HOME LOAN SYSTEM

CONTENTS

1. PROBLEM STATEMENT

Bank Savings Account Holder will apply for home loan with the bank. Following information will be provided at the time of application. Description / Address of property for which loan is required, loan amount, tenure in years (5-20), Net Monthly Salary. Also upload image of property document.

- **Loan Approval** - The system will approve loan amount up to maximum of 50 times Net monthly Salary. Mail will be sent to account holder informing about sanction of loan and monthly EMI (calculation given below). Loan Status becomes Approved.
- If requested loan amount is higher than eligible amount then account holder will have option of cancelling loan application OR going with the eligible loan amount (Loan Status = Approved). If account holder cancels loan application no Loan Account record will be created.
- When loan is approved a new Loan Account be created linking to the account holder savings account. It will also populate Loan Repayment Schedule as given below (Status = Pending). Loan interest rate will be taken as 7% PA.
- Its assumed that loan is disbursed immediately and EMI will be applicable from next month.
- EMI = [P x R x (1+R)^N]/[(1+R)^N-1] where P is the approved / eligible loan amount, R rate of interest, N is the tenure of the loan. Since we are paying monthly installments Annual Interest Rate is divided by 12, N becomes number of years x 12 months.
- EMI will be deducted from Savings Account balance every month. Its assumed account has sufficient funds to pay the EMI, paid amount will be updated in Loan Repayment Schedule. After deduction the status of loan account repayments schedule for the month will be set to Paid. Loan Account Status becomes On Going.
- After payment is deducted mail will be sent to account holder that EMI has been deducted.
- E.g. So, assuming that you take a home loan of Rs 50 lakh with an interest rate of 12% PA for a tenure of 10 years, the approximate EMI will be:

  o P = Rs 50 lakh, R = 12/100/12 or 0.01 (convert to months), N = 10 years or 120 months

  o EMI = [5000000 x 0.01 x (1+0.01)^120] / [(1+0.01)^120-1]

  o EMI = Rs 71,735.

- Once the loan is disbursed monthly repayment schedule (for whole tenure in months) will be calculated stating the principal and interest component for each EMI payment. The interest component of the EMI for the month is balance outstanding for the month x Monthly Rate of interest. Interest rounded off to rupees.

  o EMI 71735 (As above)

  o Month 1 – Interest = Balance outstanding (5000000) x 0.12 / 12 = 50000. Principal = 71735 – 50000 = 21735 o Month 2 - Interest = Balance outstanding 5000000 – 21735 ((cumulative principal paid) = 497265. Interest = 497265 x 0.12 / 12 = 49783. Principal = 71735 – 50000 = 21953

  o For tenure of 10= years = 120 month monthly repayment schedule will be created for each 120 months stating EMI, Principal Paid for the month, Interest for the month, Balance (Principal Outstanding), Paid Amount (initially zero for all), Status (Pending, Paid, Cancelled)

- Loan Foreclosure and Prepayment.

  o Account Holder can close the loan (after minimum 3 EMI have been deducted) by making full payment of the outstanding amount for that month. Monthly payment schedule for the closure month will change to reflect the paid amount = full closure amount. Subsequent months repayment schedule will get status of Cancelled. Loan Account Status becomes Closed, Principal , Interest will be set to zero.

  o Account Holder can make partial prepayment of loan of minimum 3 times the monthly EMI. In this case paid amount for the prepayment month will be adjusted against outstanding balance. Then subsequent Prepayment schedule & New EMI, Principal, Interest, outstanding from prepayment month onwards recalculated on the new outstanding amount with other parameters remaining same.

Functionalities:

This software will have following functionalities

- Online loan approval and denial status
- Customer will be able to check his/her loan approval amount online while sitting at home by accessing the database of the bank using his/her password and account no. allotted him by the bank.
- Save or view up to 1 year past history of transaction
- It will be easy for the customer to view the updates about loan amount.
- Customer can re-apply the loan amount in case a bigger amount is rejected by the loan officer.

Loan Status:

Loan Status should be checked online at home

Online record Entry:

Bank staff will input and maintain their record online. It will be easy and efficient for them to serve more and more people in less time.

Online record search:

Bank staff will easily search a record and update it if needed. Transactions will be faster even physically from the branch because it will be very easy for the bank staff to check the balance of a specific person and update its record if necessary.

Online Emi option:

Customers will be able to pay EMI every month online.

.

User Characteristics:

There are various kinds of users for the App. Usually web apps are visited by various users for different reasons.

The users include:

- Chancellor who will be acting as the controller and he will have all the privileges of administrator.

- All the persons who needs to perform banking.

Generals Constraints:

Some general constraints should be defined which will have a great part in the overall succession of the online banking project.

Hardware Requirements:

As this system is an online Web-based application so a client server will be the most suitable Organizational style for this system. Computer systems will be needed by each of the actor as well as that user must be connected to the internet. So, concisely following hardware will be needed.

- Computer systems

- Internet availability

Safety and Security:

This Project must be safe and secure because customers will directly contact their account through the internet. Software will have to identify the valid customer according to his/her bank details and password. So it is a difficult task to prevent the system by major disasters by preventing the unauthorized access to the system.

Assumptions and Dependencies:

Following are the assumptions and dependencies which are related to this Home Loan App project.

- This project is a stand-alone project so it will not affect the system where it will be embedded.
- This system will not depend on any other module. It will be a web-based so everyone will independently contact it.

Specific Requirements:

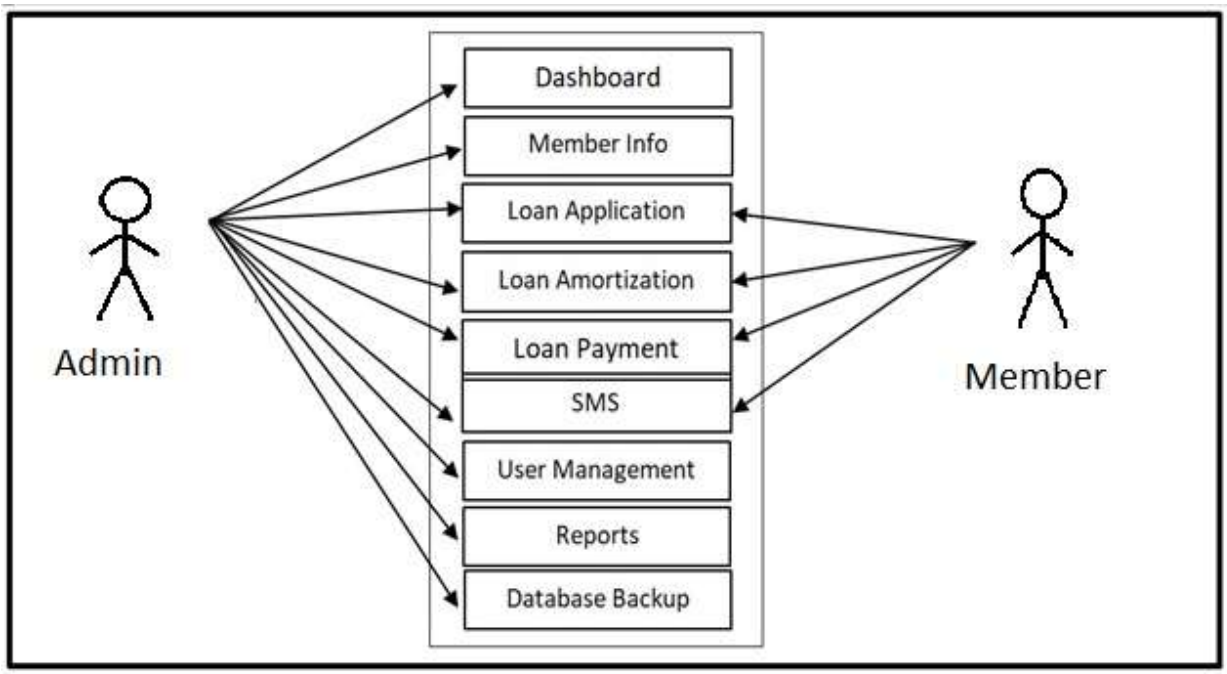
How the online banking will interact with the environment, what will be the functional and non-functional requirement. These all the steps should be defined here for providing a powerful base to the design phase. The design of the project will completely depend on the functional and non-functional requirements. So these should be defined clearly and accurately for effectiveness.

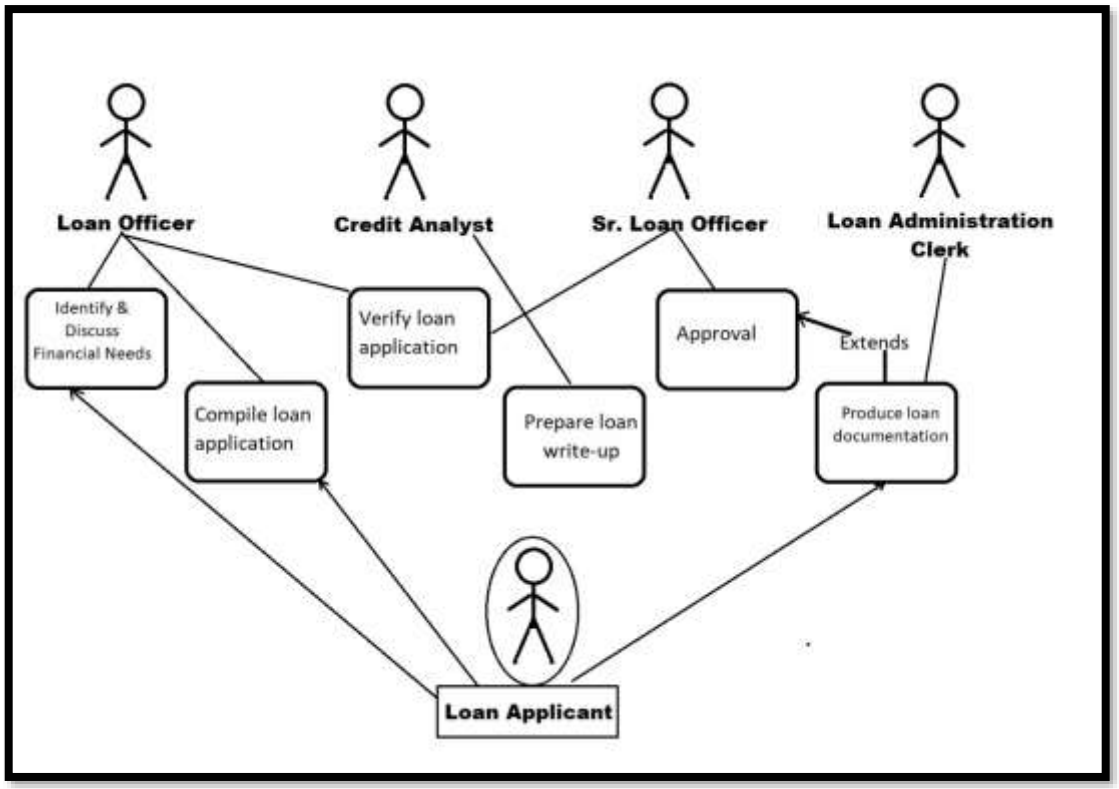
Functional Requirements:

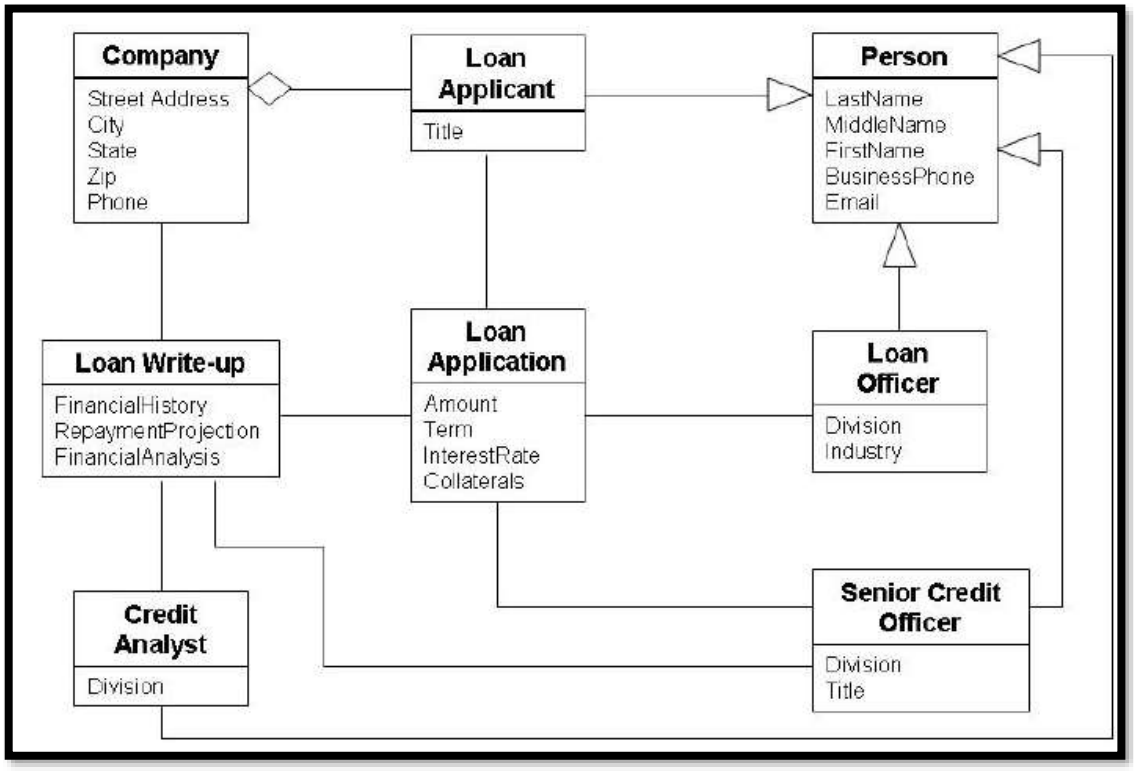
Following are the services which this system will provide. These are the facilities and functions required by the customer.

2.PROPOSED WIREFRAME

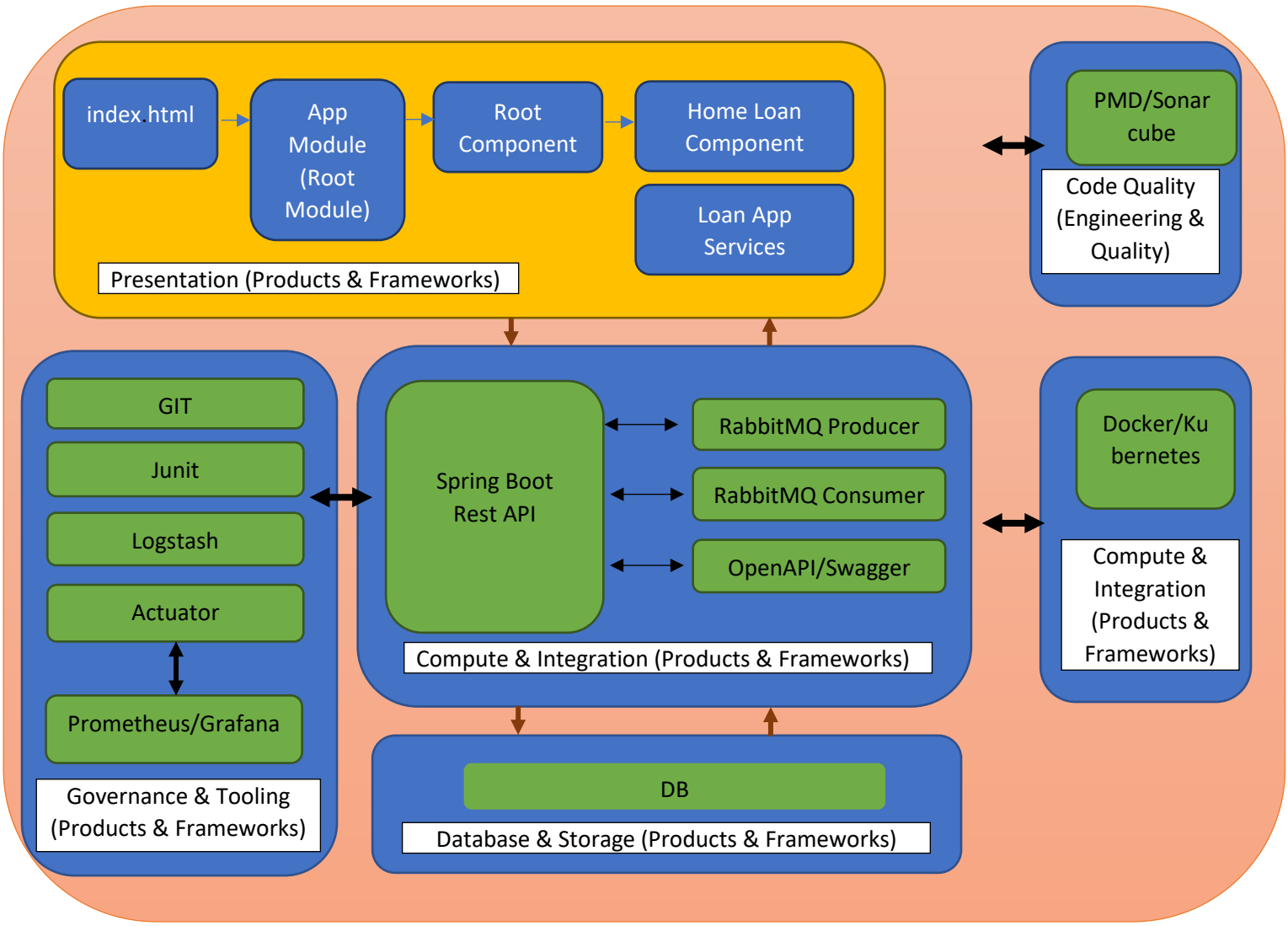
1. “Loan Management System with SMS” is an online platform that allows members to apply and request loan. In addition, they can also monitor their balance in their respective dashboard. Management of cooperative will review first the application for approval or disapproval of the request. Notification will be sent through the SMS or short messaging service feature of the system. The Use Case Diagram is a crucial tool for software engineers and stakeholders when it comes to developing a comprehensive and effective loan management solution. This type of diagram provides a visual representation of the interactions between different actors and the system being developed, making it easier to understand the different use cases, or scenarios, that the system will need to support. Whether you are a loan officer, a business owner, or a software developer, understanding the role of a LMS Use Case Diagram in the development of a loan management system is important.
2. UI needs improvisation and modification as per given use case.




Loan Officer
Credit Analyst
Sr. Loan Officer
Loan Administration Clerk
Identify & Discuss Financial Needs
Verify loan application
Approval
Extends
Compile loan application
Prepare loan write-up
Produce loan documentation
Loan Applicant

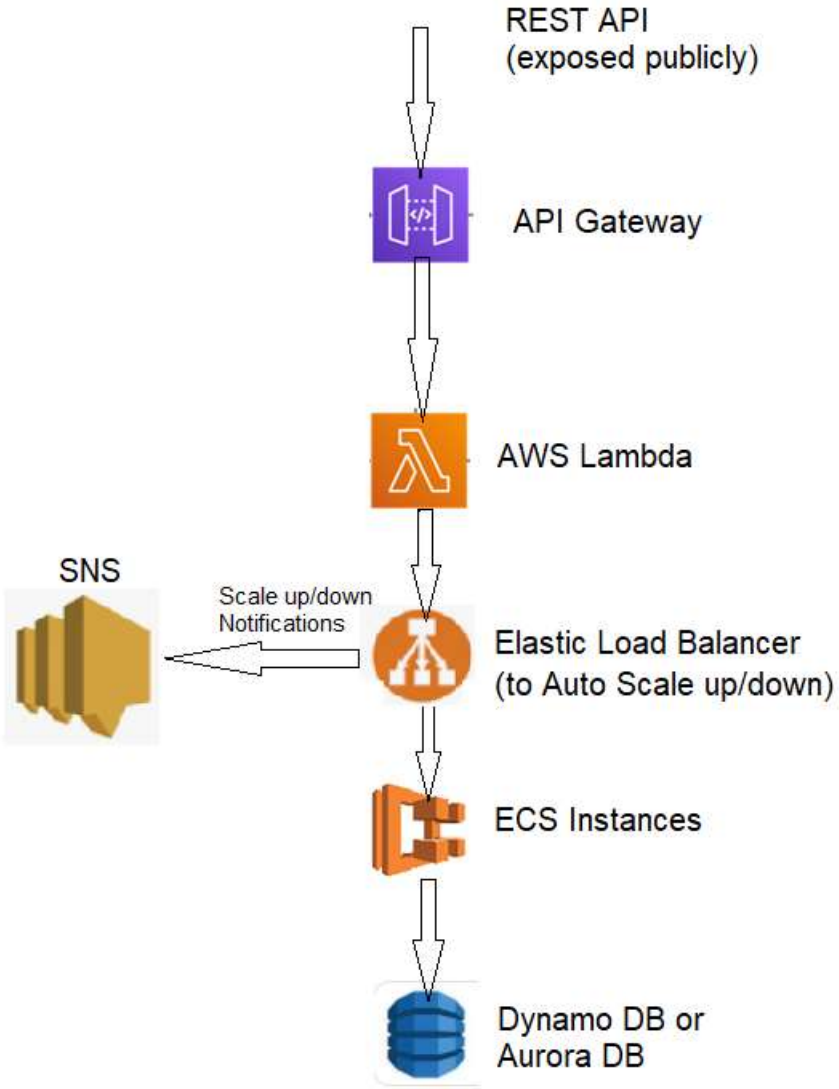

Company
Street Address
City
State
Zip
Phone
Loan Applicant
Title
Person
LastName
MiddleName
FirstName
BusinessPhone
Email
Loan Write-up
FinancialHistory
RepaymentProjection
FinancialAnalysis
Loan Application
Amount
Term
InterestRate
Collaterals
Loan Officer
Division
Industry
Credit Analyst
Division
Senior Credit Officer
Division
Title

3 APPLICATION ARCHITECTURE



4. CLOUD ARCHITECTURE

Below Diagram shows how Blog Site App can be deployed on AWS Cloud



5. TOOL CHAIN

Competency	Skill	Skill Detail
Engineering Mindset	Networking and Content Delivery	
	Ways of Working	
	Consulting Mindset	
	DevOps	
Programming Languages	Application Language	Java
Products & Frameworks	Presentation	Angular/ React & Redux

		Karma & Jasmine
	Compute & Integration	Spring Boot
		Spring Test/Selenium & Cucumber
		RabbitMQ
		Docker, Kubernetes
	Database & Storage	MongoDB
		Oracle SQL/PostgreSQL
	Governance & Tooling	Git
		Maven
		Junit
		Mockito
		Elastic Search/Logstash/Kibana
		Prometheus & Grafana
Engineering Quality	Code Quality	Sonar Cube
Platform	Cloud Tools	AWS ECS
		AWS DynamoDB/Aurora
		AWS Lambda
		AWS API Gateway
		AWS ELB(Elastic Load Balancer)
		AWS SNS

6. DEVELOPMENT FLOW

		Business Requirement		
1	**Backend**	Rest API, Database, Messaging, Log/Monitoring, Testing	**Code Submission and Evaluation, Panel Presentation**	It is mandatory to complete this MC with 60% to proceed with the next Frontend MC
2	**Front End**	Angular/React	**Code Submission and Evaluation, Panel Presentation**	It is mandatory to complete each MC with 60% to proceed with the next Cloud MC
3	**Cloud**	Compute, Identity, Compliance, Security and Content Delivery	**Code Submission and Evaluation, Panel Presentation**	It is mandatory to complete this Cloud MC with 60%

7.PROCESS-REQUIREMENT:

The following are the discussions that describe how a user uses a system to accomplish a particular goal.

- **Use Case: Dashboard**

Actor(s): Admin.
Description:

This feature is used to manage the details displayed in the dashboard.

Successful Completion:

1. The admin can search, add, update and remove dashboard details.

Alternative: The admin can access all dashboard details

Precondition: The admin will login to access and manage the dashboard

Post Condition: updated dashboard details

- **Use Case:** Member Info

Actor(s): Admin

Description:
This feature is used to manage the profile of the members registered in the system.

Successful Completion:

1. The admin can search, add, update and remove members information.

Alternative: The admin can access all members information

Precondition: New members for registration, existing members for updating

Post Condition: updated members information

- **Use Case:** Loan Application

Actor(s): Admin and Member

Description:

This feature is used to manage the loan application of members made in the system.

Successful Completion:

1. Customer can create and submit loan application.
2. Admin can search, add, update and remove loan applications by members.

Alternative: Customer can only submit and view updates of their loan application; Admin can access all of the members' loan applications.

Precondition: New application for submission and review, existing application for updating

Post Condition: accepted loan application

- **Use Case:** Loan Amortization

Actor(s): Admin and Member

Description:

This feature is used to manage the loan amortization for loans made by the members in the system.

Successful Completion:

1. Customer can view amortization for their loans.
2. Admin can search, add, update and remove a loan amortization detail using this feature.

Alternative: Customer can only view loan amortization; Admin can access all of the members loan amortization information.

Precondition: The admin and members will need to login first to access the module.

Post Condition: updated loan amortization

- **Use Case:** Loan Payment

Actor(s): Admin and Member

Description:

This feature is used to manage the payments for loans made by the members in the system.

Successful Completion:

1. Customer can pay and view their loan payments using this feature.
2. Admin can search, add, update and remove a loan payment details using this feature.

Alternative: Customer can only pay and view loan payments; Admin can access all of the members loan payment information.

Precondition: The admin and members will need to login first to access the module.

Post Condition: updated loan payment information

- **Use Case:** SMS & Email

Actor(s): Admin and Member

Description:

This feature is used to manage the SMS and E-mail in the system.

Successful Completion:

1. Customer can view SMS using this feature.
2. Admin can search, add, update and remove an SMS details using this feature.

Alternative: Customer can only view SMS; Admin can access SMS information.

Precondition: The admin and members will need to login first to access the module.

Post Condition: updated SMS

- **Use Case:** User Management

Actor(s): Admin

Description:

This feature is used to manage the users of the system.

Successful Completion:

1. The admin can search, add, update and remove user details using this feature.

Alternative: The admin can access all user details

Precondition: The admin will login to access and manage the users

Post Condition: updated users details

- **Use Case:** Reports

Actor(s): Admin

Description:

This feature is used to view and print the reports of the system.

Successful Completion:

1. Admin can view, print and export the report of the system.

Alternative: None

Precondition:

1. Admin will need to login to access the reports.

Post Condition: hard and soft copy of the report of the system.

- **Use Case:** Database Backup

Actor(s): Admin

Description:

This feature is used to manage the backup database of the system.

Successful Completion:

1. The admin can add, edit, and update database backup information.

Alternative: None

Precondition: Admin will create and connect the backup database.

Post Condition: new backup database.

User interface:

Application will be accessed through a Browser Interface. The interface would be viewed best using 1024 x 768 and 800 x 600 pixels resolution settings. The software would be fully compatible with Microsoft Internet Explorer for version 6 and above.

No user would be able to access any part of the application without logging on to the system.

8. RUBRICS/EXPECTED DELIVERABLES

a. REST API (PRODUCTS & FRAMEWORKS -> COMPUTE & INTEGRATION):

a. Use Spring Boot to version and implement the REST endpoints.

b. Implement HTTP methods like GET, POST, PUT, DELETE, PATCH to implement RESTful resources:

POST	**/api/v1.0/loan/register**	**Register as new loan applicant**
GET	**/api/v1.0/ loan/login**	**Login**
GET	**/api/v1.0/ loan /<username>/forgot**	**Forgot password**
GET	**/api/v1.0/ loan /domain/all**	**Get all loan requests on a particular domain**
POST	**/api/v1.0/ loan /<username>/add**	**Post new loan request**
PUT	**/api/v1.0/ loan /<username>/update/<id>**	**Update request**
DELETE	**/api/v1.0/ loan/<username>/delete/<id>**	**Delete request**
PUT	**/api/v1.0/ loan/<username>/modify/<id>**	**Modification in loan request**
POST	**/api/v1.0/ loan /<username>/reply/<id>**	**Reply on request added**

c. *username may be partial or complete username

d. Use necessary configuration in place for REST API in application.properties or bootstrap.properties or application.yml; whichever is applicable.

e. Package Structure for Spring Boot Project will be like com.homeloan.* with proper naming conventions for package and beans.

f. Use configuration class annotated with @Configuration and @Service for business layer.

g. Use constructor-based dependency injection in few classes and setter-based dependency injection in few classes.

h. Follow Spring Bean Naming Conventions

b. DATABASE (PRODUCTS & FRAMEWORKS -> DATABASE & STORAGE):

1. As an application developer:
   a. Implement ORM with Spring Data MongoRepository (or) use MongoDB. For complex and custom queries, create custom methods and use @Query, Aggregations (AggregationOperation, MatchOperation, AggregationResults), implementation of MongoTemplateetc as necessary.
   b. Have necessary configuration in place for REST API in application.properties or bootstrap.properties or application.yml OR Java based configuration; whichever is applicable.

c. MAVEN (TOOLING)/GRADEL:

1. As an application developer:
   a. Create the spring boot project using Maven CLI
   b. Using Maven CLI generate the project documentation, and share it as a part of deliverables

d. MESSAGING (PRODUCTS & FRAMEWORKS -> COMPUTE & INTEGRATION):

1. As an application developer:
   a. Have a centralized logging system
   b. Be able to communicate using a messaging infrastructure.
   c. Use RabbitMQ Template for communication with Springboot and topics in RabbitMQ.

d. Use RabbitMQ for messaging infrastructure and implement producers to write blogs/comment to blogs and consumers to read blogs/comments from domain.
e. Configure Springboot app to log all logging messages to RabbitMQ.
f. Configure all RabbitMQ related configuration needed for Spring Boot in *.properties or *.yml file.

e. LOG/ MONITORING (PRODUCTS & FRAMEWORKS -> GOVERNANCE & TOOLING):

1. As an application developer:

a. Containerize the complete application, which includes front-end, middleware and RabbitMQ (consumers and producers) using docker and Dockerfile (or) Kubernetes.
b. Use .dockerignore (or) Kubernetes as necessary to avoid containerizing un-necessary packages.
c. Integrate Spring Boot Actuator with Prometheus and Grafana to monitor middleware.
d. Implement logs with logstash.
e. Open the preconfigured Logstash in Kibana and check if it successfully connect to Elasticsearch Server.

f. TESTING

1. Perform proper testing using Spring Test/Selenium and Cucumber and do proper CI/CD
2. The test coverage should be of 100%
3. Test Suites should cover both positive and exception handlings

9.FRONTEND

1. Develop the front end for all user stories.
2. Implement using either Angular or React
3. Implement all the Front-End validation rules
4. Proper naming conventions and folder structures
5. Implement using proper SOLID design principles
6. Perform unit and integration testing for the front end application

10. PLATFORM

g. COMPUTE

1. Use ECS CLI (as an alternative to AWS Management Console) for container management and deployment of spring boot application.

h. COMPUTE, IDENTITY & COMPLIANCE, SECURITY& CONTENT DELIVERY

1. Use AWS Lambda and AWS Aurora to build a backend process for handling requests for Blog Site App.
2. Use Serverless Java Container using AWS ECS and run the blog site app created with Spring Boot inside AWS Lambda.
3. Use Amazon API Gateway to expose the lambda functions built in the previous step to be accessible on public internet.
4. Use AWS ELB to configure the auto-scaling container instances.
5. Configure AWS SNS to issue messages whenever a ELB scales-up and scale-down container instances

Note – Minimum two rest endpoints should be hosted in cloud

i. FRONTEND DEPLOYMENT

Deploy the Frontend solution as follows:

- Maintain the production build of Frontend application on S3 bucket
- Configure an EC2 instance to access Frontend artifacts from S3 bucket and expose it as dynamic web application
- Configure the S3 to cache
- Configure the Route 53 to register domain name to expose the Frontend solution.
- Ensure that Privacy Protection feature is enabled for Amazon Route 53 domain.

11.METHODOLOGY

j. AGILE

1. As an application developer, use project management tool along to update progress as you start implementing solution.
2. As an application developer, the scope of discussion with mentor is limited to:
    a. Q/A
    b. New Ideas, New feature implementations and estimation.
    c. Any development related challenges
    d. Skill Gaps
    e. Any other pointers key to UI/UX and Middleware Development