SRS DOCUMENT

1. Introduction

1.1 Product Scope

- DolFin is a mobile financial wellbeing application designed to empower users to manage their finances effectively and achieve their financial goals. The application will provide a comprehensive suite of features to:
- Aggregate financial data from various accounts (bank accounts, investment accounts, credit cards
- Create and manage budgets with custom categories and spending rules. Management]
- Set and track savings goals with targeted completion dates and progress visualization.
- Track investment performance (optional), including holdings, price movements, and overall portfolio value. (Consider regulatory requirements for investment advice)
- Generate personalized financial insights and recommendations leveraging AI and Machine Learning (ML) algorithms, analyzing income, expenses, savings, and financial goals.
- Deliver educational content tailored to the user's financial situation and literacy level, encompassing various financial topics like budgeting, saving, investing, and debt management.
- Foster a supportive community for peer-to-peer learning and motivation, allowing users to share experiences, ask questions, and offer encouragement.

1.2 Product Value

DolFin aims to bridge the gap between traditional financial management tools and the need for personalized guidance. By leveraging AI and user data, DolFin will offer a dynamic and engaging platform that helps users:

- Gain a holistic view of their financial health through comprehensive financial data aggregation and analysis
- Develop healthy financial habits by setting and tracking budgets, savings goals, and monitoring spending patterns.
- Make informed financial decisions based on personalized insights, recommendations, and educational content.
- Achieve their short- and long-term financial goals through effective budgeting, saving, and potential investment tracking

The primary target audience for DolFin is:

- Millennials (born 1981-1996): This demographic is digitally native and comfortable managing finances through mobile applications. They are also at a crucial stage in their financial journeys, often grappling with budgeting, saving for major life events, and beginning to invest.
- Generation Z (born 1997-2012): As this generation enters adulthood, DolFin can provide them with the tools and knowledge to build a strong financial foundation

1.4 Intended Use

DolFin is intended for individual users to manage their personal finances. It can be used for various daily financial tasks, including:

- Monitoring account balances and transactions in a centralized location through secure financial data aggregation.
- Creating and tracking budgets for different spending categories, allowing for informed financial decision-making.
- Setting and monitoring progress towards savings goals for various purposes (e.g., emergency fund, down payment on a house).
- Analyzing spending patterns to identify areas for improvement and optimize budgeting strategies. Receiving personalized financial insights and recommendations based on financial data and user behavior, helping users make informed choices.
- Accessing educational materials on various financial topics to improve financial literacy and knowledge. Connecting with other users in a safe and supportive community for advice, motivation, and peer-to-peer learning.

1.5 General Description

DolFin will be a mobile application available for download on both iOS and Android devices. The application will prioritize a user-friendly interface with a focus on minimalism and intuitive navigation, ensuring a smooth user experience across platforms.

Technical Details:

Front-End Development:

- Programming Language: ReactJS (https://react.dev/) - A popular JavaScript library for building dynamic and reusable user interfaces. ReactJS offers advantages like virtual DOM for efficient rendering and component-based architecture for modularity.
- UI Framework: Material Design or similar (Consider platform-specific guidelines for iOS and Android). Material Design provides a set of UI components and a design language that promotes consistency and ease of use.
- Back-End Development:
- Programming Language: Python (Consider other options like Java) - Python is a versatile language known for its readability, extensive libraries, and strong developer community.
- Web Framework: Flask (Consider Django or other options) - Flask is a lightweight web framework for building web applications. It offers flexibility and ease of use for the chosen functionalities.
- Database: PostgreSQL - A powerful open-source object-relational database management system (DBMS) suitable for storing and managing financial data efficiently. PostgreSQL offers features like data security, scalability, and support for complex queries.

Security Considerations:

- User authentication and authorization will be implemented using secure protocols (e.g., OAuth, JWT).
- Financial data will be encrypted at rest and in transit using industry-standard algorithms.

- Regular security audits and penetration testing will be conducted to identify and address vulnerabilities.

2. Functional Requirements

Functional Requirements: Data Management for DolFin

2.1 Data Model and Database Design

DolFin's data management will be crucial for storing, managing, and utilizing user financial data effectively. Here's a breakdown of the proposed data model and database design:

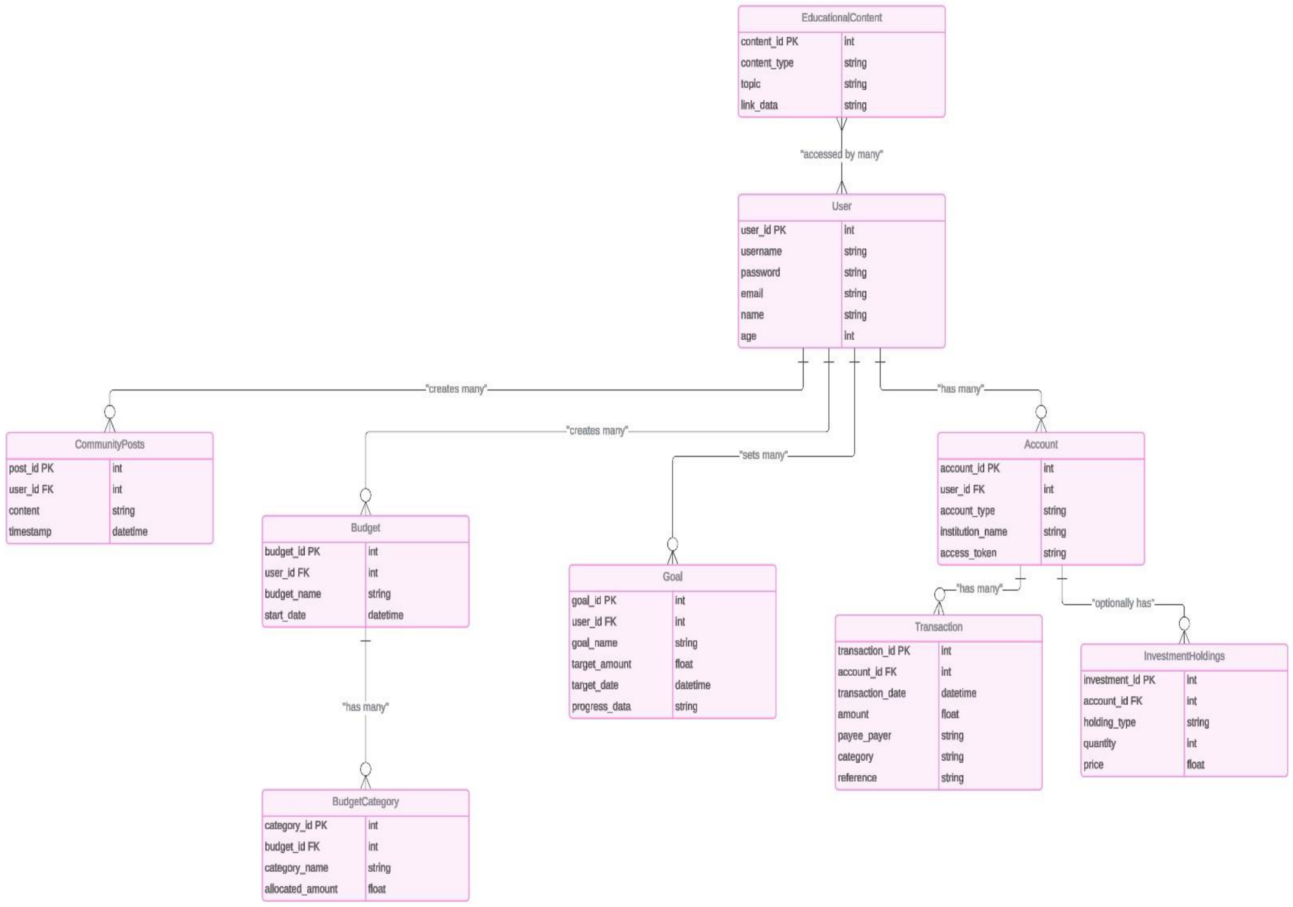
Database Engine: PostgreSQL is chosen for its robustness, scalability, and object-relational capabilities, allowing for efficient storage and retrieval of complex financial data structures.

Data Model Entities:

- The data model will consist of several key entities representing users, their financial accounts, transactions, budgets, goals, and other relevant information. Here's a breakdown of some core entities:
- Users: This entity will store user information such as username, password (hashed and salted for security), email address, and any optional profile details (e.g., name, age).
- Accounts: This entity will represent various financial accounts users connect to DolFin, including bank accounts, investment accounts, and credit cards. It will store details like account type, institution name, and a secure access token for data aggregation.
- Transactions: This entity will capture individual transactions within each connected account. It will include details like transaction date, amount, payee/payer, category (automatically assigned or user-defined), and account reference.
- Budgets: This entity will define user-created budgets with information like budget name, start date, and categories with allocated amounts.
- Goals: This entity will represent user-defined financial goals with details like goal name, target amount, target date, and progress tracking data.
- Additional Entities (Optional): Depending on functionalities, additional entities might be required, such as Investment Holdings (for tracking investment performance), Educational Content (for storing learning materials), and Community Posts (for user interactions).

Relationships between Entities:

- A User can have many Accounts.
- An Account will belong to one User.
- An Account can have many Transactions.
- A Transaction belongs to one Account.
- A User can create many Budgets.
- A Budget can have many Budget Categories.
- A User can set many Goals.
- A Goal can track progress using Transaction data.



Data Management Practices:

- Data Security: All user data, especially financial information, will be encrypted at rest and in transit using industry-standard algorithms. Secure protocols like OAuth or JWT will be implemented for user authentication and authorization. Regular security audits and penetration testing will be conducted to identify and address vulnerabilities.
- Data Accuracy: Mechanisms will be in place to ensure data accuracy during import from financial institutions via APIs. This may involve data validation, error handling, and reconciliation processes. Users can also manually review and edit imported data for additional control.
- Data Access Control: Users will have granular control over their data, allowing them to determine which accounts and information they share with the application. Role-based access control (RBAC) can be implemented within the system for different user types (e.g., standard user, administrator).
- Data Backups and Disaster Recovery: Regular backups of the database will be created and stored securely offsite to ensure data recovery in case of system failures or disasters. A disaster recovery plan will be outlined to minimize downtime and data loss.

Technical Considerations:

- Object-Relational Mapping (ORM): An ORM like SQLAlchemy can be used to simplify interaction with the PostgreSQL database from the application code. This allows developers to work with objects representing entities and their relationships, abstracting away the underlying SQL queries.
- Data Migration and Synchronization: Secure mechanisms will be implemented to handle initial data import from financial institutions and ongoing data synchronization to maintain an up-to-date view of user finances within DolFin.
- Data Analytics and Insights: The data stored in the database can be leveraged for generating personalized financial insights and recommendations for users. Machine learning algorithms can be utilized to analyze user behavior, spending patterns, and financial goals to provide actionable insights and improve the overall user experience.

2.2 Data Utilization

The data stored in the DolFin database will be used for various functionalities within the application:

- Account Aggregation and Transaction History: User financial data will be displayed in a centralized location, allowing users to easily monitor account balances, track transactions, and categorize spending.
- Budgeting and Goal Tracking: Data will be used to create budgets, allocate funds, and track progress towards financial goals. Transaction data can be automatically categorized and used to identify areas for improvement or adjust budget allocations.
- Personalized Financial Insights: Machine learning algorithms will analyze user data to generate personalized insights and recommendations. This may include identifying areas where users can save more, suggesting investment opportunities (if applicable), or providing tips for debt management.
- Educational Content Delivery: The user's financial data and behavior can be used to tailor educational content recommendations, ensuring the information aligns with their current financial situation and needs.

3. External Interface Requirements

This section will outline how DolFin interacts with external systems:

- User Interface (UI) Interfaces: The application will provide a user-friendly interface for users to interact with the functionalities.
- Financial Institution APIs: Secure APIs will be used to connect with various financial institutions, enabling data aggregation.

4. Non-Functional Requirements

This section will define performance, usability, and other critical quality attributes:

- Security: User data and financial information will be protected with robust security measures.

- Performance: The application will be responsive and deliver a smooth user experience across different devices.
- Scalability: The system should be able to accommodate a growing user base and increasing data volume.
- Reliability: The application should be highly available and minimize downtime.
- Usability: The interface should be intuitive and easy to navigate for users with varying levels of technical expertise. [Consider incorporating accessibility guidelines for users with disabilities]
- Maintainability: The codebase should be well-documented and maintainable for future enhancements.

5. Definitions and Acronyms

This section provides a list of key terms and acronyms used throughout the DolFin - Financial Wellbeing App SRS document.

- API (Application Programming Interface): A set of protocols and tools for building software applications that allows communication between different software components.
- Database: A collection of structured data organized electronically for efficient storage, retrieval, and manipulation.
- Encryption: The process of transforming data into a scrambled format that requires a secret key to decrypt and access the original information.
- Financial Institution (FI): A bank, credit union, investment firm, or other institution that provides financial services to consumers and businesses.
- Goal: A specific target amount of money a user wants to save by a certain date.
- Machine Learning (ML): A subfield of Artificial Intelligence (AI) that allows algorithms to learn and improve from data without explicit programming.
- Object-Relational Mapping (ORM): A programming technique that simplifies working with relational databases using objects that map to database tables and relationships.
- OAuth: An open-standard authorization framework that allows users to grant third-party applications access to their data without sharing their usernames and passwords.
- PostgreSQL: A powerful open-source object-relational database management system (DBMS).
- ReactJS: A popular JavaScript library for building dynamic and reusable user interfaces.
- RBAC (Role-Based Access Control): A security model that restricts user access to data and functionalities based on their assigned roles within a system.
- RESTful API: A type of API that adheres to the principles of REST (Representational State Transfer) architecture for standardized communication between web services.
- Secure Sockets Layer (SSL)/Transport Layer Security (TLS): Encryption protocols that establish secure connections between a server and a client, ensuring the privacy and integrity of data transmission.
- SQL (Structured Query Language): A standardized language for interacting with relational databases to perform tasks like data retrieval, manipulation, and definition.
- Transaction: A single financial event involving the transfer of money between accounts.
- UI (User Interface): The graphical elements of a software application that users interact with to provide input and receive information.