|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Software Requirement Specifications**  SQA BankPro  Version: [1.0]   |  |  | | --- | --- | | Project Code | N/A | | Supervisor | Mahrukh Khan | | Co Supervisor | N/A | |  |  | | Project Team | **Arsalan (22K-4614)** - Project Lead  **Sadaan Qureshi (22K-4196)** – Developer  **Qasim Naveed (22K-4380)** - UI/UX Designer | | Submission Date | 07/05/2025 | |

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Name of Person** | **Date** | **Description of change** |
| 1.0 | Arsalan | |  | | --- | | 22/02/2025 |  |  | | --- | |  | | Created the initial draft of the document. Set up the project title, team details, and document structure. Included project overview and general details. |
| 1.0 | Arsalan | 13/03/2025 | Added **Overall System Description** section, covering project background, scope, and objectives. Prepared initial assumptions and dependencies. |
| 1.0 | Sadaan Qureshi | 02/04/2025 | |  | | --- | |  |  |  |  |  | | --- | --- | --- | | |  | | --- | |  |  |  | | --- | | Added **Assumptions & Dependencies** section detailing the system's operational and technical assumptions, including mock data usage and simulated banking transactions due to the project's academic nature | | |
| 1.0 | Sadaan Qureshi | 04/04/2025 | |  | | --- | |  |  |  | | --- | | Included updates in **Functional Requirements**: Defined key functionalities such as account management, loan processing, and customer service features. | |
| 1.0 | Qasim Naveed | 02/05/2025 | Updated the **Non-functional Requirements** section, emphasizing security measures like encryption, user authentication, and system performance expectations (response time for transactions and concurrency support). |
| 1.0 | Qasim Naveed | 04/05/2025 | |  | | --- | |  |  |  | | --- | | Finalized the **Document History** section with all updates provided by each member, detailing specific contributions and updating the dates of change. | |
|  |  |  |  |

Distribution List

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | |
| Mahrukh Khan | | Supervisor |
| N/A | | Co- Supervisor |
|  | |  |

Document Sign-Off

|  |  |  |
| --- | --- | --- |
| **Version** | **Sign-off Authority** | **Sign-off Date** |
| 1.0 | Mahrukh Khan |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents**

[1. Introduction 7](#_Toc178130213)

[1.1. Purpose of Document 7](#_Toc178130214)

[1.2. Intended Audience 7](#_Toc178130215)  
1.3 Abbreviations ………………………………………………………………………………………...7

[1.4. Document Convention 7](#_Toc178130216)

[2. Overall System Description 8](#_Toc178130217)

[2.1. Project Background 8](#_Toc178130218)

[2.2. Project Scope 8](#_Toc178130219)

[2.3. Not In Scope 8](#_Toc178130220)

[2.4. Project Objectives 8](#_Toc178130221)

[2.5. Stakeholders 8](#_Toc178130222)

[2.6. Operating Environment 8](#_Toc178130223)

[2.7. System Constraints 8](#_Toc178130224)

[2.8. Assumptions & Dependencies 8](#_Toc178130225)

[3. External Interface Requirements 9](#_Toc178130226)

[3.1. Hardware Interfaces 9](#_Toc178130227)

[3.2. Software Interfaces 9](#_Toc178130228)

[3.3. Communications Interfaces 9](#_Toc178130229)

[4. Functional Requirements 10](#_Toc178130230)

[4.1. Functional Hierarchy 10](#_Toc178130231)

[4.2. Use Cases 10](#_Toc178130232)

[4.2.1. [Title of use case] 10](#_Toc178130233)

[5. Non-functional Requirements 11](#_Toc178130234)

[5.1. Performance Requirements 11](#_Toc178130235)

[5.2. Safety Requirements 11](#_Toc178130236)

[5.3. Security Requirements 11](#_Toc178130237)

[5.4. User Documentation 11](#_Toc178130238)

[6. References 12](#_Toc178130239)

[7. Appendices 13](#_Toc178130240)

1. Introduction

* 1. Purpose of Document

This document outlines the Software Requirements Specification (SRS) for **SQA BankPro**, which aims to modernize banking operations through automation and digitization. It describes the functionalities, constraints, and technical details required for developing and deploying the system.

* 1. Intended Audience
* **Project Team:** Developers, Designers, and Managers involved in the project.
* **Supervisors and Co-Supervisors:** To review and approve the system's requirements.
* **End-Users (Banks, Financial Institutions):** To understand system functionalities

**1.3 Abbreviations**

* **API:** Application Programming Interface
* **UI/UX:** User Interface / User Experience
* **DB:** Database
* **JWT:** JSON Web Token
  1. Document Convention

This document uses standard formatting with Arial font, size 10 for body text, and 16 for headings and 12 for subheadings. Headings and subheadings are clearly defined, and all sections will follow a consistent style.

1. Overall System Description
   1. Project Background

SQA BankPro is designed to automate key banking processes such as account management, loan tracking, and customer interaction, ensuring efficiency and security in a banking environment. This system addresses the limitations of traditional banking systems by offering a centralized, user-friendly solution.

* 1. Project Scope

The system covers core banking features like account management, transaction processing, and loan applications. The system will be scalable, secure, and easy to use for both administrators and customers.

* 1. Not In Scope
* Advanced financial services such as investments or cryptocurrency.
* Hardware provisioning or legacy system replacement.
  1. Project Objectives

The objective of this project is to create a seamless web-based bank management system that supports both administrative and customer-facing functionalities.**.]**

* 1. Stakeholders
* **Banks and Financial Institutions:** Primary users who will utilize the system for operational needs.
* **Bank Employees:** Admins managing accounts, transactions, and loans.
* **Customers:** End-users for accessing their accounts, requesting services, and managing transactions.
  1. Operating Environment

The system will run on a cloud-based platform with a MySQL database for data management. It will support modern browsers and mobile devices, ensuring wide accessibility

* 1. System Constraints

The following system constraints are imposed on the **SQA BankPro** project by the external environment:

* **Software Constraints**:  
  Since the project is a university demonstration, the system does not involve integration with real banking APIs or payment gateways. The banking functionalities will be simulated using mock data to demonstrate core features like account management, transactions, and loan processing.
* **Hardware Constraints**:  
  The system must be compatible with standard desktop and mobile devices. There are no specific hardware dependencies, but the system needs a stable internet connection for cloud-based deployment and access to mock data.
* **Cultural Constraints (includes language etc.)**:  
  The system will initially be designed for use in English. Future releases can include multi-language support, but this is out of scope for the current academic version.
* **Legal Constraints**:  
  Since the system does not interact with real financial data, legal constraints regarding data privacy and security (e.g., GDPR or similar regulations) are not applicable. However, the system will be designed with security measures, such as data encryption and secure login.
* **Environmental Constraints**:  
  The system will be hosted on cloud infrastructure, and there are no specific environmental constraints. However, the software should be able to run in noisy environments (e.g., universities, offices) without sound notifications, as it's focused on banking operations that do not require sound events.
* **User Constraints**:  
  The system will be designed with a simple user interface and intuitive navigation to ensure accessibility for all users, including non-technical users. While the project will not be specifically developed for children, ease of use will be a priority.
* **Off-the-shelf Components**:  
  The project might rely on standard libraries and components for frontend development (e.g., React.js, Node.js) and backend services (e.g., MySQL). These off-the-shelf components may have limitations, but these constraints will be addressed during implementation.

Assumptions & Dependencies

**Assumptions:**

* The system assumes that all users (customers and bank employees) have access to a stable internet connection to access the cloud-based banking platform.
* The system assumes that the mock data used for customer accounts, transactions, and loans is accurately simulated to represent typical banking operations.
* The project assumes that the web application will be accessed on modern web browsers (such as Chrome, Firefox, or Edge) and mobile devices with internet connectivity.

**Dependencies:**

* The system depends on **cloud-based infrastructure** (e.g., AWS or similar services) for hosting and database management. The availability and performance of the system are therefore dependent on the chosen cloud provider’s stability.
* The development of the project is dependent on the use of standard libraries and technologies such as **React.js** for the frontend, **Node.js** for the backend, and **MySQL** for database management. Any changes to these libraries might require adjustments to the system.
* The system depends on **simulated APIs** for performing mock transactions and loan processing. No real banking APIs or payment gateways are involved in this academic demonstration.

1. External Interface Requirements
   1. Hardware Interfaces

* **Devices Supported:** Desktops, laptops, and mobile devices.
* **Database Server:** Cloud-based server for storing and retrieving data.
  1. Software Interfaces
* **Frontend:** React.js for dynamic user interfaces.
* **Backend:** Node.js for handling requests and logic.
* **Database:** MySQL for storing account and transaction data.
* **Simulated APIs:** For mock data handling and transactions.

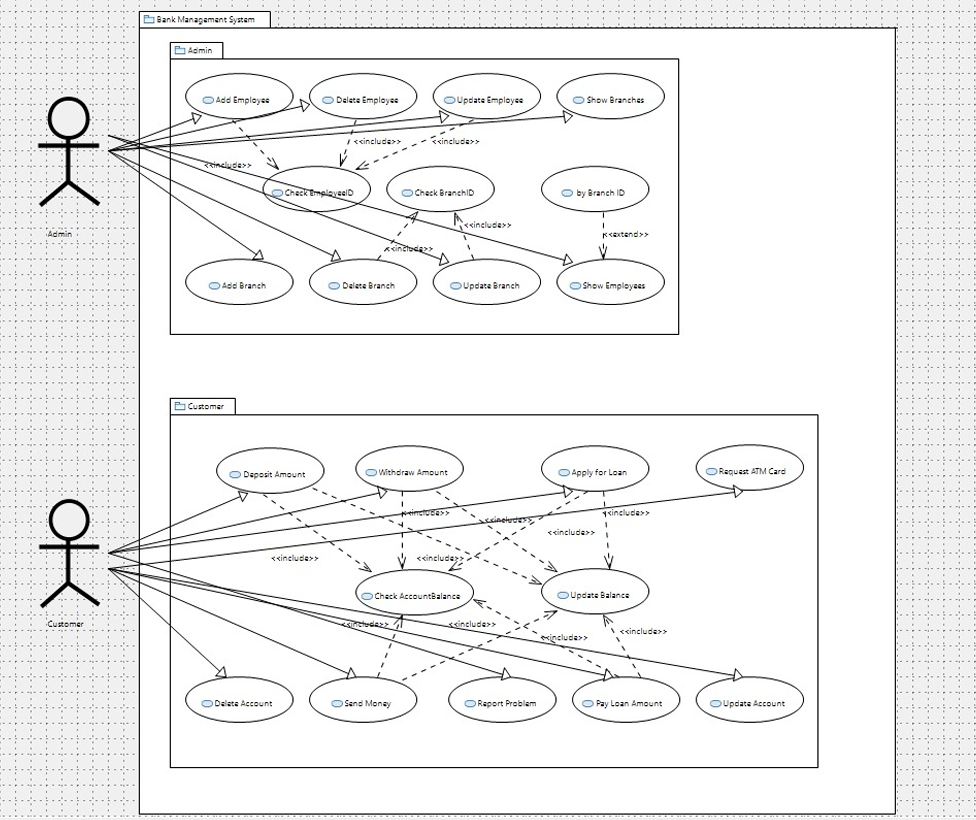
]

* 1. Communications Interfaces
* **Web Communication:** HTTP/HTTPS for secure communication between client and server.
* **No Sound Alerts:** The system uses visual notifications only.

1. Functional Requirements
   1. Functional Hierarchy

The following describes the high-level functions of the **SQA BankPro** system:

* **Customer Management**:
  + Account creation, login, and profile management.
  + Transaction management: deposit, withdrawal, fund transfer.
* **Loan Management**:
  + Application for loans, processing, and approval.
  + Loan repayment tracking.
* **Admin Management**:
  + User (employee) and branch management.
  + Transaction and report generation.
  1. Use Cases
     1. Admin and Customer Interactions



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case: Customer Login** | | | | |
| **Use case Id:** | | UC-01 | | |
| **Actors:**  Customer (user who logs into the system) | | | | |
| **Feature:** Account Management | | | | |
| **Pre-condition:** | | * Customer has an active account. * Customer knows login credentials (username and password). | | |
| **Scenarios** | | | | |
| **Step#** | **Action** | | | **Software Reaction** |
| **1.** | |  | | --- | |  |  |  | | --- | | Customer enters login credentials. | | | | The system validates the entered credentials. |
| **2.** | Customer clicks "Login" button. | | | The system authenticates the user and redirects to the dashboard. |
|  |  | | |  |
| **Alternate Scenarios** | | | | |
| **1a:** If the credentials are incorrect, the system displays an error message and prompts the user to try again.    **2a:** If the account is locked due to multiple failed attempts, the system locks the user out for a specific time and shows a lock-out message. | | | | |
| **Post Conditions** | | | | |
| **Step#** | **Description** | | | |
| **1** | |  | | --- | |  |  |  | | --- | | User is logged in and redirected to their dashboard if the credentials are correct. | | | | |
| **2** | User receives an error message if the login fails. | | | |
|  |  | | | |
| **Use Case Cross referenced** | | | * UC-02: Forgot Password * UC-03: Account Registration | |

1. Non-functional Requirements
   1. Performance Requirements

* The system should handle up to 100 concurrent users without any significant degradation in performance.
* Transaction processing (e.g., deposit, withdrawal) should be completed within 2-3 seconds.
* Account balance and loan request updates should reflect within 2 seconds of being submitted.
  1. Safety Requirements
* The system must ensure that sensitive data such as customer account details and transaction information are securely stored and protected against unauthorized access.
* The system should have appropriate error handling mechanisms to prevent crashes and ensure data consistency.
* Regular data backups should be conducted to prevent data loss in case of system failure.
  1. Security Requirements
* **Data Security**: The system must use **AES-256** encryption for storing sensitive data, such as user passwords and transaction details.
* **Authorization**: Only authorized bank employees (Admins) should have access to modify account details, employee records, or branch information.
* **Privacy**: The system must comply with basic data privacy standards. While this is a university project and doesn't process real personal data, simulated data privacy protocols should be followed to mimic real-world standards.
  1. User Documentation
* **User Manual**: A detailed user manual will be provided for both customers and bank employees.
* **Context-sensitive Help**: The system will have context-sensitive help integrated into the interface for guidance on specific sections (e.g., when accessing loan application or account details).

1. References

 **Node.js Documentation**

* **Title**: Node.js Documentation
* **Date**: Ongoing
* **Publisher**: Node.js Foundation
* **URL**: <https://nodejs.org/en/docs/>

 **React.js Documentation**

* **Title**: React.js Documentation
* **Date**: Ongoing
* **Publisher**: Facebook, Inc.
* **URL**: <https://reactjs.org/docs/getting-started.html>

 **MySQL Documentation**

* **Title**: MySQL Documentation
* **Date**: Ongoing
* **Publisher**: Oracle Corporation
* **URL**: <https://dev.mysql.com/doc/>

 **OWASP Security Principles**

* **Title**: OWASP Top 10 - Security Risks
* **Date**: 2021
* **Publisher**: OWASP Foundation
* **URL**: <https://owasp.org/www-project-top-ten/>

 **SQA Best Practices for Web Development**

* **Title**: Best Practices for Web Application Development
* **Date**: 2022
* **Publisher**: Software Quality Assurance Institute
* **URL**: <https://www.sqainstitute.org/best-practices>

1. Appendices

The **Appendices** section includes supporting details, diagrams, or other supplementary information that may be useful to the reader but would be too detailed or distracting to include in the main body of the document.

* **Appendix A: Use Case Diagram**
* **Title**: Use Case Diagram for Admin and Customer Interactions
  + This appendix contains the visual representation of the use case diagram showing interactions between the Admin and Customer roles in the **SQA BankPro** system.
* **Appendix B: Database Schema**
* **Title**: Database Schema for SQA BankPro
  + This appendix includes the schema for the MySQL database, showing the structure of tables, relationships, and data types used to store customer, transaction, and loan data.
* **Appendix C: System Architecture Diagram**
* **Title**: System Architecture for SQA BankPro
  + This appendix contains a high-level system architecture diagram that illustrates how the various components (frontend, backend, database, etc.) interact with each other.
* **Appendix D: Code Snippets**
* **Title**: Code Snippets for Key Functionalities
  + This appendix includes code examples for key functionalities such as user authentication, transaction handling, and loan application processing.
* **Appendix E: Testing Results**
* **Title**: Unit and Integration Test Results for SQA BankPro
  + This appendix provides the results of unit testing and integration testing, including any known issues and the corresponding fixes applied during the testing phase.