# Software Requirements Specification

Modern Bank Portal

Version 2.0

Date: May 18, 2025

---

Table of Contents

1. Introduction

1.1 Purpose

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

1.4 Project Scope

1.5 Definitions, Acronyms, and Abbreviations

2. Overall Description

2.1 Product Perspective

2.2 Product Features

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 Assumptions and Dependencies

3. System Features (Functional Requirements)

3.1 User Authentication

3.1.1 User Registration (Manager)

3.1.2 User Registration (Customer)

3.1.3 User Login

3.1.4 User Logout

3.2 Customer Functions

3.2.1 View Dashboard

3.2.2 Deposit Funds

3.2.3 Withdraw Funds

3.2.4 Transfer Funds

3.2.5 View Transaction Statement

3.2.5.1 Filter and Search Transactions

3.3 Bank Manager Functions

3.3.1 View Dashboard

3.3.2 View System Overview (Statistics)

3.3.3 View Registered Customers

3.3.3.1 Search and Filter Customers

3.3.4 View Specific Customer's Transaction History

3.3.5 View All Transactions

3.4 Common Features

3.4.1 Navigation (Header, Footer, Mobile Menu)

3.4.2 Data Persistence

3.4.3 Home Page

4. Non-Functional Requirements

4.1 Performance Requirements

4.2 Security Requirements

4.3 Usability Requirements

4.4 Reliability Requirements

4.5 Maintainability Requirements

5. Interface Requirements

5.1 User Interface (UI)

5.2 Software Interfaces

5.3 Hardware Interfaces (Not Applicable for current scope)

5.4 Communications Interfaces (Not Applicable for current scope)

6. Other Requirements

6.1 Data Management

---

1. Introduction

**# 1.1 Purpose**

This Software Requirements Specification (SRS) document describes the functional and non-functional requirements for the "Modern Bank Portal" web application, Version 2.0. The portal aims to provide basic banking functionalities for customers and comprehensive administrative oversight for bank managers. This document is intended for project stakeholders, designers, developers, and testers.

**# 1.2 Document Conventions**

This document uses standard formatting. "Shall" is used to indicate mandatory requirements.

**# 1.3 Intended Audience and Reading Suggestions**

This document is intended for:

Project Managers : To understand the scope and requirements.

Designers : To guide UI/UX design.

Developers : To implement the specified functionalities.

Testers : To create test cases based on the requirements.

Stakeholders : To review and approve the project's scope and functionalities.

It is recommended to read the document sequentially, though specific sections can be referred to as needed.

**# 1.4 Project Scope**

The "Modern Bank Portal" is a web-based application that allows:

Bank Managers to register, log in, view system statistics, view a list of all registered customers (with search/filter capabilities), view a specific customer's full transaction history, and view a list of all transactions within the system.

Customers can register, log in, deposit funds, withdraw funds, transfer funds to other customers, and view their transaction statements (with filtering and search capabilities).

The application features a global header with navigation, a footer, and a responsive home page. It uses browser `localStorage` for data persistence in its current iteration and is structured as a single-page application (SPA) built with HTML, CSS, and JavaScript.

Out of Scope for Version 2.0:

\* Real-time database integration or backend server.

\* Advanced security features beyond basic password management (e.g., two-factor authentication, encryption of data at rest/in transit beyond HTTPS).

\* Loan applications or management.

\* Profile editing for users (e.g., changing password, name).

\* Administrative user management by super-admins.

\* Automated password recovery mechanisms.

\* Integration with external financial systems or APIs.

\* "Flag Account" feature for managers.

**# 1.5 Definitions, Acronyms, and Abbreviations**

SRS: Software Requirements Specification

UI: User Interface

UX: User Experience

SPA: Single-Page Application

HTML: HyperText Markup Language

CSS: Cascading Style Sheets

JS: JavaScript

Portal: Refers to the Modern Bank Portal web application.

User: Can be a Bank Manager or a Customer.

Account ID: Unique identifier for a customer's bank account.

Transaction ID: Unique identifier for a financial transaction.

CDN: Content Delivery Network

DOM: Document Object Model

---

**2. Overall Description**

# 2.1 Product Perspective

The Modern Bank Portal is a standalone, client-side web application designed to demonstrate and simulate basic banking operations. It operates entirely within the user's web browser, using `localStorage` for data storage. In its current version, it does not interact with any external systems or backend services.

**# 2.2 Product Features**

The major features of the Modern Bank Portal are:

\* A modern and attractive home page with clear calls to action.

\* Global header with navigation links and user authentication status.

\* Global footer with informational links and copyright.

\* Responsive mobile menu for navigation on smaller screens.

\* Separate registration processes for Bank Managers and Customers.

\* Secure login for registered users.

Customer Dashboard :

\* View account balance.

\* Deposit funds.

\* Withdraw funds.

\* Transfer funds to other customers.

\* View transaction statements with filtering (by date range, type) and search (by description).

Bank Manager Dashboard :

\* View system overview with key statistics (total customers, total funds, total transactions, transactions today).

\* View a list of all registered customers with search and filter capabilities.

\* View a specific customer's full transaction history.

\* View a comprehensive list of all transactions made by all customers.

\* Client-side data validation for forms.

\* Responsive design for usability across different screen sizes.

\* Session persistence for logged-in users (until logout or browser storage is cleared).

**# 2.3 User Classes and Characteristics**

**2.3.1 Bank Manager**

**Characteristics:** Assumed to be internal bank staff with administrative privileges. Familiar with basic computer and web usage.

**Responsibilities:** Overseeing customer accounts, system-wide transactions, and monitoring overall system health via statistics.

**Permissions :**

\* Register as a Manager.

\* Login as a Manager.

\* View system statistics (Overview tab).

\* View a list of all registered customers.

\* Search/filter the customer list by name, email, or Account ID.

\* Select a customer from the list to view their detailed transaction history.

\* View a comprehensive list of all transactions made by all customers.

\* Logout.

**2.3.2 Customer**

**Characteristics:** General public users who wish to perform basic banking operations. Assumed to have basic web literacy.

**Responsibilities:** Managing their personal bank account.

**Permissions :**

\* Register as a Customer (including an initial deposit).

\* Login as a Customer.

\* View their account balance.

\* Deposit funds into their account.

\* Withdraw funds from their account (up to the available balance).

\* Transfer funds to another customer's account.

\* View their transaction history (statement).

\* Filter their transaction statement by date range and transaction type.

\* Search their transaction statement by description.

\* Logout.

**# 2.4 Operating Environment**

The Modern Bank Portal is a web application and shall operate in the following environments:

Client-Side: Modern web browsers that support HTML5, CSS3, JavaScript (ES6+), and Web

Storage (`localStorage`). Examples include:

\* Google Chrome (latest versions)

\* Mozilla Firefox (latest versions)

\* Microsoft Edge (latest versions)

\* Apple Safari (latest versions)

Network: An internet connection is required to load the initial application files (HTML, CSS from CDN, JS). Once loaded, core operations (data manipulation) are client-side.

**# 2.5 Design and Implementation Constraints**

Technology Stack: The application shall be built using HTML, CSS (including Tailwind CSS via CDN and custom styles in `style.css`), and vanilla JavaScript (in `[script.js](http://script.js)`).

Data Storage: All user and transaction data shall be stored in the browser's `localStorage`. This implies data is local to the browser and not shared across devices or sessions on different browsers.

Security: Due to `localStorage` usage, the application is not suitable for handling real sensitive financial data. Passwords are stored as plain text in `localStorage`.

Single Page Application (SPA): The application shall function as an SPA, with views dynamically rendered on the client-side.

External Libraries: Tailwind CSS (CDN) and Google Fonts (CDN) are the primary external libraries. No other major JavaScript frameworks are used.

**# 2.6 Assumptions and Dependencies**

\* Users have JavaScript enabled in their web browsers.

\* Users have a modern web browser compatible with the specified technologies.

\* The Tailwind CSS CDN and Google Fonts CDN are accessible.

\* Users understand that data stored is local to their browser and can be lost if browser data is cleared.

---

**3. System Features (Functional Requirements)**

**# 3.1 User Authentication**

(Requirements FR-AUTH-001 to FR-AUTH-024 remain largely the same as v1.0, covering Manager Registration, Customer Registration, Login, and Logout. Key aspects include form fields, validation, unique email, password length, initial deposit for customers, credential verification, session establishment, and redirection.)

**# 3.2 Customer Functions**

**3.2.1 View Dashboard**

(Requirements FR-CUST-001 to FR-CUST-004 remain largely the same, covering display of name, account ID, balance, and navigation tabs.)

FR-CUST-004.1: The customer dashboard navigation shall include tabs for Deposit, Withdraw, Transfer Funds, and View Statement.

**3.2.2 Deposit Funds**

(Requirements FR-CUST-005 to FR-CUST-008 remain largely the same, covering deposit form, amount validation, balance update, transaction recording, and success message.)

**3.2.3 Withdraw Funds**

(Requirements FR-CUST-009 to FR-CUST-014 remain largely the same, covering withdrawal form, amount validation, balance check, transaction recording, and success/error messages.)

**3.2.4 Transfer Funds**

FR-CUST-015: The system shall provide a form for transferring funds to another customer.

FR-CUST-016: The transfer form shall require the Recipient's Account ID and the Transfer Amount.

FR-CUST-017: The system shall validate that the Recipient's Account ID is not empty.

FR-CUST-018: The system shall validate that the Transfer Amount is a positive numeric value.

FR-CUST-019: The system shall validate that the Transfer Amount does not exceed the sender's current balance.

FR-CUST-020: The system shall prevent a customer from transferring funds to their own Account ID.

FR-CUST-021: The system shall verify that the Recipient's Account ID exists and belongs to a registered customer.

**FR-CUST-022: Upon successful transfer:**

\* The system shall deduct the amount from the sender's balance.

\* The system shall add the amount to the recipient's balance.

\* The system shall record a 'transfer-out' transaction for the sender, including recipient details in the description.

\* The system shall record a 'transfer-in' transaction for the recipient, including sender details in the description.

\* The system shall update the displayed balance on the sender's dashboard.

\* The system shall display a success message to the sender.

FR-CUST-023: If any validation fails (e.g., insufficient funds, invalid recipient), the system shall display an appropriate error message.

FR-CUST-024: The system may optionally display the recipient's name upon valid Account ID entry before confirming the transfer.

**3.2.5 View Transaction Statement**

FR-CUST-025: The system shall allow the customer to view their transaction statement.

FR-CUST-026: The transaction statement shall display a list of all transactions for the customer's account.

FR-CUST-027: Each transaction entry shall display: Date, Description, Amount (with indication of type), and Balance After Transaction.

FR-CUST-028: Transactions shall be listed in reverse chronological order (most recent first) by default.

FR-CUST-029: If there are no transactions, or no transactions match applied filters, the system shall display an appropriate message.

**# 3.2.5.1 Filter and Search Transactions**

FR-CUST-030: The system shall provide controls to filter the transaction statement by a date range (Start Date, End Date).

FR-CUST-031: The system shall provide a dropdown to filter the transaction statement by transaction type (All, Deposit, Withdrawal, Transfer In, Transfer Out).

FR-CUST-032: The system shall provide a text input to search the transaction statement by keywords in the transaction description (case-insensitive).

FR-CUST-033: The system shall provide an "Apply Filters" button to re-render the statement based on selected filter criteria.

FR-CUST-034: The system shall provide a "Reset Filters" button to clear all filter criteria and display all transactions.

**# 3.3 Bank Manager Functions**

**3.3.1 View Dashboard**

FR-MAN-001: Upon successful login, the manager shall be presented with their dashboard.

FR-MAN-002: The manager dashboard navigation shall include tabs for "Overview", "View Customers", and "All Transactions".

FR-MAN-002.1: The "Overview" tab shall be the default active tab upon manager login.

**3.3.2 View System Overview (Statistics)**

FR-MAN-003: The "Overview" tab shall display key system statistics.

FR-MAN-004: The system shall calculate and display the "Total Registered Customers".

FR-MAN-005: The system shall calculate and display the "Total Funds Held" across all customer accounts.

FR-MAN-006: The system shall calculate and display the "Total Number of Transactions Processed".

FR-MAN-007: The system shall calculate and display the "Number of Transactions Processed Today".

**3.3.3 View Registered Customers**

FR-MAN-008: The "View Customers" tab shall allow the manager to view a list of all registered customers.

FR-MAN-009: The customer list shall display: Account ID, Name, Email, Current Balance, and an action button (e.g., "View Txns") for each customer.

FR-MAN-010: If there are no registered customers or no customers match the search criteria, the system shall display an appropriate message.

**# 3.3.3.1 Search and Filter Customers**

FR-MAN-011: The system shall provide a text input field for the manager to search customers by Name, Email, or Account ID (case-insensitive).

FR-MAN-012: The system shall provide a "Search" button to apply the search term and filter the customer list.

FR-MAN-013: The system shall provide a "Reset" button to clear the search term and display all customers.

**3.3.4 View Specific Customer's Transaction History**

FR-MAN-014: The manager shall be able to initiate viewing a specific customer's transaction history from the "View Customers" list (e.g., by clicking a "View Txns" button).

FR-MAN-015: Upon selection, the system shall display a dedicated view showing the selected customer's Name and Account ID.

FR-MAN-016: This view shall list all transactions for the selected customer, similar to the customer's statement view (Date, Description, Amount, Balance After), sorted in reverse chronological order.

FR-MAN-017: If the selected customer has no transactions, an appropriate message shall be displayed.

FR-MAN-018: A "Back to Customer List" button shall be provided to return the manager to the main "View Customers" tab, preserving any previous search/filter state on that list if feasible.

**3.3.5 View All Transactions**

FR-MAN-019: The "All Transactions" tab shall allow the manager to view a list of all transactions across all customer accounts.

FR-MAN-020: Each transaction entry shall display: Date, Customer Name, Account ID, Transaction Type (Deposit, Withdrawal, Transfer In, Transfer Out), Description, and Amount.

FR-MAN-021: Transactions shall be listed in reverse chronological order.

FR-MAN-022: If there are no transactions in the system, the system shall display an appropriate message.

**# 3.4 Common Features**

**3.4.1 Navigation (Header, Footer, Mobile Menu)**

FR-COM-001: The system shall provide a global header persistent across all views when a user is not on the initial home page sections (like registration/login forms).

FR-COM-001.1: The global header shall display the portal logo ("ModernBank"), which links back to an appropriate home/dashboard view.

FR-COM-001.2: The global header shall display navigation links (e.g., "Features", "Security", "Contact Us") for public-facing pages.

FR-COM-001.3: The global header shall display the logged-in user's welcome message (e.g., "Welcome, [User's First Name]") and a "Logout" button when a user is authenticated.

FR-COM-001.4: The system shall provide a responsive mobile menu (hamburger icon) for accessing navigation links on smaller screens.

FR-COM-002: The system shall provide a global footer persistent across all views.

FR-COM-002.1: The footer shall contain informational links (e.g., About Us, Contact) and copyright information, including the current year.

FR-COM-003: "Back to Home" buttons shall be available on registration and login pages, leading to the main public home page.

**3.4.2 Data Persistence**

FR-COM-004: All user data (registrations, balances) and transaction data shall be persisted in the browser's `localStorage`.

FR-COM-005: The logged-in state of a user (`currentUser`) shall be persisted in `localStorage` to maintain the session across page refreshes.

**3.4.3 Home Page**

FR-COM-006: The system shall display a modern and attractive home page as the initial entry point.

FR-COM-007: The home page shall feature a hero section with a title, subtitle, and illustrative imagery.

FR-COM-008: The home page shall provide clear primary calls to action for customer registration ("Open Your Account") and customer login ("Access Account").

FR-COM-009: The home page shall provide secondary options for manager registration and login.

---

**4. Non-Functional Requirements**

**# 4.1 Performance Requirements**

NFR-PERF-001: UI transitions, view rendering, and tab switching shall be smooth and occur within 1-2 seconds on typical client hardware.

NFR-PERF-002: Form submissions and data updates (e.g., balance changes, registrations) shall be processed and reflected in the UI within 2-3 seconds.

NFR-PERF-003: Loading and filtering of customer lists and transaction lists (up to several hundred entries) shall be completed within 3-4 seconds.

**# 4.2 Security Requirements**

NFR-SEC-001: User inputs for registration and login (especially passwords) shall be handled over HTTPS if deployed to a live server.

NFR-SEC-002: The application shall implement client-side input validation for all forms to prevent common data entry errors and guide users.

NFR-SEC-003 : (Limitation) User passwords are stored as plain text in `localStorage`. This is a known security risk for this demonstration version and should be addressed with hashing and a secure backend in future production versions.

NFR-SEC-004 : (Limitation) All data in `localStorage` is accessible to anyone with physical or remote access to the user's browser and its developer tools.

**# 4.3 Usability Requirements**

NFR-USAB-001: The user interface shall be clean, modern, professional, and intuitive, following common web design conventions.

NFR-USAB-002: All interactive elements (buttons, input fields, tabs, table rows where applicable) shall provide clear visual feedback (e.g., hover states, focus states).

NFR-USAB-003: Error messages shall be clear, concise, user-friendly, and displayed in proximity to the error source or a designated message area.

NFR-USAB-004: Success messages shall confirm user actions.

NFR-USAB-005: The application shall be responsive and provide an optimal viewing and interaction experience on common desktop, tablet, and mobile screen sizes.

NFR-USAB-006: Forms shall be easy to understand and complete. Required fields shall be indicated.

NFR-USAB-007: Navigation shall be consistent and predictable.

NFR-USAB-008: Filter and search functionalities shall be easy to use and understand.

**# 4.4 Reliability Requirements**

NFR-REL-001: The application should function correctly under normal usage conditions within supported browsers.

NFR-REL-002: Data stored in `localStorage` shall persist between browser sessions, provided the user does not clear their browser data.

NFR-REL-003: The application should gracefully handle invalid user inputs by displaying appropriate error messages rather than crashing or producing unexpected behavior.

**# 4.5 Maintainability Requirements**

NFR-MAIN-001: The JavaScript code (`script.js`) shall be well-commented, organized into logical functions and sections for better understanding and modification.

NFR-MAIN-002: The HTML structure (`index.html`) shall be semantic and well-organized.

NFR-MAIN-003: The CSS (`style.css`) shall be organized, with clear comments for custom styles, complementing the Tailwind CSS utility classes.

---

**5. Interface Requirements**

**# 5.1 User Interface (UI)**

NFR-UI-001: The UI shall be developed using HTML and styled with Tailwind CSS (via CDN) and custom CSS in `style.css`.

NFR-UI-002: The UI shall adhere to the visual design specified (modern, professional look and feel, defined color palette, typography with 'Inter' font).

NFR-UI-003: The UI shall be responsive, adapting to different viewport sizes.

NFR-UI-004: Key UI elements include:

\* Global Header: Logo, navigation, user status/logout, mobile menu.

\* Global Footer: Informational links, copyright.

\* Home Page: Hero section with text, image, and distinct call-to-action groups.

\* Forms: For registration, login, deposit, withdrawal, and transfer.

\* Dashboards: For Manager and Customer, with tabbed navigation.

\* Manager Overview: Stat cards.

\* Manager Customer List: Search/filter controls, customer table with action buttons.

\* Manager Specific Customer Transactions: Customer details, transaction table, and back button.

\* Customer Statement: Filter/search controls, transaction table.

\* Tables: For displaying lists of customers and transactions, with clear headers and zebra-striping.

\* Buttons: Clearly styled for primary, secondary, outline, and specific actions (e.g., `btn-view-transactions`).

\* Input Fields: Consistently styled with focus and error states, including smaller variants for filter bars.

\* Message Areas: For displaying success and error feedback.

**# 5.2 Software Interfaces**

NFR-SINT-001: The application will interact with the browser's Web Storage API (`localStorage`) for data persistence.

NFR-SINT-002: The application will use CDNs for Tailwind CSS and Google Fonts. No other external software interfaces are required for this version.

**# 5.3 Hardware Interfaces**

Not applicable for the current scope of this client-side web application.

**# 5.4 Communications Interfaces**

Not applicable for direct communications interfaces beyond standard HTTP/HTTPS for loading resources.

---

**6. Other Requirements**

**# 6.1 Data Management**

OR-DATA-001: User data (managers and customers) shall be stored as an array of objects in `localStorage`. Each user object will contain relevant attributes (ID, name, email, password, role, balance for customers, and accountId for customers).

OR-DATA-002: Transaction data shall be stored as an array of objects in `localStorage`. Each transaction object will contain relevant attributes (transactionId, userId, accountId, type, description, amount, date, balanceAfter).

OR-DATA-003: Unique IDs (Account ID, Transaction ID, User ID) shall be generated client-side using a combination of a timestamp and random numbers.

OR-DATA-004: Data integrity for calculations (e.g., balance updates during deposits, withdrawals, transfers) must be maintained.

OR-DATA-005 : (Limitation) There is no backup or recovery mechanism for data stored in `localStorage` beyond what the browser provides. Data loss can occur if browser data is cleared by the user.

OR-DATA-006: Dates for transactions shall be stored in ISO format and displayed in a user-friendly localized format.

---

End of Document