**Global Bank User Management Application**

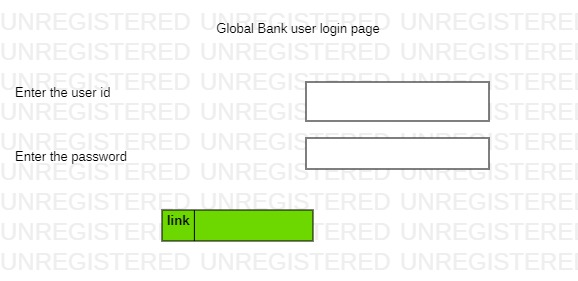
**Description:**

Global Bank allows bank customers to log in and performs simple bank transactions. This application will allow performing the below operations:

1. As a user he can register him/her selves to the application
2. As a user, he/she can be able to apply for a loan in a specific branch.
3. As a user he/she can be able to perform withdraw or deposit amount
4. As a user can be able to view all his / her selected transactions ie., withdraw or deposit for a specific period.
5. Display all transactions for a specific period

**Proposed Wireframes**:

1. A standard login screen to validate user id (customer\_number) and password



1. Screen for user operations menu

Table

Description automatically generated with low confidence

1. Screen for users to register themselves

Table

Description automatically generated

1. Screen for Applying Loans

Graphical user interface, text, application

Description automatically generated

1. Transaction management

Graphical user interface, text, application

Description automatically generated

1. View Statement

Graphical user interface, text, application

Description automatically generated

**3. Toolchain**

Databases: MySQL

Presentation or View Layer: React

Backend processing: Spring and Springboot

Database Connectivity: Spring data JPA

Version control systems: Git

Build Tools: Maven

**Development Flow**

The application development should be completed in 40 hours, as per the below order

Phase -1 : Backend Development: Backend Tasks – Code Project panel presentation

Phase -2: Frontend Development: Frontend Tasks – Code project panel presentation

**Business Requirement:**

There will be 5 main user stories required to be implemented covering the below use cases:

**User Stories**

|  |  |
| --- | --- |
| User story Id | Us-01 |
| User story title | User Login |
| User Story Details | 1. User should provide the user id(customer\_number) and password for validation. |
| Acceptance Criteria | 1. Both username and password are mandatory, if not provided, error messages should be displayed. 2. Successful validation should redirect to menu page, unsuccessful validation should redirect to registration page. |

|  |  |
| --- | --- |
| User story Id | Us-02 |
| User story title | User Menu |
| User Story Details | Display the user/customer menu |
| Acceptance Criteria | Null |

|  |  |
| --- | --- |
| User Story Id | Us\_03 |
| User Story Title | User registration |
| User story Details: | 1. As a new user he/she should be able to register them selves by providing below details: 2. Firstname 3. Lastname 4. Middlename 5. Customer\_city 6. Customer\_contact-no 7. Occupation 8. Customer\_date\_of\_birth |
| Acceptance Criteria | 1. Front end is stable, neat and user friendly 2. All fields are mandatory and not filled respective error messages should be displayed. 3. Upon successful insertion of data, it should redirect to login page. |

|  |  |
| --- | --- |
| User\_Story id | Us\_04 |
| User story Title | Apply for Loan |
| User Story Details: | As a registered user, he / she should be able to apply for a loan and enter the loan amount. User should be able to provide the following details for the loan:   1. Enter the loan amount 2. Select the serving branch |
| Acceptance Criteria | Upon selecting this apply for loan page,   1. Customer number should be populated in the page. 2. Branch must be selected from the pop menu only. 3. Max loan amount is 100000 only and no negative values are accepted |

|  |  |
| --- | --- |
| User\_Story\_id | Us\_05 |
| User Story Title | Transaction Management |
| User Story Details | As a registered user, he or should be able to perform the withdraw and deposit amounts. Here deposit is logical no physical currency is used for this purpose.  User should be able to provide the following details:   1. Enter the amount 2. Select the type of transaction like withdraw or deposit |
| Acceptance Criteria: | Upon redirecting to this page, the following validations are to be enforced:   1. Customer account details should be populated 2. Date of application should be displayed as per the system date 3. Transaction id must be generated incrementally and automatically 4. Transaction type must be selected from a popup menu, and it is only either withdraw or deposit. |

|  |  |
| --- | --- |
| User Story Id | Us\_06 |
| User Story Title | View Transactions |
| User Story Details: | As a registered user, he/she should be able to view all his or her transactions like withdrawals and deposits:   1. Enter the transaction type 2. Select the period of transaction |
| Acceptance Criteria: | Upon redirecting to this page, the following validations should keep in place:   1. Customer account details should be populated. 2. Date of the transaction should be generated automatically from the system date. 3. Medium of the transaction should be selected from drop-down menu. 4. Transaction type also should be selected from drop down menu and it should be either deposit or withdraw only. |

Note: A separate service component must be created to call the spring boot backend services and all the validations or processing regarding the use case should be done at the backend only.

**Backend Layer Userstories**

|  |  |
| --- | --- |
| User story Id | Us-01 |
| User story title | User Login processing |
| User Story Details | 1. Should be able to extract the values from request body using @RequestBody 2. Read the user details from database using spring data jpa and validate it with the UI values. 3. After validating should send response to React UI 4. Must use GET Method of communication |
| Acceptance Criteria | 1. Once user validation is done, view must return the main menu in react 2. If validation fails, view must return to registration page. 3. All validations must be performed at backend only |

|  |  |
| --- | --- |
| User story Id | Us-02 |
| User story title | User Registration processing |
| User Story Details | 1. Should be able to extract the values from request body object 2. Read the user registration values from UI and pass it to service layer further to dao layer to perform insertion of record in database 3. Should return the model object after successful insertion of data 4. Must use POST method of communication |
| Acceptance Criteria | 1. Once user validation is done, view must return the main menu 2. If validation fails, view must return to registration page. |

|  |  |
| --- | --- |
| User story Id | Us-03 |
| User story title | Loan Application |
| User Story Details | 1. Should be able to extract the values from request body object 2. Read the loan application details from UI and pass it to service layer further to dao layer to perform insertion of record in database 3. Should return the Boolean object after successful loan submission 4. Must use POST method of communication |
| Acceptance Criteria | 1. Loan application can be accepted from the account holders only 2. Branch must be selected from the drop list only |

|  |  |
| --- | --- |
| User story Id | Us-04 |
| User story title | Transaction Management |
| User Story Details | 1. Should be able to read transactional ie user account\_number, transaction\_type and period details from UI 2. Search from the transactional\_details database based on the values read from the UI 3. Should return all transaction data |
| Acceptance Criteria | 1. If from is ahead of to date it should throw exception and must be handled. 2. Transaction type must be either withdraw or deposit |

**Database Layer**

|  |  |
| --- | --- |
| User story | User Story Details |
| Us\_01 | 1. DB Schema creation and setup in mysql database 2. Spring boot project setup creation. 3. Develop the post method api to read data from view page. 4. Use appropriate DTO objects for view and data integration 5. Use spring data jpa for connecting to databases. |
| Us\_02 | 1. Set up the appropriate methods to perform functions like user validation, user registration, transaction management, transaction details. |

**Presentation:**

1. No custom CSS, UX framework like bootstrap must be used
2. An Appropriate GoF design pattern should be implemented to compose and process the data received from backend APIs
3. SOLID principles should be implemented to develop reusable and modular components
4. UI app should have appropriate client-side validations
5. UI app should have the latest versions of available imported packages and libraries

**Methodology:**

Agile-based development methodology should be used to track and manage the progress of the whole process. As a developer, it is expected to update the Agile tools like JIRA with status updates and impediments (Optional)

Day wise plan for user stories

|  |  |
| --- | --- |
| Day -1 | Database Layer Us\_01 , Us\_02 , Frontend US\_01 |
| Day-2 | Backend US\_01 and Frontend Us\_02 |
| Day-3 | Frontend US\_03, Frontend US\_04, Backend Us\_02 and Backend US\_03 |
| Day-4 | Frontend US\_05 Frontend US\_06  Backend Us\_04 |
| Day- 5 | Unit test cases, testing and ppt preparation. |