Loan Management System

Functional Specification Document

Technology Stack: .NET Framework

Deployment Target: AWS

Last Updated: 08/06/2025

## Introduction

### Purpose

To outline the functionalities, workflows, and integrations of a Loan Management System that automates the loan lifecycle from application to repayment.

### Scope

This system manages user roles (Admin and Borrower), loan applications, approvals, repayments, notifications, and reporting, and integrates with third-party APIs for payments and notifications.

## System Overview

**Loan Lifecycle Coverage:**

1. Application
2. Evaluation & Approval
3. Disbursement
4. Repayment Tracking
5. Status Notifications
6. Closure

**User Types:**

1. Visitor (unregistered user)
2. Borrower
3. Admin

## Functional Requirements

**1. Authentication and Authorization**

* User Registration:

Users (borrowers and admins) can create accounts using their full name, email, and password.

Upon registration, the system assigns a default role (borrower by default).

Input validation includes email format, password strength, and duplicate checks.

* Login & Session Management:

Users log in using email and password.

System authenticates via JWT tokens.

Tokens are used for session management and access control.

* Role-Based Access Control:

Admins can access management endpoints (users, loans, logs).

Borrowers can access personal loan applications and repayments.

Unauthorized access returns HTTP 401/403 responses.

**2. Loan Application**

* Loan Form Submission:

Borrowers submit loan applications with amount, duration, and purpose.

System validates user eligibility and checks for any active/pending loans.

* Credit Score Evaluation:

System fetches credit score using external APIs for example TransUnion.

If score is below a configurable threshold, the application is auto-flagged for review or rejection.

* Application Tracking:

Borrowers can view the status of their applications, ie Pending, Approved, Rejected, Disbursed.

**3. Loan Approval Workflow**

* Admin Review:

Admins view a list of all pending applications, sorted by risk profile or date.

Admins can approve or reject each application, optionally leaving comments.

* Status Update:

Upon admin action, the system updates loan status and logs the decision in the audit trail.

* User Notification:

Borrower receives notification via email/SMS with the decision and comments.

**4. Repayment Tracking**

* Repayment Submission**:**

Borrowers can record repayments via form or through the integrated M-pesa payment gateway.

Payment method, amount, and timestamp are logged.

* Balance Calculation:

The system recalculates outstanding balance and updates repayment schedule dynamically.

* Overdue Alerts:

If a payment is missed, the system flags the loan as late and sends a reminder.

**5. Notifications**

* Automated Alerts:

Users are notified on key events:

Loan approved/rejected

Payment due or overdue

Payment received

* Channels:

Email via SendGrid and SMS via Twilio

* Read Status:

Users can mark notifications as read in their dashboard.

**6. Reporting**

* Admin Dashboard:

Displays real-time metrics: active loans, pending approvals, total disbursed, repayments received.

* Downloadable Reports:

PDF/Excel reports on loan performance, repayment trends, and user activity.

**7. External API Integrations**

* Credit Scoring API Integration:

Triggered during loan application.

Sends borrower identity to scoring API and receives risk profile.

* Payment Gateway Integration:

Used to collect repayments and disburse approved loans.

Logs transaction status (success/failure) and reference ID.

* Email/SMS API Integration:

Centralized notification handler invokes third-party APIs for message delivery.

Error handling for failed deliveries with retry logic.

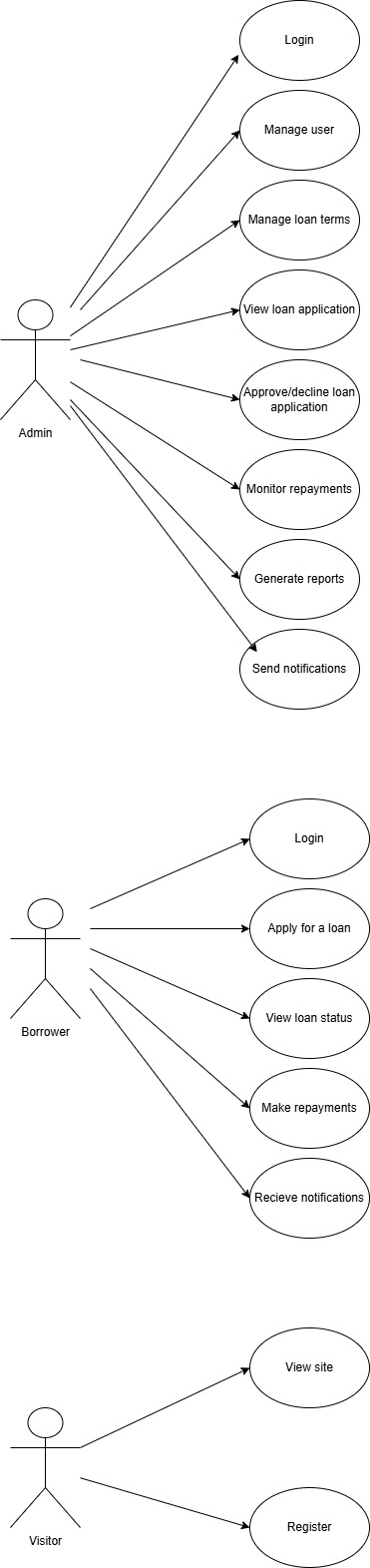
## User Roles & Permissions

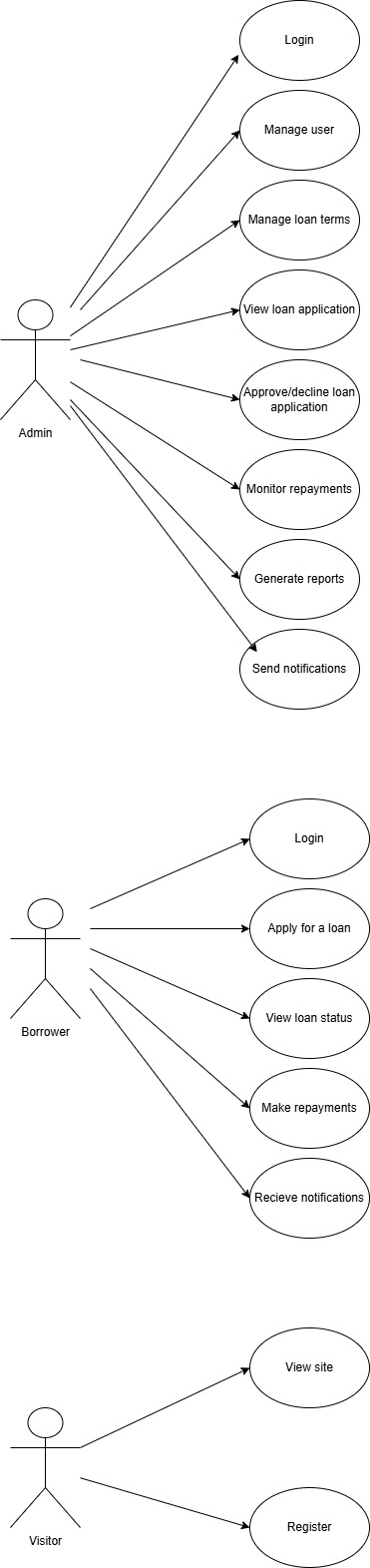
|  |  |  |  |
| --- | --- | --- | --- |
| Functionality | Visitor | Borrower | Admin |
| Register/Login | yes | yes | yes |
| Apply for Loan | no | yes | no |
| View Own Loans | no | yes | yes |
| Approve/Reject Loans | no | no | yes |
| View All Users | no | no | yes |
| View Reports | no | no | yes |
| Receive Notifications | no | yes | yes |

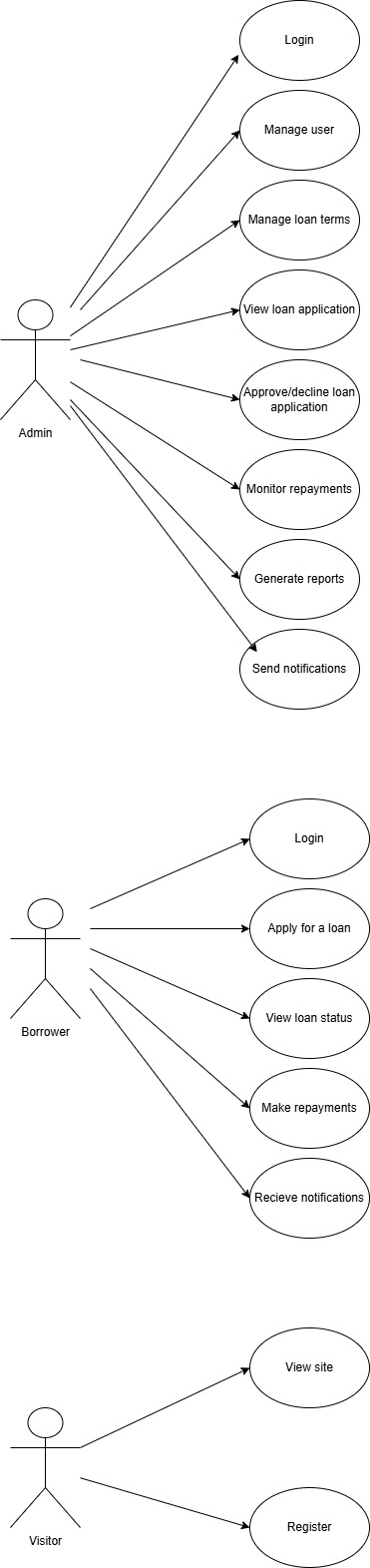
## System Workflows and Diagrams

**Use Case Diagram**

Depicts the functional requirements of the system by showing the interactions between users (actors) and the system.

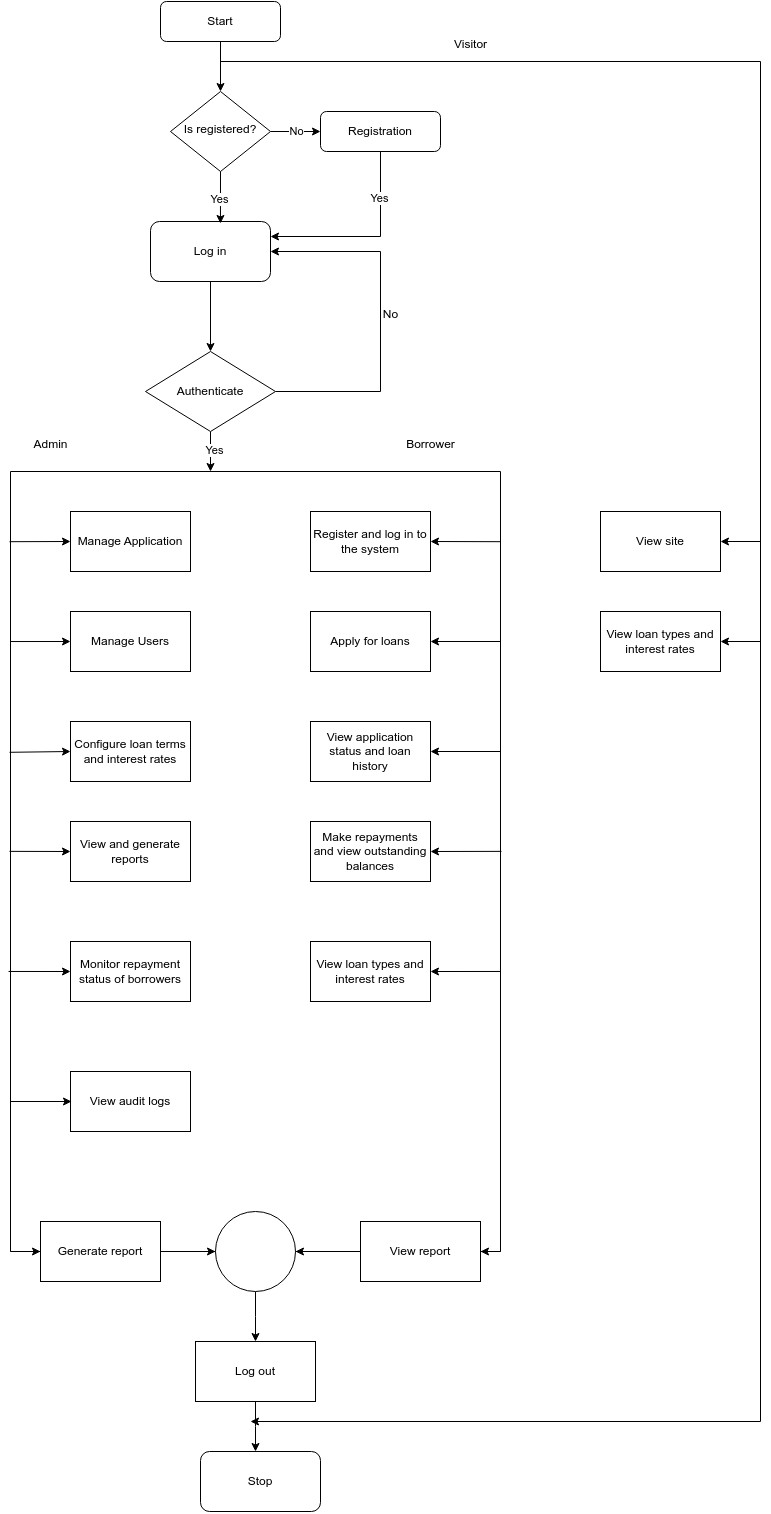






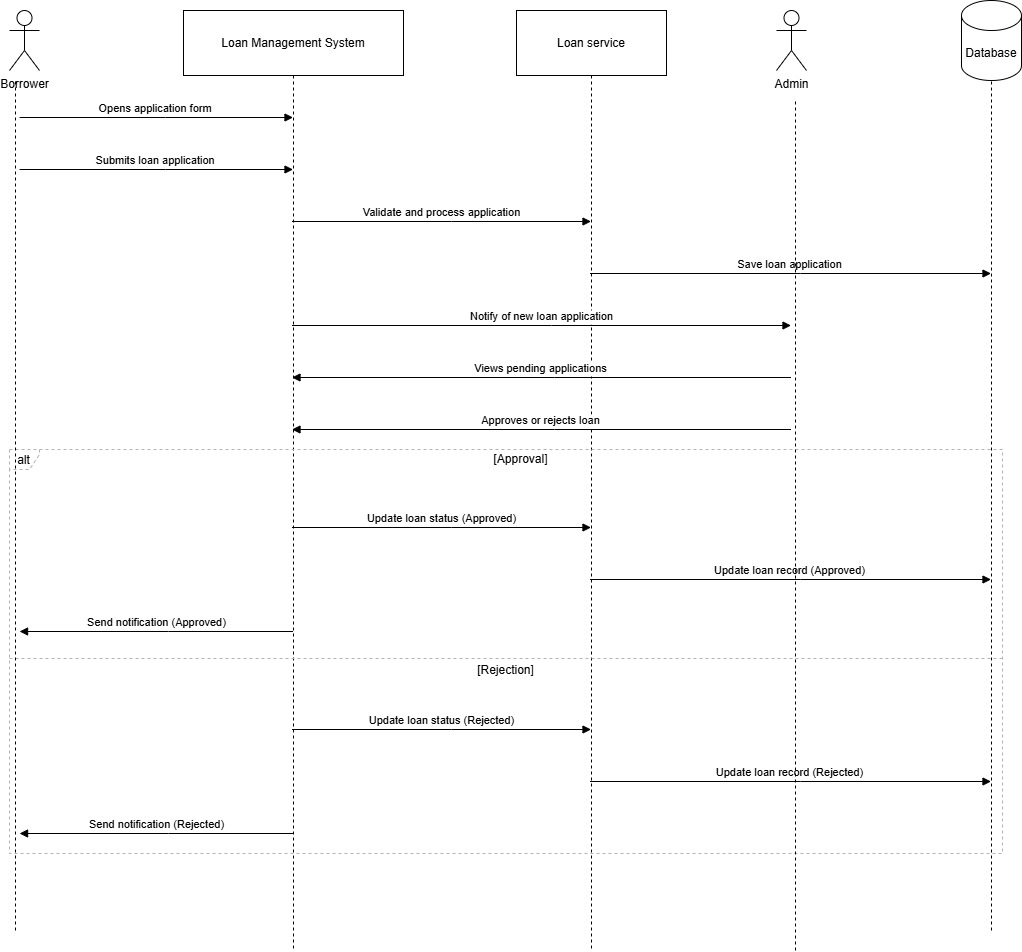
**Activity Diagram**

Demonstrates the step-by-step flow of processes like loan application and approval.



**Unified Modeling Language (UML)**

Depicts the functional requirements of the system by showing the interactions between users (actors) and the system.



## External API Integration

1. **Payment Gateway Integration ( M-Pesa)**

This will facilitate real-time loan disbursements and collection of repayments. It initiates disbursement to the borrower’s mobile wallet (M-Pesa). The API also accepts repayments via mobile payment transactions.

The API verifies transaction success, failure, or timeout. It also returns transaction ID and confirmation code for tracking.

Transactions are authenticated via API keys and webhook validation.

Failed payments trigger retries and user alerts.

1. **Notification Services (e.g., Twilio, SendGrid)**

This API will ensure timely communication with users through email and SMS. It will be used to:-

1. Send OTP during registration/login .
2. Notify users about loan approvals, rejections, disbursements.
3. Send automated payment reminders and overdue alerts.

The API will accept recipient, message content, and delivery channel and log delivery status and timestamps. Failed sends are logged with retry logic and admin notification.

1. **Credit Scoring API (e.g., TransUnion)**

It will assess borrower risk level during loan application. Upon application, the system sends user identity (ie., national ID) to the credit scoring API. Receives a score, risk category, and optional recommendations.

The API system uses the returned score to auto-flag high-risk applications. Admins will view credit report details during the approval process.

Data will be transmitted securely using HTTPS and bearer tokens. Sensitive identity data is encrypted.

## Non-Functional Requirements

1. **Performance**

The system must support at least 50 concurrent active users without degradation in performance.

Average response time for API calls should be less than 2 seconds under normal load.

1. **Security**

Authentication: Use of JSON Web Tokens to manage secure sessions.

Transport Security: All communication must occur over HTTPS with TLS encryption.

Data Validation: All user input must be validated server-side to prevent SQL injection, XSS, and other attacks.

Role-Based Access Control (RBAC): Ensure access to resources is restricted based on user role (admin, borrower).

1. **Scalability**

The system will be deployed on Amazon Web Services (AWS) and designed to scale both horizontally (load balancing) and vertically (resource allocation) as demand increases.

Microservice-ready architecture for future modular expansion.

1. **Logging and Auditing**

All user actions, API access, and administrative tasks must be logged with timestamps.

Logs should be securely stored and accessible only to administrators.

Logs are required for compliance auditing, debugging, and monitoring.

## Assumptions and Constraints

1. **Assumptions**
2. The primary user base is small to medium-sized, with an expected range of 100–500 active users.
3. All users have stable internet access to interact with the system and to allow external API communication.
4. Borrowers have access to a mobile number or email for notifications and identity verification.
5. **Constraints**
6. Internet connectivity is mandatory for system functionality due to reliance on external APIs (e.g., credit scoring, payment gateways).
7. Real-time payment processing and credit verification are dependent on the uptime and responsiveness of third-party services (e.g., M-Pesa).
8. The system will initially support only English as the interface language.
9. Budget and development time constraints may limit implementation of advanced features such as dynamic credit modeling or AI-based fraud detection in the current phase.