**Bank Management System**

**Software Requirements Document**

**V 2.0**



Table of Contents

[1.0 Introduction 3](#_Toc482974035)

[1.1 About this document 3](#_Toc482974036)

[1.1.1 Purpose & Scope of the document 3](#_Toc482974037)

[1.1.2 Intended Audience 3](#_Toc482974038)

[1.2 About the Software System 3](#_Toc482974039)

[1.2.1 Purpose 3](#_Toc482974040)

[1.2.2 Scope of the system 4](#_Toc482974041)

[1.2.3 Exclusions 5](#_Toc482974042)

[1.2.4 System Perspective 5](#_Toc482974043)

[1.2.5 System diagram 6](#_Toc482974044)

[1.2.6 Architecture diagram 6](#_Toc482974045)

[1.2.7 System Environment 7](#_Toc482974046)

[1.2.8 User Characteristics 8](#_Toc482974047)

[1.2.9 Impact of the System 8](#_Toc482974048)

[2.0 System Requirements 9](#_Toc482974049)

[2.1 Functional Requirements 9](#_Toc482974050)

[2.1.1 Home Module (Common for all the requirements) 9](#_Toc482974051)

[2.1.2 Perform Transaction 9](#_Toc482974052)

[2.1.3 View Transactions 11](#_Toc482974053)

[2.1.4 Apply Home Loan 12](#_Toc482974054)

[2.1.5 Get Home Loan details 13](#_Toc482974055)

[2.1.6 Apply Educational Loan 14](#_Toc482974056)

[2.1.7 Get Education Loan details 15](#_Toc482974057)

[2.2 Nonfunctional Requirements 16](#_Toc482974058)

[2.2.1 UI Requirements: Inn 16](#_Toc482974059)

[3.0 Requirements / Use Case Summary Table 17](#_Toc482974060)

[4.0 Make /Buy analysis 17](#_Toc482974061)

[4.1 Reusable components 17](#_Toc482974062)

[5.0 Annexure 17](#_Toc482974063)

# Introduction

## About this document

### Purpose & Scope of the document

The purpose of the software requirements document is to systematically capture requirements for the project and the system “Bank Management System” to be developed. Both functional and non-functional requirements are captured in this document. It also serves as the input for the project scoping

The scope of this document is limited to addressing the requirements from a user, quality, and non-functional perspective. It is recommended that design aspects are not added in this document

### Intended Audience

Project Team

## About the Software System

The client would like to develop an independent application **Bank Management System** (BMS) application in order to automate the process of managing the activities of bank like Performing Transactions, View Transaction details, Apply loan, Get loan details.

The following section will cover aspects related to Bank Management System.

### Purpose

BMS is a system used for managing the activities of bank like Performing Transactions, View Transaction details, Apply loan, Get loan details.

The following are the three modules in the system

1. Transaction Module
   * Perform Transactions
   * View Transactions
2. Home Loan Module
   * Apply Home Loan
   * Get Home Loan details
3. Educational Loan Module
   * Apply Educational Loan
   * Get Educational Loan details

### Scope of the system

The scope of the system is explained through its modules as follows-

1. Transaction Module

* Perform Transactions - will be used by registered users to make Transactions on their accounts. The system stores the details of the Transaction types Deposit, Withdrawal.
* View Transactions - used by customers to view the transaction details.

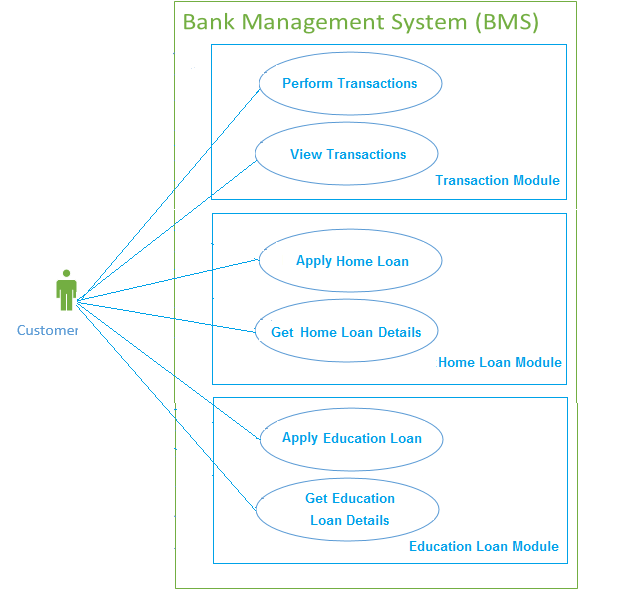
1. Home Loan Module - used by customers to apply home loan and get the loan details.

* Apply Home Loan - will be used by registered users to apply home loan into the system. The system stores the loan details in the system along with the account details.
* Get Home Loan Details - used by customers to get the loan details.

1. Educational Loan Module - used by customers to apply Educational loan and get the loan details.

* Apply Educational Loan - will be used by registered users to apply Educational loan into the system. The system stores the loan details in the system along with the account details.
* Get Educational Loan Details - used by customers to get the loan details.

**Use Case Diagram**



### Exclusions

The system will operate only on the modules discussed above and will not include any additional functionality.

### System Perspective

Bank Management System is an independent software system developed to manage the activities like Perform Transaction, View Transaction details, Apply Loan, Get Loan details using the Java architecture.

### System diagram

Perform Transaction

View Transactions

Apply Home Loan

Database

Start

Get Home Loan Details

Apply Educational Loan

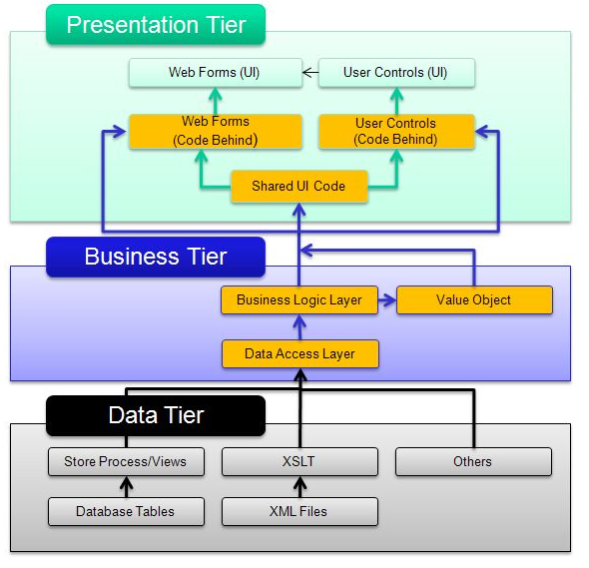
Get Educational Loan Details

### Architecture diagram

#### Physical Architecture:

A physical architecture is an arrangement of physical elements, (system elements and physical interfaces) that provides the designed solution for a product, service, or enterprise. It is intended to satisfy logical architecture elements and system requirements. Bank Management System follows a three layered architecture namely presentation layer, business logic layer and data access layer.

* **Presentation Tier** is the tier in which the users interact with an application. Presentation Tier contents Shared UI code, Code Behind and Designers used to represent information to user.
* **Business Tier** is mainly working as the bridge between Data Tier and Presentation Tier. All the Data passes through the Business Tier before passing to the presentation Tier. Business Tier is the sum of Business Logic Layer, Data Access Layer and Value Object and other components used to add business logic.
* **Data Tier** is basically the server which stores all the application’s data. Data tier contents Database Tables, XML Files and other means of storing Application Data.



### System Environment

Bank Management System application will be operated from the client server with parallel processor support. When a user connects to the Web Server, the Web Server will interact with the Database after processing the business logic to transfer data to and from a database.

Kindly refer to the table below for Skill set to be used to implement the requirements.

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Requirement** | **Skill set to be used** |
| 1 | Perform Transactions  Apply Home Loan  Apply Educational Loan | Spring MVC 3.x, Hibernate 4.x, SQL |
| 2. | View Transactions  Get Home Loan details  Get Educational Loan details | HTML, JSP/Servlet, REST WebServices, JSON, SQL. |

### User Characteristics

The application should be user friendly; hence the user should not need any software and hardware knowledge. The user may be new to the client products having never used them before.

The types of users likely to use the system are as follows,

* Client
* Developers, testers, other associates in the project

### Impact of the System

This is a new product which is developed for internal users. Expected impact of the product is to automate existing manual processes in order to make them more efficient and cost effective.

# System Requirements

## Functional Requirements

### Home Module (Common for all the requirements)

|  |  |
| --- | --- |
| Bank Management System  Req-2.1.1 | Home Page |
| **Functional Requirements** | When the users enter the following link **http://localhost/BankManagementSystem** in the browser, Home page appears which will have a short welcome message; a brief introduction to the purpose of the Bank Management System, the user can choose appropriate options. Under home page, we need below links-   1. Perform Transaction. 2. View Transaction details. 3. Apply Home Loan 4. Get Home Loan Details 5. Apply Educational Loan 6. Get Educational Loan Details |

### Perform Transaction

|  |  |
| --- | --- |
| Bank Management System  Req-2.1.2 | Customer – Perform Transaction  **Skills to be used: Spring MVC 3.x, Hibernate 4.x, SQL** |
| **Functional Requirements** | * Customer should be able to perform Transactions on his account. * Any given point of time, the Customer should have the option to go for home page, by clicking home. |
| **UI Interface Screen** | This should capture the following fields   * A/c Number – Represents Account number of the A/c holder. * Transaction id – Represents the Transaction id (auto populated). * Transaction Type – Represents the type of transaction. * Amount – Represents the amount to be debited or credited. * Description – Some remarks giving description about the Transaction.   **Transaction Confirmation Screen**   * Display the success message. * Balance - Represents the balance amount after the transaction is complete. |
| **Trigger** | Registered user triggers the functionality once he/she wants to credit (deposit) or debit (withdraw) the amount into the account. |
| **Pre-Conditions / Assumptions** | Create table **USER\_DETAILS** and **TRANSACTION\_DETAILS** as per theDatabase Designgiven in the **System Design document (Refer section 5.0).**  Populate data in **USER\_DETAILS** table using insert query to insert user details.   * A/c Number – Represents Account number of the A/c holder. * A/c Type – Represents the Account type (Salary/Saving). * A/c Holder Name – Name of the Account Holder. * Balance – Balance amount**.** |
| **Post Conditions** | The Transaction details will be generated and stored in Transaction table, balance details will be updated in User table. |
| **Success End Condition** | On success, a message should be displayed “**Transaction Successfully Completed. The balance available in your Account is XXX”.**  (XXX should be balance after the transaction is complete) |
| **Failed End Condition** | The system should throw an error message if the validation is wrong. The error message and error code needs to be identified during design. The validation and business rules are mentioned in the business rules section. |
| **Steps & Actions** | * Customer clicks on Perform Transaction link. * Based on the amount entered, the Balance amount in master table is credited (for transaction type ‘Deposit’) or debited (for transaction type ‘Withdrawal’). * On Submit, the system needs to update the balance amount in User table and insert transaction details in Transaction table. |
| **Business Rules & Validations** | * Transaction id should be automatically generated. It should be numeric of 10 digit. * Account number should be a numeric of 16 digits. * Transaction type should be Deposit, Withdrawal. * While withdrawing the amount check the balance amount in User table is greater than the withdrawal amount. * In case of withdrawal, minimum balance to be maintained as per below table.  |  |  | | --- | --- | | **Account Type** | **Minimum Balance** | | Saving | 5000 | | Salary | 0 |  * If the account type in User table is salary account then the balance zero is allowed otherwise the Error message stating “Remaining balance must be greater than the   5000 Rupees” needs to be displayed.   * Use Spring framework to design the requirement. Use hibernate as persistence layer to persist the data to DB. |

### View Transactions

|  |  |
| --- | --- |
| Bank Management System  Req-2.1.3 | Customer – View Transactions  **Skills to be used: HTML, JSP/Servlet, REST WebServices, JSON, SQL.** |
| **Functional Requirements** | Customer should be able to view transaction details.  Any given point of time, the Customer should have the option to go for home page, by clicking home. |
| **UI Interface Screen** | **View Transaction details Screen:**  Generate all values using database.  This screen should capture the following fields-   * A/c Number – Represents Account number of the A/c holder. * Transaction id – Represents the Transaction id. * View button   **View Statements Screen:**  Customer transactions should be rendered in a table format with below columns.   * A/c Number, Transaction id, Transaction Type, Description, Balance |
| **Trigger** | Customer triggers the functionality of generating their transaction details. |
| **Pre-Conditions / Assumptions** | None. |
| **Post Conditions** | The customer should be able to view Transaction details. |
| **Success End Condition** | * Display the Transaction details in a table format. |
| **Failed End Condition** | If the customer’s search criteria do not match with the details stored in the system a message will be displayed to the user “**No records found.**” |
| **Steps & Actions** | * Display screen with search criteria. * All details should be retrieved from database. |
| **Business Rules and Validations** | **Business Rules & Validations for viewing statement:**   * At least one field in search criteria is required. * Account number should be a numeric of 16 digits. * Write web service which takes search criteria as input and return required transaction details from database. Use JSON format to transfer the results. |

### Apply Home Loan

|  |  |
| --- | --- |
| Bank Management System  Req-2.1.4 | Apply Home Loan |
| **Functional Requirements** | When the Customer logs in, he should be able to apply for Home loan.  Any given point of time, the Customer should have the option to go for home page, by clicking home. |
| **User Interface Requirements** | This should capture the following fields   * A/c Number – Represents Account number of the A/c holder. * Loan Amount – Represent the Loan Amount * Loan Apply Date – Represent the Loan Apply Date. * Duration of the loan - Represents the tenure for which the loan is applied for. * Annual Income – Represents the Annual Income * Company Name – Represents the Company Name * Designation – Represents the Designation * Total Exp – Represents the Total Experience * Exp with Current company – Represents the Experience with Current company |
| **Trigger** | Registered users trigger the functionality once he/she needs loan from the bank. |
| **Pre-Conditions / Assumptions** | Create table **USER\_DETAILS** and **HOME\_LOAN\_DETAILS** as per theDatabase Designgiven in the **System Design document (Refer section 5.0).**  Populate data in **USER\_DETAILS** table using insert query to insert user details. |
| **Post Conditions** | The loan details will be generated and stored in **HOME\_LOAN\_DETAILS** table. |
| **Success End Condition** | On success a message should be displayed “**Loan applied successfully. Your Loan Account number is XXXX”.**  XXXX is 16 digit Loan account number which is generated automatically. |
| **Failed End Condition** | The system should throw an error message if the validation is wrong. The error message and error code needs to be identified during design. The validation and business rules are mentioned in the business rules section. |
| **Steps & Actions** | * System should display all the necessary fields in the screen. * System needs to store the loan details into **HOME\_LOAN\_DETAILS** table and return the Loan Account Number. |
| **Business Rules & Validations** | * All fields are mandatory. * The Home Loan id should be automatically generated by the system and format should be ‘HL-XXX’. XXX should be last 3 digits of the A/c number. * Loan Account number will be generated automatically and should be a numeric of 16 digits. * Loan apply date should not be lesser than system date. * Loan should be sanctioned only if the A/c number entered is valid account number that is present in database (**USER\_DETAILS** table**)**. * Loan should be sanctioned only if annual income of the father is 10% of Loan amount. * Duration of Loan can be 5,10,15,20 years only. |

### Get Home Loan details

|  |  |
| --- | --- |
| Bank Management System  Req-2.1.5 | Customer – Get Home Loan details  **Skills to be used: HTML, JSP/Servlet, REST WebServices, JSON, SQL.** |
| **Functional Requirements** | Customer should be able to view home loan details.  Any given point of time, the Customer should have the option to go for home page, by clicking home. |
| **UI Interface Screen** | **Get Home Loan details Screen:**  Generate all values using database.  This screen should capture the following fields-   * Loan ID – Represents Home Loan Id. * Loan Account number – Represent the Loan Account number. * Get Loan Details button.   **Get details Screen:**  Customer Loan details should be rendered in a table format with below columns.   * Loan Account number, Account Holder Name, Loan ID, Loan Amount, Designation and Company Name. |
| **Trigger** | Customer triggers the functionality of generating their transaction details. |
| **Pre-Conditions / Assumptions** | None. |
| **Post Conditions** | The customer should be able to view Transaction details. |
| **Success End Condition** | * Display the Loan details in a table format. |
| **Failed End Condition** | If the customer’s search criteria do not match with the details stored in the system a message will be displayed to the user “**No records found.**” |
| **Steps & Actions** | * Display screen with search criteria. * All details should be retrieved from database. |
| **Business Rules and Validations** | **Business Rules & Validations for viewing statement:**   * At least one field in search criteria is required. * Loan Account number should be a numeric of 16 digits. * Loan id should be in format ‘HL-XXX’. XXX numeric of 3 digits. * Write web service which takes search criteria as input and return required transaction details from database. Use JSON format to transfer the results. |

### Apply Educational Loan

|  |  |
| --- | --- |
| Bank Management System  Req-2.1.6 | Apply Educational Loan |
| **Functional Requirements** | When the Customer logs in, he should be able to apply for Educational loan.  Any given point of time, the Customer should have the option to go for home page, by clicking home. |
| **User Interface Requirements** | This should capture the following fields   * A/c Number – Represents Account number of the A/c holder. * Loan Amount – Represent the Loan Amount * Loan Apply Date – Represent the Loan Apply Date. * Duration of the loan - Represents the tenure for which the loan is applied for. * Course Fee – Represents the Course Fee * Course name – Represents the Course name * Father Name – Represents the Father’s Name * ID No – Represents the Identification Card Number * Annual Income – Represents father’s Annual Income |
| **Trigger** | Registered users trigger the functionality once he/she needs loan from the bank. |
| **Pre-Conditions / Assumptions** | Create table **USER\_DETAILS** and **EDU\_LOAN\_DETAILS** as per theDatabase Designgiven in the **System Design document (Refer section 5.0).**  Populate data in **USER\_DETAILS** table using insert query to insert user details. |
| **Post Conditions** | The loan details will be generated and stored in **EDU\_LOAN\_DETAILS** tableand return the Loan Account Number. |
| **Success End Condition** | On success a message should be displayed “**Loan applied successfully. Your Loan Account number is XXXX”.**  XXXX is 16 digit Loan account number which is generated automatically. |
| **Failed End Condition** | The system should throw an error message if the validation is wrong. The error message and error code needs to be identified during design. The validation and business rules are mentioned in the business rules section. |
| **Steps & Actions** | * System should display all the necessary fields in the screen. * System needs to store the loan details into **EDU\_LOAN\_DETAILS** table and return the Loan Account Number. |
| **Business Rules & Validations** | * All fields are mandatory. * The Educational Loan id should be automatically generated by the system and format should be ‘EL\_XXXXX’. XXXXX should be first 5 digits of the ID No. * Loan Account number will be generated automatically and should be a numeric of 16 digits. * Father name should contain only alphabets and space. * Duration of Loan can be 5 or 10 years only. * Course Fee cannot be greater than 20 lacks. |

### Get Education Loan details

|  |  |
| --- | --- |
| Bank Management System  Req-2.1.7 | Customer – Get Educational Loan details  **Skills to be used: HTML, JSP/Servlet, REST WebServices, JSON, SQL.** |
| **Functional Requirements** | Customer should be able to view Education loan details.  Any given point of time, the Customer should have the option to go for home page, by clicking home. |
| **UI Interface Screen** | **Get Educational Loan details Screen:**  Generate all values using database.  This screen should capture the following fields-   * ID No – Represents the Identification Card Number * Loan Account number – Represent the Loan Account number. * Get Loan Details button.   **Get Loan details Screen:**  Customer Loan details should be rendered in a table format with below columns.   * Account Holder Name, Loan ID, Loan Amount, Couse Name, Designation and Father Name. |
| **Trigger** | Customer triggers the functionality of generating their transaction details. |
| **Pre-Conditions / Assumptions** | None. |
| **Post Conditions** | The customer should be able to view Transaction details. |
| **Success End Condition** | * Display the Loan details in a table format. |
| **Failed End Condition** | If the customer’s search criteria do not match with the details stored in the system a message will be displayed to the user “**No records found.**” |
| **Steps & Actions** | * Display screen with search criteria. * All details should be retrieved from database. |
| **Business Rules and Validations** | **Business Rules & Validations for viewing statement:**   * At least one field in search criteria is required. * Loan ID should be in EL\_XXXXX format. XXXXX should be numeric of 5 digits. * Write web service which takes search criteria as input and return required transaction details from database. Use JSON format to transfer the results. |

## Nonfunctional Requirements

### UI Requirements: Inn

1. The front end should be user-friendly and pleasant.
2. Any error message or exception displayed to the user should be user-readable (and not technical).
3. All entered values should be validated.
4. The UI screens can be designed with any theme color along with the logo of the company.
5. Exceptions should not be shown on console.
6. No debug message should come on console.
7. All the errors and exceptions should be logged to log file.
8. Log file path should be configurable.
9. Database connections should be configurable. They should not be hard coded.

# Requirements / Use Case Summary Table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Business Requirement ID** | **Software Requirements ID** | **Short Description** | **Requirement Provider (Originator)** | Priority | **Complexity** | **Requirement type** | **Remarks** |
| BR001 | SR001 | Perform Transaction | Client | High | Medium | UI screen & backend processing | - |
| BR002 | SR002 | View Transactions | Client | Medium | Medium | UI screen & backend processing | - |

# Make /Buy analysis

## Reusable components

The BMS can be made reusable as such to handle the performing transactions and generating statements as it forms the base for other activities involved in banking system.

# Annexure

NA