System Requirements Specification Index

For

e-Loan

Application

Version 1.0

IIHT Pvt. Ltd.

IIHT Ltd, No: 15, 2nd Floor, Sri Lakshmi Complex, Off MG Road, Near SBI LHO,

Bangalore, Karnataka – 560001, India

interest@iiht.com

**E-LOAN APPLICATION**

**System Requirements Specification**

**Project Abstract**

**E-Loan** Application is simple Spring boot RESTful application with MySQL, where it allows customers to apply for Loan Online, and same would be processed by loan clerk and sanctioned by manager from bank.

**Following is the requirement specifications**:

|  |  |
| --- | --- |
|  | E-LOAN |
|  |  |
| USERS |  |
| 1 | Admin |
| 2 | Manager |
| 3 | Loan Clerk |
| 4 | Customer |
|  |  |
| Admin Functionalities |  |
|  |  |
| 1 | Can create new Manager and Loan Clerk users and assign roles |
| 2 | Can get the list of all registered loan clerks and managers |
|  |  |
|  |  |
| Customer Functionalities |  |
| 1 | Can register itself |
| 2 | Can apply for Mortgage loan on a property |
|  | *While applying fo loan, following information is required* |
|  | *a. Loan Name (Description)* |
|  | *b. Loan Application number (Auto generated)* |
|  | *c. Loan Amount requested* |
|  | *d. Loan Application Date* |
|  | *e. Business structure (Individual/Organization)* |
|  | *f. Billing Indicator (Salaried person or not)* |
|  | *g. Tax indicator (Tax payer or not)* |
|  | *h. Contact Details (Contact Address, mobile, email)* |
| 3 | Can track status of Loan Application |
| 4 | Can get the details of all the loans applied by a particular customer |
|  |  |
| Loan Clerk Functionalities |  |
| 1 | Can list all Loan Application to be processed |
| 2 | Can Process Loan Application After Receiving and physical verification with following information added to it |
|  | a. Acres of land |
|  | b. Land Values in Rs |
|  | c. Appraised By (Name of third party appraiser) |
|  | d. Valuation Date |
|  | e. Address of property |
|  | f. Suggested amount of loan that can be released on property |
|  |  |
|  |  |
| Manager Functionalities |  |
| 1 | Can list all Loan Application processed by Loan clerk |
| 2 | Can Accept or reject a loan application ( with remarks) |
| 3 | If Loan is sanctioned, following information should be furnished |
|  | *a. Loan Amount Sanctioned* |
|  | *b. Term of loan ( Duration )* |
|  | *c. Payment start Date* |
|  | *d. Loan closure Date (Auto calculated from Payment start date and Term)* |
|  | *e. Monthly payment (Calculated)* |
|  | *Formulae:* |
|  | *a. Term payment amount = (Sanctioned loan amount ) \* (1 + interest rate/100) ^ (term of loan)* |
|  | *b. Monthly payment = (Term payment amount ) / (Term of loan)* |
|  |  |

**Software Requirements:** Spring Boot, Restful Web Services, MySQL, Postman.

**Assumptions, Dependencies, Risks / Constraints**

* While applying loan by customer, if customer id does not exist then operation should throw custom exception.
* While fetching loan status, if loan application id does not exist then operation should throw custom exception.
* While fetching all loan details of a customer, if customer id does not exist then operation should throw custom exception.
* While processing loan by loan clerk, if clerk id does not exist then operation should throw custom exception.
* While processing loan by loan clerk, if loan application id does not exist then operation should throw custom exception.
* While processing loan by loan clerk, if loan has already been processed then operation should throw custom exception.
* While sanctioning or rejecting loan by manager, if manager id does not exist then operation should throw custom exception.
* While sanctioning or rejecting loan by manager, if loan application id does not exist then operation should throw custom exception.
* While sanctioning or rejecting loan by manager, if loan has already been rejected or sanctioned then operation should throw custom exception.
* For all rest endpoints receiving @RequestBody, validation check must be done and must throw custom exception if data is invalid
* All the business validations must be implemented on DTO object only
* All the database operations must be implemented on entity object only
* The conversion from entity to dto and vice-versa can be done in the service layer.
* Do not change, add ,remove any existing methods in controller and service layer
* In service layer, private utility methods can be added for conversions or reusability functionality.
* In Repository interfaces, custom methods can be added as per requirements.

**URL’s exposed in the controller**

**AdminController**

|  |  |
| --- | --- |
| **URL Exposed** | **Purpose** |
| /admin/register-clerk | Register a loan clerk |
| /admin/register-manager | Register a manager |
| /admin/all-clerks | Fetches the list of all registered loan clerks |
| /admin/ all-managers | Fetches the list of all registered managers |

**CustomerController**

|  |  |
| --- | --- |
| **URL Exposed** | **Purpose** |
| /customer/register | Register a new user |
| /customer/apply-loan/{customerId} | Apply for loan |
| /customer/loan-status/{loanAppId} | Fetches details of applied loan |
| /customer/loan-status-all/{customerId} | Fetches details of all loans applied by a customer |

**ClerkController**

|  |  |
| --- | --- |
| **URL Exposed** | **Purpose** |
| /clerk/all-applied | Fetches list of applied loans not yet processed |
| /clerk/process/{clerkId}/{loanAppId} | Process the loan to forward to Manager |

**ManagerController**

|  |  |
| --- | --- |
| **URL Exposed** | **Purpose** |
| /manager/all-processed | Fetches list of applied loans processed by loan clerk |
| /manager/reject-loan/{managerId}/{loanAppId} | Reject loan application by providing remark |
| /manager/sanction-loan/{managerId}/{loanAppId} | Sanction loan by providing sanction info |

**Business Validations**

* User firstName is not null, min 3 and max 100 characters.
* User lastName is not null, min 3 and max 100 characters.
* User email is not null, min 3 and max 100 characters and in proper email format.
* User mobile is not null, min 10 and max 10 characters.
* LoanName is not null, min 3 and max 100 characters.
* LoanAmount is not null, and not 0;
* acresOfLand is not null, and not 0;
* landValue is not null, and not 0;
* suggestedAmountOfLoan is not null, and not 0;
* addressOfProperty is not null, min 3 and max 150 characters;
* loanAmountSanctioned is not null, and not 0
* termOfLoan is not null, and not 0.

**Considerations**

1. For Role of Users three possible values must be used
2. Customer
3. Clerk
4. Manager
5. For status of loan 4 possible numeric values must be used

|  |  |
| --- | --- |
| 0 | Applied |
| 1 | Processed |
| 2 | Sanctioned |
| -1 | Rejected |

1. For Database configuration customize the following keys in application.properties file

*spring.datasource.username=root*

*spring.datasource.password=*

**Note**

1. **Need to implement ONLY controller and service methods in**
2. **DO NOT ADD ANY method or CHANGE any Rest End-Point pattern.**