Tribhuvan University

Faculty of Humanities and Social Sciences

A Project Report

On

Loan Approval Prediction System Using Random Forest Algorithm

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

Submitted to

Department of Computer Application

Samriddhi college

Lokanthali, Bhaktapur

Submitted by

Aanchal Neupane [111302001]

Anup Kasula [111302004]

Under the Supervision of

Mr. Jeeban Dhungel

October, 2023

Tribhuvan University

Faculty of Humanities and Social Sciences

Samriddhi College

Supervisor's Recommendation

I hereby recommend that this project prepared under my supervision by **Aanchal Neupane, Anup Kasula** entitled "**Loan Approval Prediction System**" in partial fulfillment of the requirements for the degree of Bachelor of Computer Application is recommended for the final evaluation.

____

Signature

Jeeban Dhungel

Supervisor

BCA Department

Smariddhi College

Tribhuvan University

Faculty of Humanities and Social Sciences

Samriddhi College

LETTER OF APPROVAL

This is to certify that this project prepared by Aanchal Neupane and Anup Kasula entitled **"Loan Approval Prediction System"** 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 degree.

____________________ **SIGNATURE of Supervisor** Jeeban Dhungel BCA Department Samriddhi College	____________________ **SIGNATURE of** **HOD/Coordinator**
____________________ **SIGNATURE of** **Internal Examiner**	____________________ **SIGNATURE of** **External Examiner**

Acknowledgement

This project is prepared in the partial fulfillment of the requirement for the degree of Bachelor in Computer Application (BCA). The satisfaction and success of completion of this task would be incomplete without heartfelt thanks to people whose constant guidance, support and encouragement made this work successful. On doing this undergraduate project we have been fortunate to have help, support and encouragement from many people we would like to acknowledge them for their cooperation.

Our first thanks goes to Tribhuvan University for designing such a worthy syllabus and making us do this project. Our next batch of thanks goes to the faculty of Management of Samriddhi College without whose help our project would have been impossible. This list includes Principal of Samriddhi College, **Mr. Sandeep Shrestha.** Our very sincere and heartfelt thanks go to **Mr. Jeeban Dhungel** our project supervisors who constantly guided us through the project time period. Without his guidance, our project would have been impossible. Last but not the least we want to thank every direct and indirect hands that were involved in completion of this project.

This project has been a wonderful experience where we have learnt and experienced many beneficial things.

Aanchal Neupane [111302001]

Anup Kasula [111302004]

Abstract

Loan prediction is a critical area in banking and insurance, where data analysis plays a pivotal role to understand patterns in the data. Loan prediction involves the use of machine learning methods such as Random Forest, implemented in Python with tools like Google Colab.

The project “Loan Approval Prediction System” focuses on using machine learning, specifically, Random Forest to predict loans. The machine learning model is created using Python and implemented through a Flask API. The frontend of the project is designed using Next.js, while the backend is developed using Node.js. The system delivers accurate results with reasonable loss during training and validation. The results suggest that the model has good accuracy. The system can improve this work by focusing on achieving even higher accuracy in the future.

Key Words: Loan , Prediction System, Random Forest

Table of Contents

List of Tables

List of Figures

List of Abbreviation

ANN	: Artificial Neural Networks (ANN)
API	: Application Program Interface
CSS	: Cascading Style Sheet
GDRP	: General Data Protection Regulation
HTML	: Hyper Text Markup Language
I/O	: Input/Output
JS	: Java Script
PC	: Personal Computer
SQL	: Structured Query Language
SVM	: Support Vector Machine
UI	: User Interface
UML	: Unified Modeling Language
WSGI	:Web Server Gateway Interface

Chapter 1 . Introduction

1.1 . Introduction

With the increasing number of customers approaching banks and financial institutions for loans, the risk of loan defaulting has become a significant concern. Loan approval is a crucial decision for banks as it directly impacts their profitability and stability. To mitigate this risk, banks are turning to advanced technologies such as machine learning and computer science to predict the probability of defaulting for potential borrowers. By analyzing various factors, these technologies enable banks to assess an individual's capacity to repay a loan within the proposed terms. And, Laon Approval Prediction is a similar system that which perform predictive analysis to make informed decisions and minimize the chances of non-performing loans, ultimately safeguarding the stability of the banking sector and reducing the risk of bankruptcy.

The project focuses on designing an intuitive user interface that allows loan officers or financial institutions as well as general users to input applicant information, such as personal details, income, credit history, and employment status. This data serves as the input for the loan approval prediction algorithm i.e., Random Forest Algorithm, which analyzes various factors to generate a prediction of loan approval or rejection. The system aims to provide a clear and understandable prediction result, enabling loan officers to make well-informed decisions based on the algorithm's output. Additionally, the UI may include features such as data validation, error handling, and notifications to ensure data accuracy and enhance user experience throughout the loan application process.

The Loan Approval Prediction project not only benefits lending institutions but also applicants seeking loans. By leveraging historical loan data and advanced analytics, the system can assess loan applications objectively and fairly. This approach reduces bias and subjectivity in the decision-making process, providing applicants with a transparent and reliable loan approval prediction. Moreover, the project contributes to the overall efficiency of loan processing, potentially reducing turnaround time for loan approvals and enhancing customer satisfaction. The accurate prediction of loan approvals can assist applicants in making informed financial decisions and improving their chances of successful loan applications.

1.2 . Problem Statement

The problem this project aims to address is the difficulty faced by applicant when issuing the loan. It also helps loan officer to find the person who is capable to pay loan. The loan approval process in banks and financial institutions is a critical decision-making task that involves assessing the creditworthiness and risk associated with potential borrowers. The increasing number of loan applicants further exacerbates the challenge of accurately and efficiently evaluating loan requests. The traditional approach heavily relies on manual assessment by loan officers, leading to inconsistencies and inefficiencies in the decision-making process. Additionally, the risk of approving loans to customers with a high probability of defaulting poses a significant threat to the financial stability of the institutions. So, introducing the loan approval prediction system can be a useful tool for loan approval for the developing country like Nepal.

1.3 . Objectives

The objectives of Loan Approval Prediction System are:

i. To predict loan by using machine learning Random Forest Algorithm.

ii. To provide information about personal loan.

1.4 . Scope and Limitation

The scope of a Loan Approval Prediction System is that it helps banks and lenders decide whether to approve or reject loan applications. It uses data to make predictions about whether a person is likely to repay a loan or not using Random Forest machine learning algorithm.

However, it also has limitations. It provides loan approval status based on general condition but not specific to the any banking particular banking rule. While it's a useful tool, it's not completely reliable, and lenders should also take other factors into account when deciding on loans.

1.5 . Development Methodology

System development is a process through which a product will get completed or a product gets rid from any problem. Software development process is described as a number of phases, procedures and steps that gives the complete software. It follows series of steps which is used for product progress. The development method followed in this project is waterfall model.

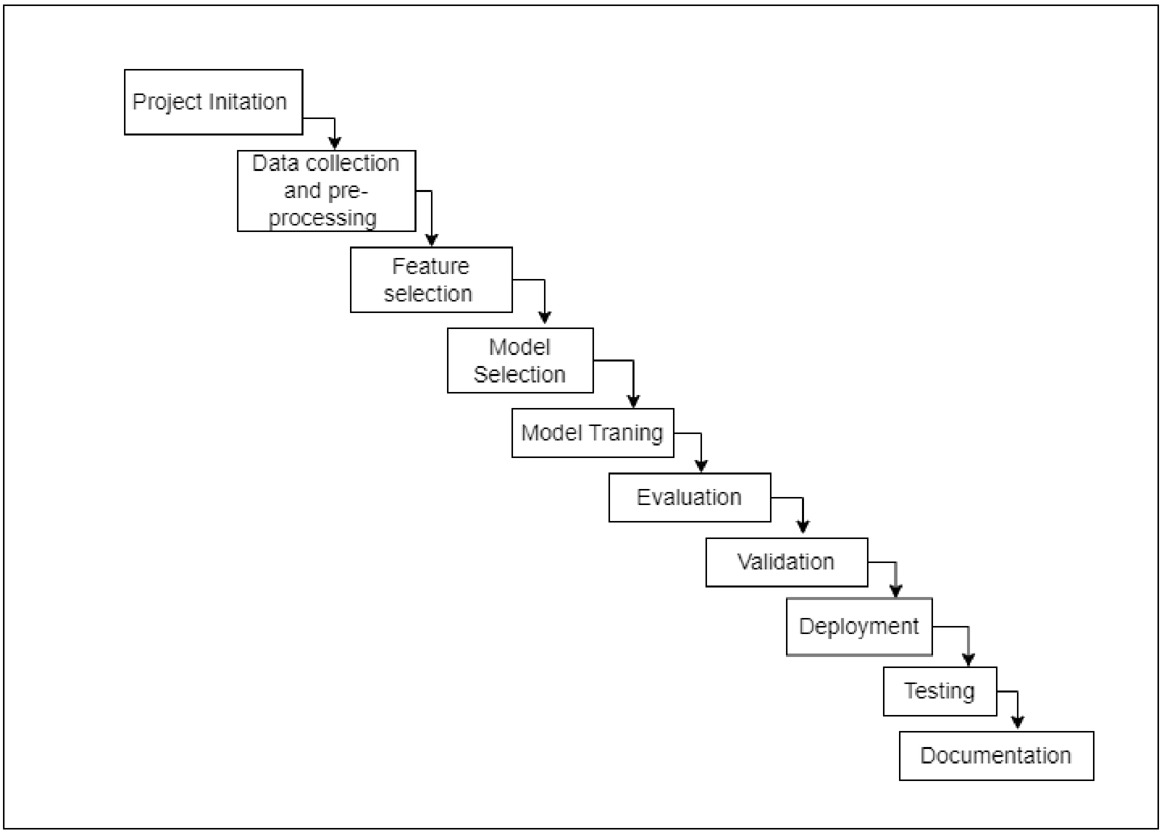

Figure 1. 1 Waterfall Model of Loan Approval Prediction System

The development methodology for a Loan Prediction System typically involves several key steps:

i. Project Initiation: The project's goals, objectives, and scope are identified and gather initial requirements.

ii. Data Collection and Preprocessing: Historical loan data was collected, and it was cleaned and preprocessed to eliminate inconsistencies and missing values. This step is crucial because the quality of data significantly affects the accuracy of predictions.

iii. Feature Selection: Relevant features (variables) were identified from the dataset, and new features were created if necessary. Feature selection was employed to enhance the model's performance and reduce complexity.

iv. Model Selection: The appropriate machine learning or statistical models for loan prediction were chosen. Common choices, depending on the complexity of the problem, included logistic regression, decision trees, random forests, or neural networks.

v. Model Training: A portion of the data was used to train the selected model, and this involved adjusting model parameters to fit the data accurately.

vi. Model Evaluation: The model's performance was evaluated using metrics like accuracy, precision, recall, and F1-score.

vii. Validation: The model's hyperparameters were fine-tuned to optimize its performance.

viii. Deployment: Once a satisfactory model was achieved, it was integrated into the Loan Approval System. An API or user interface was created for interacting with the model.

ix. Testing: The entire system was thoroughly tested to ensure it functioned as expected and delivered accurate loan predictions.

x. Documentation and Testing: The entire development process was documented, including data sources, model architecture, and deployment details. A maintenance plan was implemented for regular updates and monitoring of the model's performance.

1.6 . Report Organization

The report consists of four chapters. The “Introduction” is the first chapter which contains whole information about the project “Loan Approval Prediction System”. The introduction chapter contains the introduction, problem statement, objectives and scope of the project. The second chapter includes the background study and literature review. The chapter three is known as system analysis and design which include many sub chapters. The sub chapters inside the chapter three are: system analysis and system design which is further divided into sub topics. The system analysis includes requirement analysis (functional analysis and non-functional analysis), feasibility study (Technical feasibility, operational feasibility, economic feasibility and scheduling feasibility), object diagram and sequence diagram. The system design consists of state modeling, dynamic modeling and deployment modeling the project. The fourth chapter includes implementation and testing with result analysis. The last chapter contains two parts: Conclusion and Future recommendation.

Chapter 2 . Background Study and Literature Review

2.1. Background Study

In Nepal, the loan approval prediction process in banks and financial institutions traditionally involves manual assessment by loan officers. They evaluate loan applications based on various factors, including the applicant's credit history, income, collateral, and other relevant financial information. The decision-making process is subjective and heavily relies on the experience and judgment of the loan officers.

In recent years, there has been a growing interest in leveraging technology and data-driven approaches for loan approval prediction in Nepal. Some financial institutions have started exploring the use of machine learning and data analytics techniques to enhance the accuracy and efficiency of loan assessment. But, still there is not a platform for general people or applicant to predict either they are applicable for issuing the loan or not.

The "Loan Approval Prediction System" project has a primary objective: to bridge the existing gap in the loan approval process within Nepal's financial institutions. Traditional methods involve manual assessment by loan officers, which can be time-consuming, subjective, and reliant on human judgment.

The scope of the project encompasses the development of a robust machine learning model, specifically utilizing the Random Forest algorithm, implemented in Python and accessible through a Flask API. Random Forest is a versatile machine learning algorithm known for its ability to handle complex datasets and improve predictive accuracy by aggregating predictions from multiple decision trees. It works by creating a collection of decision trees and combining their results to make robust and accurate predictions in various applications, from classification to regression tasks.

The project's architecture includes Python, Flask API, Next.js, and Node.js as key components, forming a cohesive foundation for its development. Python is widely preferred for developing machine learning models due to its extensive libraries, simplicity, and rich ecosystem, making it an ideal language for data analysis, model training, and deployment. Flask API is a lightweight and flexible framework for building web APIs in Python, facilitating the creation of web-based applications with ease [1]. Next.js is a powerful JavaScript framework that simplifies server-side rendering and provides an efficient, optimized, and user-friendly approach for building modern web applications. Node.js, which serves as a JavaScript runtime, is employed to efficiently handle data processing and manage server-side operations within the project [2].

2.2. Literature Review

With the increase in banking sector many people are applying for loans in bank. All these loans are not approvable. The main income of bank assets comes from gain earned from loans. The main objective of banks is to invest their assets in safe customers. Today many banks approve loan after many processes of verification and validation but still there is no surety that selected customer is safe or not. Therefore, the "Loan Approval Prediction System Using Machine Learning" by Associate Professor and Head of Department of Computer Science and Engineering from Hyderabad, India concluded it is important to apply various techniques in banking sector for selecting a customer who pays loan on time [3].

The paper with title "Bank Loan Approval Prediction Using Machine Learning Techniques" by Ndayisenga, Theoneste concluded that financial institutions should use machine learning techniques because it saves money and time for both sides. [4].

In depth analysis of credit scoring in various fields with research paper "Credit Scoring, Statistical Techniques and Evaluation Criteria" by Hussein A. Abdou and John Pointon in 2011 concluded that there is no single overall best classification technique for credit scoring models [5].

The top 10 important features from the models were selected and then used in the modeling process to test the stability of binary classifiers by comparing their performance on separate data in 2018 by authors of paper "Credit Risk Analysis Using Machine and Deep Learning Models" observe that the tree-based models are more stable than the models based on multilayer artificial neural networks. [6]

Mostafa Yousofi Tezerjan and Azizollah Memariani author of "A hybrid model for credit scoring in complex systems" concluded that using machine learning system will allow the lending bank to be informed of the outcome of the current situation and credit history of its customers, which constitutes the customer [7].

The article title "Loan default prediction using decision trees and random forest" concluded that "the Random Forest algorithm demonstrated superior performance compared to the Decision Tree algorithm, achieving significantly greater accuracy". [8].

Paper introduced by Lifeng Zhou and Hong Wang* in 2012 with title Loan Default Prediction on Large Imbalanced Data Using Random Forests shows that parallel random forests can greatly improve random forests' efficiency during the learning process. [9].

Svraka, and Nalica introduced the article “Bank Loan Prediction Using Machine Learning Techniques” which conclude models typically assign a numerical score (1 or 0), which represents the creditworthiness of the borrowers based on historical data [10].

The article purposed by D. J. HAND and W. E. HENLEY with the title “Statistical Classification Methods in Consumer Credit Scoring” concluded the statistical work in credit scoring and credit control to date has focused on the conceptually relatively straightforward aspect of constructing improved discriminating rules [11].

The paper “Secure Loan Prediction System Using Artificial Neural Network (ANN)” conclude that the use of a decision support system and the expert system has helped to solve major issues in the banking industry and the world at large and also increasing the accuracy of the decision [12].

To determine the maximum relevant features, i.e. the factors that have the most impact on the prediction outcome, various ML algorithms such as Random Forest, Support Vector Machine, K-Nearest Neighbor and Logistic Regression, were used. These mentioned algorithms are evaluated with the standard metrics and compared with each other. The random forest algorithm achieves better accuracy. This paragraph is the conclusion of the article “Comparative Analysis of Customer Loan Approval Prediction using Machine Learning Algorithms” [13].

.

Chapter 3 . System Analysis and Design

3.1. System Analysis

The system analysis process of loan approval prediction sytem involves conducting a requirement analysis and feasibility study.

3.1.1. Requirement Analysis

a. Functional Requirement

Functional requirements are product features or functions those developers must implement to enable users to accomplish their tasks. The functional requirement of our project:

i. Login/Register

Loan approval predictor allows the user to create a new account or log into an existing one.

ii. View Applicant

The admin of the Loan approval predictor can view the list of the applicant or the user how predict the loan approval using this system.

iii. Input loan specification

Loan approval predictor enables the user to input the specifications. So, that they can get information about their loan request approval.

iv. Predict Loan

The loan approval predictor informs about approval loan by using machine learning model.

v. Download Individual Prediction.

The loan approval predictor enables the user to download a report containing the predicted status and other details of their own.

vi. Download Prediction.

The loan approval predictor enables the admin to download a report containing the predicted price and other details of all users.

vii. Manage Blog.

The admin of the system can insert, update and delete the bog through the admin dashboard.

viii. Feedback

Users can send feedback, and admin can view and assess it for improvements and enhancements.

ix. Logout

The system allows the user to log out from their account.

x. Manage Bank Information

The admin can insert the bank loan information by using admin dashboard.

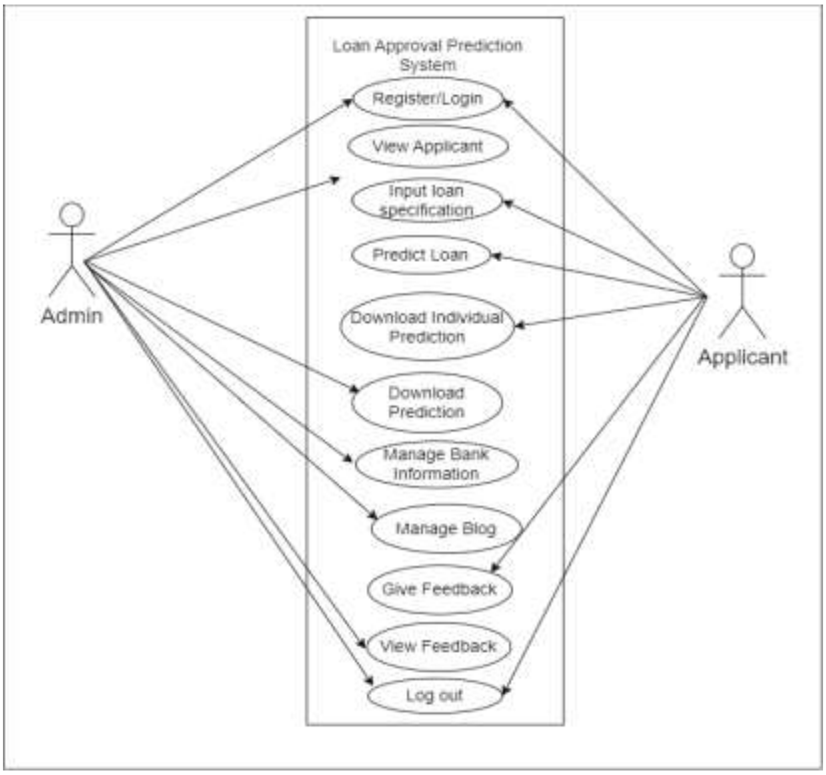

Figure 3. 1 Use Case Diagram of Loan Approval Prediction System

b. Non-Functional Requirement

Non-functional requirements specify the qualities and characteristics that a loan approval prediction system should possess.

i. Performance

The system has fast response times to ensure efficient processing of loan predictions. It handles a high volume of loan information and large datasets without significant performance degradation.

ii. Availability

The system can operate 24 hours per week and 365 days a year. As long as the user, do not shut down the desktop.

iii. Usability

The user interface is intuitive, user-friendly, and easy to navigate for loan officers and other authorized users.

iv. Portability

Loan approval prediction shall run in any platform.

v. Compliance

The system complies with relevant regulatory requirements and industry standards, such as data protection regulations (e.g., GDPR) and financial industry regulations.

3.1.2. Feasibility Study

i. Technical

The technologies chosen, such as Next Js for the frontend and Django for the backend, are widely used and well-documented. There are sample online resources, tutorials, and communities available to assist with any technical challenges that may arise during development. The machine learning random forest algorithm, being a well-established technique, can be implemented using various libraries and frameworks in Python. With the required knowledge and skills in web development, machine learning, and programming, the project can be successfully executed. So, this project is technically feasible.

ii. Operational

The project is operationally feasible. Within the timeframe the project can be developed as it is a college final year project. The required tasks of data collection, preprocessing, model training, and frontend-backend development can be accomplished efficiently. Effective project management, resource allocation, and communication among team members will be essential to ensure successful completion within the allocated time.

iii. Economic

As this is a college final year project without a specified budget, economic feasibility is not a major concern. The project can be implemented using open-source tools and resources, which are freely available, and hosting costs can be minimized by opting for low-cost or free hosting options. Since the necessary hardware and software resources are typically available within the college or personal computers, the project can be completed without significant economic expenses.

iv. Schedule

The development process is scheduled to encompass the design phase and extend until the end of the semester, provides with approximately six months. While achieving a flawless final system within this timeframe might be challenging. But, following waterfall model with exact mindset allows us to allocate enough time to create an initial, functional version of the software. The rough schedule of the project is shown below (using Gantt chart):

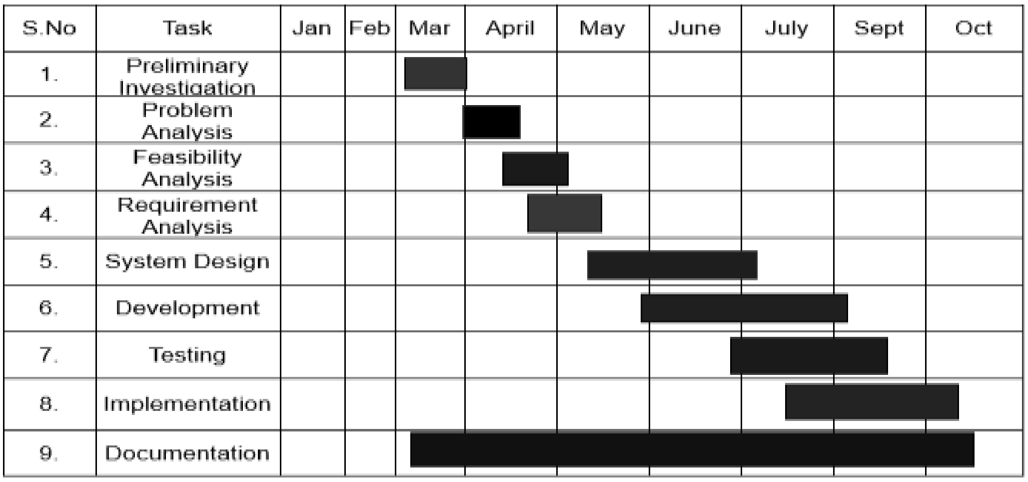

Figure 3. 2 Gantt Chart of Loan Approval Prediction System

The above Gantt Chart for the Loan Approval Prediction System illustrates the timeline for completing each phase of the project development life cycle.

3.1.3. Object Diagram

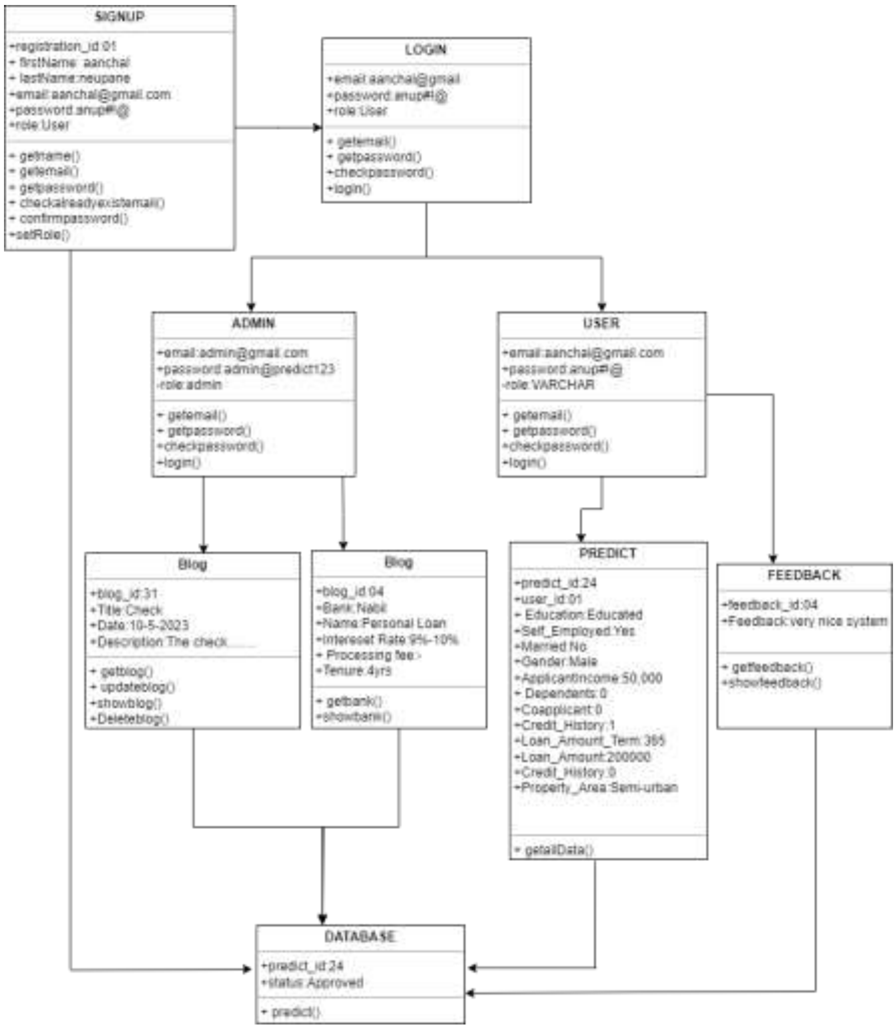

Figure 3. 3 Object Diagram of Loan Approval Prediction System.

The provided object diagram display the components and connections within the "Loan Approval Prediction System." It consists of various objects, including "Signup," "Login," "User" (specifically, the "Applicant" role), "Admin," "Blog," "Feedback," and "Prediction." These objects represent key elements of the system and their interrelationships, offering a visual representation of how they interact and function within the system's framework.

3.1.4. Sequence Diagram

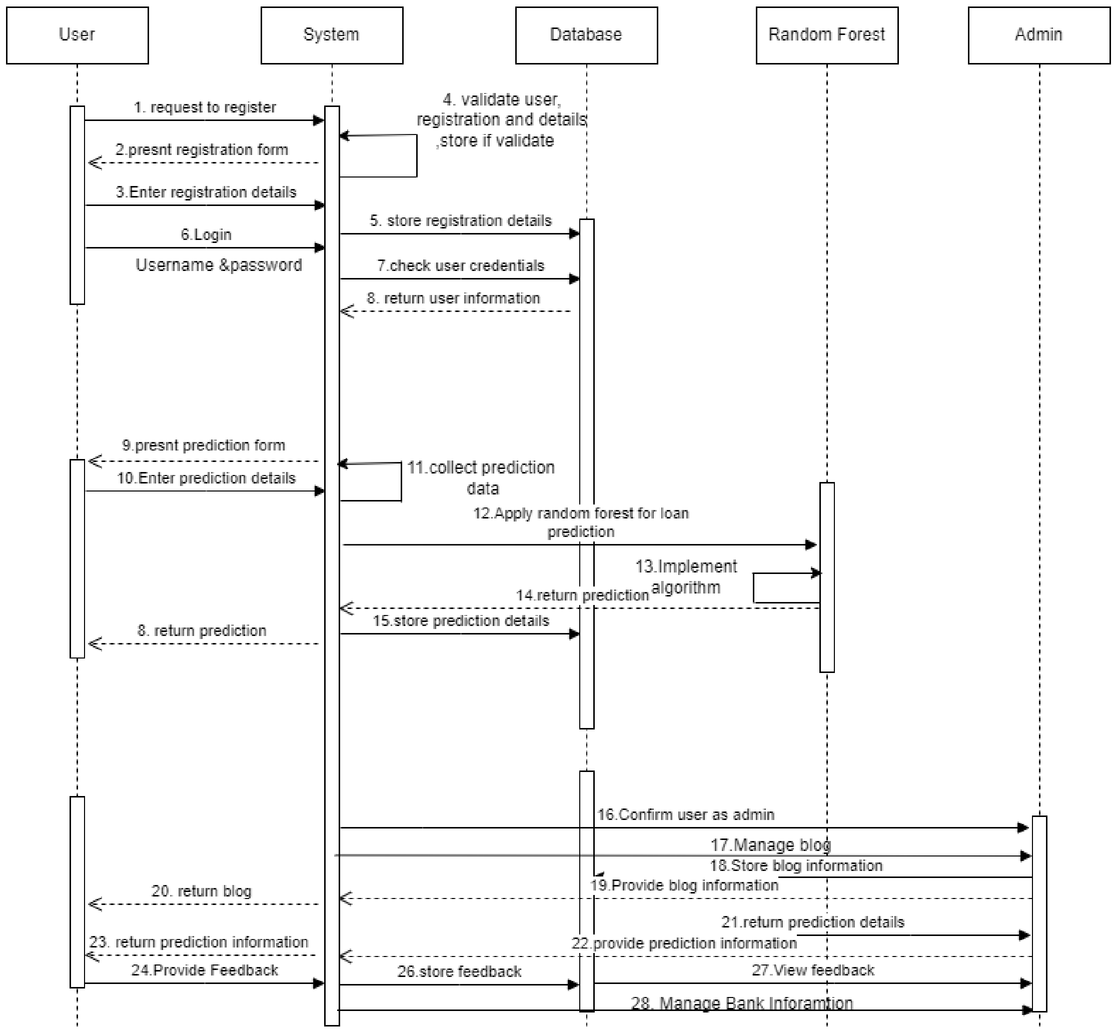

Figure 3. 4 Sequence diagram of Loan Approval Prediction System

The given sequence diagram shows the interaction between different components of the Loan Approval Prediction System using Random Forest.

The diagram shows the interactions between the user, system, database, random forest, and admin. The user initiates the interaction by filling prediction form to find out their status about approval or disapproval of Loan.

The admin can log into the system, add blog about loan information and view all the applicant status about loan. Overall, this sequence diagram provides a high-level overview of the interactions between different components of the Loan Approval Prediction System.

3.1.5. Activity Diagram

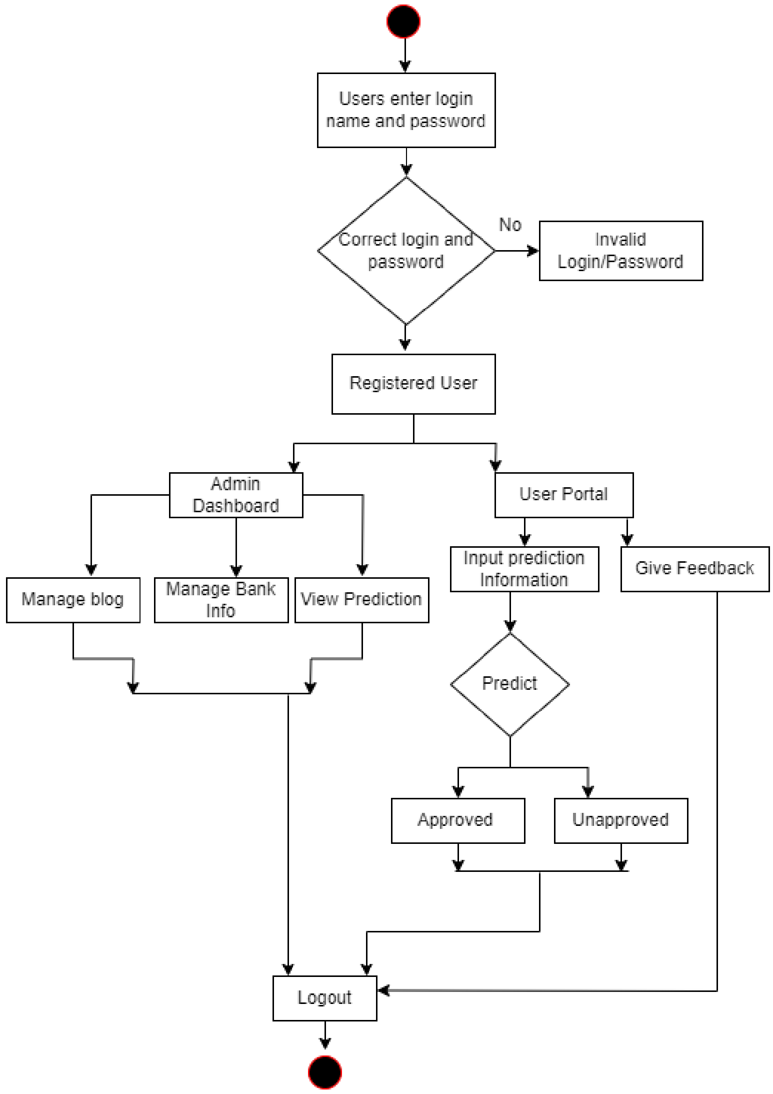

Figure 3. 5 Activity diagram of Loan Approval Prediction System using Random Forest

The activity diagram of Loan Approval Prediction System starts with the Login activity. After the user logs in, the system checks if the user is registered in the database. If not, the system prompts the user to register. Once the user is registered, the system checks whether the user is an admin or a regular user. If the user is an admin, the system allows the admin to manage blog and view information of users with status of loan approval. Once the admin completes the activities, they can log out of the system.

If the user is a regular user, the system provides prediction form where user can predict the loan approval. If the status is approved than approved message is popup and if not than unapproved message is obtained. Once the user perform prediction, they can log out of the system.

3.2. System Design

System design consist the process how system plan was created for a software or hardware system, including its components and how they interact in the Loan Approval Prediction System.

3.2.1. State Diagram

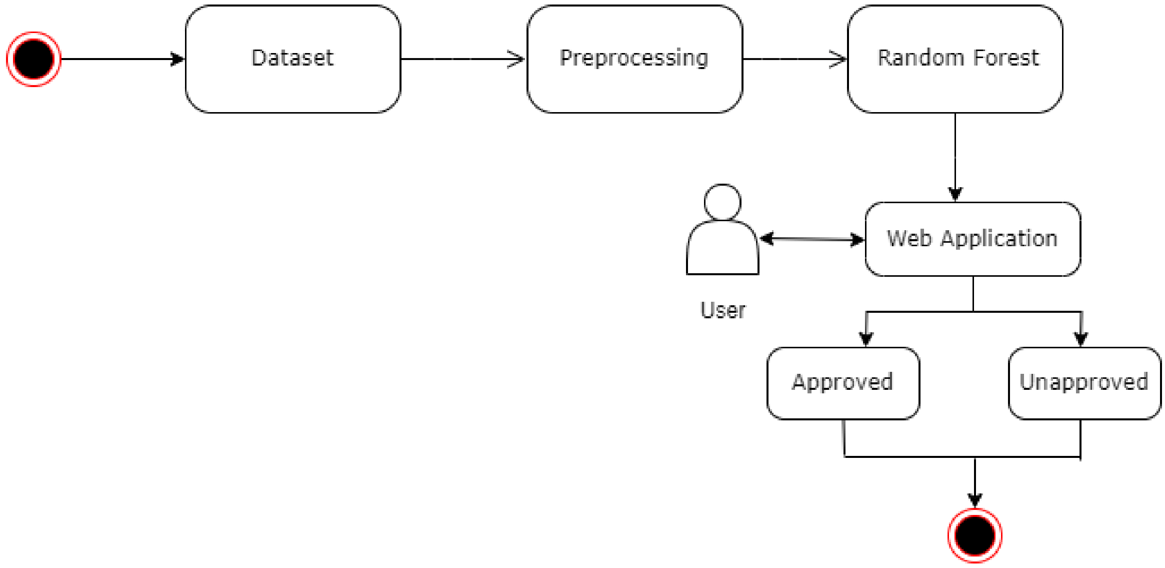

Figure 3. 6 State diagram of Loan Approval Prediction System using Random Forest.

The above state diagram illustrates the various states of "Loan Approval Prediction System" that a system or entity can be in and how transitions between these states occur in response to certain events or conditions.

The system starts in the Preprocessing state, where the raw loan application data is cleaned and prepared for training the random forest model. Once the data is preprocessed, the system transitions to the ML model with Random Forest state. In this state, the random forest model is trained on the preprocessed data. After the model is trained, the system moves to the "Web Application" state. In this state, the system is ready to accept loan applications from users and predict whether each application is likely to be approved.

When a user submits a loan application, the system transitions to the Approved or Unapproved state, depending on the prediction of the random forest model. If the model predicts that the loan is likely to be approved, the system transitions to the Approved state. Otherwise, the system transitions to the Unapproved state.

3.2.2. Component Diagram

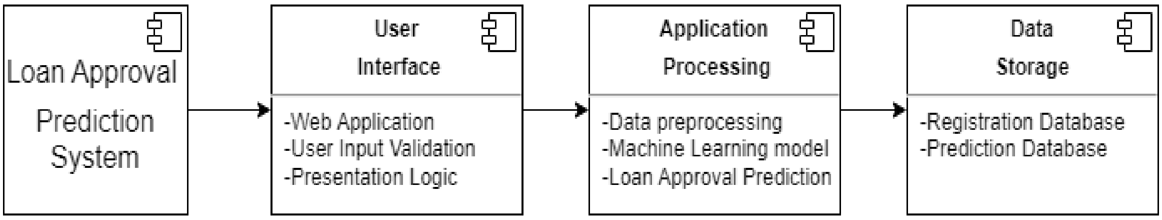

Figure 3. 7 Component Diagram of Loan Approval Prediction System.

A component diagram for a loan approval prediction system visually represents the software components used in the system and shows how they interact and depend on each other. The user interface handles user interactions and includes a web application for users to submit loan applications. It also manages user input validation and presentation logic. The application processing component is responsible for processing loan applications. It includes sub-components like data preprocessing and a machine learning model for prediction. The data storage component stores various data relevant to the system, such as loan application details and user profiles. It typically includes databases to manage this information.

3.2.3. Deployment Diagram

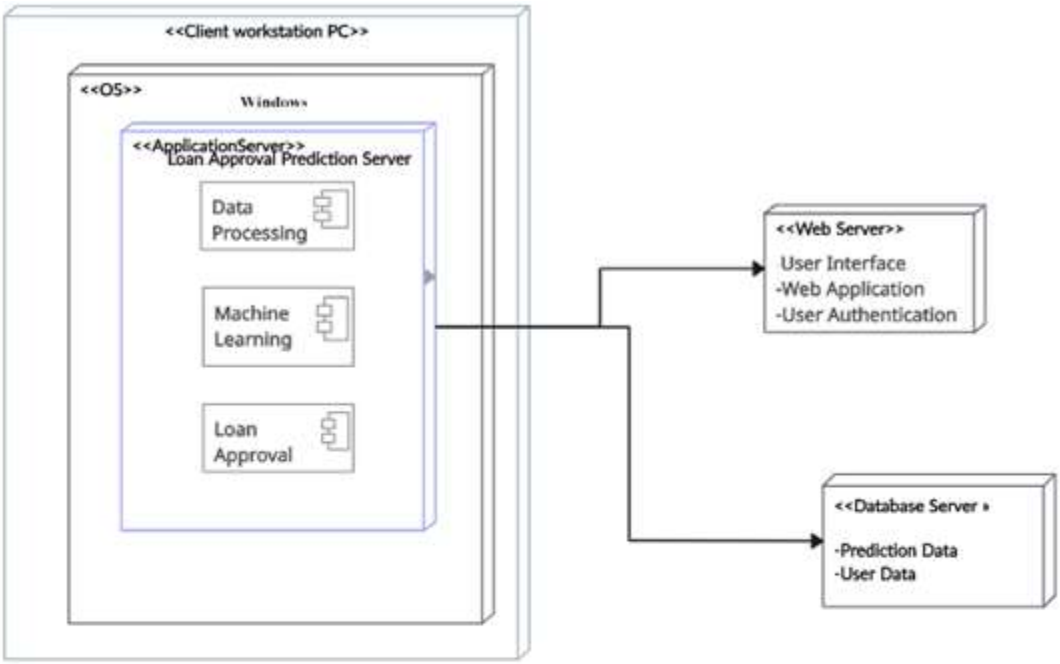

Figure 3. 8 Deployment Diagram of Loan Approval Prediction System.

A deployment diagram shown above illustrates the physical deployment of software components and hardware nodes in a system architecture of “Loan Approval Prediction System”. It shows how different components of a web application work together to process a loan application.

The Client Workstation PC is the user's computer, where they can access the loan prediction web page. The web page is hosted on a Web Server, which is responsible for delivering the web page to the user's browser.

When the user submits their prediction form, the web server forwards the data to an Application Server. The application server is responsible for processing the loan prediction and making a decision on whether or not to approve it.

The application server stores the loan prediction data and user information in a Database Server. This data can be used to generate reports and track loan performance.

3.2.4. Algorithm Details

Random Forest is a popular machine learning algorithm used for both classification and regression tasks. It is an ensemble learning technique that combines the predictions of multiple decision trees to improve overall prediction accuracy and reduce overfitting. At the core of a Random Forest are decision trees. Decision trees are simple, tree-like structures that make decisions by splitting the data based on feature values. Random Forest can handle missing values in the dataset without much preprocessing.

Step 1 : The class is initialized with three parameters: n_estimators (number of decisions trees in the ensemble), max_depth (maximum depth of each decision tree), and random_state (seed for random number generation).

Step 2 : An empty list, estimators, is created to store the trained decision trees.

Step 3 : The fit method is used to train the random forest classifier on the input data X and corresponding labels y. It iterates n_estimators times to create multiple decision trees.

Step 4 : np.random.choice is use to ensure diversity among the trees inside fit method.

Step 5 : DecisionTreeClassifier is created inside fit method with the specified max_depth and random_state.

Step 6 : Train the decision tree on the selected features by calling the fit method with the appropriate subset of the input data X and labels y.

Step 7 : Append the trained decision tree, along with the selected features, to the estimators list, forming the ensemble of decision trees for the random forest classifier inside fit method.

Step 8 : Create an predict method with empty array, predictions, to store the predictions from each decision tree in the ensemble.

Step 9 : Iterate over each decision tree in estimators in predict method.

Step 10 : Obtain the final predictions by applying majority voting along each row of the predictions array.

Step 11 : Use np.apply_along_axis and np.argmax(np.bincount(x)) to select the most frequent class label in predicts method.

Step 12 : Return the majority vote predictions.

Mathematically,

The mathematical representation of the Random Forest algorithm is as follows:

Let h1(x), h2 (x), ...,hT (x) be a set of T decision trees. The prediction of the Random Forest for a new data point x is given by:

y^(x)=model (h 1 (x), h 2 (x), ...,h T (x))

where mode is the function that returns the most frequent value in a list.

Each decision tree h t (x) is trained on a random sample of the training data, and a random subset of features is considered at each split. This technique, known as bootstrapping, helps to reduce overfitting and improve the generalization performance of the model.

The decision tree is calculated by calculating information gain from target attribute and for remaining attribute entropy is calculated to find the gain of the node.

Information Gain (IG) = -P/(P+N) Log2(P/(P+N)) -N/(P+N) Log2(P+N) where P = positive result, N = negative result.

Entropy (Attribute) = Information Gain (IG) * Probability

Gain = Information Gain of Target Attribute – Entropy (Attribute)

Chapter 4 . Implementation and Testing

4.1. Implementation

4.1.1 Tools Used

No special hardware requirements were necessary to develop the loan approval prediction system using model based on Random Forest algorithm. The system was built and tested on a 64-bit computer running Windows. This configuration was sufficient to handle the data processing and model training tasks required for the project.

The various software tools used to develop this project are as follows:

a. CASE tools

CASE tools used in Loan Approval Prediction System:

i. Figma

Figma is used to design wireframes for the initial case. Icons, logos, and assets can be easily created using Figma.

ii. Visual Studio Code

Visual Studio Code is used because it is a code editor that is employed for redefining and optimizing the development and debugging of modern web and cloud applications. Instant productivity is achieved by us through features such as syntax highlighting, bracket-matching, auto-indentation, box-selection, snippets, and more provided by VS Code.

iii. Google colab

Feature extraction, data preprocessing, and the implementation of the Random Forest Algorithm were executed in Google Colab due to its provision of free access to computing resources, facilitating the creation of the model.

b. Programming Languages

i. JavaScript:

Next.js a popular React framework for constructing server-rendered web applications is used for frontend of "Loan Approval Prediction System". Node.js a server-side JavaScript run-time environment capable of executing asynchronous I/O through its event-driven architecture is used for backend development.

ii. Python:

Essential packages like SKlearn, NumPy, Pandas, Matplotlib, and the Flask framework are utilized for feature extraction, data preprocessing, training ,testing and calculation of performance measure.

iii. Flask:

Flask API is used to deploy machine learning model with “Loan Approval Prediction System”.

c. Database Platforms

The structured query language (SQL) has been used to interact with the database, allowing for easy handling of tables and execution of various data operations. It is reliable, fast, and widely supported, making it a preferred choice for many database-driven applications.

4.1.2 Implementation Details of Module

a. Predictive module

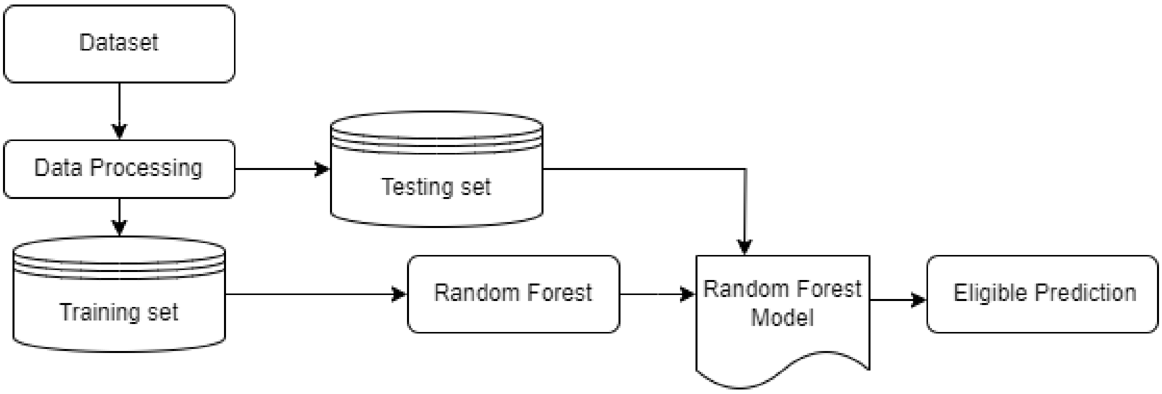

Figure 4. 1 Block Diagram of Loan Approval Prediction System

The diagram illustrates how the data set is processed and divided into the training set and testing set. The training set is utilized to train the random forest model. Once the model has been trained, it is assessed on the testing set. The model's performance on the testing set is then utilized to evaluate how effectively the model will generalize to new data.

1. Dataset

The datasets being used were obtained from Kaggle, and they were in the form of CSV files named "LoanApprovalPrediction.csv," containing information about applicants, along with a target attribute. The dataset consisted of 13 features: Loan_id, Gender, Married, Dependents, Education, Self_Employed, ApplicantIncome, CoApplicantIncome, LoanAmount, Loan_Amount_Term, Credit_History, Property_Area, and Loan_Status. The dataset was imported in Google colab by using pandas which is the libabry of python for working with dataset. The pandas have function to analyze, clean, explore and manipulate the data.

```
df.head()
```

Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
Male	No	0.0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
Male	Yes	1.0	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
Male	Yes	0.0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
Male	Yes	0.0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
Male	No	0.0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y

Figure 4. 2 Loan Approval Prediction System Dataset

2. Data Processing

The data were processed using various steps to clean, preprocess and format the data so that it can be used in machine learning model. The task like checking null value, handling missing value and encoding categorical data were done.

The steps implemented in data processing in "Loan Approval Prediction Model" are:

a. The information about columns of dataset were printed using df.info method.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 598 entries, 0 to 597
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            598 non-null    object
 1   Gender             598 non-null    object
 2   Married            598 non-null    object
 3   Dependents         586 non-null    float64
 4   Education          598 non-null    object
 5   Self_Employed      598 non-null    object
 6   ApplicantIncome    598 non-null    int64
 7   CoapplicantIncome  598 non-null    float64
 8   LoanAmount         577 non-null    float64
 9   Loan_Amount_Term   584 non-null    float64
 10  Credit_History     549 non-null    float64
 11  Property_Area      598 non-null    object
 12  Loan_Status        598 non-null    object
dtypes: float64(5), int64(1), object(7)
```

Figure 4. 3 Information about columns or features for preprocessing data

b. The count of null values for each column in the DataFrame 'df' were calculated and printed.

```
# Get the null values in the DataFrame
null_values = df.isnull().sum()

print(null_values)
```

```
Loan_ID               0
Gender                0
Married               0
Dependents           12
Education             0
Self_Employed         0
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount           21
Loan_Amount_Term     14
Credit_History       49
Property_Area         0
Loan_Status           0
dtype: int64
```

Figure 4. 4 Count null values of each column in Dataframe

c. Count the number of categorical variables in a DataFrame.

```
obj = (df.dtypes == 'object')
print("Categorical variables:",len(list(obj[obj].index)))
```

```
Categorical variables: 7
```

Figure 4. 5 Count the number of categorical variables in a DataFrame

d. Drop the unique and completely not correlated column.

```
df.drop(['Loan_ID'],axis=1,inplace=True)
```

Figure 4. 6 Drop the unique and completely not correlated column.

e. Use scikit-learn's LabelEncoder to transform categorical variables in a DataFrame into numerical labels.

```
# Import label encoder
from sklearn import preprocessing
# label_encoder object knows how
# to understand word labels.
label_encoder = preprocessing.LabelEncoder()
obj = (df.dtypes == 'object')
for col in list(obj[obj].index):
  df[col] = label_encoder.fit_transform(df[col])
```

Figure 4. 7 Transform categorical variables in a DataFrame into numerical labels

f. Fill the missing values (NaNs) in each column of a DataFrame df with the mean of that respective column and then checking for any remaining missing values.

```
for col in df.columns:
 df[col] = df[col].fillna(df[col].mean())
df.isna().sum()
```

Figure 4. 8 Fill the missing values in each column of a DataFrame

3. Feature selection.

A DataFrame `df` is used to define the feature matrix `X` and the target variable `y` for a machine learning task. The feature matrix `X` is created by selecting specific columns from the DataFrame, including 'Gender,' 'Married,' 'Dependents,' 'Education,' 'Self_Employed,' 'ApplicantIncome,' 'CoapplicantIncome,' 'LoanAmount,' 'Loan_Amount_Term,' 'Credit_History,' and 'Property_Area.' These columns contain the input features used for making predictions. The target variable `y` is defined as 'Loan_Status,' which represents the variable to be predicted or classified. This setup prepares the data for training a machine learning model, with `X` containing the input features and `y` representing the target variable that the model will predict.

```
X = df[['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome',
        'LoanAmount', 'Loan_Amount_Term', 'Credit_History', 'Property_Area']]
y = df['Loan_Status']
```

Figure 4. 9 Selected features form dataset after data preprocessing

4. Training set and Testing set

The `train_test_split` function from scikit-learn is used to split a dataset into training and testing sets. It takes input features X and corresponding target values y as inputs. The `test_size` parameter is set to 0.2, meaning 20% of the data will be allocated to the testing set, while 80% will be used for training. The `random_state` parameter, set to 42, ensures reproducibility by seeding the random number generator. After execution, `X_train` and `y_train` contain the training data, while `X_test` and `y_test` hold the testing data. This partitioning allows for the evaluation of machine learning models on an independent dataset to assess their performance and generalization capabilities.

```
# Split the data into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 4. 10 Split data into training and testing sets

5. Random Forest

The custom `RandomForestClassifier` class, shows the behavior of a random forest ensemble model. It initializes with parameters like the number of decision trees (n_estimators), maximum depth of trees (max_depth), and random state. In the `fit` method, it trains multiple decision trees on random subsets of features from the input data (X) and stores these trees in the ensemble. The `predict` method then combines the predictions of individual trees through majority voting, where each tree's output is considered, and the class with the most votes is chosen as the final prediction for each input sample. The custom class approximates the behavior of a random forest classifier by using a collection of decision trees for classification.

```
class RandomForestClassifier:
    def __init__(self, n_estimators=100, max_depth=None, random_state=None):
        self.n_estimators = n_estimators
        self.max_depth = max_depth
        self.random_state = random_state
        self.estimators = []
    def fit(self, X, y):
        np.random.seed(self.random_state)
        for _ in range(self.n_estimators):
            # Randomly select a subset of features
            selected_features = np.random.choice(X.shape[1], size=int(np.sqrt(X.shape[1])), replace=False)
            # Create a decision tree classifier
            dt = DecisionTreeClassifier(max_depth=self.max_depth, random_state=self.random_state)
            # Train the decision tree on the selected features
            dt.fit(X[:, selected_features], y)
            # Append the trained decision tree to the ensemble
            self.estimators.append((selected_features, dt))
    def predict(self, X):
        # Make predictions for each decision tree in the ensemble
        predictions = np.zeros((X.shape[0], len(self.estimators)))
        for i, (selected_features, dt) in enumerate(self.estimators):
            predictions[:, i] = dt.predict(X[:, selected_features])
        # Aggregate predictions using majority voting
        majority_votes = np.apply_along_axis(lambda x: np.argmax(np.bincount(x)), axis=1, arr=predictions)
        return majority_votes
```

Figure 4. 11 Random forest algorithm in model

6. Random Forest Model

A Random Forest classifier is instantiated with specific configuration settings. The `n_estimators` parameter is set to 7, indicating that the ensemble will consist of 7 decision trees. The `criterion` parameter is set to 'entropy,' specifying that the decision trees will use the entropy criterion to measure impurity. The `random_state` is set to 7, ensuring reproducibility. Next, the `fit` method is called on the `rf_classifier` object, which trains the ensemble of decision trees on the provided training data (`X_train` and `y_train`). The ensemble model will use the entropy-based impurity measure and consists of 7 decision trees.

```python
# Instantiate a Random Forest classifier
rf_classifier = RandomForestClassifier(n_estimators=7,
                                       criterion='entropy',
                                       random_state=7)

# Fit the classifier on the training data
rf_classifier.fit(X_train, y_train)
```

Figure 4. 12 Instantiate a Random Forest classifier

7. Eligible Prediction

The trained Random Forest classifier `rf_classifier` is used to make predictions on a set of test data `X_test`. The `predict` method of the classifier is called with the test data as input, and the resulting predictions are stored in the variable `y_pred`. These predictions represent the model's output for the test data and can be used for further analysis and evaluation of the classifier's performance.

```python
# Use the trained model to make predictions on the test data
y_pred = rf_classifier.predict(X_test)
```

Figure 4. 13 Make predictions on model using test data set

b. Admin Module

The Admin Module is an important component of our system that provides administrative privileges and control over various aspects of the application. Here are the implementation details:

i. View User Predictions:

The admin has the ability to access and view all predictions made by users within the system. This feature allows the admin to gain insights into the trends and patterns of user predictions.

ii. Manage Blog Posts:

Within the admin module, there is a functionality that enables the admin to manage the content of the blog section. They can upload new blog posts to share information or updates, delete outdated posts, and update existing ones to keep the content relevant and informative.

iii. Manage Bank Information:

The system administrator can input bank loan details via the admin dashboard, which will then be displayed on the loan details page of the website.

iv. View feedback:

The administrator has the ability to access and review feedback provided by system users. This information serves as valuable input for potential modifications and improvements to the system in the future.

c. User Module

The User Module serves as a platform for authorized users to make predictions about loan approvals and retrieve their historical prediction records. Here are the details:

i. Authorized User Login:

To access the system's prediction capabilities, users must log in with their authorized credentials. This login process ensures that only authenticated individuals can use the prediction service.

ii. View Past Predictions:

Users also have the ability to access and review their past loan approval predictions. This feature allows users to track the accuracy of their previous predictions and assess their financial decisions over time.

iii. Give Feedback

Users of the system have the opportunity to provide feedback based on their experiences within the system. This feedback mechanism allows them to share their thoughts, opinions, and observations about their interactions with the system. They can use this feature to offer insights, suggestions, or comments to help enhance and improve the system for both themselves and other users.

4.2. Testing

The purpose of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements. The following things were done during the process of "Loan Approval Prediction System" testing:

- Tests were planned before testing begun.
- The entire tests were prepared as per users' requirements.
- Analytical tools were used to develop test cases.
- A testing strategy was adopted and applied.

4.2.1. Test cases for Unit Testing

Unit testing is done to validate that each unit of the software code performs as expected. It is done during the coding phase of an application by the developers. This test is done to determine the working of the individual modules. Loan Approval Prediction System includes the various modules that are tested at the development processes.

Table 4. 1 Test case for Login

Test case id	1		
Test case description	User Login		
Prerequisites	User must be registered Enter the valid username and password. Click login		
Test scenario 1	User enter a wrong username		
Test data	Username: abc@gmail.com Password: student		
Step	Expected output	Actual Result	Pass/fail
1	Incorrect information	Username or Password is Invalid	pass
Test scenario 2	User enter a right password		
Test data	Username: example@gmail.com Password: abcd		
Step	Expected output	Actual Result	Pass/fail
1	Correct information	Username or Password is Invalid	Pass
Test scenario 3	User enter all details successfully		
Test data	Username: example@gmail.com Password: example1		
Step	Expected output	Actual Result	Pass/fail
1	User account login	Login Successful	Pass

Table 4. 2 Test case for admin login

Test Case ID	2		
Test Description	Login as admin		
Prerequisites:	1.Valid id and password		
Test Scenario	1. Enter the valid id and password. 2. Click login		
Test Data	User name: admin@gmail.com Password: admin@prediction123		
Step	Expected Result	Actual Result	Pass/Fail
1.	Logged in.	Login Successful.	Pass

Table 4. 3 Test case for blog management by admin

Test Case ID	3		
Test Description	Add the blog		
Prerequisites:	All the field must be filled.		
Test Scenario	1.Enter the title and description 2. Click submit button. 3.Click update button. 4.Update the title. 5.Click Delete button.		
Test Data	Title: Loan rate of Nabil Bank Ltd. Description: From Shrawan 24, Nabil Bank Ltd has decided…		
Step	Expected Result	Actual Result	Pass/Fail
1. 2. 4.	Blog Added and Published. Title Updated Delete Success	Blog Added and Published. Title Updated Delete Success	Pass Pass Pass

4.2.2. Test case for system testing

Since this test concentrates on each unit of the software as implemented in source code, the test was conducted keeping every small module in consideration. The following is the report of a few bugs that were overcome during the test and those remain unchecked.

Table 4. 4 Test case for system testing.

S.N.	Bugs:	Solution:
1	Lack of input validation leading to unexpected inputs. **Error description:** Invalid Input.	Validate the input data in proper way.
2	System predict that an applicant will be approved for a loan, but they are actually denied. **Error description:** Incorrect predictions.	Train the model with error free data, select appropriate feature and avoid overfitting.
3	The system crashes when a user tries to submit a loan application with incomplete information. **Error description:** Unexpected result.	Handle incomplete form properly. Notify the user about the information that is missing.

Most of the error is caused by human because human develop system and manipulation is done by human themselves only, hence nobody can deny the possibility of having committing some or other mistake or the existence of bugs. Even though, extreme condition should be taken care for maintaining the data and respective entries of the system for the testing of report. A set of test materials is nothing more than a list of possible problems in a program and a set of procedure for determining whether the problems actually exist and are significant or not.

4.3. Result Analysis

The result analysis of the loan approval prediction system involves assessing the accuracy and performance of the predictions made by the system. This analysis typically includes evaluating metrics such as precision, recall, F1 score, and accuracy to determine how well the model is performing in classifying loan prediction as approved or unapproved.

Confusion Matrix:

The confusion matrix is a powerful tool for assessing the performance of classification models. It can help us to understand the strengths and weaknesses of our models and to identify areas where they need to be improved. Various performance metrics, including accuracy, precision, recall, and F1 score, can be computed using the confusion matrix. The confusion matrix of the Loan Approval Prediction System is shown below:

Figure 4. 14 Confusion Matrix of Loan Approval Prediction System.

Accuracy: The accuracy of the classifier is the percentage of loan applications that it correctly predicted.

Mathematically,

Accuracy = (TP + TN) / (TP + TN + FP + FN) =0.825

An accuracy of 0.825 means that a machine learning model is correctly predicting the target value 82.5% of the time. It means the model has average performance for performing prediction.

Precision:

The precision of the classifier is the percentage of loan applications that it predicted to be approved that were actually approved.

Mathematically,

Precision = TP / (TP + FP) = 0.84530

A precision of 0.84530 means that, of all the instances that the model predicted to be positive, its correct about 84.53% of the time. The precision value shows that the model is relatively good at predicting true positive.

Recall:

The recall of the classifier is the percentage of loan applications that were actually approved that the classifier predicted to be approved.

Mathematically,

Recall = TP / (TP + FN) =0.916167.

A recall of 0.91616 means that the model correctly identified 91.616% of all the positive instances. The model is quite effective at identifying most of the true positive cases.

F1 score:

The F1 score is a harmonic mean of precision and recall. It is a good metric to use when both precision and recall are important.

Mathematically,

F1 score = 2 * (Precision * Recall) / (Precision + Recall) = 0.87931.

An F1 score of 0.87931 means that the model is performing well overall. It is able to identify both true positives and true negatives with a high degree of accuracy.

Receiver operating curve:

The Area Under the Curve (AUC) is a critical metric in machine learning that quantifies the performance of a binary classification model. It plots the true positive rate (TPR) against the false positive rate (FPR) at different classification thresholds. A higher AUC indicates a better classifier. A perfect classifier would have an AUC of 1.0, while a random classifier would have an AUC of 0.5.

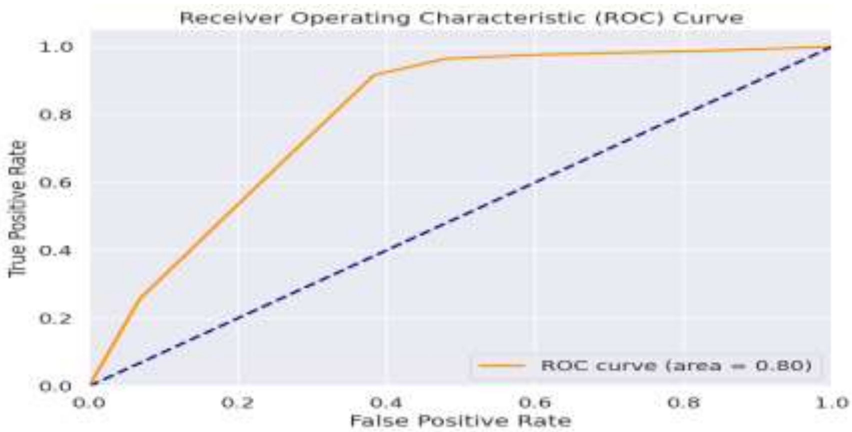

Figure 4. 15 ROC curve of Loan Approval Prediction System

An AUC of 0.80 indicates that it is able to distinguish between positive and negative instances with 80% accuracy. This is a good performance, but it is not perfect. There is still room for improvement, and the classifier could be made more accurate by using more data, training for longer, or using a different machine learning algorithm.

Chapter 5 . Conclusion and Future Recommendation

5.1. Conclusion

In conclusion, a project that incorporates Random Forest for Loan Approval Prediction is developed. Loan Approval Prediction System uses machine learning to predict the approval or disapproval of loan. It uses data management to store and organize the data of users, blog and prediction information. Furthermore, the prediction model was integrated with Flask, a lightweight Python web framework, to improve the user experience and offer a user-friendly interface for system interaction. The system allows the administrator to manage the blog post and view user with loan status by providing easy to use interface. Due to the power of machine learning, loan approval prediction system successfully predicts loan approval outcomes with a good degree of accuracy.

5.2. Future Recommendation

There are many different machine learning algorithms like linear regression, support vector machine, K-nearest neighbor and more that can be used for loan approval prediction. The experiment can be performed with different algorithms to see which one performs best on dataset and can implement the algorithm with higher accuracy. Another way to further improvement is to use more data and feature to train model to improve accuracy. The ensemble learning can be implemented which is the technique that combines the predictions of multiple machine learning models to produce a more accurate prediction. The system can consider both individual user information and financial and risk assessment parameters to make accurate and compliant predictions about loan approval. This helps banks to streamline the loan approval process and make better decisions about which loans to approve.

References

[1] P. S. Lokhande and F. Aslam, "Efficient way of web development using python and flask," aiktclibrary.org, 2015.

[2] N. Chhetri, "A Comparative Analysis of Node.js (Server-Side JavaScript)," St. Cloud State, 2016.

[3] G. S. S. P. L. Srinivasa Murthy, "Loan Approval Prediction System Using Machine Learning," *Journal of Innovation in Information Technology,* p. 4, 2020.

[4] T. NDAYISENGA. [Online]. Available: http://www.dr.ur.ac.rw/bitstream/handle/123456789/1437/THEONESTE%20NDAYISENGA.pdf?sequence=1&isAllowed=y.

[5] H. A. a. J. Pointon, " Credit scoring, statistical techniques and evaluation criteria," *Intelligent Systems in Accounting, Finance & Management,* vol. 18, pp. 2-3, 2011.

[6] M. P. G. D. H. B. Addo, "Credit Risk Analysis Using Machine and Deep Learning Methods," *Finance & Management,* p. 4, 2018.

[7] M. Y. T. &. A. S. Samghabadi, "A hybrid model for credit scoring in complex systems," *ScienceDirection,* p. 3, 2013.

[8] A. K. C. K. R. J. a. P. N. Mehul Madaan1, "Loan default prediction using decision trees and random forest," *IOP Conference Series: Materials Science and Engineering,* p. 13, 2021.

[9] H. W. Lifeng Zhou, "Loan Default Prediction on Large Imbalanced Data," *TELKOMNIKA Indonesian Journal of Electrical Engineering,* vol. 10, p. . 1519~1525, 2012.

[10] N. &. Švraka, "Bank Loan Prediction Using Machine Learning," *University of Rwanda,* p. 10, 2018.

[11] H. &. Henley, "Statistical Classification Methods in Consumer Credit," *Journal of Royal Statistical Society,* vol. 35, 1997.

[12] O. O. A. R. O. J. O. O. A. A. A. MARION O. ADEBIYI1, "SECURED LOAN PREDICTION," *Journal of Engineering Science and Technology,* vol. 17, 2022.

[13] P. Tumuluru, L. R. Burra, M. Loukya, S. Bhavana, H. CSaiBaba and N. Sunanda, "Comparative Analysis of Customer Loan Approval Prediction using Machine Learning Algorithms," *ieeexplore,* 2022.

Appendices

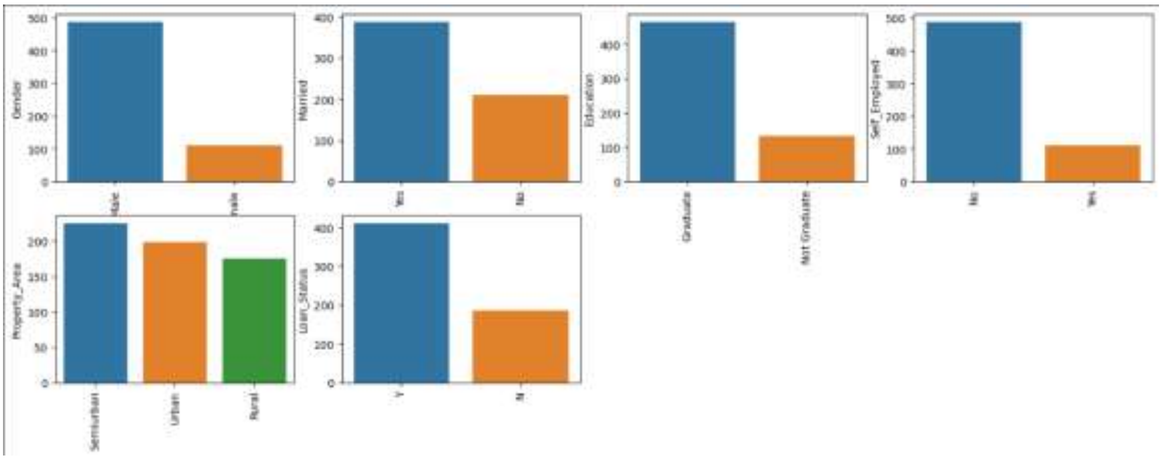

Figure 1: Bar Diagram of Loan Prediction Model

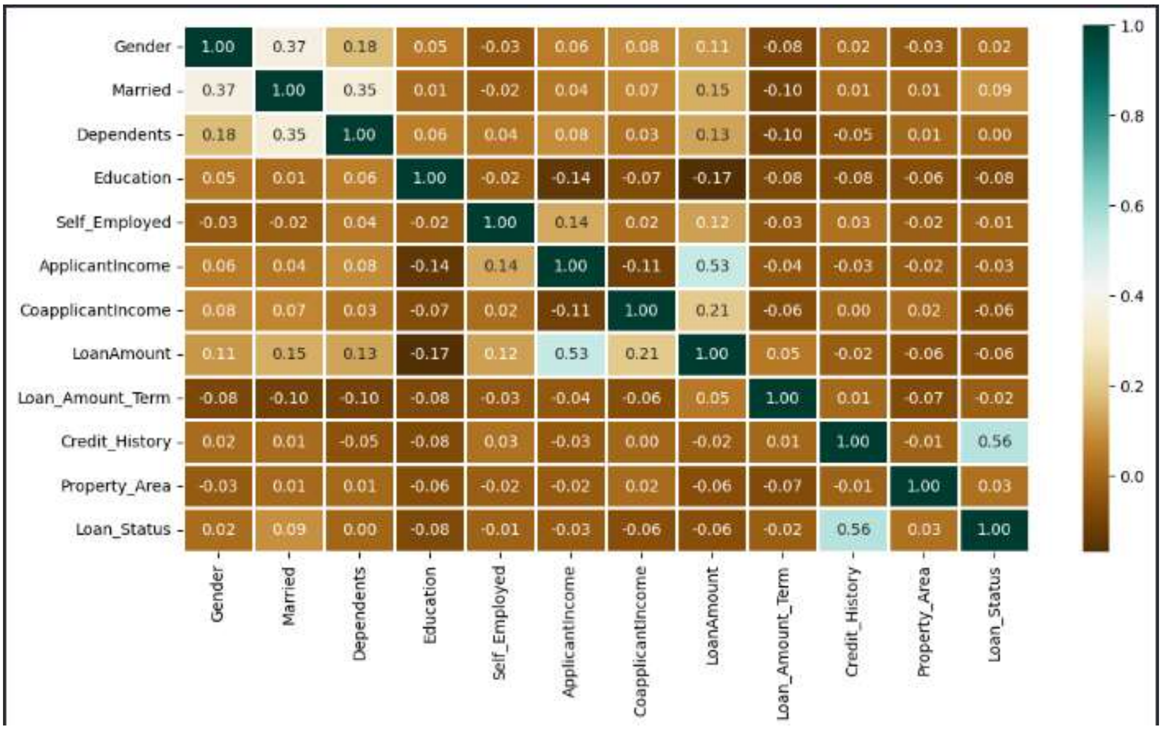

Figure 2 : Heat Map of Loan Prediction Model

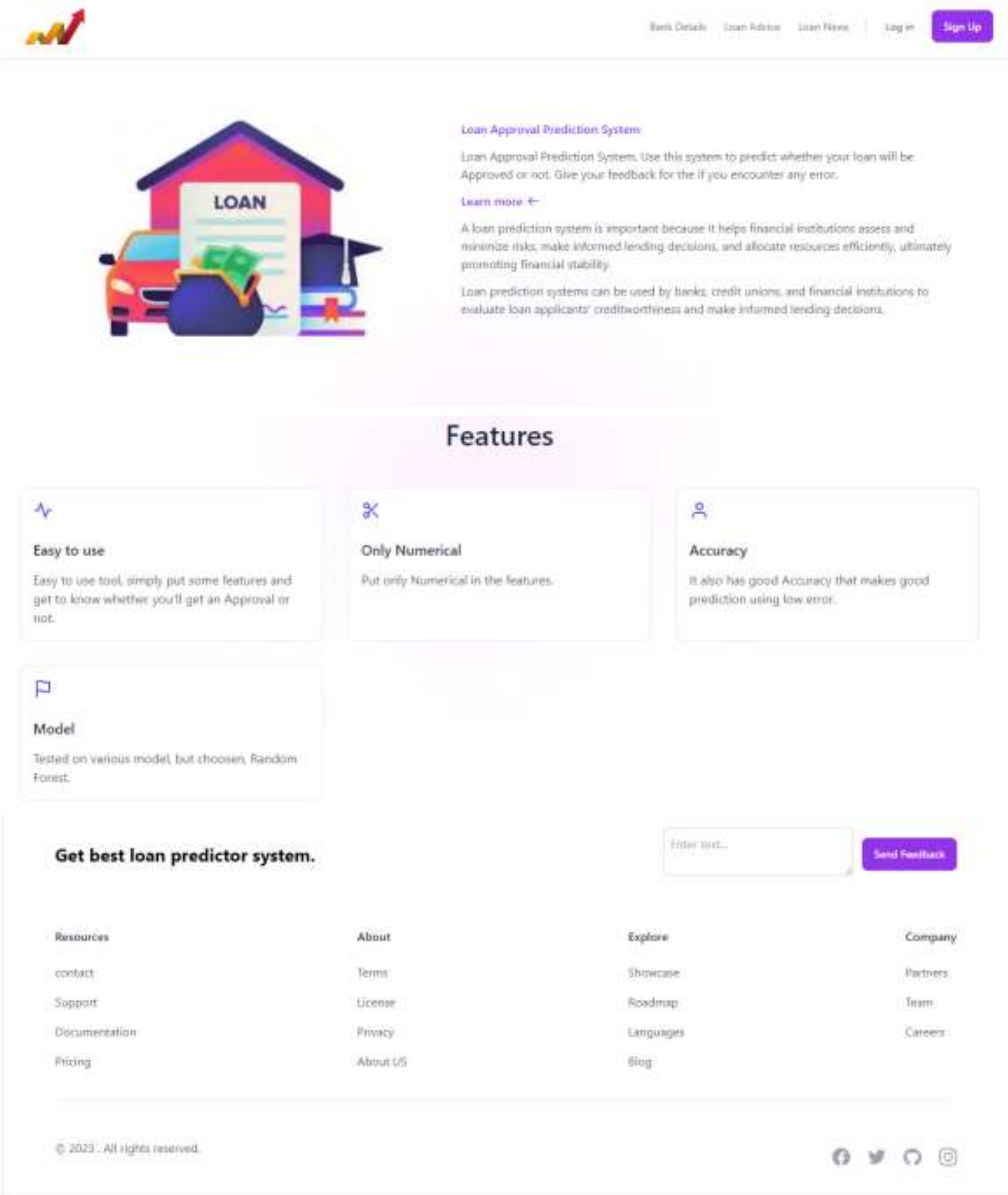

Figure 3 : Homepage of Loan Prediction System

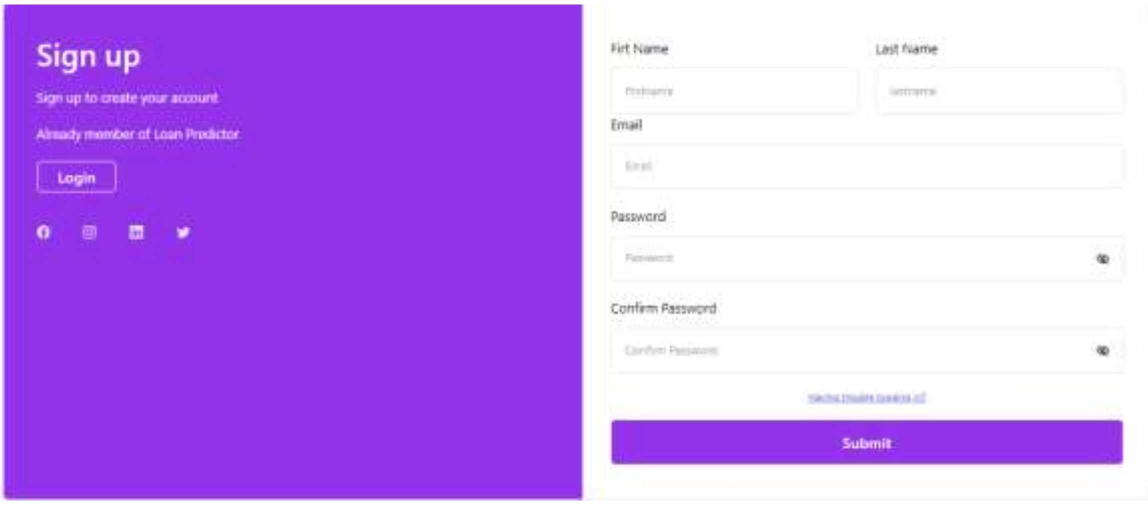

Figure 4 : Signup Form of Loan Prediction System

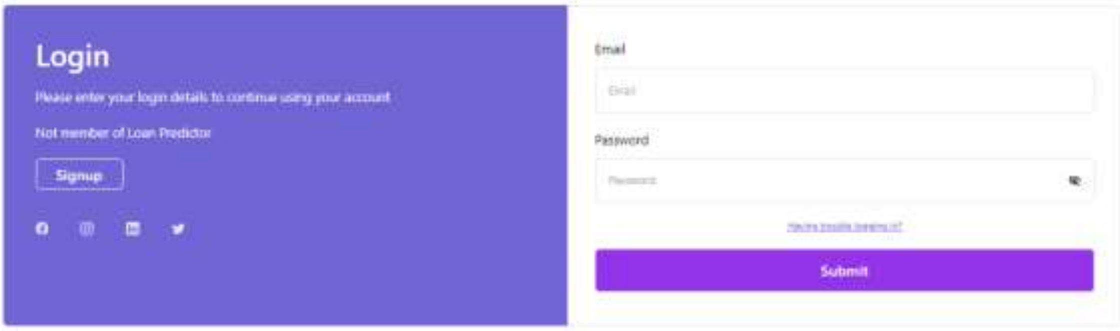

Figure 5 : Login Form of Loan Prediction System

Figure 6 : Navbar after user Login

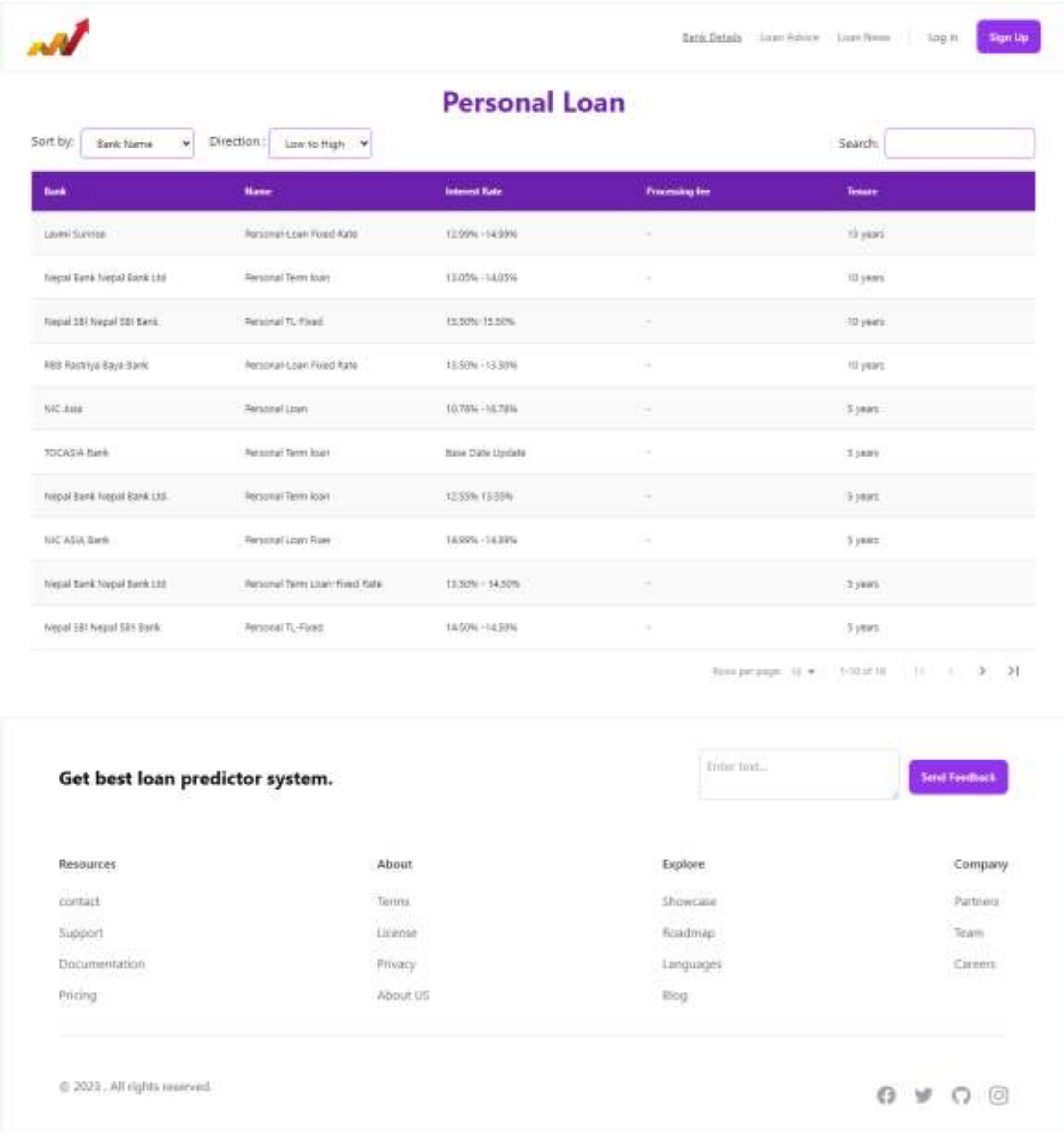

Figure 7 : Loan Details Page

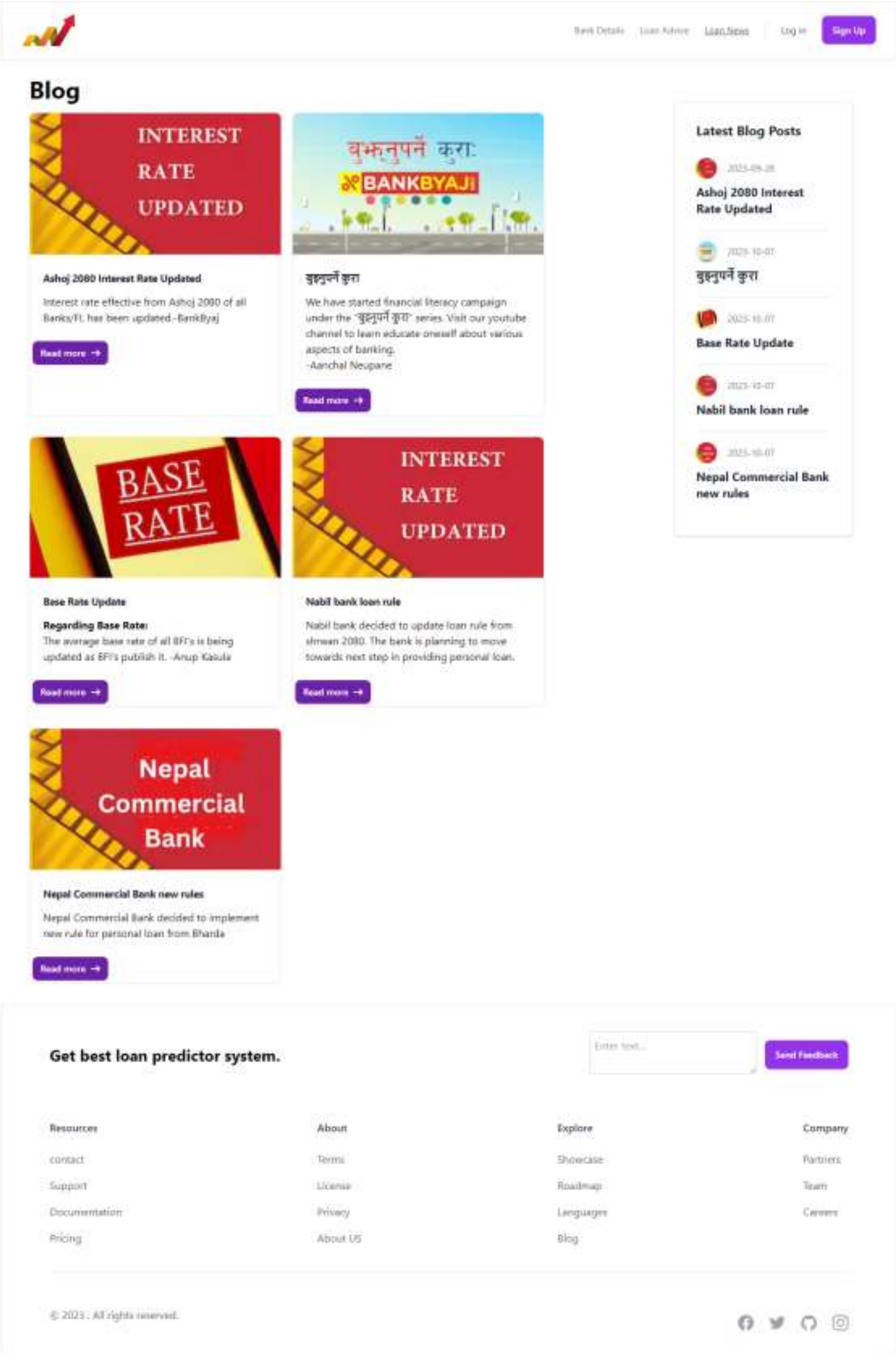

Figure 8 : Blog Page

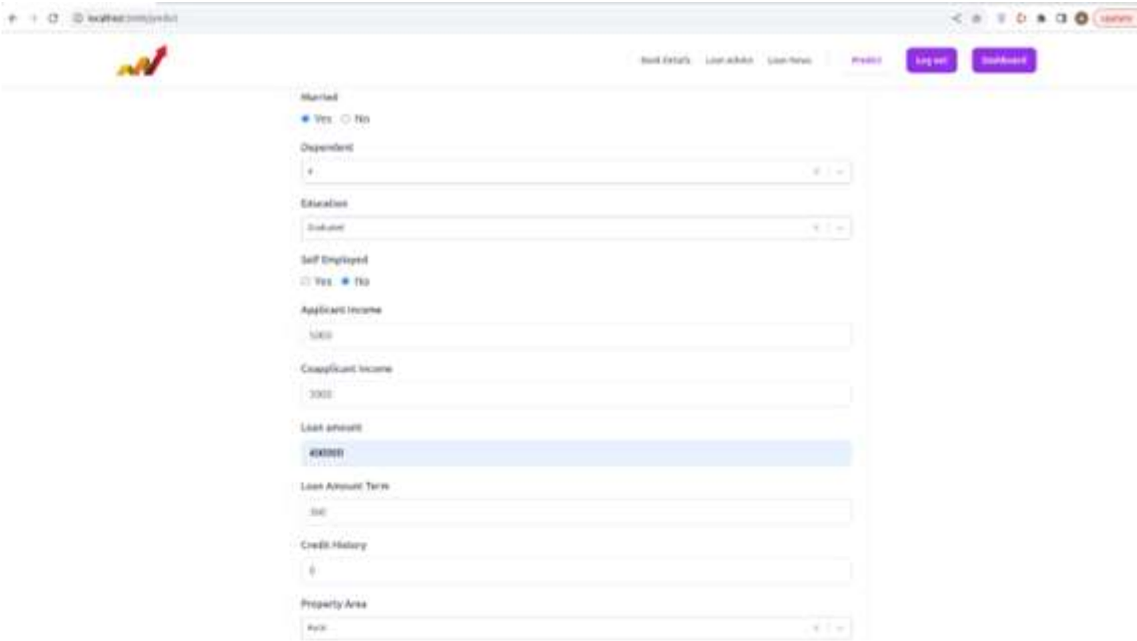

Figure 9 : Loan Prediction Form

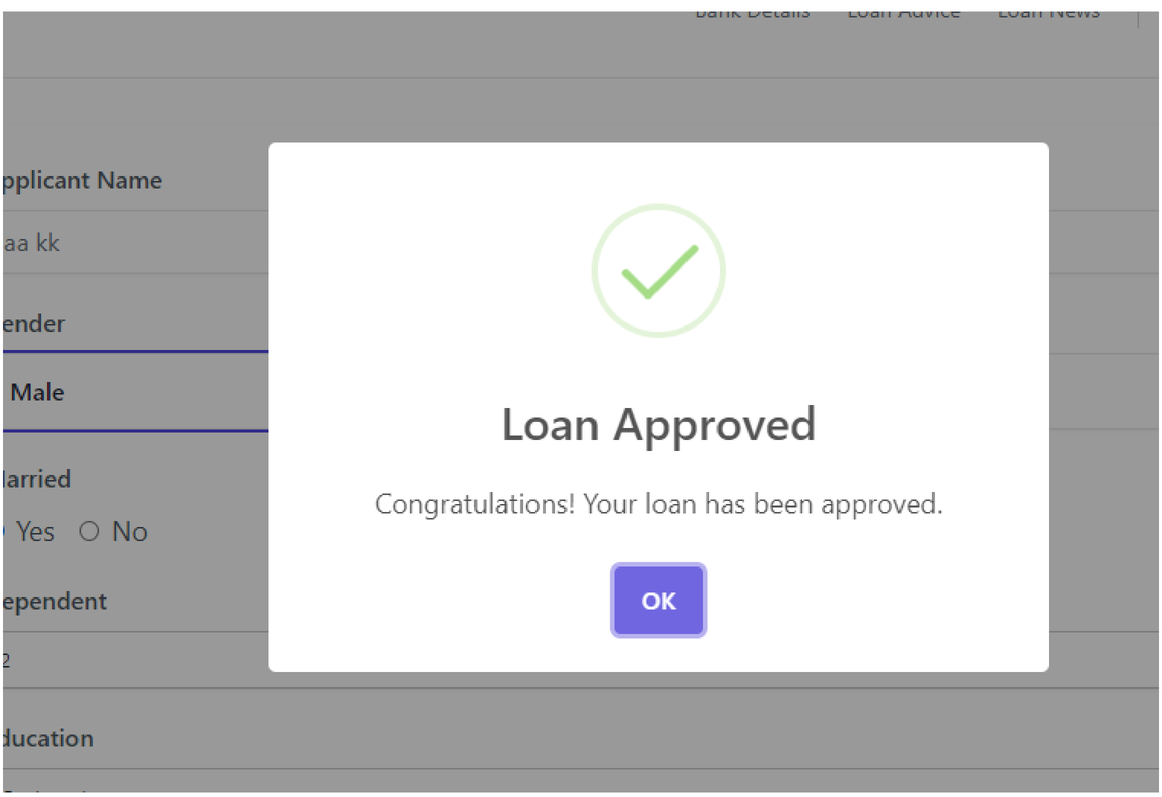

Figure 10 : Loan Approved Message Popup

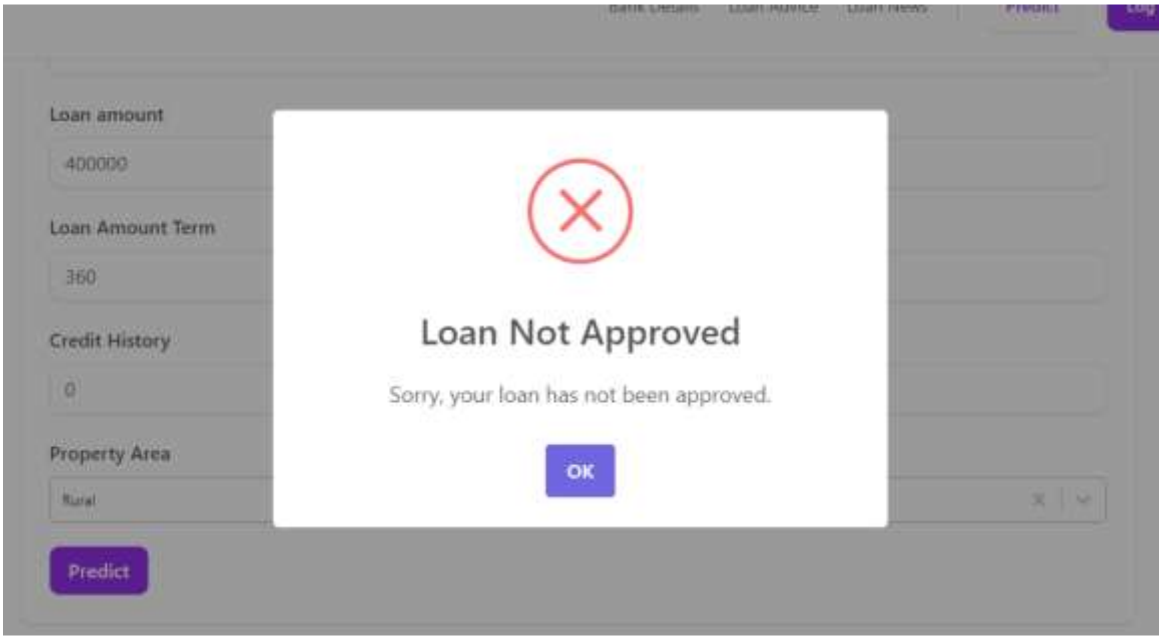

Figure 11 : Loan Not Approved Message Popup

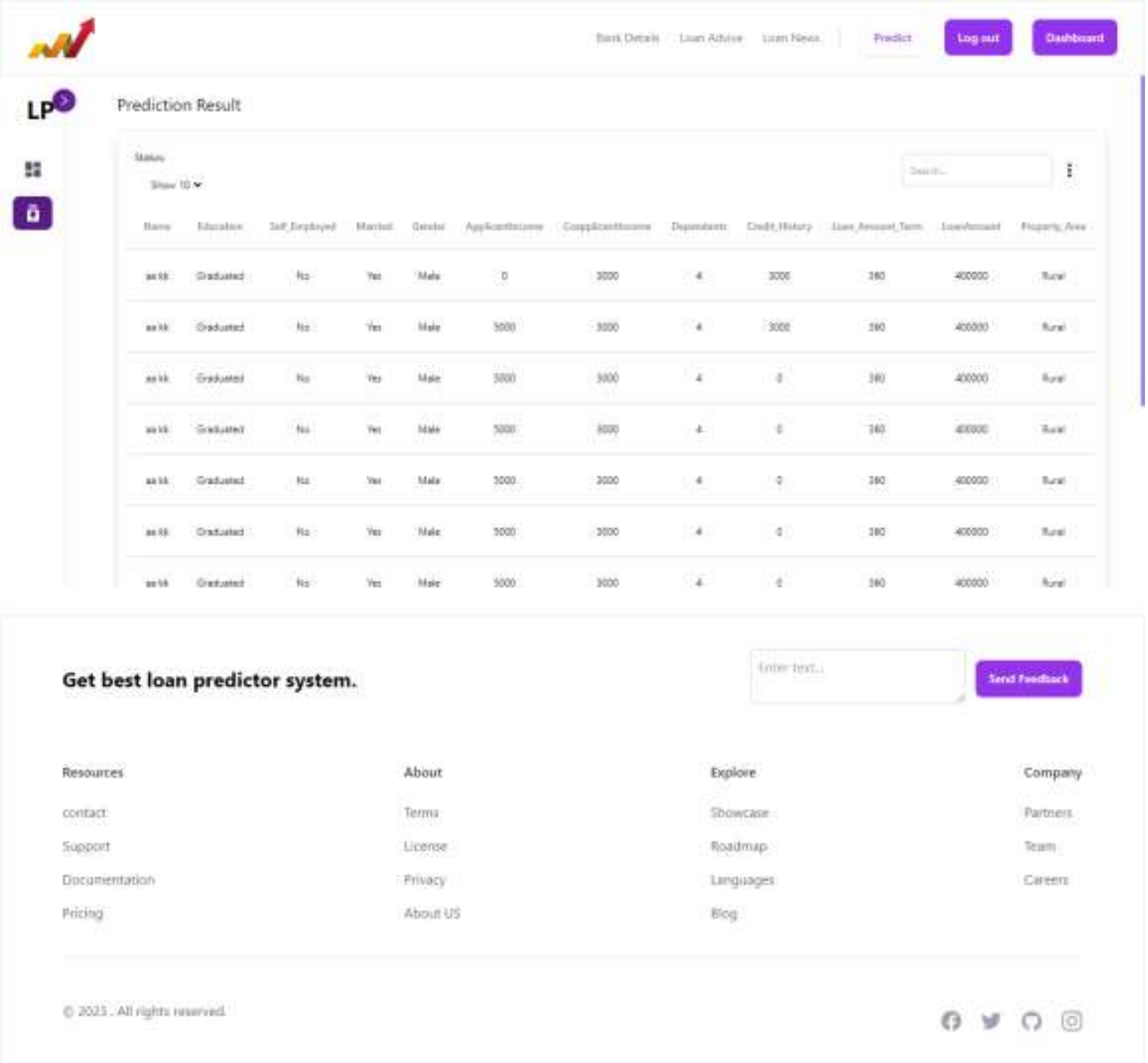

Figure 12 : User Prediction Results

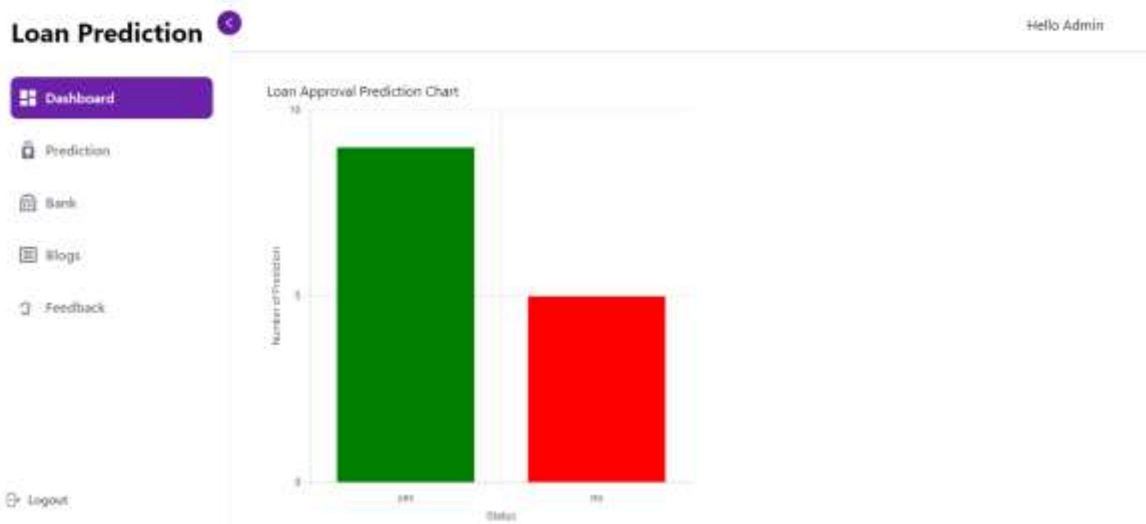

Figure 13 : Admin Dashboard of Loan Prediction System

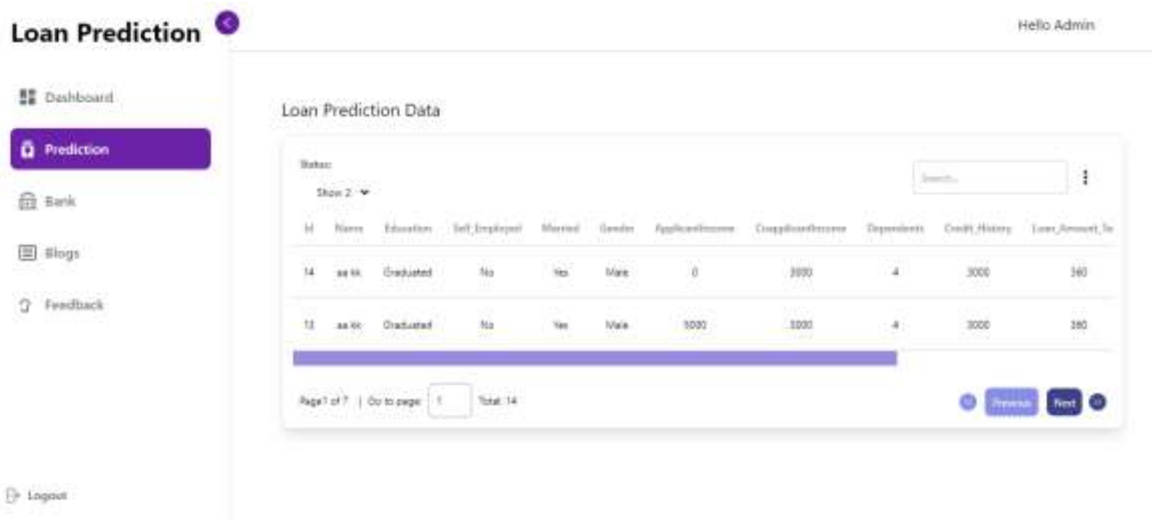

Figure 14 : All user Prediction Result in Admin

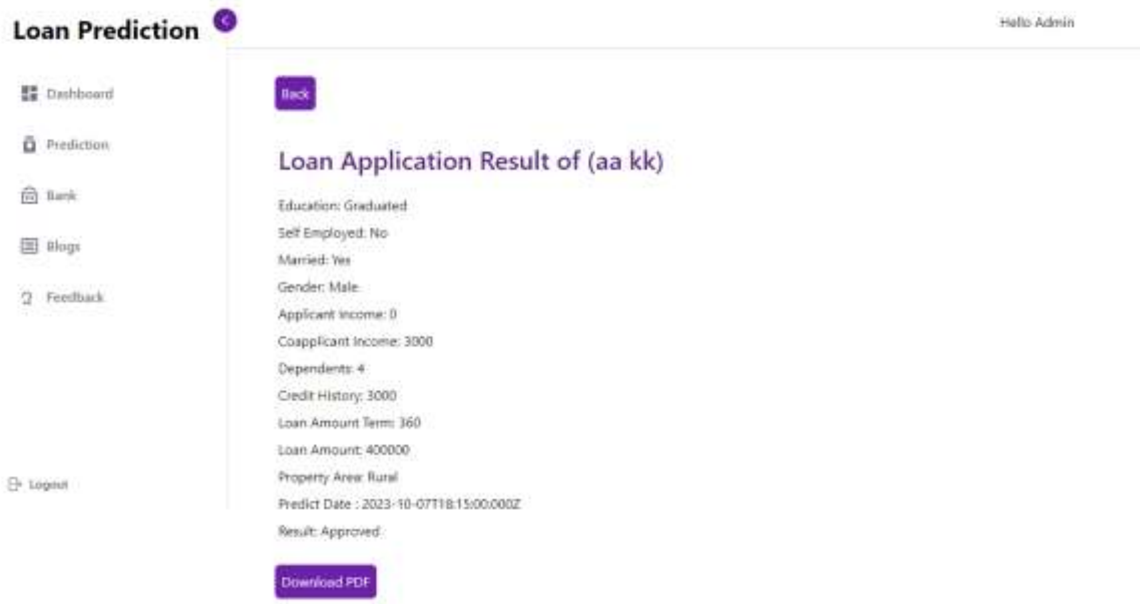

Figure 15 : Single User Loan result

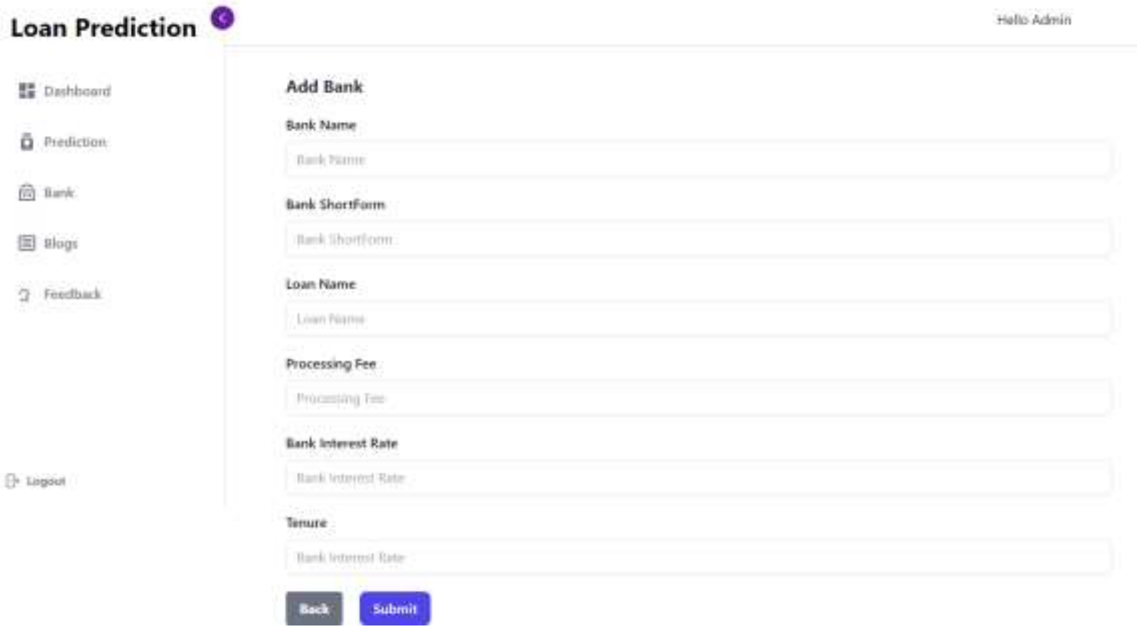

Figure 16 : Add Bank and Loan Details of Bank by Admin

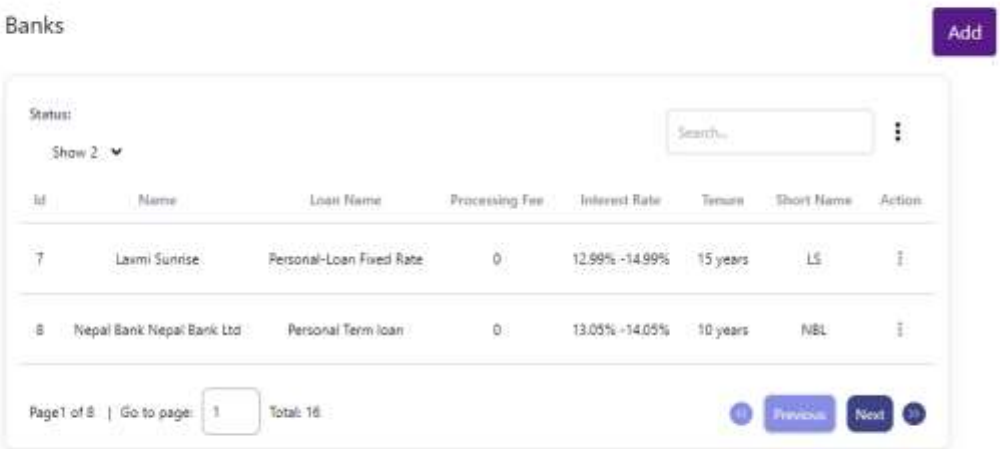

Figure 17 : Bank Details view in admin panel

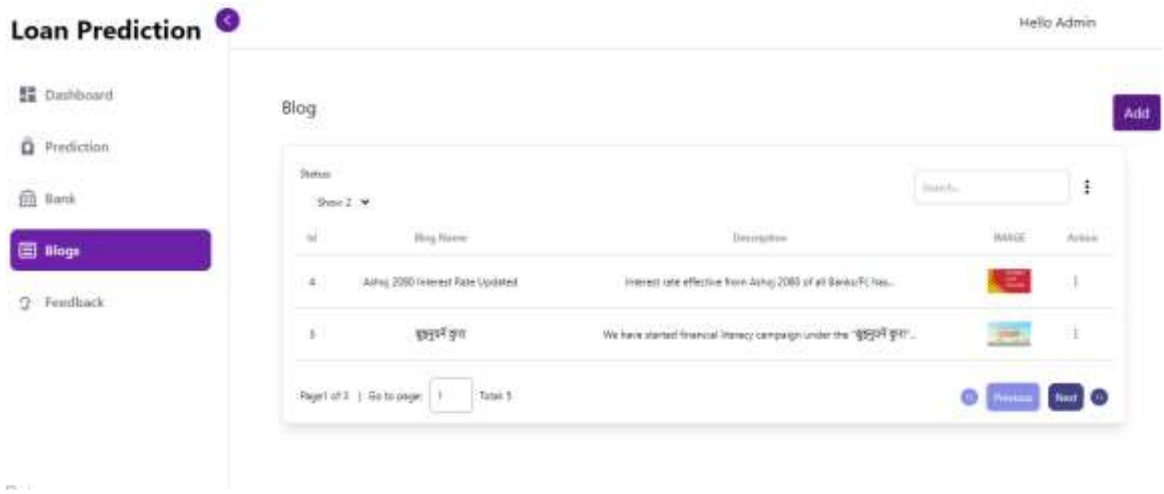

Figure 18 : Blogs data view in admin panel

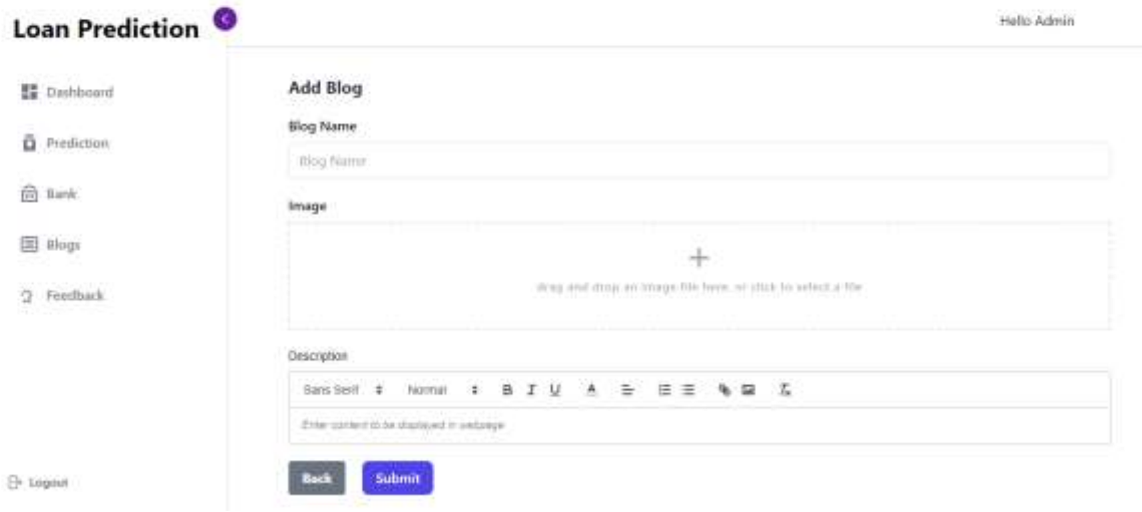

Figure 19 : Add Blogs form

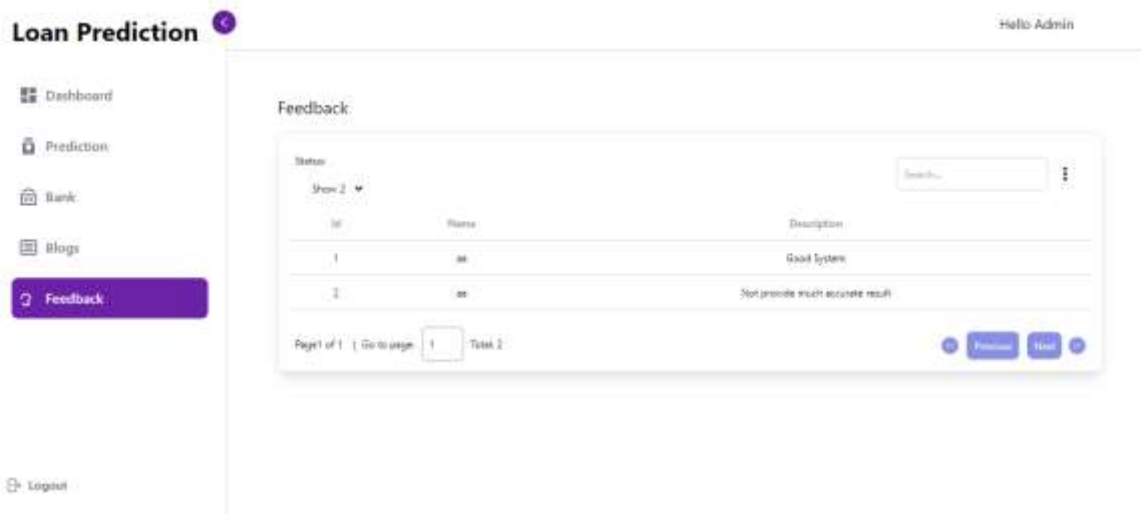

Figure 20 : User feedback