**Loan Management System Documentation**

**Title:** Loan Management System Documentation  
**Author:** Aditya Kumar Mahalik  
**Date:** 18/10/2024

**Table of Contents**

1. Introduction
2. Features
3. Requirements
4. Installation
5. Usage
6. API Endpoints
7. Error Handling
8. Conclusion

**1. Introduction**

The Loan Management System is a web application built using Flask for the backend, HTML/CSS/JavaScript for the frontend, and MySQL as the database. This system allows users to register, apply for loans, check the status of their applications, and allows admins to approve or reject loan applications.

**2. Features**

* **User Registration & Login**: Secure registration and login functionality with email validation, password hashing, and session management.
* **Loan Application**: Users can apply for loans with validations on loan amount, tenure, and purpose.
* **Loan Status API**: Users can check the status of their loan applications, ensuring access to only their applications.
* **Admin Dashboard**: Admins can approve or reject loans with proper role-based access control.
* **Error Handling & Notifications**: Comprehensive error messages for both frontend and backend operations.

**3. Requirements**

**Backend Dependencies**:

* Flask==2.3.3
* Flask-SQLAlchemy==2.5.1
* mysqlclient==2.1.1 (or PyMySQL==1.0.2)
* Werkzeug==2.3.3
* Flask-Migrate==3.1.0
* Flask-Login==0.6.2
* Flask-WTF==1.0.1
* email-validator==1.3.1

**Frontend Dependencies**:

* HTML/CSS/JavaScript for form handling and interaction.

**4. Installation**

1. **Clone the repository**:

bash

Copy code

git clone [repository\_url]

cd loan\_management\_system

1. **Create a virtual environment** (optional but recommended):

bash

Copy code

python -m venv venv

source venv/bin/activate # On Windows use: venv\Scripts\activate

1. **Install dependencies**:

bash

Copy code

pip install -r requirements.txt

1. **Create the MySQL database**:
   * Follow the instructions in the backend setup section to create a database and user.
2. **Run the application**:

bash

Copy code

python app.py

**5. Usage**

* **User Registration**: Navigate to /register to create an account.
* **User Login**: Navigate to /login to log in.
* **Apply for a Loan**: After logging in, go to /apply\_loan to submit a loan application.
* **Check Loan Status**: Users can check the status at /loan\_status.
* **Admin Actions**: Admins can approve or reject loans via /admin/approve\_loan/<loan\_id>.

**6. API Endpoints**

* **POST /register**: Register a new user.
  + **Request Body**: { "email": "user@example.com", "password": "your\_password", "name": "Your Name" }
* **POST /login**: Log in an existing user.
  + **Request Body**: { "email": "user@example.com", "password": "your\_password" }
* **POST /apply\_loan**: Apply for a loan.
  + **Request Body**: { "loan\_amount": 100000, "loan\_tenure": 2, "purpose": "Home Renovation" }
* **GET /loan\_status**: Retrieve loan status for the logged-in user.
* **POST /admin/approve\_loan/<loan\_id>**: Approve or reject a loan application.
  + **Request Body**: { "decision": "approve" } or { "decision": "reject" }

**7. Error Handling**

The system provides user-friendly error messages for:

* Invalid input during registration and login.
* Loan application errors, including duplicate applications.
* Unauthorized access to certain endpoints.

Backend logs errors for further investigation using Flask's logging system.

**8. Conclusion**

The Loan Management System provides a robust framework for managing loan applications. With proper validation, error handling, and admin functionalities, it ensures a secure and efficient process for both users and administrators.