Software Requirements Specification

for

Microbanking and Interest Management System

Version 1.0 approved

HERATH H.M.M.P.B. 230243D

KARIYAPPERUMA K.M.N.S. 230317J

SAJIV R. 230559C

SILVA T.D.R. 230616B

WEERASEKARA R.V. 230694J

Group 27

07.08.2025

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this project is to design a backend database system for B-Trust's Microbanking and Interest Management System (MIMS). This system will support digital banking operations such as account management, deposits, withdrawals, fixed deposits, and interest tracking, while ensuring data integrity and operational efficiency across multiple branches and agents, using a secure light QA UI (Quality Assurance Testing) and backend database. This SRS outlines system requirements, both functional and nonfunctional, to support rural communities, offering banking services that are accessible, high-performance, and secure from the full range of the back office database and its interactions.

1.2 Document Conventions

This SRS adheres to the IEEE standard for software requirements specifications. Key terms specific to the Microbanking and Interest Management System (MIMS), such as "Customer," "Agent," and "Transaction," are capitalized to denote their significance. Each requirement is uniquely identified with a prefix (e.g., REQ-1, REQ-2) and assigned a priority level: High (H), Medium (M), or Low (L). High-priority requirements focus on critical system functions, such as transaction processing and interest calculation, while medium- and low-priority requirements address supplementary features. Headings are formatted in bold, and italics are used for emphasis and placeholders. A standard font is used throughout the document. All requirements are written to be concise, unambiguous, and verifiable.

1.3 Intended Audience and Reading Suggestions

This document outlines the Software Requirements Specification (SRS) for the Microbanking and Interest Management System (MIMS), a banking system for B-Trust. It is intended for project managers, developers, testers, technical writers, end users, and compliance officers.

- **Chapters 1-3** provide an overview of the system and are recommended for all readers.
- **Chapter 4** details functional requirements—important for developers, testers, and writers.
- **Chapter 5** covers non-functional requirements and performance standards—for developers and compliance teams.

Readers can begin with the overview and then focus on sections relevant to their role.

1.4 Product Scope

The MIMS system aims to assist B-Trust in its efforts to provide accessible and dependable banking services to the underserved rural areas of Sri Lanka. Customers can now manage their accounts through servicing agents, and the system automates account management, transaction processes, and interest computations. Bank staff can also generate critical reports. Significant improvements can also be accomplished in operational efficiency, transaction tracking, and the support of complex multi-tiered savings plans for children through seniors, and even joint accounts.

Aligned with B-Trust's financial inclusion objectives, the system provides dependable and accessible banking services. Additional details, if any, can be found in a separate vision and scope document. This document concentrates on the backend database and QA interface requirements.

1.5 References

• IEEE Standard 830-1998, IEEE Recommended Practice for Software Requirements Specifications.

1.6 *Overview of Document*

This Software Requirements Specification (SRS) captures all the details pertaining to the Microbanking and Interest Management system (MIMS) that was designed and developed for B-Trust, a microfinance bank in Sri Lanka.

The document has been split into three main chapters to assist both the stakeholders and the developers. Chapter 1, Introduction, explains the purpose, scope, and structure of the SRS while contextualizing B-Trust's desire to digitize banking processes for enhanced financial inclusion and streamlined operations. In Chapter 2, Overall Description, the system's environment, the user classes, the system's major functions, along with the constraining factors, are presented to give an overview of the role of MIMS in managing the savings accounts, fixed deposits, and transactions. Chapter 3 outlines the External Interface Requirements operations, and Chapter 4 outlines the Key Functional features of the MIMS., The last chapter defines the nonfunctional requirements of MIMS to ensure secure, efficient, and scalable system performance in line with regulatory standards.

2. Overall Description

2.1 Product Perspective

The Microbanking and Interest Management System (MIMS) is a standalone system designed to replace B-Trust's manual, paper-based microfinance operations. Built from the ground up, MIMS serves as the core digital backend for the institution, with no dependencies on existing software.

Key Features:

- Savings account management
- Fixed deposit handling
- Customer registration
- Transaction logging
- Automated interest processing
- Report generating

Primary Users: Internal staff (agents and managers) accessing the system via a simple interface for data entry and validation.

Future Scalability: The modular architecture allows for potential expansion, such as online customer access via web or mobile platforms—though this is excluded from the current release.

2.2 Product Functions

The Microbanking and Interest Management System (MIMS) provides the following core functionalities:

1. User & Account Management

- Register and manage branches, agents, and customers.
- Open savings accounts under customizable account plans.

2. Transaction Processing

- Process deposits and withdrawals with timestamped audit logs.
- Support fixed deposits linked to primary savings accounts.
- Restrict transactions based on account eligibility and available balance.

3. Automated Operations

- Calculate and credit monthly interest automatically.
- Maintain a complete transaction history per customer.

4. Reporting & Analytics

Generate predefined reports, including:

- Agent-wise transaction summaries
- Fixed deposit status reports
- Customer activity reports

- Monthly interest distribution statements

2.3 User Classes and Characteristics

User Class	Responsibilities	Access Level
Admin	Create branches, agents, plans, and configure system-wide settings	Full access
Branch Manager	Monitor branch performance, approve transactions, and oversee agents	High-level access
Agent	Register customers, handle savings and FD operations, and perform transactions	Operational Access

- All users are assumed to have basic computer literacy. Admins and testers are expected to have technical proficiency to manage system configurations or validations.

2.4 Operating Environment

- **Client Interface**: Web-based UI (HTML/CSS/JavaScript)
- **Database**: MySQL / PostgreSQL
- **Server OS**: Linux or Windows
- **Client Devices**: Windows PCs for agents and testers
- **Network**: Operates over LAN or VPN-secured connection
- **Browser Support**: Chrome, Firefox ,Edge (latest versions)

2.5 Design and Implementation Constraints

The system must adhere to the following technical and operational constraints:

Database & Transaction Integrity

- ACID compliance for all financial transactions (Atomicity, Consistency, Isolation, Durability).
- Foreign keys and indexing are mandatory for data integrity and query performance.

Account Rules

- Fixed deposits (FDs): Limited to one FD per savings account.

- Savings plans: Must enforce:
    1. Minimum balance requirements.
    2. Age-based eligibility criteria.

Interest Calculation

- Computed monthly based on a standardized 30-day cycle.

Security & Access

- Role-based access control (RBAC) for internal staff (agents, managers).
- No customer-facing portal in the current release (internal users only).

2.6 User Documentation

The following comprehensive documentation will be provided with the system:

1. Banking Agent Manual

- Daily transaction procedures
- Account management workflows
- Error resolution guide

2. Administrator Guide

- Account plan configuration
- Branch setup and management
- Interest rate adjustment

3. Technical Specifications

- ER Diagram (Visual database representation)
- Schema Documentation (Table structures and relationships)

4. QA Testing Package

- Test case scenarios
- Validation checklists
- UAT guidelines

5. Demonstration Materials

- Sample customer dataset
- Test transaction templates

2.7 Assumptions and Dependencies

Operational Assumptions

- All banking transactions occur only during official business hours
- Agents are solely responsible for the accurate data entry of:
    1. Customer information
    2. Transaction details
- Customer interactions occur exclusively through agents (no direct system access)

Financial Rules

- Monthly interest calculations use a **fixed 30-day cycle**
- All interest crediting occurs **automatically** per configured schedules

System Dependencies

- **Proper administrative configuration** of:
    1. Account plans
    2. Interest rates
    3. Branch parameters
- **Future compatibility** with (but no current support for)
- Mobile banking applications
- Online customer portals

3. External Interface Requirements

3.1 User Interfaces

Role-Based Access:
UI components and features will be enabled or disabled based on the agent's role (e.g., account creation, transaction management, FD approval).

- The interface will follow a clean and minimalistic design approach.
- Responsive layout and styling, ensuring compatibility across devices.
- Each screen will include a consistent top navigation bar with the primary options
- Error messages will be displayed inline, directly beneath the relevant input field.
- All error messages will appear in red text to clearly indicate issues.
- Confirmation modals will be shown before performing critical actions such as deleting records or approving fixed deposits.
- The UI will support keyboard navigation to enhance accessibility for all users.

3.2 Hardware Interfaces

- The system is designed to run on standard desktop or laptop hardware.

- No special hardware dependencies are required.
- The application will operate through modern web browsers.
- No direct hardware integration is necessary for core functionalities.
- Optionally, the system may support:
- USB thermal receipt printers for transaction confirmations.
- Use of standard print drivers for printer compatibility.

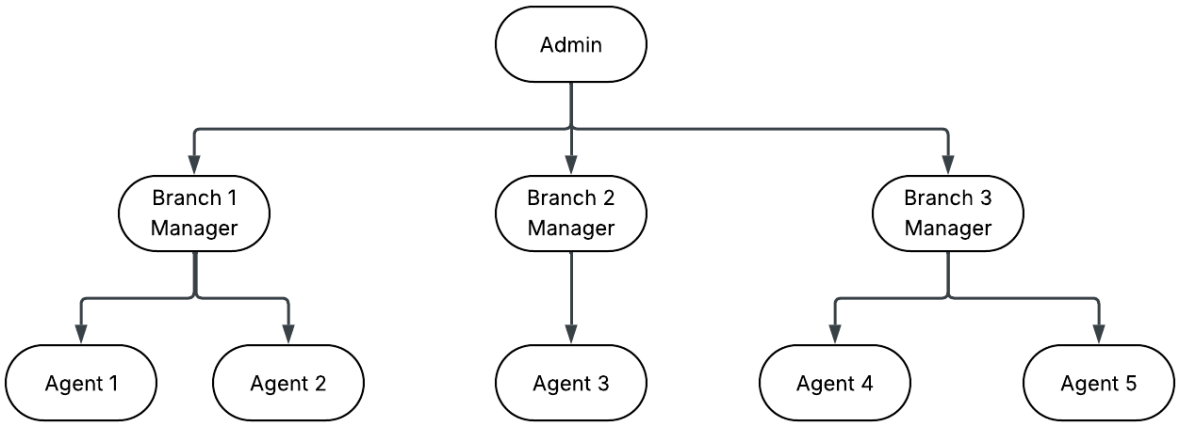
3.3 Software Interfaces

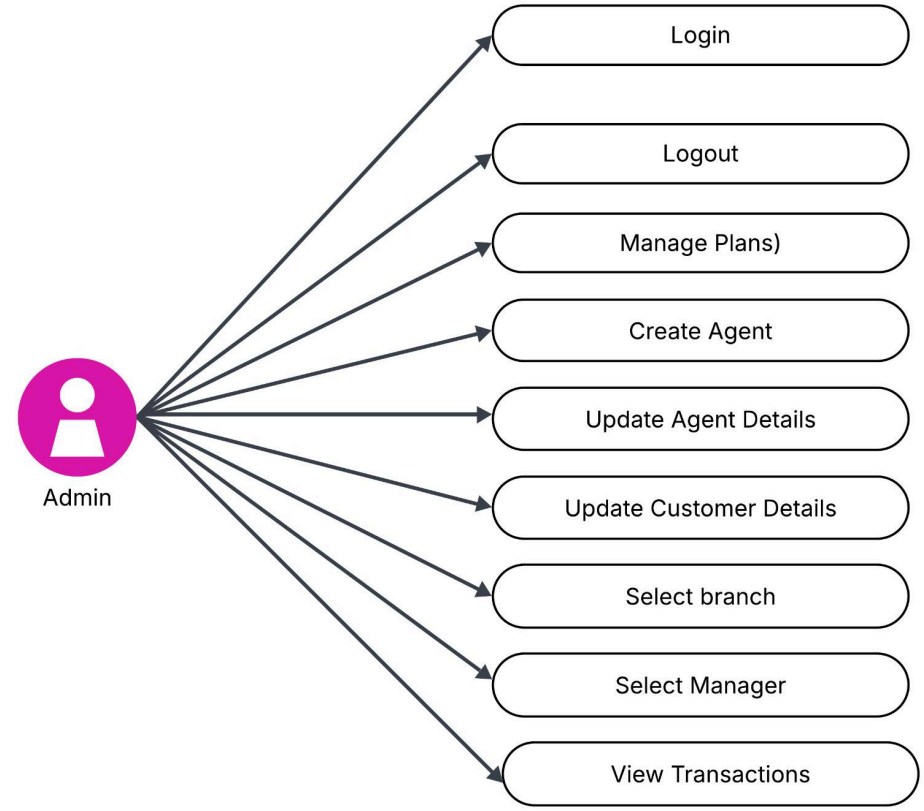
- Backend: Developed using Python FastAPI, exposing RESTful APIs for frontend interaction. Data is exchanged in JSON format.
- Frontend: Communicates with the backend using REST APIs via axios . All user actions trigger API calls.
- Database: Uses PostgreSQL/MySQL. FastAPI connects using an ORM or raw SQL. Data is securely stored and managed.
- Authentication: Implements JWT-based authentication. Access to routes is controlled by user roles (e.g., agent, manager, admin).
- Tools: Uses Tailwind CSS for frontend design, React vite for frontend.

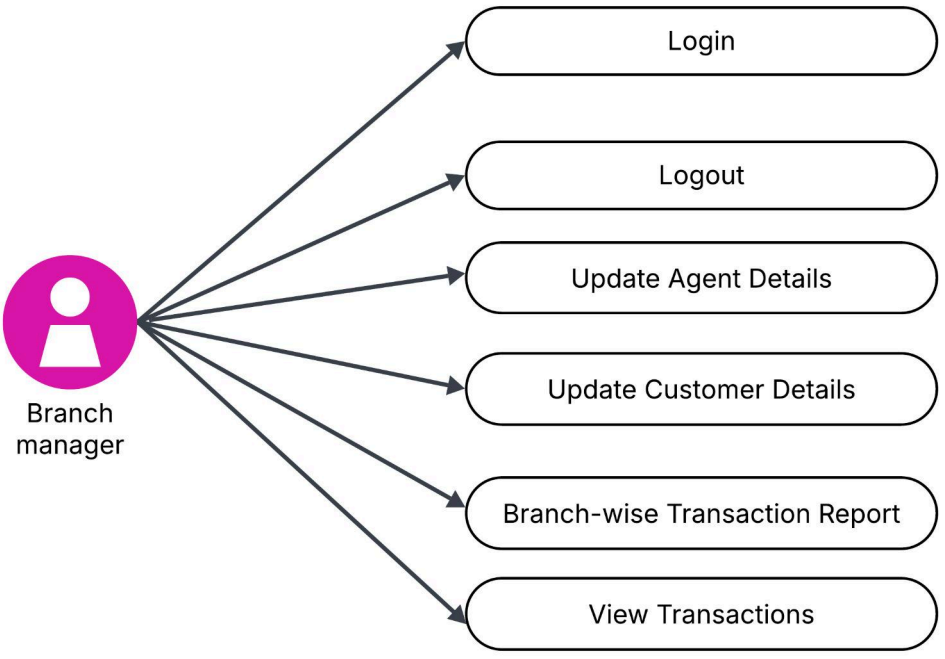
3.4 Communications Interfaces

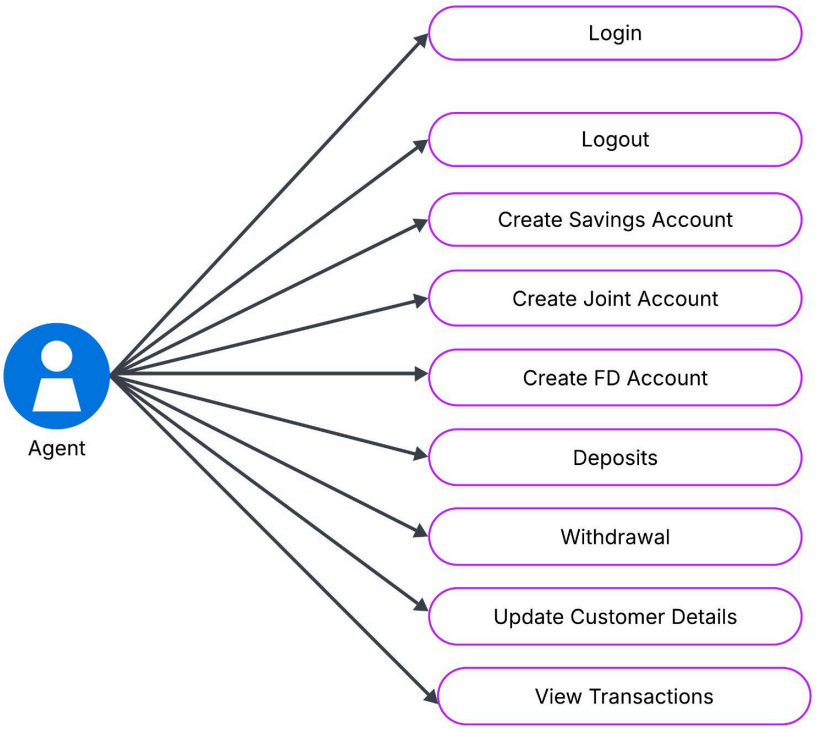
- Protocol: All communication between frontend and backend uses HTTP/HTTPS.
- API Format: Data is exchanged in JSON format via RESTful API endpoints.
- Authentication: JWT tokens are sent in the Authorization header with each API request.
- Client Requests: Frontend sends POST, GET, PUT, DELETE requests to interact with backend services.
- Error Handling: Errors are returned as standardized JSON responses with appropriate HTTP status codes.

4. System features




Login
Logout
Manage Plans)
Create Agent
Update Agent Details
Update Customer Details
Select branch
Select Manager
View Transactions
Admin


Login
Logout
Update Agent Details
Update Customer Details
Branch-wise Transaction Report
View Transactions
Branch manager



4.1. Login

Trigger: The user clicks on the “Sign in” button on the “Home” page.

Precondition: The user must have an account and user credentials

Basic Path:

1. The user clicks on the “Sign in” button on the “Home” page.
2. The system displays the sign-in form.
3. The user enters their credentials and submits them.
4. The credentials are verified against the system.
5. Then the system sends an OTP number to the user’s email account while enabling the
6. space for entering the OTP.
7. The user enters the OTP in the given place and submits it.
8. The system verifies the OTP and navigates to the corresponding page.

Alternate Path:

- If the user logs into his account for the first time, the system will ask to change his password. Then the user enters his new password. The system will validate the password. If the validation is successful, the system will continue the process from step 4.
- If the credentials do not match, the system will display an error message.

- If the OTP does not match, the system will give another 2 chances for the user to enter the correct OTP. If that step fails, the system will update the user logs as “OTP failed”, send an email to the manager about it and block the user account for an hour.

Postcondition: The user logs in to the account or remains logged out.

4.2. Logout

Trigger: The user will click the “Sign out” button on his account page.

Precondition: The user should be logged into his account.

Basic Path:

1. The user clicks on the “Sign out” button.
2. The system displays a sign-out confirmation box.
3. The user submits “Yes”.
4. The system navigates the user to the “Home” page and signs out the user.

Alternate Path:

- If the user submits “No”, the system will close the confirmation box, and the user will remain logged in

Postcondition: The user logs out of the account or remains logged in.

4.3. Create Account

Trigger: Agent accesses the “Create Account” page.

Precondition: The agent logged into the account.

Basic Path:

1. The agent selects the “Create Account” option.
2. The system displays the “Create Account” page.
3. The agent selects the customer from the existing list.
4. The agent chooses the account type: “Savings Acc”, “FD”, “Joint Acc”
5. The agent enters the required account details.
6. The agent submits the form.
7. The system creates the account and confirms the creation.

Alternate Path:

- If the account type is “Savings”, the system prompts the agent to select a plan type, and the minimum balance will be validated with respect to the selected savings plan.
- If there’s missing or invalid information, the system prompts the agent to correct it.

- If the customer already has an account of that type **in that particular branch**, the system will display an error message.

Postcondition: The account is created and linked to the selected customer.

4.4. Branch-wise Transaction Report

Trigger: agent accesses the “Transaction Report” page.

Precondition: The agent is logged into the account and belongs to a specific branch.

Basic Path:

1. The agent selects the “Transaction Report” option.
2. The system displays the report page with filter options.
3. The agent selects the desired duration for the report.
4. The agent submits the filter criteria.
5. The system generates and displays the report.

Alternate Path:

- If no transactions match the selected duration, the system displays a message indicating no results found.
- If the agent doesn’t select any filter options or any desired duration, the system will ask to select at least one of them.
- If the agent only selects a filter option and doesn’t select any desired duration, the system will add a duration of a month as the default and generate the report, and display it.
- If the agent only selects a desired duration and doesn’t select any filter options, the system will add “All” as the default and generate the report, and display it.
- If the system doesn’t have data to make reports for the selected filter options and the duration, the system will display that data that isn't available for that combination.

Postcondition: The transaction report for the agent’s branch is generated and displayed.

4.5. Update agent Details

Trigger: Admin/Manager accesses the “Update agent Details” page.

Precondition: The Manager is logged into the account to change the agent details.

Basic Path:

1. The Manager selects the “Update agent Details” option.
2. The system displays the agent details form with current information.
3. The manager updates the relevant fields (e.g., mobile, email, and address).
4. The Manager submits the updated information.
5. The system asks to confirm the updating process by the manager’s user password
6. The manager enters his user password.
7. The system saves the changes and confirms the update.

Alternate Path:

- If any required fields are left blank, the system prompts the agent to fill them in.
- If the confirmation fails, details will not be updated in the system, and the system will display an error message

Postcondition: The agent’s details are updated in the system.

4.6. Deposits

Trigger: agent accesses the “Cash Deposit” page.

Precondition: The agent is logged into the account.

Basic Path:

1. The agent selects the “Cash Deposit” option.
2. The agent fills the form with the client's data and submits it after the client hands over the money.
3. The system asks to confirm the cash depositing process by the agent’s user password
4. The agent enters the agent user password.
5. The system updates the cash depositing and confirms the process.

Alternate Path:

- If the client does not have an account in the bank or the entered account number is invalid, the process will not start, and the system will display an error message.
- If the data filled in the form is invalid or empty, the system will not let the agent submit the form.
- If the agent’s user password is invalid, the system will not process the cash deposit and display an error message.

Postcondition: The agent deposits cash for the customer.

4.7. Branch-wise Logs

Trigger: The manager accesses the “Branch Logs” page.

Precondition: The manager is logged into the account.

Basic Path:

1. The manager selects the “Branch Logs” option.
2. The system displays all the logs for the last 30 days, which are relevant to the branch in tabular format with filter options to select the duration and the type of logs (Default: duration = “Last 30 days”, type = “All”).
3. The manager selects filter options for the duration and the type of log and submits the selected filter options.
4. The system displays all the logs in tabular format that are relevant to the selected filter options.

Alternate Path:

- If the branch doesn’t log for the default values, the system will display a message that the logs are empty.
- If the manager submits the filter options without selecting a filter option, the system will set the unselected filter options to default values and handle the submit action.

Postcondition: The manager can see his branch’s logs as he prefers.

4.8. Create FD

Trigger: Agent accesses the “Create FD” page.

Precondition: The agent is logged into the account

Basic Path:

1. The agent selects the “Create FD” option.
2. The system fetches the available FD plans.
3. The agent selects a savings account from the list of existing savings accounts of the customer.
4. The agent selects the FD plan and amount according to the customer’s purpose
5. The system processes the request and creates an FD.

Alternate Path:

- If the customer does not have a savings account, the system displays a message instructing the agent to create a new savings account for the relevant customer.
- If the FD amount is higher than the available savings amount (considering the minimum balance that should be maintained by different savings account types), the system displays a message saying insufficient funds.

Postcondition: The amount is debited from the account, and the FD is created.

4.9. Update Customer Details

Trigger: Agent accesses the “Update Customer Details” page.

Precondition: The agent is logged into the account.

Basic Path:

1. The agent selects the “Update Customer Details” option.
2. The system displays the customer details form with current information.
3. The agent updates the relevant fields (e.g., address, contact details).
4. The agent submits the updated information.
5. The system asks to confirm the updating process.
6. The system saves the changes and confirms the update.

Alternate Path:

- If any required fields are left blank, the system prompts the agent to fill them
- If the confirmation fails, details will not be updated in the system, and the system will display a message saying that the changes are not being saved.

Postcondition: The customer’s details are updated in the system, or remain unchanged.

4.10. View Transactions

Trigger: Agent/Manager/Admin accesses the “View Transactions” page.

Precondition: The agent is logged into the account.

Basic Path:

1. The agent selects the “View Transactions” option.
2. The system displays the list of accounts associated with the customer.
3. The agent selects an account to view its transaction history.

4. The system retrieves and displays the transaction history for the selected account for the last 30 days, and displays an option menu to select the desired duration
5. The agent selects a duration and submits.
6. The system retrieves and displays the transaction history for the selected account for the selected duration.

Alternate Path:

- If there are no transactions for the selected account, the system displays a message indicating no transactions found.

Postcondition: The transaction history for the selected account/branch is displayed.

4.11. Withdrawal

Trigger: Agent accesses the withdrawal page of the relevant customer(will simulate on the website).

Precondition: The customer must have an account created.

Basic Path:

1. The agent initiates a withdrawal according to the customer's request.
2. The system displays the available balance.
3. The agent enters the amount to withdraw.
4. The system processes the withdrawal request.
5. The system updates the account balance and confirms the transaction.

Alternate Path:

- If the withdrawal amount exceeds the available balance, the system displays an error message.

Postcondition: The account balance is updated, and the withdrawal is confirmed.

4.12. Deposit

Trigger: Agent accesses the deposit page of the relevant customer(Simulated on the website).

Precondition: None

Basic Path:

1. The agent initiates a deposit through the CDM.
2. The agent enters the account number of the relevant customer.
3. The name of the account holder is displayed.
4. The system prompts the agent to enter the amount to deposit.
5. The customer confirms the deposit.
6. The system processes the deposit request.
7. The system updates the account balance and confirms the transaction.

Alternate Path:

- If the displayed is not the expected/the account number is invalid, the agent can go back
- If the deposit amount exceeds the allowed limit, the system displays an error message

Postcondition: The account balance is updated, and the deposit is confirmed.

4.13. Manage Plans

Trigger: The admin accesses the “Manage Plans” page.

Precondition: The admin is logged into the account.

Basic Path:

1. The admin accesses the “Manage Plans” page.
2. The system displays the available and the unavailable plan lists separately.
3. Admin selects the plan to be removed or restored.
4. The system asks the admin to confirm it.
5. Admin confirms it.
6. System updates the “availability” field and confirms the changes.

Alternate Path:

- If the admin doesn’t confirm, the system will cancel the pending process of updating the plan’s availability.

Postcondition: The selected plan is removed or restored.

4.14. Create agent

Trigger: The admin accesses the “Create agent” page.

Precondition: The admin is logged into the account.

Basic Path: The admin accesses the “Create agent” page.

1. The system displays the form to create a new agent
2. The admin fills the fields, including the user name field of the form
3. The system generates a random password and creates the new agent and the new agent user accoun,t and confirms it.

Alternate Path:

- If the form has empty fields, the system will notify the admin to fill those empty fields.
- If the username already exists, the system will notify the admin to use a new username.

Postcondition: The new agent is created.

4.15. Select Manager

Trigger: The admin accesses the “Select Manager” page.

Precondition: The admin is logged into the account.

Basic Path:

1. The admin accesses the “Select Manager” page.
2. The system displays the list of agents who are not managers and the list of managers.
3. The admin selects an agent.
4. The system enables the list of branches with no branch managers.
5. The admin selects the branches with no managers.
6. The system will ask for the confirmation of the admin.
7. The admin confirms.
8. The system adds the selected agent as the manager to the selected branch.

Alternate Path:

- If the admin doesn’t confirm the update, the system will cancel the pending process.
- If the admin selects a manager, the system will change its process of updating to remove the manager and continue the task of the confirmation of the admin.
- If the admin confirms, the selected manager will be removed from the manager position of his branch. Otherwise the system will cancel the pending process.

Postcondition: The branch manager is updated (removed or added).

4.16. Select Branch

Trigger: The admin accesses the “Select Branch” page.

Precondition: The admin is logged into the account.

Basic Path:

1. The admin accesses the “Select Branch” page.
2. The system displays the list of available branches.
3. The admin selects a Branch.
4. The admin selects the branches with no managers.

Postcondition: The branch is selected to view

5. Nonfunctional Requirements

This section addresses the non-functional requirements of the Microbanking and Interest Management System (MIMS). This encompasses performance, security, safety, and system properties required for a strong, scalable, and secure banking system. The design meets specifications by the **Central Bank of Sri Lanka (CBSL)** and the **Sri Lanka Personal Data Protection Act (PDPA)**.

5.1 Performance Requirements

The MIMS must offer responsive performance, efficient processing of data, and scalable design in an effort to address rural banking needs.

- The system backend will use PostgreSQL or MySQL, which provides full ACID compliance, row-level locking, and efficient indexing.
- To enable seamless banking services under heavy loads, the system should support a minimum of 100 concurrent agent sessions without any decrease in performance.
- Deposits, withdrawals, and account operations must be completed within 2 seconds.
- For fixed deposits, the monthly interest computations will be completed within 60 seconds for up to 1000 records.
- Reports such as transaction summaries and customer statements must be generated within 5 seconds for datasets of up to 500 records.
- The database must be capable of storing and handling a minimum of 50,000 customer profiles, 100,000 transactions, and 10,000 active Fixed Deposits.
- Database indexing strategies must be optimized for high-read workloads with compound indexes on fields like `customer_id`, `agent_id`, and `transaction_date`.
- The system should maintain an average response time of 3 seconds or less during peak hours (9 AM – 1 PM, 3 PM – 5 PM).

- System availability shall be a minimum of 99.5% with all planned maintenance carried out outside office hours.

5.2 Safety Requirements

The system must ensure that all operations are validated and data is handled securely to prevent data loss, corruption, and improper modification.

- All transactions are recorded with full metadata. (timestamp, agent ID, customer ID, and reference number)
- Unauthorized data manipulation is strictly prohibited and should trigger security alerts.
- The system will perform daily backups and support point-in-time recovery(PITR) to restore the database to a consistent state when a system failure is detected.
- Access to execute logs will only be given to admins..
- The logout activity is logged, ensuring that all session terminations are recorded for audit and security purposes.
- All operations must enforce business rule validation via stored procedures and CHECK constraints.
- Operations that violate safety policies (like overdrafts, multiple FDs on one savings account) will be automatically rejected and logged.
- Schema changes and log access must be traceable and recorded in PostgreSQL/ MySQL general logs and binary logs, with protection against tampering.

5.3 Security Requirements

Security is of the highest importance in a financial system. The MIMS must protect sensitive data, enforce authentication and authorization, and defend against threats.

- All of the users (agents, branch managers, and admins) must authenticate via secure (username/password and multi-factor authentication).
- **Password security**: Passwords are never stored in plain text, and they will be salted and hashed and stored in the database.
- If a user who is already signed in tries to go back to the login page and is successful, then the user has to sign in again in order to access the application’s data.
- **Session security:** Session expires after 15 minutes of inactivity and session regeneration upon login to prevent session fixation.
- After 3 failed login attempts, the user account will be locked to prevent brute-force attacks.
- The logout process ensures that the user’s session is securely terminated, preventing unauthorized access to the account after logout.
- Role-based access control must ensure that only authorized users have access to sensitive operations.
- **Rate Limiting**: Implement rate limiting to protect against brute force and credential stuffing attacks
- Steps have been taken to prevent SQL injection attacks in the creation of the queries.
- All data in transit will be encrypted using SSL/TLS, and data at rest will be encrypted using AES-256 where applicable.

- API endpoints will be protected with JWT tokens, and tokens will be scoped by role.
- Compliance with **Sri Lanka PDPA** and **CBSL security guidelines** is mandatory.

5.4 Software Quality Attributes

These attributes ensure that the system is flexible and future-proof.

Availability:
- The system needs to achieve 99.5% uptime, excluding planned maintenance, ensuring uninterrupted access.

Reliability:
- ACID properties will ensure that transactions are reliable and resistant to partial failures or race conditions.

Usability:
- The Interface is designed for agents with basic digital literacy.
- Minimal training required for agents.

Maintainability:
- Codebase must follow a modular architecture with reusable components and developer documentation.
- In-code documentation and automated API documentation.

Scalability:
- The system must support horizontal scaling of agents, branches, and customer data without redesigning the architecture.

Portability:
- Application deployable on both Linux and Windows servers**,** using platform-independent libraries.
- Frontend/browser support includes Chrome, Firefox, and Edge (latest versions)

Testability:
- The system shall include automated unit, integration, and regression tests with at least 80% code coverage.

Flexibility:
- Business rules (interest rates, account types, eligibility) are configurable via the admin dashboard or config files.

Interoperability:
- RESTful APIs designed to integrate with mobile apps, SMS gateways, and external financial audit tools

5.5 Business Rules

These domain-specific rules guide how the system handles banking activities.

- Every customer is assigned to an agent during onboarding, but can transact with any agent; the system logs the actual handling agent per transaction.
- Customers are able to open several savings accounts, but can only have one active Fixed Deposit per savings account.
- Joint accounts must involve two or more registered customers with shared permissions.
- Withdrawals are permitted only when the minimum balance is maintained and the age and plan eligibility requirements are met.
- Interest on FDs is calculated and credited monthly based on a 30-day cycle using stored procedures and credited to the linked savings account.
- Overdrafts are strictly prohibited and enforced at the database and application layers.
- All transactions must be logged with a unique reference ID, timestamp, agent ID, customer ID, and transaction type.
- Transactions are irreversible once committed. Reversal must be handled as a separate, compensating transaction, subject to manager approval.
- Business rule violations must result in an error message and block the operation from proceeding.
- Schema and configuration must allow for the addition of new account types, interest plans, and eligibility conditions, and the disabling of obsolete services without affecting existing records.

Appendix A: Glossary

Term	Definition
Admin	The term “Admin” refers to the Director Board of B-Trust Bank System
Agent	A banking representative assigned to a customer, responsible for facilitating transactions and account management at a B-Trust branch.

Branch	A regional service location of B-Trust where customers register and perform banking activities.
Customer	An individual or group (for joint accounts) registered with B-Trust, eligible to open savings accounts or fixed deposits.
Fixed Deposit (FD)	A financial product where a customer deposits a lump sum for a fixed term (6 months, 1 year, or 3 years) to earn interest at a specified rate.
Interest Credit	A transaction that records the monthly interest earned on a fixed deposit, credited to the linked savings account.
Joint Account	A savings account shared by multiple customers, allowing deposits and withdrawals by any account holder, subject to minimum balance requirements.
MIMS	Microbanking and Interest Management System, the software system developed to manage B-Trust's banking operations, including accounts, transactions, and reports.
Savings Account	A deposit account offered by B-Trust under various plans (Children, Teen, Adult, Senior, Joint) with specific interest rates and minimum balance requirements.
Transaction	A recorded financial activity, such as a deposit, withdrawal, or interest credit, logged with a timestamp, type, and reference number.