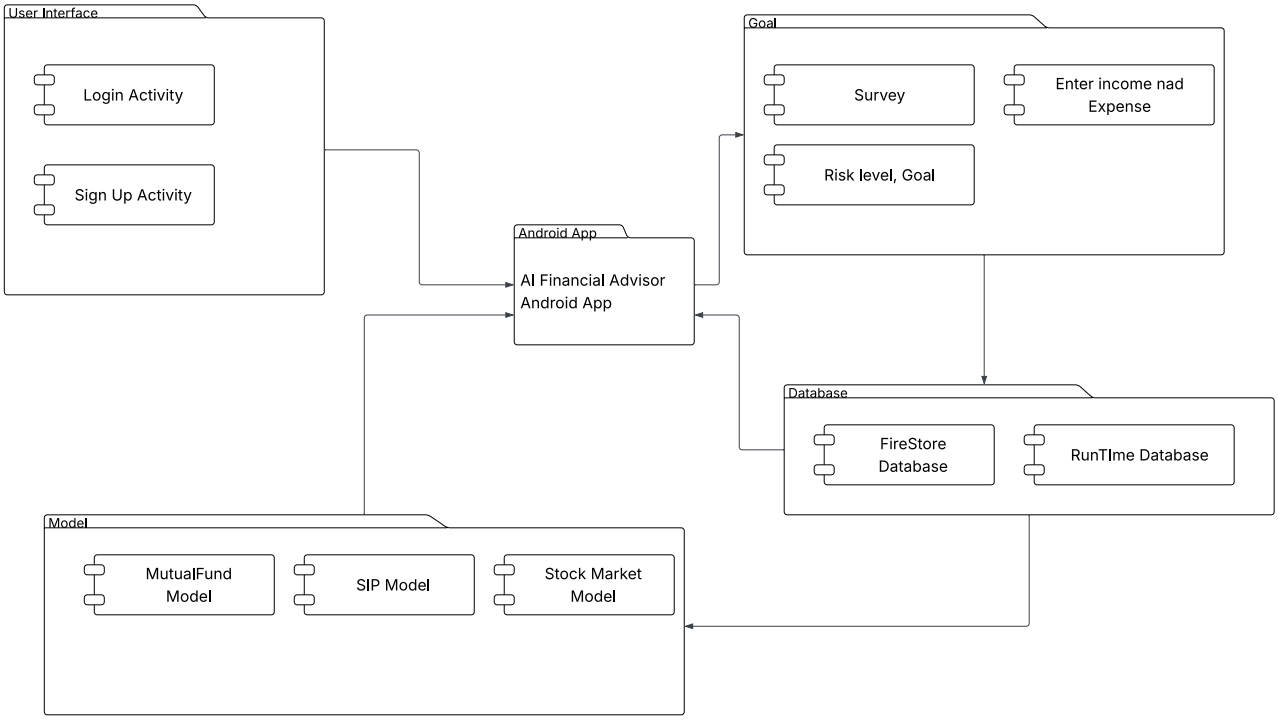


d. State-chart diagram



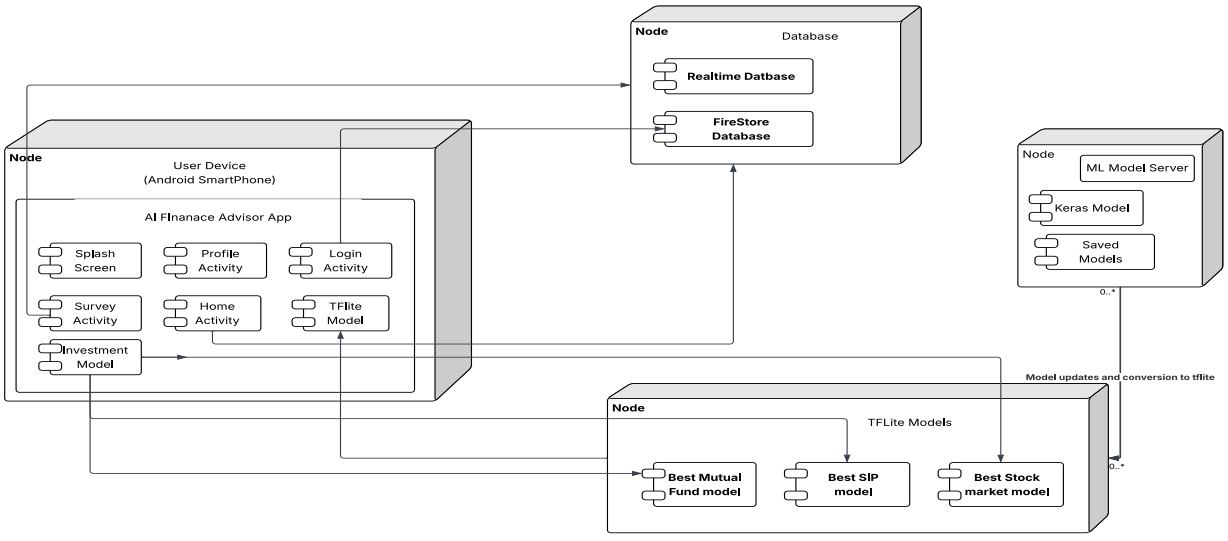
4.3. Implementation Design

a. Component Diagram



4.4. Environment Design

a. Deployment Diagram



5. References

•**Android Developer Documentation**, *Google Developers (2024).*
Provides detailed guidelines for building secure, efficient, and responsive Android applications using Java, XML, and Android Studio tools. Available at: https://developer.android.com

TensorFlow Lite Documentation, *TensorFlow Organization (2024).*
Offers comprehensive instructions for implementing, optimizing, and deploying machine learning models on mobile and embedded devices. Available at: https://www.tensorflow.org/lite

Firebase Developer Guide, *Google Firebase (2024).*
Explains the integration of Firebase Authentication, Realtime Database, and Cloud Storage for mobile applications, ensuring security and data synchronization. Available at: https://firebase.google.com/docs

IEEE Std 830-1998, *IEEE Recommended Practice for Software Requirements Specifications.*
Defines standardized guidelines for preparing, structuring, and validating Software Requirement Specifications, ensuring clarity and completeness in documentation.

Maarek, S. (2023). "Machine Learning with TensorFlow Lite on Android Devices."
Discusses practical methods of integrating TensorFlow Lite models into Android apps, focusing on optimization and inference efficiency for real-time predictions.

OpenAI Research (2024). "AI-Driven Financial Analytics and Planning Systems."
Presents recent advancements in artificial intelligence applications for financial analysis, investment forecasting, and intelligent decision support systems.

K. Jain and R. Patel (2023). "Design of Intelligent Financial Planning Systems using AI Models." ***International Journal of Computer Applications, Vol. 186(7).***
Explores AI-based financial management tools and the use of supervised learning techniques for predicting savings and investment strategies.

S. Mehta, A. Roy, and V. Sharma (2022). "Personal Finance Management through Predictive Analytics." ***IEEE Transactions on Computational Finance, Vol. 5(3).***
Discusses algorithms for predicting user spending patterns, improving budget planning accuracy, and providing data-driven financial advice.