

**Tribhuvan University**

**Faculty of Humanities and Social Sciences**

**LOAN MANAGEMENT SYSTEM (SAHAKARI)**

**A PROJECT REPORT**

**Submitted to**

**Department of Computer Application**

**Ratna Rajyalaxmi Campus**

***In partial fulfillment of the requirements for the Bachelors in Computer Application***

Submitted by

Lokesh Raj Bhatt (6-2-59-217-2020)

08/2024

Under the supervision of

**Ananda KC**



**Tribhuvan University**

**Faculty of Humanities and Social Sciences**

**Ratna Rajyalaxmi Campus Pradarsanimarg Kathmandu**

**SUPERVISOR’S RECOMMENDATION**

I hereby recommend that this project prepared under my supervision by “Lokesh Raj Bhatt (6-2-59-217-2020)” entitled “**LMS**” in partial fulfillment of the requirements for the degree of Bachelor of Computer Application is recommended for the final evaluation.

-------------------------------------

Signature

Ananada KC

**SUPERVISOR**



**Tribhuvan University**

**Faculty of Humanities and Social Sciences**

**Ratna Rajyalaxmi Campus Pradarsanimarg Kathmandu**

**LETTER OF APPROVAL**

This is to certify that this project prepared by LOKESH RAJ BHATT entitled “**LMS**” in partial fulfillment of the requirements for the degree of Bachelor in Computer Application has been evaluated. In our opinion it is satisfactory in the scope and quality as a project for the required

|  |  |
| --- | --- |
| **Signature of Supervisor**    ---------------------------------    Ananda KC  Ratna Rajya Laxmi Campus | **Signature of HOD**    ----------------------------------------  Coordinator, Department of Bachelor in  Computer Application  Ratna Rajya Laxmi Campus |
| **Signature of Internal Examiner**    ----------------------------------------  Ratna Rajya Laxmi Campus | **Signature of External Examiner**    --------------------------------------- |

# ABSTRACT

The Loan Management System (Sahakari) is a comprehensive, web-based solution designed to digitize and streamline the loan management processes within cooperatives. This system addresses the inefficiencies of traditional paper-based methods, such as handwritten records, manual file systems, and the need for physical meetings to process loans and verify KYC details. By automating these processes, the system enhances data accuracy, reduces processing times, and improves the overall operational efficiency of cooperatives. The system supports multiple user roles, including members, loan officers, and administrator, each with specific functionalities to manage loan applications, approvals, disbursements. The Loan Management System (Sahakari) aims to provide a transparent, secure, and user-friendly platform that not only meets the current needs of cooperatives but also scales with their growth. This documentation outlines the design, implementation, and benefits of the system, providing a valuable resource for developers and end-users alike. Key features of the Sahakari system include role-based access for members, loan officers, and administrator, ensuring secure and efficient operations. Members can apply for loans, track their loan status, and make repayments online. Loan officers are equipped to review and approve loan applications, oversee disbursements, and manage collections. Administrators have comprehensive control over generating reports, monitoring system performance, and maintaining user accounts.

***Keywords:* Sahakari, KYC, LMS, administrator, Officers, Members, approvals, disbursements**

# ACKNOWLEDGEMENT

Apart from the efforts of ourselves, the success of any project depends largely on the encouragement and guidance of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project. I would like to express our deep sense of greatest appreciation to **Mr. Ananda KC.** I’m thankful of his tremendous support and help. We felt motivated and encouraged every time we attend project meetings. Without his encouragement and guidance, this project would not have materialized. The guidance and support received from all the members how contribute and who are contributing to this project, was vital for the success of project. I’m grateful for his consultant support and help.

# Table of Contents

[ABSTRACT iii](#_Toc178371166)

[ACKNOWLEDGEMENT iv](#_Toc178371167)

[Table of Contents v](#_Toc178371168)

[List of Abbreviations vii](#_Toc178371169)

[List of figures viii](#_Toc178371170)

[List of Tables ix](#_Toc178371171)

[Chapter 1: Introduction 1](#_Toc178371172)

[1.1 Introduction 1](#_Toc178371173)

[1.2 Problem Statement 1](#_Toc178371174)

[1.3 Objectives 2](#_Toc178371175)

[1.4 Scope and Limitation 2](#_Toc178371176)

[1.5 Report Organization 2](#_Toc178371177)

[Chapter 2: Background Study and Literature Review 3](#_Toc178371178)

[2.1 Background Study 3](#_Toc178371179)

[2.2 Literature Review 4](#_Toc178371180)

[Chapter 3: System Analysis and Design 5](#_Toc178371181)

[3.1 System Analysis 5](#_Toc178371182)

[3.1.1 Requirement Analysis 5](#_Toc178371183)

[3.1.2 Feasibility Analysis 8](#_Toc178371184)

[3.1.3 Object Modelling: 10](#_Toc178371185)

[3.1.4 Dynamic Modelling: 12](#_Toc178371186)

[3.1.5 Process Modelling: 14](#_Toc178371187)

[3.2 System Design 15](#_Toc178371188)

[3.2.1 Refinement of Classes and Objects: 15](#_Toc178371189)

[3.2.2 Component Diagram: 16](#_Toc178371190)

[3.2.3 Deployment Diagram: 17](#_Toc178371191)

[3.3 Algorithm details 19](#_Toc178371192)

[Chapter 4: Implementation and Testing 21](#_Toc178371193)

[4.1 Implementation of system 21](#_Toc178371194)

[4.1.1 Tools Used: 21](#_Toc178371195)

[4.1.2 Implementation details of modules: 22](#_Toc178371196)

[4.2 Testing 23](#_Toc178371197)

[4.2.1 Test Cases for Unit Testing: 23](#_Toc178371198)

[4.2.2 Test Cases for System Testing: 25](#_Toc178371199)

[Chapter 5: Conclusion and Future Recommendations 28](#_Toc178371200)

[5.1 Conclusion 28](#_Toc178371201)

[5.2 Lessons Learned/Outcome 29](#_Toc178371202)

[5.3 Future Recommendations 30](#_Toc178371203)

[References 31](#_Toc178371204)

[Appendices 32](#_Toc178371205)

# List of Abbreviations

|  |  |  |
| --- | --- | --- |
| LMS |  | Loan Management System |
| CSS |  | Cascade Style Sheet |
| HTML |  | Hyper Text Markup Language |
| UI |  | User Interface |
| UX |  | User Experience |
| MVC |  | Model View Controller |
| PHP |  | Hypertext Preprocessor |
| OTP |  | One Time Password |
| SQL |  | Structured Query Language |
| UML |  | Uniform Modelling Language |
| MySQL |  | My Structured Query Language |

# List of figures

[Figure 3. 1 Use Case diagram of LMS 5](#_Toc178365653)

[Figure 3. 3 : Class diagram of LMS (Sahakari). 10](#_Toc178365654)

[Figure 3. 4: Object diagram of LMS (Sahakari) 11](#_Toc178365655)

[Figure 3. 5: State diagram of LMS {Sahakari) 12](#_Toc178365656)

[Figure 3. 6: Sequence diagram of LMS. 13](#_Toc178365657)

[Figure 3. 7: Activity diagram of LMS (Sahakari) 14](#_Toc178365658)

[Figure 3. 8: Component diagram of LMS (Sahakari) 16](#_Toc178365659)

[Figure 3. 9: Deployment diagram of LMS (Sahakari). 17](#_Toc178365660)

# List of Tables

[Table 3. 1: Gantt Chart of LMS (Sahakari) 9](#_Toc178365943)

[Table 4. 1 Test Case 1-Registration Form 23](#_Toc178365946)

[Table 4. 2 Test Case 2-Login Form 24](#_Toc178365947)

[Table 4. 3 Test Case 3: End-to-End Member Verification and KYC Approval 25](#_Toc178365948)

[Table 4. 4 Test Case 4: End-to-End Loan Application approvement and Disbursement 26](#_Toc178365949)

[Table 4. 5 Test Case 5: Data Encryption and RBAC test 27](#_Toc178365950)

# Chapter 1: Introduction

## 1.1 Introduction

The Loan Management System (Sahakari) is designed to streamline and automate the process of managing loans within a cooperative or financial institution. The system provides a comprehensive solution for handling the entire lifecycle of a loan, from application to disbursement. The system is tailored to meet the specific needs of cooperatives, with features that support multiple user roles, including members, loan officers, and administrator.

The Loan Management System (Sahakari) is designed to streamline and automate the process of managing loans within a cooperative or financial institution. The system provides a comprehensive solution for handling the entire lifecycle of a loan, from application to disbursement, repayment tracking, and reporting. The system is tailored to meet the specific needs of cooperatives, with features that support multiple user roles, including members, loan officers, and administrators.

## 1.2 Problem Statement

The current loan management process in cooperatives is highly inefficient and prone to errors due to its reliance on paper-based records, handwritten documentation, and manual file systems. These outdated methods require members and staff to physically attend meetings for loan applications, approvals, and KYC verifications, leading to delays, increased workload, and potential inaccuracies. The lack of a centralized, automated system hampers the cooperative's ability to manage loans effectively, impacting overall operational efficiency and member satisfaction The lack of a centralized system makes it difficult to track loan statuses, and generate accurate reports.

## 1.3 Objectives

* To facilitate exact KYC form and its approval process.
* To generate the loan application and approval process.
* To facilitate efficient loan disbursement.
* To provide real-time reporting and analytics for better decision-making.

## 1.4 Scope and Limitation

**Scope:**

* Implementation of loan application, approval, disbursement.
* Role-based access control for members, loan officers, and administrators.
* Real-time reporting and analytics.

**Limitations:**

* The system is designed specifically for cooperative societies and may require customization for other types of financial institutions.
* Initial deployment is web-based; mobile applications are not within the scope of this project.

## 1.5 Report Organization

**Chapter1** includes introduction of the System LMS (Sahakari) with problem statement objectives and scope and limitation.

**Chapter 2** includes the background study of LMS (Sahakari) and some literature review of other LMS (Sahakari) systems.

**Chapter 3** includes the system analysis, system design (OOP approach) along with object and class diagram, component diagram and deployment diagram of the LMS (Sahakari).

**Chapter 4** includes about the tools used in this system and the testing that are done.

**Chapter 5** includes about the outcome of this system as well as the future recommendations for the LMS (Sahakari).

# Chapter 2: Background Study and Literature Review

## 2.1 Background Study

In today's rapidly advancing technological landscape, every industry strives to simplify and enhance human life through innovative solutions. The financial sector, particularly in cooperative societies, has seen significant growth in online systems designed to streamline complex processes. The Loan Management System (Sahakari) is a prime example of such innovation. It is a web-based platform that caters to the needs of cooperative members by efficiently managing loan applications, approvals, disbursements. The system is designed to automate and optimize the loan management process, making it easier for members, loan officers, and administrators to perform their tasks.

The Loan Management System (Sahakari) aims to provide a seamless experience for both members and loan staff. Members can easily apply for loans, track the status of their applications, and manage their repayments through a user-friendly interface. On the other hand, loan officers can review applications, validate member information, calculate loan terms, and monitor loan performance with minimal effort. The system also allows administrators to oversee the entire loan management process, ensuring transparency and efficiency.

The system is designed with a simple and intuitive interface, making it accessible to users of all technical backgrounds. Predefined functions and streamlined input processes minimize the need for multiple screen interactions, allowing users to quickly complete tasks without confusion. This design philosophy ensures that both members and loan officers can efficiently use the system, regardless of their familiarity with technology.

In summary, the Loan Management System (Sahakari) not only simplifies the loan management process within cooperative societies but also enhances user experience through its secure, scalable, and easy-to-use platform. This system represents a significant step forward in the digital transformation of financial services in the cooperative sector.

## 2.2 Literature Review

Cooperative can be effective institutional arrangement in breaking the vicious cycle of poverty in the rural socio-economic context. Under effective supervision, if cooperative can be well managed and strengthened; cooperative can potentially strengthen the domain of transferability of rural community and there forwards to contribute to sustainable reduction of poverty. The cooperatives had played the significant role in defining and sustaining the lives of the communities. Therefore, the rural communities had lot to benefit from undertaking cooperative programs. The major roles [1] of the cooperatives to the rural communities included the poverty reduction, employment creation, improved food security, women empowerment and human capital development.

The financial sector has witnessed a significant transformation with the advent of technology, leading to the development of sophisticated loan management systems (LMS). These systems are designed to streamline the lending process, enhance financial inclusion, and improve operational efficiency in financial institutions only for Sahakari. Loan management systems play a critical role in the efficient functioning of financial Sahakari. They automate and manage the entire loan lifecycle, from application and disbursement to repayment and closure.

LMS helps reduce manual errors, minimize operational costs, and ensure regulatory compliance, thereby enhancing the overall efficiency and reliability of financial operations [5]. Recent advancements in financial technology (FinTech) have significantly improved LMS capabilities. For instance, artificial intelligence (AI) and machine learning (ML) algorithms are increasingly being integrated into LMS to assess creditworthiness, predict loan defaults, and personalize loan products for members [6]. These technologies enable Sahakari to make data-driven decisions and offer tailored financial solutions, thereby fostering financial inclusion and member satisfaction.

A user-friendly LMS can streamline the loan application process, making it faster and more accessible for members. Digital platforms allow members to apply for loans remotely, track their application status in real-time, and receive instant notifications, thus improving the overall member experience [7]. LMS helps in mitigating risks associated with lending by providing robust tools for credit assessment and monitoring. integrated risk management.

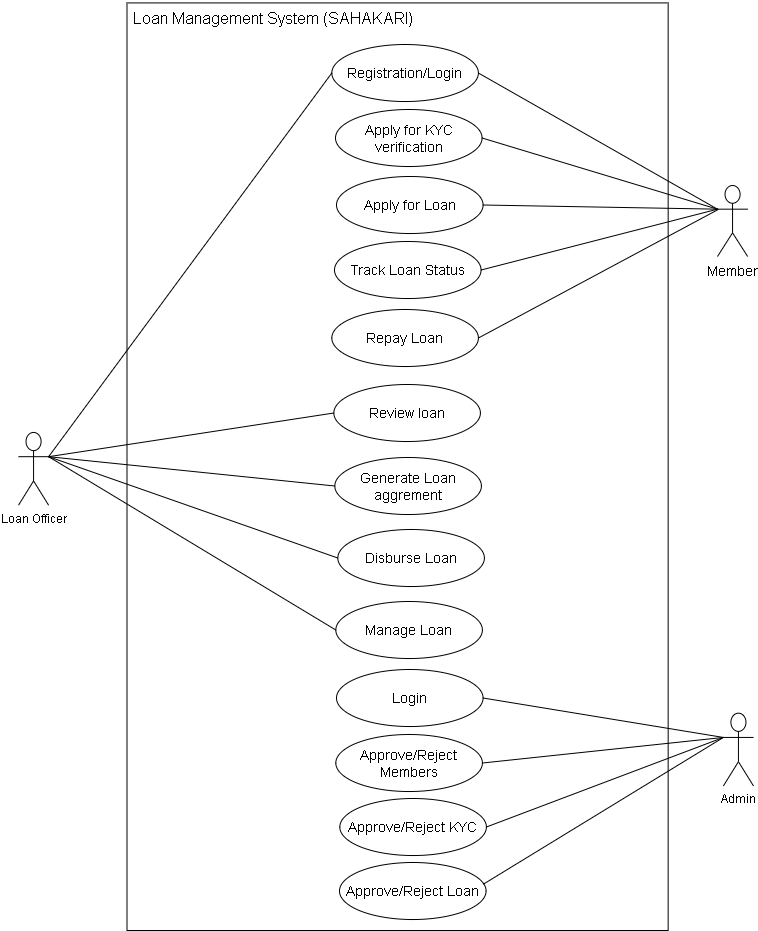
# Chapter 3: System Analysis and Design

## 3.1 System Analysis

### 3.1.1 Requirement Analysis

**Functional requirements:**

**Use Case Diagram of Loann Management system (Sahakari)**

****

**Figure 3. 1 Use Case diagram of LMS**

Functional requirements provide overview of system. The system LMS has following functional requirements:

Member Module

* Member should be able to register, login, logout from the system.
* Member should be able to apply KYC verification form and Loan application form.
* Member should be able to edit their detail KYC and loan application details.

Loan Staff Module

* Staff should able to register, login, logout from the system.
* Staff should be able to review members applied loan applications.
* Staff should be able to generate loan agreement and disburse loan amount.

Admin Module

* Admin should be able to login, logout from system.
* Admin should be able to approve members and members KYC if it applicable.
* Admin should be able to approve Loan were applied by member and reviewed by staff.

**Non-functional requirements:**

**Performance Requirement:**

The system must be optimized to handle multiple simultaneous transactions efficiently. It should process loan applications, approvals, and repayments with minimal latency, ensuring a smooth and responsive user experience even during peak usage times.

**Security Requirement:**

The system must implement robust security measures to protect sensitive data, including member personal information, loan details, and financial transactions. This includes secure authentication mechanisms, data encryption, regular security audits, and protection against threats such as unauthorized access, data breaches, and SQL injection attacks.

**Usability Requirements:**

The system must be designed with a user-friendly interface that is easy to navigate for users with varying levels of technical expertise. The interface should be intuitive, with clear instructions, minimal input requirements, and consistent design patterns to ensure a positive user experience.

**Scalability Requirement:**

The system must be capable of scaling to accommodate growth in the number of users, transactions, and data volume. This includes both horizontal scaling (adding more servers) and vertical scaling (upgrading existing hardware) to maintain performance and reliability as the cooperative expands.

**Reliability Requirement:**

The system must be highly reliable, ensuring continuous availability and minimal downtime. This includes implementing redundant systems, automated backups, and disaster recovery plans to ensure that the system remains operational even in the event of hardware failures or other unexpected issues.

**Maintainability Requirement:**

The system must be designed for easy maintenance and updates. This includes writing clean, modular code with comprehensive documentation, allowing developers to quickly identify and fix bugs, add new features, or update existing functionality without disrupting the system's operation. Regular updates and patches should be easy to deploy, ensuring that the system remains secure and up-to-date with the latest technology standards.

**Legal Requirement:**

The system must adhere to all relevant legal and regulatory requirements, including data protection laws, financial regulations, and industry standards. This includes ensuring compliance with local and international laws regarding the collection, storage, and processing of personal and financial data.

### 3.1.2 Feasibility Analysis

**Technical feasibility:**

The Loan Management System (Sahakari) is technically feasible, utilizing Laravel for robust application development and MySQL for secure and efficient data management. The system is scalable, integrates well with existing banking systems, and includes strong security measures like encryption and role-based access control.

**Operational feasibility:**

Operationally, the system aligns with the daily activities of cooperative societies, streamlining processes like member registration and loan management. With training and support, the system can be smoothly adopted, improving efficiency and reducing manual effort.

**Economic feasibility:**

Economically, the system is a sound investment. It reduces operational costs by automating tasks, minimizes errors, and enhances member satisfaction. The scalable design ensures long-term cost-effectiveness, making it a valuable asset for cooperative societies.

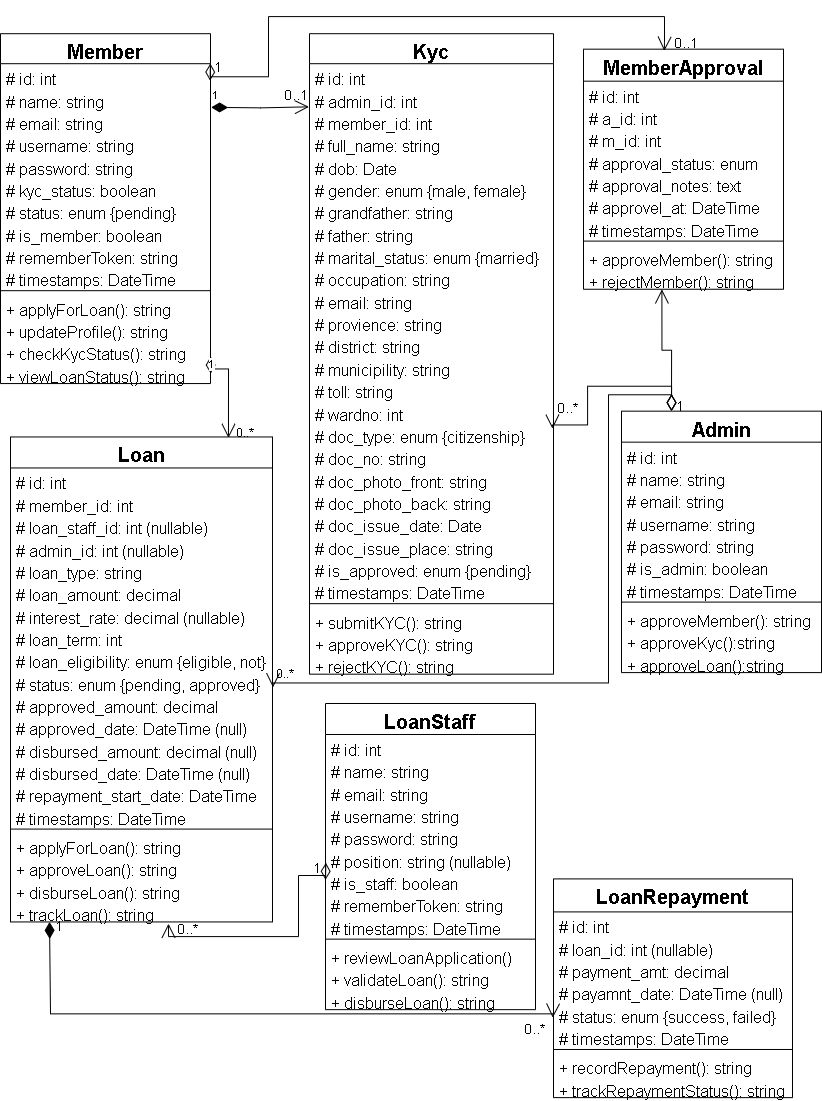
**Schedule Feasibility**

Here is the Gantt chart showing the probability of the project being completed within its schedule time limits by a planned time due date.

**Table 3. 1:** **Gantt Chart of LMS (Sahakari)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Phases of project development** | **Year 2081-2082** | | | | | | | | | | | | | | |
| **Month June** | | | | | | | | | | | | | | |
| **June** | | | | **July** | | | | **August** | | | | **September** | | |
| **W1** | **W2** | **W3** | **W4** | **W1** | **W2** | **W3** | **W4** | **W1** | **W2** | **W3** | **W4** | **W 1** | **W2** | **W3** |
| **Project selection** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Planning and Analysis** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Proposal Defense** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Design** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Coding** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Testing** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Documentation** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

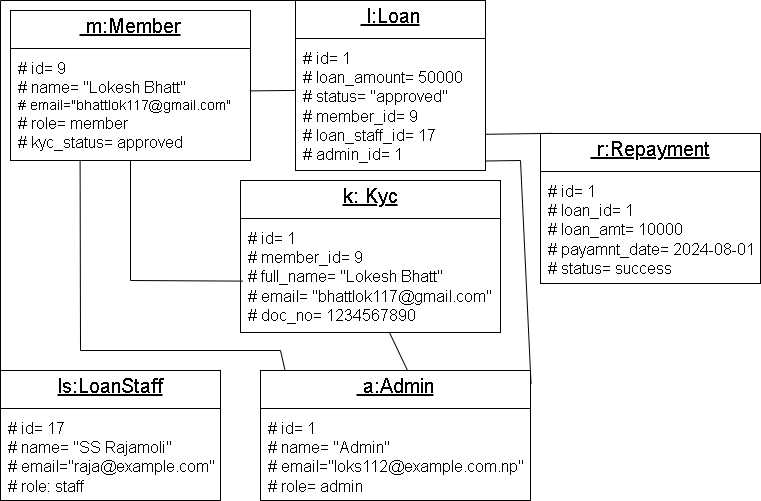
3.1.3 Object Modelling:



**Figure 3. 2 : Class diagram of LMS (Sahakari).**

Object modeling in the Loan Management System (Sahakari) plays a crucial role in understanding the system's structure, entities, and the relationships between them. This model represents the system's core objects, encapsulating both their attributes and behaviors to ensure smooth operation across different roles (member, staff, admin). Here above picture class diagram shows a reduced block diagram that highlights the essential elements and interactions in LMS. The figure illustrates the functions and interactions of a number of entities, such as members, staffs, admin, loans, and KYCs.

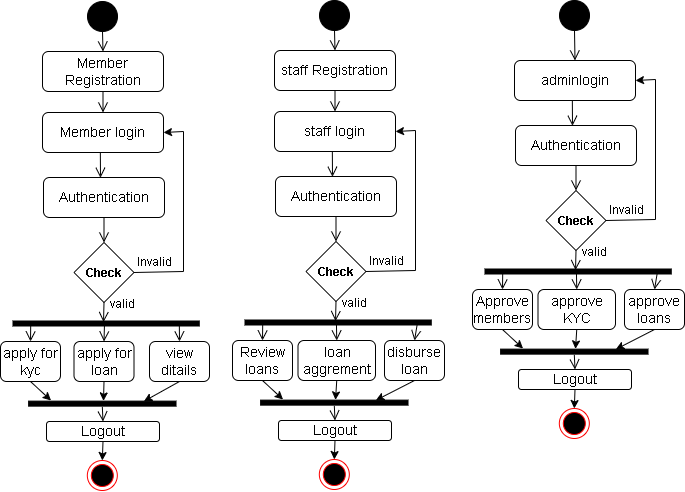
The Member object represents the cooperative members who interact with the system to apply for loans, manage their accounts, and track loan status. The LoanStaff object is responsible for reviewing loan applications, verifying documents, and making approval decisions. The KYC object handles the Know Your Customer verification process, storing and managing the identification details submitted by members. The Loan object encapsulates all details related to a loan, including the loan amount, interest rate, repayment schedule, and status. Finally, admin abject oversees the system’s operations, approve members, approve KYCs, and approve loan applications applied by members.



**Figure 3. 3: Object diagram of LMS (Sahakari)**

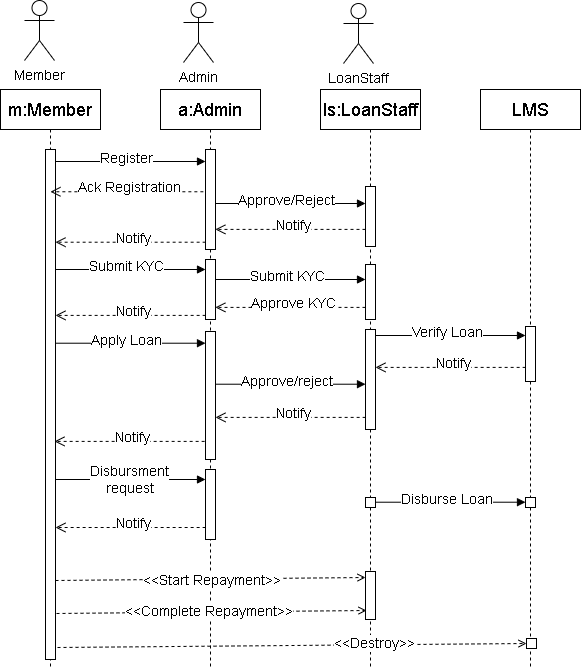
3.1.4 Dynamic Modelling:

A state diagram illustrates the different states of an object in the system and the transitions between these states as a result of events or actions. For instance, the lifecycle of a loan application is represented through different states, from creation to approval to disbursement. Loan State Transitions:



**Figure 3. 4: State diagram of LMS {Sahakari)**

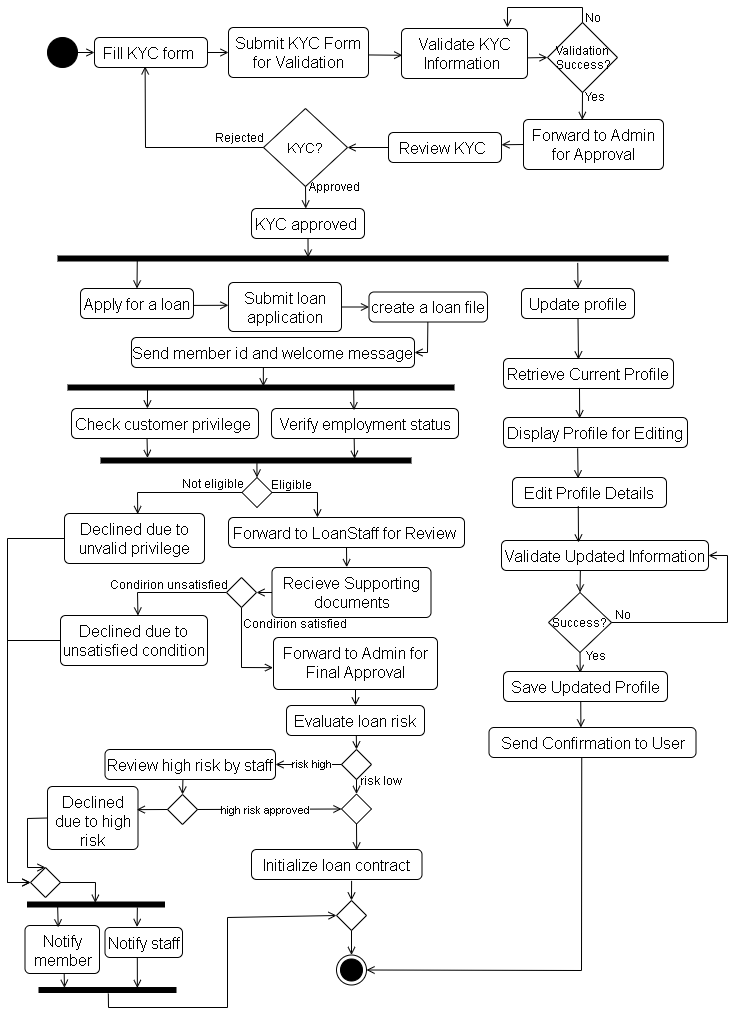
A sequence diagram shows how objects communicate with each other in a specific scenario or use case over time. It depicts the interactions between different entities (members, staff, admin) and system components, highlighting the sequence of messages exchanged during an operation such as loan application or KYC approval.



**Figure 3. 5: Sequence diagram of LMS.**

3.1.5 Process Modelling:

Activity diagrams detailing the workflows within the system:



**Figure 3. 6: Activity diagram of LMS (Sahakari)**

Process modeling in the Loan Management System (Sahakari) represents the workflow and business processes involved in handling various tasks, such as loan applications, KYC approval, loan disbursements, etc. It aims to define, analyze, and document the flow of information and tasks in the system. This step is critical to understanding how the different components of the system interact to complete specific processes, ensuring that the business rules are met and the system runs efficiently.

The above figure shows that detailed description about Loan Management System here initially member can register on the system then fill him/her KYC detail after KYC can be approved by admin member can be eligible for applying loan after members successful loan application staff can review their loan to check the loan is eligible or not. If loan is not eligible discard such loan otherwise evaluates its risk and waiting for admin approval. After successful loan approval the loan can be disbursed by staff.

## 3.2 System Design

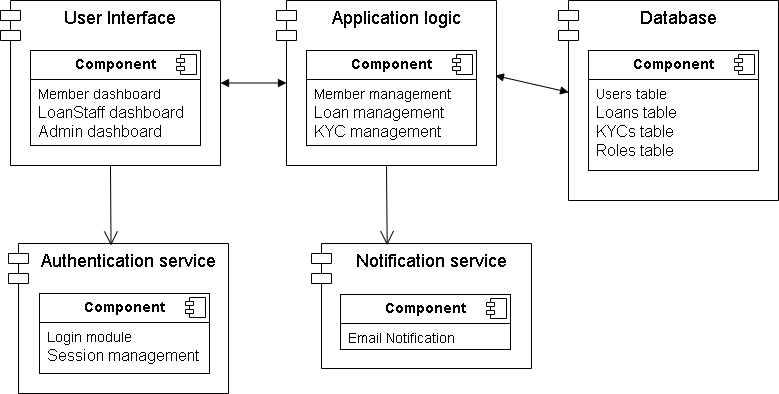
System design explain about the high-level architecture and detailed components of the system are refined and implemented. This phase focuses on defining how the system will meet the specified requirements, ensuring scalability, performance, and maintainability.

### **3.2.1 Refinement of Classes and Objects:**

In the refinement of classes and objects for the Loan Management System (Sahakari), the initial object models developed during the analysis phase are expanded and detailed to ensure they align with the system’s specific requirements. This process involves adding attributes, defining methods, and specifying the relationships between different classes. For example, in the **User** class, which serves as a base class for all users in the system (members, staff, and admin), attributes like user\_id, name, email, password, role, and kyc\_status are further refined with appropriate data types and constraints. Methods such as login(), logout(), and register() are defined to handle user authentication and profile management. The inheritance structure is also optimized—common functionality shared across all users is placed in the User base class, while specific behaviors, such as applyForLoan() for members or approveLoan() for admin, are delegated to their respective subclasses.

Relationships between key classes, like LoanApplication, KYC, and User, are precisely defined to ensure efficient data handling. For instance, the **LoanApplication** class, which handles the loan submission process, includes attributes such as loan\_type, loan\_amount, loan\_term, and status. It interacts with the **Member** class, allowing users to submit loan applications and view statuses. The system also optimizes relationships, ensuring that each user can have multiple loan applications, but each application is linked to one specific user. Polymorphism and inheritance are employed to streamline functionality, reducing redundancy by sharing common features across classes while allowing unique methods for specialized tasks. This refinement ensures a robust, maintainable system architecture that supports the system's complex workflows, such as member loan applications, admin approvals, and loan disbursements by staff.

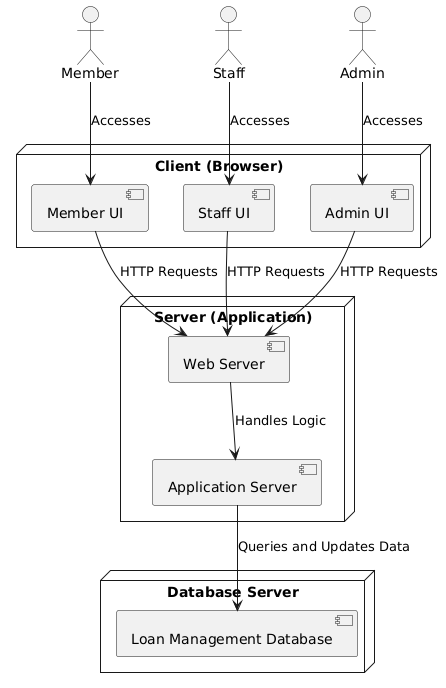
### **3.2.2 Component Diagram:**



**Figure 3. 7: Component diagram of LMS (Sahakari)**

The component diagram for the Loan Management System (Sahakari) provides a high-level overview of the system's architecture, illustrating the key components and their interactions. This diagram features several essential components, including the User Interface (UI), which serves as the front-end for members, staff, and admin users to interact with the system. The Business Logic Layer contains the core functionalities, such as user authentication, loan application processing, and KYC management, implementing the necessary business rules. Interfacing with the Business Logic Layer is the Data Access Layer, responsible for managing CRUD operations on the persistent data stored in the Database, which includes all records related to users, loans, and KYC applications. Additionally, if applicable, the Notification System component handles alerts and notifications for users regarding their loan applications and KYC statuses, while Third-Party Services may be integrated for functionalities like payment processing and document verification. The above component diagram effectively outlines how each part of the system communicates, facilitating a better understanding of the overall structure and enabling easier maintenance and scalability as the system evolves.

### **3.2.3 Deployment Diagram:**



**Figure 3. 8: Deployment diagram of LMS (Sahakari).**

The deployment diagram for the Loan Management System (Sahakari) illustrates the physical deployment of the system components across various hardware nodes. This diagram provides a visual representation of how the software components are distributed across the hardware environment, showcasing the relationships between different servers, client devices, and any external services.

At the core of the deployment diagram is the **Application Server**, which hosts the main application logic and handles incoming requests from users. This server runs the web application that enables members, staff, and admins to interact with the system through a web browser or mobile application. Additionally, a **Database Server** is depicted, which stores all persistent data related to users, loans, and KYC applications, ensuring data integrity and efficient access.

Client devices, such as desktops, laptops, and mobile devices, connect to the application server via the internet, enabling users to access the system remotely. The deployment diagram may also include any third-party services, such as payment gateways or external KYC verification services, indicating how the system integrates with these external components.

Furthermore, the diagram highlights the network infrastructure, including firewalls and load balancers, which enhance security and manage traffic to ensure optimal performance. By providing a clear visualization of the system's deployment architecture, this diagram assists stakeholders in understanding how the Loan Management System is hosted and accessed, facilitating better planning for system scalability, maintenance, and performance optimization.

## 3.3 Algorithm details

**Funding Algorithm**

The Funding Algorithm is a mathematical model used to assess and determine the funding amount for loan applicants based on various financial parameters. The goal is to ensure that the calculated funding is feasible for both the lender and the borrower, minimizing the risk of default while maximizing the potential for timely repayment. Below is a detailed breakdown of the algorithm with calculations.

**Funding Algorithm Overview:**

**Input Parameters**

Loan Amount Requested (L) – The amount the applicant wants to borrow.

Interest Rate (r) – The annual interest rate for the loan, expressed as a decimal.

Loan Term (T) – The loan duration in years.

Applicant Income (I) – The annual income of the applicant.

Debt-to-Income Ratio (DTI) – The percentage of the applicant's income that can reasonably be allocated to debt repayments.

**Output Parameters**

Approved Loan Amount (A) – The final loan amount approved by the algorithm.

Monthly Installment (M) – The monthly payment required for the loan term.

Eligibility Score (E) – A score indicating the applicant's likelihood of successful repayment.

**Algorithm Steps**

1. Calculate the Maximum Loan Capacity  
The maximum loan capacity is derived from the Debt-to-Income Ratio (DTI), ensuring that the applicant can manage monthly payments within their income constraints.  
  
Max Loan Capacity (MLC) = I × DTI × T

2. Determine the Monthly Installment  
Using the loan amount, interest rate, and loan term, calculate the monthly payment the applicant will need to make to repay the loan.  
M = (L × (r / 12)) / (1 - (1 + (r / 12))^(-12 × T))

3. Eligibility Score Calculation  
The eligibility score is based on the applicant's ability to repay the loan. It considers both the applicant's income and the calculated monthly installment. Higher scores indicate a greater likelihood of loan approval.  
E = (I / M) × 100

4. Approve Loan Amount (A)  
Based on the Max Loan Capacity, Monthly Installment, and Eligibility Score, the algorithm approves an appropriate loan amount. If the requested loan amount L exceeds MLC, then A will be adjusted downwards to meet capacity limits.  
A = min(L, MLC)

**Example Calculation**

Suppose an applicant requests a loan amount of RS. 50,000 with an interest rate of 5%, a loan term of 5 years, and an annual income of RS. 60,000. The DTI ratio is set at 0.3, or 30%.  
1. Max Loan Capacity (MLC): MLC = 60,000 × 0.3 × 5 = 90,000  
  
2. Monthly Installment (M): Using the formula for M, we find: M ≈ 943.56  
  
3. Eligibility Score (E): E ≈ 53.08  
  
4. Approved Loan Amount (A)  
Since the requested amount L = 50,000 is less than the Max Loan Capacity 90,000, the approved amount A is 50,000.

# Chapter 4: Implementation and Testing

## 4.1 Implementation of system

### 4.1.1 Tools Used:

**Laravel**

Laravel was used as the primary backend framework due to its robust features, including MVC architecture, blade templating engine, routing, and built-in security features. It streamlined the development process by providing tools for database management, authentication, and encryption.

**MySQL**

MySQL was selected as the database management system for storing the application data, including user information, loan details, and KYC data. Its compatibility with Laravel and its ability to handle complex queries efficiently made it an ideal choice.

**Tailwind CSS (CSS Framework)**

Tailwind CSS was used for designing the frontend of the system. Its utility-first approach enabled the creation of a responsive, modern, and user-friendly UI without writing extensive custom CSS. Tailwind ensured that the interface remained clean and adaptable across different screen sizes.

Other tools and technologies that played a crucial role in the implementation include:

* **Composer:** A dependency management tool for PHP that was used to manage Laravel packages and libraries.
* **Git:** A version control system that enabled collaborative development and version tracking. GitHub was used as the remote repository to host and share the project's codebase.
* **XAMPP:** An open-source cross-platform web server solution that includes Apache, MySQL, and PHP, used for local development and testing.

### 4.1.2 Implementation details of modules:

**User Management Module**

The User Management module is responsible for handling all user-related functionalities, including member registration, role-based authentication, and session management. It is implemented through the UserController, which contains methods like register(), login(), and logout(). The register() function processes new user registrations, while the login() function authenticates users based on their roles (member, staff, or admin) and redirects them to their respective dashboards. The system uses middleware for role-based access control (RBAC), ensuring that only users with the appropriate role can access specific sections of the application. The logout() function terminates the user session, securing the system by ensuring that unauthorized users cannot retain access.

**KYC Management Module**

The KYC (Know Your Customer) Management module facilitates the submission, review, and approval of KYC documentation by members, staff, and admin. It is controlled by the KYCController, which includes methods such as submitKYC() for members to submit their KYC information, reviewKYC() for staff to review and verify the details, and approveKYC() for admin to approve or reject applications. The KYC status is updated in both the users and kycs tables upon approval. Input validation ensures that KYC forms are submitted with all the required fields. The module plays a critical role in preventing fraudulent activity by verifying the identity of each member.

**Loan Application Module**

The Loan Application module enables members to apply for loans and staff to review and process these applications. It is managed by the LoanController, which contains methods like applyLoan() for members to submit loan applications, reviewLoan() for staff to evaluate the application, and approveLoan() for admin to finalize the decision. Loan eligibility is checked through a helper function validateLoan(), which assesses the member's credit score, loan term, and amount. Upon approval, the disburseLoan() method disburses the approved amount and updates the loan's status in the system. This module integrates seamlessly with the loan approval workflow to ensure efficient loan processing.

**Admin Dashboard Module**

The Admin Dashboard module offers administrative control over the system, allowing admin users to manage members, KYC approvals, and loan applications. The AdminController contains functions like viewMembers() to display a list of members, approveMember() to accept or reject member registrations, and approveLoan() to handle loan application approvals. The admin interface provides a streamlined view of the system’s overall status, including pending KYC and loan applications. Role-based access ensures that only admin users can access this module, while regular members and staff cannot interfere with administrative operations.

## 4.2 Testing

### 4.2.1 Test Cases for Unit Testing:

The following are detailed test cases for some of the critical components of the Loan Management System (Sahakari).

**Table 4. 1** **Test Case 1-Registration Form**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Project Name: Loan Management System | | | | | | |
| Test Case | | | | | | |
| Test Case ID: TC\_1 | | | | Test Designed by: Lokesh Raj Bhatt | | |
| Test Priority: High | | | | Test designed date: 2024-09-20 | | |
| Module Name: Sign up | | | | Test Executed by: Lokesh Raj Bhatt | | |
| Test Title: Register new user into the system | | | | Test Execution date: 2024-09-20 | | |
| Description: Test the Loan Management System’s registration page. | | | | | | |
| Pre-condition: User should have all the necessary details | | | | | | |
| Dependencies: | | | | | | |
| Step | Test Steps | Test Data | Expected Result | Actual result | Status | Note |
| 1 | Navigate to the  registration page |  | Registration page should open | As expected,  i.e. the user  is navigated to the  registration page | Pass |  |
| 2 | Provide all the  required  information | Full name=Lokesh Raj Bhatt,  [Email=lokesh117@gmail.com](mailto:Email=lokesh117@gmail.com),  Username= Loks  Password=Lokesh123%%%%,  Confirm password= Lokesh123%%%%, | The information  should be entered | The student’s information is expected | Pass |  |
| 3 | Click on the  register button |  | The user should be  registered into the system | The user is registered as expected | Pass |  |
| Post-condition: Users information is validated and successfully registered into Loan Management System. | | | | | | |

**Table 4. 2 Test Case 2-Login Form**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Project Name: Loan Management System (LMS) | | | | | | |
| Test Case | | | | | | |
| Test Case ID: TC\_2 | | | | Test Designed by: Lokesh Bhatt | | |
| Test Priority (Low/Medium/High): High | | | | Test Designed Date: 2024-09-20 | | |
| Module Name: Login | | | | Test Executed by: Lokesh Bhatt | | |
| Test title: verify login with valid username and password | | | | Test Execution date: 2024-09-20 | | |
| Description: Test the Loan Management system’s login page | | | | | | |
| Pre-condition: User has valid username and password | | | | | | |
| Dependencies: | | | | | | |
| Step | Test Steps | Test Data | Expected Result | Actual Result | Status | Notes |
| 1 | Navigate to login page |  | Login page should open | The user is navigated to the login page | Pass |  |
| 2 | Input valid username | Username: Loks | The username can be entered | As expected, | Pass |  |
| 3 | Input valid password | Password:  Lokesh123%%%% | The password can be entered | As expected, | Pass |  |
| 4 | Click on Login button |  | User should be able to  login into the system | As expected, | Pass |  |
| Post-conditions:  Users are validated with database and successfully logged into Loan Management Systems. | | | | | | |

### 4.2.2 Test Cases for System Testing:

**Table 4. 3** **Test Case 3: End-to-End Member Verification and KYC Approval**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Project Name: Loan Management System | | | | | | |
| Test Case | | | | | | |
| Test Case ID: TC\_3 | | | | Test Designed by: Lokesh Bhatt | | |
| Test Priority: High | | | | Test designed date: 2024-09-21 | | |
| Module Name: Admin | | | | Test Executed by: Lokesh Bhatt | | |
| Test Title: End-to-End Member Verification and KYC Approval | | | | Test Execution date: 2024-09-21 | | |
| Description: To ensure a new member can successfully register, submit KYC documents, and get verified by the admin. | | | | | | |
| Pre-condition: The member registration form is functional. KYC document upload functionality is working. Admin access to review and approve/reject KYC is implemented. | | | | | | |
| Dependencies: | | | | | | |
| Step | Test Steps | Test Data | Expected Result | Actual result | Status | Note |
| 1 | Navigate to the registration page and fill out the registration form with valid details | Name=Lokesh Raj Bhatt  Email=bhattlok117@gmail.com  Username=Loks Password=Lokesh123%%%% | User account is successfully created, and the system redirects to the KYC document submission page or dashboard with pending KYC status. | As expected,  Registration completed then KYC fill up page will open. | Pass |  |
| 2 | Upload KYC documents such as ID proof, address proof, and other required documents | ID Proof= National ID card, Citizenship  Address= actual address. Document Format: PDF, JPG, PNG | KYC documents are uploaded successfully, and a confirmation message is displayed indicating the KYC submission is under review. | The KYC detail is uploaded successfully and its status is pending due to under review | Pass |  |
| 3 | Log in as the admin | Username= Admin  Password= newpassword123 | Admin successfully logs into the dashboard, where pending KYC requests are visible. | The admin login successfully and navigate to admin dashboard | Pass |  |
| 4 | Approve or Reject KYC |  | If approved system update member kyc\_ststus approved | Member can approve and KYC can approve easily. | pass |  |

**Table 4. 4** **Test Case 4: End-to-End Loan Application approvement and Disbursement**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Project Name: Loan Management System | | | | | | |
| Test Case | | | | | | |
| Test Case ID: TC\_4 | | | | Test Designed by: Lokesh Bhatt | | |
| Test Priority: High | | | | Test designed date: 2024-09-21 | | |
| Module Name: Admin, Staff | | | | Test Executed by: Lokesh Bhatt | | |
| Test Title: End-to-End Loan approval and loan Disbursement | | | | Test Execution date: 2024-09-21 | | |
| Description: To ensure a user can successfully apply for a loan, and the admin can approve and staff can disburse it. | | | | | | |
| Pre-condition: The loan application form is functional. The admin dashboard and loan approval process are implemented. The member must be verified (KYC completed) before applying for a loan. | | | | | | |
| Dependencies: | | | | | | |
| Step | Test Steps | Test Data | Expected Result | Actual result | Status | Note |
| 1 | Log in as a verified member | Email=bhattlok117@gmail.com  Username=Loks Password=Lokesh123%%%% | Member successfully logs in and is redirected to the dashboard. | Member easily access their dashboards to apply loan. | Pass |  |
| 2 | Navigate to the "Apply for Loan" section and fill out the loan application form. | Loan amount= 4000000  Loan type= home  Loan term=3 | The loan application is submitted successfully, and the status is updated to on the member dashboard. | The loan is uploaded successfully | Pass |  |
| 3 | Log in as the admin | Username= Admin  Password= newpassword123 | Admin successfully logs into the dashboard, where pending loan requests are visible. | The admin login successfully and navigate to admin dashboard | Pass |  |
| 4 | Approve or Reject Loan | Select the "Approve" or "Reject" option, and if rejected, provide a reason for rejection. | If approved, the loan status changes to "Approved" and the disbursement process begins. If rejected, the member is notified, rejected | Loan can approved easily. | pass |  |

**Table 4. 5** **Test Case 5: Data Encryption and RBAC test**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Project Name: Loan Management System | | | | | | |
| Test Case | | | | | | |
| Test Case ID: TC\_5 | | | | Test Designed by: Lokesh Bhatt | | |
| Test Priority: High | | | | Test designed date: 2024-09-22 | | |
| Module Name: Admin, Staff | | | | Test Executed by: Lokesh Bhatt | | |
| Test Title: Data encryption and RBAC test | | | | Test Execution date: 2024-09-22 | | |
| Description: To ensure that sensitive data, such as KYC information and loan details, are encrypted before storage. | | | | | | |
| Pre-condition: encryption methods are implemented. The RBAC system is implemented with predefined roles. | | | | | | |
| Dependencies: | | | | | | |
| Step | Test Steps | Test Data | Expected Result | Actual result | Status | Note |
| 1 | Insert a new loan record with sensitive data | Member id=12, loan amount=906090, | Data should be encrypted in the database | Only authorized users can decrypt and view the information. | Pass |  |
| 2 | Intercept a request where sensitive data is being transmitted | Loan amount= 4000000  Loan type= home  Loan term=3 | Data should be encrypted during transmission, ideally via HTTPS. | Applied data are encrypted | Pass |  |
| 3 | Log in as the admin | Username= Admin  Password= newpassword123 | Admin should have full access to these features without restrictions. | Admin have full access of these features. | Pass |  |
| 4 | Login as member | Username= loks  Password= Lokesh123%%%% | Member should only have access to personal data and member-specific features. Access to admin features should be denied. | Member can access personal data and features. | pass |  |
| Post-conditions:  The sensitive data are encrypted successfully applied on Loan Management Systems. | | | | | | | |

# Chapter 5: Conclusion and Future Recommendations

## 5.1 Conclusion

The successful implementation of the Loan Management System (Sahakari) marks a significant milestone in modernizing and streamlining the cooperative's operations. This project was conceived with the primary goal of creating an efficient, user-friendly, and secure platform that would enhance the cooperative’s ability to manage loans, from application to disbursement, and to streamline the member approval process.

Throughout the project, the development team focused on ensuring that the system met all the specified requirements, including the management of member information, loan applications, KYC processes, and multi-role access control. The system was built using Laravel, a robust PHP framework, coupled with MySQL for database management and various frontend technologies to create an intuitive user interface. The implementation of this system has automated many manual processes, reduced the risk of errors, and improved overall operational efficiency.

One of the key achievements of the project is the comprehensive integration of various modules, which allows different user roles—members, loan staff, and admins—to interact with the system seamlessly. The members can easily apply for loans, check their status, and make repayments, while the loan staff can efficiently review applications and manage loan disbursements. Admins are provided with powerful tools to oversee the entire operation, ensuring that all processes are conducted smoothly and securely.

The system’s rigorous testing, including unit and system testing, ensured that all components work together flawlessly, minimizing the likelihood of operational disruptions. The implementation of distinct session management for different roles and the careful handling of sensitive data, such as KYC information, further contributed to the system’s security and reliability.

## 5.2 Lessons Learned/Outcome

The development of the Loan Management System (Sahakari) provided valuable insights and practical lessons that contributed significantly to the project's success. These insights are crucial for future projects and for the continued improvement of the current system. Below are the key lessons learned and outcomes from the project:

**1. Importance of Clear Requirements Gathering:**

One of the early challenges faced during the project was the need for clear and comprehensive requirements. Initial ambiguities in user roles, specific functionalities, and data management needs led to some rework in the design phase. This experience underscored the importance of detailed requirements gathering and validation with stakeholders before starting the development process. It taught the team to place more emphasis on creating thorough documentation and using prototypes to ensure that everyone involved had a shared understanding of the project scope.

**Outcome:** The team improved communication with stakeholders, leading to a more aligned development process and a system that met all user expectations.

**2. Balancing Security with Usability:**

Ensuring data security, especially with sensitive information like KYC details and financial transactions, was a critical concern throughout the project. However, this need sometimes conflicted with creating a user-friendly experience. For example, implementing strong password policies and multi-factor authentication had to be balanced with ease of use for members who may not be tech-savvy.

**Outcome:** The team found that involving users early in the design phase to test security features helped strike a balance between security and usability. Implementing features like tooltips and user guides also helped members navigate the system without compromising on security.

## 5.3 Future Recommendations

The Loan Management System (Sahakari) has laid a strong foundation for efficient loan management, but there are several opportunities for further enhancement. One significant area for improvement is the development of mobile applications. With an increasing number of users relying on smartphones, a dedicated mobile app would offer greater convenience, enabling members to manage their loans, submit applications, and track payments directly from their devices. Additionally, integrating the LMS with other financial systems, such as banking platforms, credit scoring agencies, and digital payment gateways, would streamline processes like loan disbursement and repayment, and automate credit assessments. This integration would not only enhance operational efficiency but also provide a seamless experience for members.

Another key area for future development is the expansion of the system's analytics and reporting capabilities. By incorporating advanced analytics, the cooperative could gain deeper insights into financial performance, member behavior, and potential risks, enabling more informed decision-making. Automation of the loan underwriting process is another recommendation, as it would allow the system to instantly evaluate loan applications based on predefined criteria, reducing approval times and ensuring consistency in decision-making. Additionally, implementing a member rewards program could increase engagement and loyalty by offering incentives for timely loan repayments and active participation in cooperative activities.

Expanding multi-language support within the system would make it more accessible to a diverse member base, catering to those who prefer to interact in languages other than English. Finally, the importance of ongoing security cannot be overstated. Regular security audits, updates, and staff training are essential to protect sensitive financial data and ensure compliance with the latest security standards. By addressing these areas, the Loan Management System (Sahakari) can continue to evolve, offering enhanced services to its members while maintaining its competitive edge in the financial sector.

# References

|  |  |
| --- | --- |
| [1] | S. e. al, "LMS," online pdf, 2020. |
| [2] | J. &. Wang, "Loan System," john and wang, 2019. |
| [3] | Brown, "Digital platforms," brown, 2018. |
| [4] | S. e. al, "Loan system," p. 20, 25 07 2019. |
| [5] | L. a. credit, "Bharat Ram Dhungana," *access credit in nepal,* p. 30, 2018. |
| [6] | K. a. Devkota, "Financial Inclusion," 2018. |
| [7] | R. a. Gurung, "Inadequate risk management," p. 30, 2019. |
| [8] | S. a. Khanal, "Types of loan," *Loan types in nepal,* p. 20, 2017 and 2020 . |
| [9] | A. Regmi, "Policy Reforms," p. 30, (2019) and (2020). |
| [10] | T. a. Shrestha, "NPL Loans," p. 35, 2016 and 2019. |

# Appendices

