---

1. INTRODUCTION

1.1 Purpose

This document outlines the design and functionality of the SecureBank Online Banking Portal, a web-based platform that allows users to manage their bank accounts, perform transactions, and access financial services securely.

1.2 Scope

The SecureBank portal will provide a user-friendly interface for personal banking customers to:

- View account balances and transaction history.
- Transfer funds between accounts or to external recipients.
- Pay bills and manage recurring payments.
- Update personal profile information.
- Access customer support.

1.3 Target Audience

- Bank customers (individuals with savings, checking, or loan accounts).
- Bank administrators (for backend monitoring).
- Development and QA teams.

---

2. SYSTEM OVERVIEW

2.1 Objectives

- Provide a secure, reliable, and intuitive online banking experience.
- Reduce the need for in-person banking visits.
- Ensure compliance with financial regulations (e.g., GDPR, PCI-DSS).

2.2 Key Features

- Multi-factor authentication (MFA) for secure login.
- Real-time account updates.
- Transaction history with filters and search.
- Responsive design for desktop and mobile use.

---

3. SCREENS AND USER INTERFACE

3.1 Login Screen

- **Description:** Entry point for users to access their accounts.

- **Elements:**
    - Username/Email field.
    - Password field.
    - “Forgot Password?” link.
    - “Login” button.
    - MFA verification (OTP sent to phone/email).
- **Wireframe Notes:** Clean design with bank logo at the top, centered login form.

3.2 Dashboard

- **Description:** Overview of user’s financial status post-login.
- **Elements:**
    - Account summary (list of accounts with balances).
    - Recent transactions (last 5).
    - Quick actions (Transfer Funds, Pay Bills).
- **Wireframe Notes:** Card-based layout for accounts, horizontal scroll for transactions.

3.3 Account Details

- **Description:** Detailed view of a selected account.
- **Elements:**
    - Account type (e.g., Savings, Checking).
    - Current balance.
    - Transaction history table (date, description, amount).
    - Filter options (by date, type).
- **Wireframe Notes:** Tabular layout with pagination for history.

3.4 Transfer Funds

- **Description:** Interface for transferring money.
- **Elements:**
    - From Account dropdown.
    - To Account/Recipient field (internal or external).
    - Amount input.
    - “Confirm Transfer” button.
- **Wireframe Notes:** Stepper design for multi-step confirmation.

3.5 Bill Payments

- **Description:** Manage and pay bills.
- **Elements:**
  - Add payee form (name, account number).
  - List of saved payees.
  - Payment amount and date selector.
  - “Pay Now” button.
- **Wireframe Notes:** Accordion-style list for payees.

3.6 Profile Settings

- **Description:** Manage user profile and security settings.
- **Elements:**
  - Personal info (name, email, phone).
  - Change password option.
  - Enable/disable MFA.
  - Notification preferences.
- **Wireframe Notes:** Form layout with save/cancel buttons.

---

4. USE CASES

4.1 User Login

- **Actor:** Bank Customer
- **Precondition:** User has a registered account.
- **Steps:**
  1. User enters username and password.
  2. System validates credentials.
  3. User receives OTP for MFA.
  4. User enters OTP and logs in.
- **Postcondition:** User is redirected to Dashboard.

4.2 View Account Balance

- **Actor:** Bank Customer
- **Precondition:** User is logged in.
- **Steps:**
  1. User selects an account from the Dashboard.

2. System displays Account Details screen.

- **Postcondition:** User views balance and transaction history.

4.3 Transfer Money

- **Actor:** Bank Customer
- **Precondition:** User has sufficient funds.
- **Steps:**
    1. User navigates to Transfer Funds screen.
    2. User selects source and destination accounts.
    3. User enters amount and confirms.
    4. System processes transfer and notifies user.
- **Postcondition:** Funds are transferred, balance updated.

4.4 Pay Bills

- **Actor:** Bank Customer
- **Precondition:** User has added a payee.
- **Steps:**
    1. User navigates to Bill Payments screen.
    2. User selects payee and enters amount.
    3. User confirms payment.
    4. System processes payment and sends confirmation.
- **Postcondition:** Bill is paid, balance updated.

---

5. FUNCTIONAL REQUIREMENTS

- The system must support user authentication via MFA.
- The system must allow transfers between accounts with real-time updates.
- The system must store transaction history for at least 12 months.
- The system must integrate with third-party payment gateways for bill payments.

---

6. NON-FUNCTIONAL REQUIREMENTS

- **Performance:** Page load time < 2 seconds.
- **Security:** End-to-end encryption (TLS 1.3), compliance with PCI-DSS.
- **Scalability:** Support up to 1 million concurrent users.

- **Availability:** 99.9% uptime.

---

7. ASSUMPTIONS AND CONSTRAINTS

- **Assumptions:** Users have internet access and a registered bank account.
- **Constraints:** Limited to English language in Version 1.0; no support for cryptocurrency transactions.

---

8. APPENDIX

- Mock wireframes (to be added in design phase).
- API endpoints (to be defined by development team).
