# Design Document

## Section 1 - Project Description

### 1.1 Project

Project Name: Banking System

### 1.2 Description

Brief Overall Description:  
The main goal of this project is to create a simple banking system that allows users to create accounts, make deposits and withdrawals, and view transaction histories through an easy-to-use interface.

### 1.3 Revision History

|  |  |  |
| --- | --- | --- |
| Date | Comment | Author |

## Section 2 - Overview

### 2.1 Purpose

Focus:  
To develop a user-friendly banking system module for basic banking transactions. The intended audience includes developers, stakeholders, and users.

### 2.2 Scope

Scope of the Module:  
The module includes user account creation, secure login, deposit and withdrawal functionalities, transaction history viewing, and balance inquiries. Excluded features are integration with real banking APIs, advanced financial services, and multi-currency support.

### 2.3 Requirements

#### 2.3.1 Functional Requirements

- Requirement 1: The system shall allow users to create new accounts with personal details and an initial deposit.  
- Requirement 2: The system shall provide secure login and registration with password encryption.  
- Requirement 3: The system shall allow users to deposit money into their accounts.  
- Requirement 4: The system shall allow users to withdraw money from their accounts.  
- Requirement 5: The system shall allow users to view their transaction history.  
- Requirement 6: The system shall allow users to check their current account balance.

#### 2.3.2 Non-Functional Requirements

- Performance: The system should be able to efficiently manage one user performing actions (like logging in, making deposits, or checking balances) at a time.  
- Having simple backup procedures to restore the system quickly in case of failure.

#### 2.3.3 Technical Requirements

- Programming Languages: JavaScript (frontend), Java (backend)  
- Frameworks: React.js (frontend), Spring Boot (backend)  
- Tools: Git (version control), Maven (build automation)

#### 2.3.4 Security Requirements

- Authentication: The system shall use password encryption for all user logins.  
- Data Encryption: All sensitive data shall be encrypted both in transit and at rest.

#### 2.3.5 Estimates

|  |  |  |
| --- | --- | --- |
| # | Description | Hrs. Est. |

## Section 3 - System Architecture

### 3.1 Overview

High-Level Overview:  
The system architecture consists of a frontend developed with React.js and a backend powered by Spring Boot. The backend manages data storage, user authentication, and transaction processing.

### 3.2 Architectural Diagrams

- Component Diagram  
- Sequence Diagram  
- Data Flow Diagram (DFD)  
- Deployment Diagram  
- Class Diagram  
- Use Case Diagram

## Section 4 - Data Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
| Table | Field | Notes | Type |

## Section 5 – Data Design

### 5.1 Persistent/Static Data

Logical Data Model:  
- User  
 - Attributes: UserID, Username, Password, Email, Role  
 - Relationships: One-to-Many with Accounts  
- Account  
 - Attributes: AccountID , UserID, Balance  
 - Relationships: Many-to-One with Users

## Section 6 - User Interface Design

### 6.1 User Interface Design Overview

High-level requirements and mockups for user account creation, login, deposit/withdrawal forms, and transaction history views.

### 6.2 User Interface Navigation Flow

Diagram illustrating the flow from one screen to the next.