Introduction

1.1 Purpose

The purpose of this document is to define the software requirements for a money transfer application. The application should allow users to send and receive money securely, view their transaction history, set up recurring payments, and manage their account information.

1.2 Scope

The money transfer application should be developed for use on mobile devices and desktop computers. It should be designed for use by individuals and small businesses.

1.3 Definitions, Acronyms, and Abbreviations

SRS: Software Requirements Specification

API: Application Programming Interface

Functional Requirements

2.1 User Management

The application should allow users to create an account and log in securely. Users should be able to reset their password if they forget it.

2.2 Transfer Money

Users should be able to send money to other users of the application. Users should be able to specify the recipient's account information and the amount to be transferred.

2.3 Receive Money

Users should be able to receive money from other users of the application. Users should be able to specify their own account information to receive funds.

2.4 Transaction History

The application should allow users to view their transaction history. The transaction history should include the date, time, amount, and details of each transaction.

2.5 Recurring Payments

The application should allow users to set up recurring payments. Users should be able to specify the amount, frequency, and duration of the recurring payment.

2.6 Account Management

Users should be able to manage their account information, including their name, contact information, and payment information.

2.7 Security

The application should implement security features to protect user data and transactions. Security features should include encryption, authentication, and error handling.

Non-Functional Requirements

3.1 Performance

The application should be responsive and performant. It should have fast load times and quick transaction processing.

3.2 Usability

The application should be easy to use and intuitive. Users should be able to complete tasks quickly and efficiently.

3.3 Reliability

The application should be reliable and should not experience downtime or crashes.

3.4 Scalability

The application should be scalable and able to handle increasing numbers of users and transactions.

User Interface

4.1 Design

The application should have a modern and user-friendly design. The design should be consistent across all platforms.

4.2 Navigation

The application should have intuitive navigation. Users should be able to easily find the features and functions they need.

4.3 User Input and Feedback

The application should provide clear and helpful feedback to users when they complete actions or encounter errors.

System Architecture

5.1 Hardware

The application should be designed to run on mobile devices and desktop computers.

5.2 Software

The application should be developed using modern software development practices and tools.

5.3 API

The application should expose an API that allows third-party developers to build integrations with the application.

Data Management

6.1 Database Schema

The application should use a database to store user data and transaction information. The database schema should be designed for scalability and performance.

6.2 Data Backups

The application should implement a data backup strategy to prevent data loss.

6.3 Data Retention Policies

The application should have data retention policies that comply with legal and regulatory requirements.