A Major Project Report

on

LOAN APPROVAL ESTIMATION DEPLOYING DEEP LEARNING AND BLOCKCHAIN TECHNOLOGIES

Submitted in partial fulfilment of the requirements for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

By

A. SAI VISHWANTH	**20EG105101**
JAYA SRAVANI JANAPAREDDY	**20EG105119**
T. MANIKANTA	**20EG105156**
V. CHANDANA	**20EG105160**

Under the Guidance of

Dr. G. BALAKRISHNA

Assistant Professor

Department of Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Venkatapur(V), Ghatkesar(M), Medchal(D) - 500083

2023 - 2024

ANURAG UNIVERSITY | SCHOOL OF ENGINEERING

CERTIFICATE

This is to certify that the Project Report entitled **"LOAN APPROVAL ESTIMATION DEPLOYING DEEP LEARNING AND BLOCKCHAIN TECHNOLOGIES"** being submitted by A. Sai Vishwanth, Jaya Sravani Janapareddy ,T. Manikanta, V. Chandana bearing the Hall Ticket number 20EG105101, 20EG105119 , 20EG105156, 20EG105160 respectively in partial fulfillment of the requirements for the award of the degree of the **Bachelor of Technology** in **Computer Science and Engineering** to **Anurag University** is a record of bonafide work carried out by them under my guidance and supervision from 2023 to 2024.

The results presented in this report have been verified and found to be satisfactory. The results embodied in this report have not been submitted to any other University for award of any other degree or diploma.

Signature of Supervisor	Signature Dean CSE,
Dr. G. Balakrishna	Dr. G. Vishnu Murthy
Assistant Professor	

External Examiner

ACKNOWLEDGMENT

We would like to express our sincere thanks and deep sense of gratitude to project supervisor **Dr. G. Balakrishna** for his constant encouragement and inspiring guidance without which this project could not have been completed. His/her critical reviews and constructive comments improved our grasp of the subject and steered to the fruitful completion of the work. His/her patience, guidance and encouragement made this project possible.

We would like to express our special thanks to **Dr. V. Vijaya Kumar**, Dean School of Engineering, Anurag University, for their encouragement and timely support in our B. Tech program.

We would like acknowledge our sincere gratitude for the support extended by **Dr. G. Vishnu Murthy**, Dean, Dept. of CSE, Anurag University. We also express our deep sense of gratitude to Dr. **V V S S S Balaram,** Academic coordinator, **Dr. Pallam Ravi**, Project in-Charge, **Dr. G. Prabhakar Raju**. Project Co-Ordinator and Project review committee members, whose research expertise and commitment to the highest standards continuously motivated us during the crucial stage of our project work.

A. Sai Vishwanth
(20EG105101)

Jaya Sravani Janapareddy
(20EG105119)

T. Manikanta
(20EG105156)

V. Chandana
(20EG105160)

DECLARATION

We hereby declare that the Report entitled "**LOAN APPROVAL ESTIMATION DEPLOYING DEEP LEARNING AND BLOCKCHAIN TECHNOLOGIES**" submitted for the award of Bachelor of Technology Degree is our original work and the Report has not formed the basis for the award of anydegree, diploma, associate ship or fellowship of similar other titles. It has not been submitted to any other University or Institution for the award of any degree or diploma.

20EG105101: A. Sai Vishwanth

20EG105119: Jaya Sravani Janapareddy

20EG105156: T. Manikanta

20EG105160: V. Chandana

Place: Hyderabad
Date:

ABSTRACT

The convergence of deep learning and blockchain technologies presents a promising avenue for revolutionizing the loan approval process in the financial sector. This explores the integration of these cutting-edge technologies to enhance the accuracy, transparency, and security of loan approval estimation. By leveraging deep learning algorithms, vast amounts of heterogeneous data, including applicant information, credit histories, and financial records, can be analysed in real-time to assess creditworthiness with unprecedented precision.

This study aims to provide a comprehensive overview of the architecture, design principles, and implementation strategies involved in deploying deep learning and blockchain technologies for loan approval estimation. By examining the synergistic benefits of these technologies, financial institutions can optimize their lending processes, minimize risks, and foster greater trust and transparency with borrowers. Ultimately, the integration of deep learning and blockchain technologies has the potential to transform the loan approval landscape, ushering in a new era of efficiency, security, and accessibility in the financial industry. It gives various methodologies aimed at forecasting loan defaults, with a specific focus on the efficacy of machine learning models. Leveraging a rich dataset obtained from Kaggle, we embark on a meticulous analysis and comparative examination of these methodologies to discern the most robust and reliable strategies, the utilization of blockchain technology ensures the integrity and immutability of the entire loan approval process. Through the deployment of smart contracts, self-executing agreements stored on the blockchain, various aspects of the loan approval process can be automated, streamlining operations and enhancing efficiency. Additionally, the decentralized nature of blockchain technology eliminates the need for intermediaries, reducing processing times and operational costs associated with traditional loan approval processes.

Our investigation entails a thorough scrutiny of factors such as age, credit history, loan amount, and duration, aiming to elucidate their respective roles in predicting loan defaults. Through this comprehensive analysis, we endeavour to identify the key variables that wield significant influence over default prediction accuracy. Moreover, our findings underscore the imperative of considering a diverse array of customer characteristics beyond rudimentary financial data in the loan evaluation process.

Furthermore, this research underscores the pivotal significance of incorporating additional factors, such as the purpose of the loan and the borrower's intent, in enhancing prediction accuracy. By harnessing advanced machine learning techniques and embracing a holistic approach to risk assessment, financial institutions can refine their lending strategies and mitigate the risk of non-performing assets, thus safeguarding their financial stability and long-term profitability. Through an exhaustive examination of various factors, we strive to unravel the complex web of variables that impact loan default prediction. Our findings highlight the critical importance of delving beyond surface-level financial indicators and embracing a multidimensional perspective by incorporating a diverse array of customer characteristics into predictive models. The evaluation of credit worthiness enhancing the accuracy and reliability of loan default predictions and this study underscores the necessity for financial institutions to continually innovate and refine their approaches to credit evaluation and risk management and ensuring their long-term success and stability in banking industry.

Table of Contents

Table of Figures

Table of Tables

CHAPTER-1. INTRODUCTION

The application of cutting-edge technology like Deep Learning and Blockchain is driving a dramatic change in the loan approval landscape. The collaborative project "Loan Approval Estimation Deploying Deep Learning and Blockchain Technologies" aims to improve the effectiveness, precision, and security of these procedures. Deep Learning algorithms can sort through large datasets, find intricate patterns, and make remarkably accurate predictions about the future. This makes them a valuable tool for decision-making when it comes to loan approvals and it seeks to transform the way loan approval procedures are conducted, putting the banking sector on the verge of a technological revolution.

In this project an innovative solution at the intersection of blockchain technology and machine learning. The economic repercussions, particularly impacting small and medium enterprises (SMEs) in India, have accentuated the need for alternative financing solutions. This study explores the integration of blockchain and machine learning to establish a peer-to-peer lending framework for micro-loans, with a particular focus on mitigating the impact of frequent lockdowns on traditional bank loan applications. The proof-of-concept involves the creation of a decentralized web application utilizing the Ethereum blockchain, aiming to provide swift, secure, and contactless financial support in the aftermath of the pandemic [1]

The pivotal role of loans in a bank's revenue, underscoring the complexity of accurately assessing loan requests and managing associated risks. With a focus on machine learning techniques, including K-Nearest Neighbours, Decision Trees, and the novel Support Vector Classifier, the study aims to predict loan approval. The discussion centres on the inherent credit risk in lending, the reliance on credit scores for applicant eligibility, and the need for effective machine learning methodologies to identify suitable candidates and mitigate credit risk. The multifaceted nature of human decision-making in banking, influenced by non-quantifiable factors, adds complexity to the challenge. The introduction sets the stage for exploring diverse machine learning algorithms and their potential in revolutionizing the loan approval process.[2]

In response to the evolving dynamics of the banking sector, the challenge of discerningly approving loans in the face of a burgeoning number of applicants has become increasingly pronounced. this study introduces a pioneering machine learning approach to address this issue. The research delves into the intricacies of predicting loan approval by leveraging extensive data from previous records, aiming to automate and refine the selection process. The paper explores a diverse set of machine learning models, including Decision Trees, Random Forest, Support Vector Machine, Linear Models, Neural Network, and AdaBoost, evaluating their parameters and efficacy in predicting loan safety. The ultimate goal is to contribute to the banking industry's quest for more efficient, data-driven methodologies in the realm of loan approval.[3]

The current financial crisis has prompted a reassessment of loan approval practices globally. This work advocates for leveraging state-of-the-art machine learning techniques to predict loan approvals. Unlike traditional methods relying solely on income and age, our approach considers additional factors like the customer's occupation and existing EMIs. Outliers are addressed, and features are analyzed to enhance model reliability. The best-performing models are Gradient Boosting and Random Forest, integrated using a Voting classifier, yielding good performance. Implemented in Python 3.8.3 using Pandas, NumPy, Matplotlib, and Scikit-Learn, this work aims to provide a reliable solution to the loan approval problem in the financial industry.[4]

Predicting loan defaulters in the banking system, emphasizing the role of credit lines in a bank's income. It discusses the importance of choosing effective predictive analytics methods, with a focus on the Naïve Bayes model's superior performance in loan forecasting. The introduction provides context for the study, emphasizing the need for automated loan eligibility processes based on customer details. The motivation centers around the crucial role of loan approval and recovery in banking organizations, highlighting the challenge of predicting loan repayment. The literature survey references two papers exploring machine learning algorithms for predicting loan approval, with a particular emphasis on the Naïve Bayes model's better performance in one of the studies.[5]

In the contemporary banking landscape, where financial institutions offer a myriad of products, the primary revenue stream often hinges on credit lines and the accruing interest from loans. The profitability of a bank is intricately tied to the management of loans, necessitating a nuanced understanding of customer behaviour in repayment. Addressing this imperative, the present study delves into the critical realm of predicting loan defaulters, a phenomenon with significant implications for mitigating Non-Performing Assets (NPAs). By employing advanced predictive analytics, particularly the Logistic Regression model, this research endeavours to explore and compare various methodologies, drawing on data sourced from Kaggle, to enhance the precision of identifying potential defaulters and, consequently, fortify the financial health of banking institutions.[6]

Furthermore, there is a revolutionary opportunity when using blockchain technology in financial inclusion initiatives. Blockchain can help with cheaper prices, faster settlement times, and better customer experiences for both domestic and international payments, especially in nations where a sizable portion of the population lacks access to banking. This promotes the creation of a legislative framework that encourages early-stage involvement and development in order to fully realize the potential of blockchain technology to improve financial inclusion. [7]

Researchers have developed advanced fraud detection methods to combat internet fraud in online finance. They utilize cutting-edge computer tech and data analytics, employing algorithms like XGBoost and deep neural networks. Their model, tested on a large lending institution dataset, shows high accuracy and generalization. This work marks a substantial step in strengthening fraud prevention and showcases deep learning's potential in defending against cyber threats in finance.[8]

CHAPTER-2 . LITERATURE SURVEY

Deep learning approaches have brought to significant advancements in credit risk assessment innovations. In order to forecast credit risk in the financial industry, offers a novel model that uses an Adaptive Binarized Spiking Marine Predators ‘Neural Network. The accuracy of this model is exceptional, providing strong support for the use of sophisticated neural networks in financial decision-making, it enhancing the model with blockchain technology ensures data security, while integrating a user-friendly interface enhances accessibility and usability.[9]

In parallel, a paper on the research highlights the usefulness of machine learning methods like Random Forest, Naive Bayes, Decision Tree, and KNN, offering an insightful viewpoint on the models' suitability for use in banking systems but there is an absence of user-friendly interface.[10]

The breadth of research and the promise for these technologies to simplify financial procedures are reflected in the study's coverage using decision tree and how well they anticipate loan approvals and it is a user-friendly application.[11]

To forecast loan acceptance while demonstrating the efficacy and simplicity of SVM, logistic regression, and other machine learning techniques, their work also recognizes the need for better user interfaces to encourage wider adoption and ease of use in practical applications and easy to implement.[12]

The methodologies and practical implementation, aim in to understand the effectiveness of these tools in real-world banking scenarios where model can anticipate outcomes and is quickly adaptable to a wide range of inputs using Logistic regression and random forest [13].

The primary objective of their research is to improve the loan approval process by utilizing historical data for predicting loan approval outcomes. By leveraging Deep learning methods like ANN and data mining trained on historical data, the authors aim to provide a framework for predicting whether a new loan applicant should be granted approval or not. Their emphasis on accurate predictions, prudent fund management, and overall profitability contributes to the ongoing discourse on optimizing the loan approval process.[14]

This research provides significant insights at the intersection of machine learning and the banking sector, laying the groundwork for further exploration and the application of advanced predictive analytics to enhance loan approval systems.[15]

It is a real-time loan approval classification method using deep learning, specifically an auto-encoder, which outperformed traditional binary classifiers like SVM in terms of F1 score. This framework presents promising potential for loan providers seeking to select creditworthy candidates in real-time, and future research will explore advanced deep learning techniques to further enhance performance.[16]

The methodology of the paper involves a hybrid approach combining Convolutional Neural Network (CNN) with Support Vector Machine (SVM), Random Forest (RF), and Decision Tree (DT) classifiers for credit risk prediction.[17]

This research provides the machine learning in finance, and ensemble learning techniques, including studies on specific algorithms like Random Forest, Decision Tree and XGBoost. It should highlight challenges, limitations, recent advancements, and the current state of research in loan prediction and machine learning applications in finance. [18]

The literature survey reviews studies on credit forecasting systems, emphasizing data processing, customer trait consideration, and model comparisons to enhance loan approval processes using machine learning and deep learning techniques along with models containing blockchain.

Reference No	Strategies	Advantages	Disadvantages
[9]	Assessment of Credit Risk using an Adaptive Binarized Spiking Marine Predators' Neural Network	Using deep learning techniques which perform with high accuracy	Enhancing the model with blockchain technology ensures data security while integrating a user-friendly interface enhances accessibility and usability
[10]	Random Forest, Naive Bayes, Decision Tree, and KNN.	Random forest gives the highest value of accuracy (83%)	The absence of a user-friendly interface is a significant drawback of this model.

[11]	Decision Tree	User-friendly application.	The accuracy can still be improved using deep learning techniques. The current application is 81% accuracy
[12]	Logistic regression, Random forest, SVM, K Neighbors, Decision tree	Easy to implement	The absence of a user-friendly interface is a significant drawback of this model.
[13]	Logistic regression, Random forest	model can anticipate outcomes and is quickly adaptable to a wide range of inputs	Interpretability issues And it has Risk of over Reliance
[14]	Artificial Neural Networks, Supervised learning, unsupervised learning, Data mining	It gives efficiency and improves financial outcomes to banking sector	Gives sensitivity to external factors and it is robustness
[15]	Random forest, SVM, Decision tree, logistic regression	SVM gives the highest accuracy in prediction loan approval	Gives the interoperability, computational complexity
[16]	Auto encoder Deep Neural Network - three encoders and three decoders, totalling six hidden layers	Excellent recall value and accuracy is also good	Not an end to end application such that user can access the front end

[17]	hybrid model, combining Convolutional Neural Network (CNN) feature extraction with Support Vector Machine (SVM), Random Forest (RF), and Decision Tree (DT) classifiers, for credit risk prediction.	Hybrid model leverages CNN for feature extraction and ensemble learning with SVM, RF, and DT, enhancing credit risk prediction.	Increased complexity, computational intensity, and reduced interpretability due to integrating multiple classifiers in the hybrid model.
[18]	Strategy of this is data collection data preprocessing and training the models using XGBoost, decision tree and random forest	Streamlines loan approval process by automating eligibility assessment based on predefined criteria.	Potential for bias in algorithm design and data inputs could lead to unfair outcomes or discrimination.

Table 2.1 : Comparative Table

Summary of the survey all things considered, "Loan Approval Estimation Deploying Deep Learning and Blockchain Technologies" represents a progressive method of modernizing conventional loan approval processes. This project heralds a new era in financial services, where technology-driven solutions enable banks to navigate the complexities of credit risk management with unprecedented agility and assurance. It does this by combining the analytical power of Deep Learning with the strong security features of Blockchain.

CHAPTER-3. PROPOSED METHOD

3.1 Problem Statement

The problem statement pertains to the inefficiencies that financial institutions, such as housing finance companies, banks, and NBFCs, encounter in the loan approval procedure. Currently, these institutions evaluate loan applications through manual processes, which results in lengthy wait periods and security concerns. In addition, the current systems are frequently antiquated, which exacerbates the time lag.

Considering these obstacles, the proposal proposes the integration of Blockchain and Deep Learning technologies to develop a solution that is both more efficient and secure. The objective is to automate the loan approval procedure by utilizing machine learning algorithms in conjunction with a range of customer characteristics, including but not limited to sex, marital status, income, and credit history. The implementation of this automation would not only streamline the procedure but also guarantee a more precise evaluation of loan eligibility.

In addition, the proposal emphasizes the importance of a holistic strategy that seamlessly integrates the entire loan approval procedure. In order to develop a resilient and effective solution that satisfies the requirements of the contemporary banking sector, this integration is vital. By integrating Blockchain and Deep Learning technologies, the proposed solution intends to significantly reduce processing times, improve security, and increase overall efficiency with the intention of transforming the loan approval procedure.

3.1.1 Main Objectives

- Develop a model for loan approval prediction.
- Implement a data storage system to enhance transparency and guarantee the tamper-proof integrity of the data.
- Create a user-friendly interface for seamless interaction and accessibility.

3.2 Proposed System

The proposed methodology offers an end-to-end user-friendly application. Users can input parameters into the system, which then feeds them into a trained deep learning model to predict whether a loan should be approved or not. The results of this prediction, along with the input parameters, are securely stored on the blockchain. This ensures permanence and tamper-proofing of the data, maintaining its integrity and reliability over time.

3.2.1 Proposed Architecture

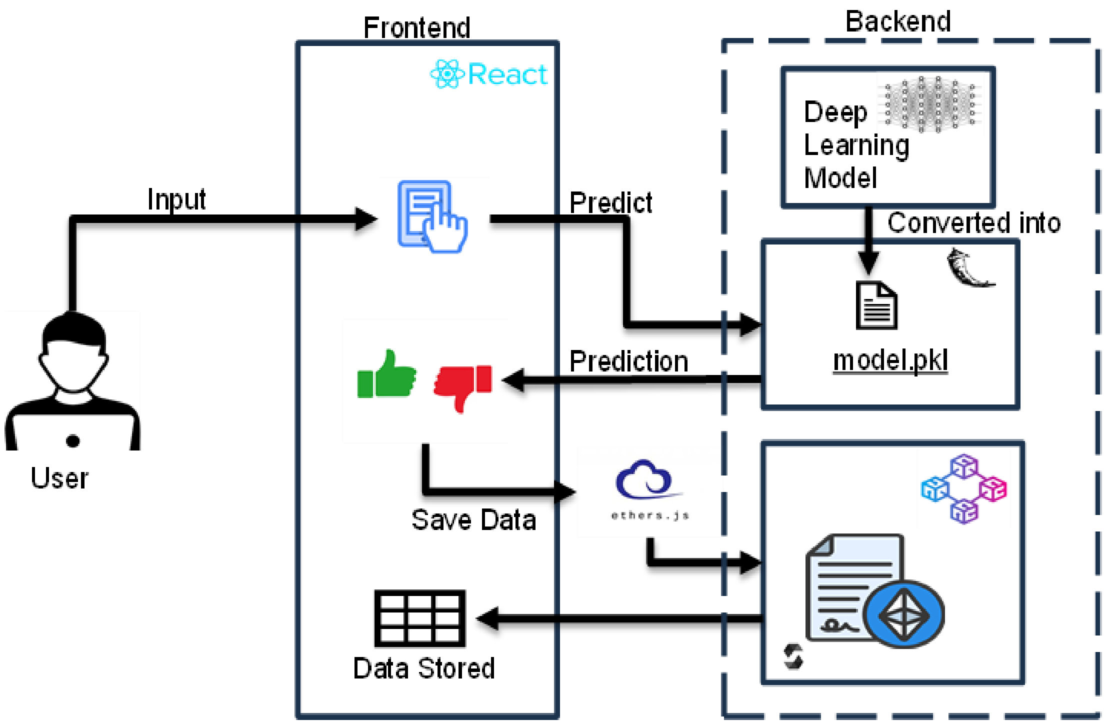


Figure 3.1: Architecture for Loan Approval Estimation

The information storage and end-to-end loan estimation system is made possible by the proposed architecture represented in figure 3.1. It commences with a React.js-based user interface in which a bank employee inputs a variety of data points, including the individual's name, application number, age, income, loan grade, duration of employment, loan interest rate, loan percentage of income, and credit score. When the "predict" icon is clicked, the request is transmitted to the Python-based infrastructure that is implemented using Flask.

A deep learning model, which is contained within a pickle file, is utilised in the backend to evaluate the information presented and produce a loan approval prediction (which can be labelled as 'yes' or 'no'). Returns to frontend.

The frontend and backend interact when the bank employee selects the "save data" icon. The backend is linked to the JavaScript and integrates with the MetaMask extension present on the webpage. To authenticate requests for accessing and engaging with the Ethereum blockchain, MetaMask is implemented.

After authorization is obtained through MetaMask, the request is forwarded to the Solidity code, more precisely the Ethereum blockchain-implemented data storage contract.

The data that is furnished is securely stored on the blockchain. In essence, the interface presents the blockchain-preserved data in the form of a tabular representation. This brings an end to the entire process, establishing a secure and efficient framework for loan estimation and data storage.

3.2.3 Deep Learning Model

The credit risk assessment model being suggested use a neural network framework with many layers to examine input data, which includes characteristics such as age, income, job duration, loan purpose, and loan rating. This model is taught to forecast the acceptance or denial of loans by minimising a designated loss function via the optimisation of weights and biases throughout the training phase. The model's performance is assessed using evaluation measures like as accuracy, precision, recall, and F1 score. The results of this model provide useful information for decision-makers in the lending industry, assisting in the efficient control of credit risk.

A. Data Collection

The dataset used for building and evaluating the credit risk assessment model is sourced from Kaggle, and it is named the "Credit Risk" dataset. This dataset contains diverse information about individuals, including attributes like age, income, employment length, loan intent, loan grade, loan amount, interest rate, loan status, loan percentage of income, and credit history length. Each entry in the dataset corresponds to an individual's loan application, making it a valuable resource for training and assessing the proposed credit risk model. It's important to acknowledge Kaggle as the platform providing this dataset, which has been utilized for developing and testing the credit risk assessment model discussed in the previous interactions.

person_age	person_income	person_home_ownership	person_emp_length
loan_intent	loan_grade	cb_person_cred_hist_length	loan_int_rate
loan_status	loan_amnt	cb_person_default_on_file	loan_percent_income

Table 3.1 : Columns in dataset

B. Pre-Processing

1. Handling Missing Values:
    - Identify missing values in the 'person_emp_length' and 'loan_int_rate' columns using df.isnull().sum().
    - Impute missing values by replacing them with the mean values of their respective columns. This ensures that the dataset is complete and ready for analysis
2. Drop Columns:
    - Remove unnecessary columns ('loan_intent', 'person_home_ownership', 'cb_person_default_on_file') from the dataset. This step helps reduce dimensionality and focuses on relevant features for credit risk assessment.
3. Handling Outliers:
    - Visualize outliers in selected numerical features ('person_age', 'person_emp_length','loan_percent_income','cb_person_cred_hist_leng th') using box plots.
    - Remove outliers by filtering the data within the interquartile range (IQR). This helps mitigate the impact of extreme values that could affect the model's performance.
4. Analysis of outliers:

    We tackled outliers using the IQR method, focusing on the middle 50% of data points and flagging those outside this range as potential outliers. By honing in on the central data dispersion, we aim to reduce the impact of extreme values on our analysis, as depicted in Figure 3.2. This approach ensures a more accurate and dependable dataset for training our model, leading to improved reliability and performance.

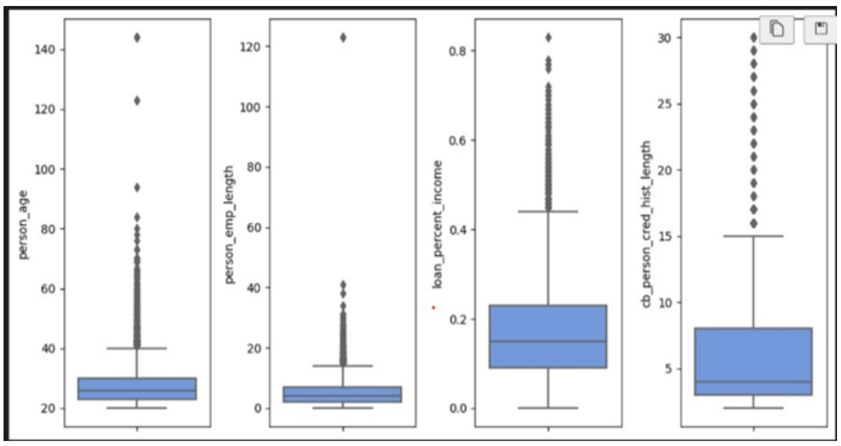
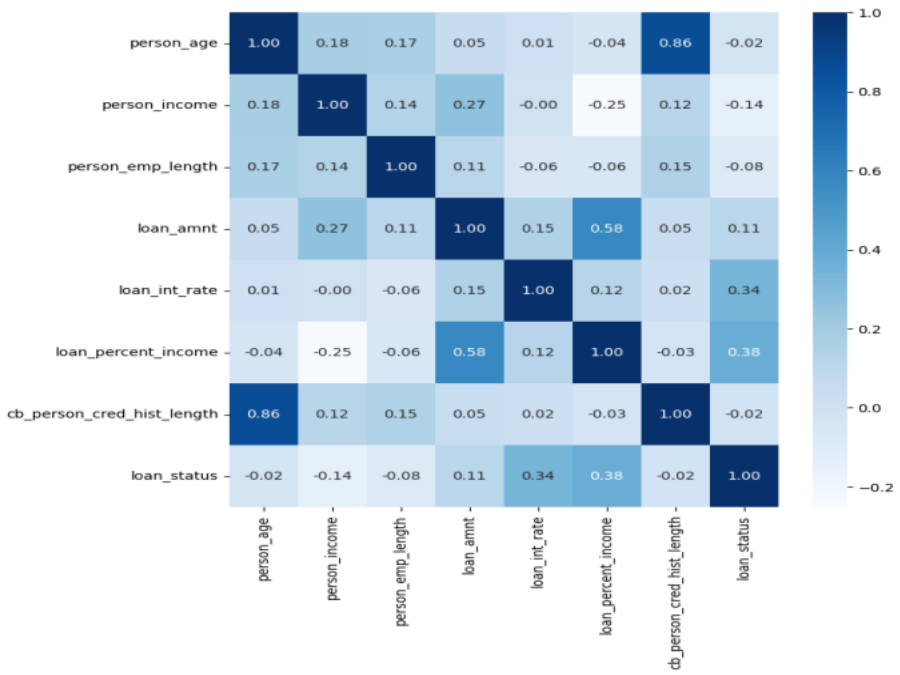


Figure 3.2: Analysis of Outliers

5. Convert categorical data to numerical data
6. <u>Credit Risk correlation matrix:</u>

   The correlation matrix in the dataset quantifies linear relationships between variables, visualized through a heatmap in Figure 3.3. Darker shades of blue in the heatmap indicate stronger correlations. Positive values signify positive linear relationships, negative values denote negative linear relationships, and values close to zero suggest weak or no linear correlation between variables.



	person_age	person_income	person_emp_length	loan_amnt	loan_int_rate	loan_percent_income	cb_person_cred_hist_length	loan_status
person_age	1.00	0.18	0.17	0.05	0.01	-0.04	0.86	-0.02
person_income	0.18	1.00	0.14	0.27	-0.00	-0.25	0.12	-0.14
person_emp_length	0.17	0.14	1.00	0.11	-0.06	-0.06	0.15	-0.08
loan_amnt	0.05	0.27	0.11	1.00	0.15	0.58	0.05	0.11
loan_int_rate	0.01	-0.00	-0.06	0.15	1.00	0.12	0.02	0.34
loan_percent_income	-0.04	-0.25	-0.06	0.58	0.12	1.00	-0.03	0.38
cb_person_cred_hist_length	0.86	0.12	0.15	0.05	0.02	-0.03	1.00	-0.02
loan_status	-0.02	-0.14	-0.08	0.11	0.34	0.38	-0.02	1.00

Figure 3.3: Correlation matrix

7. <u>Feature Selection and Finalization:</u>

   A thoughtful feature selection process was undertaken to identify the most relevant attributes for credit risk assessment. Leveraging techniques like mutual information regression, we prioritized features such as 'person_age,' 'person_income,' 'person_emp_length,' 'loan_amnt,' 'loan_int_rate,' 'loan_percent_income,' and 'cb_person_cred_hist_length.' These features were deemed to have the most substantial impact on our credit risk model, resulting in a streamlined and optimized set of variables for effective predictive modelling.

8. Pie chart of loan status:

   The pie chart figure 3.4 visually represents the distribution of responses, breaking them down into two parts. The larger portion, comprising 79.1%, is represented by the blue section, indicating a substantial majority of responses falling into this category. In contrast, the yellow section accounts for a smaller share, specifically 20.9%, highlighting its comparatively lesser representation in the overall responses. The chart clearly conveys the significant dominance of the blue category in the dataset.

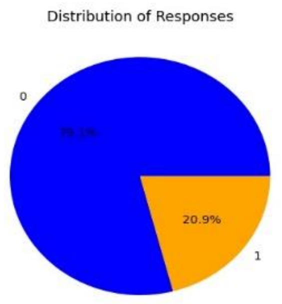


Figure 3.4: Loan Status

C. Model Selection:

A Sequential neural network model was selected for credit risk prediction due to its capacity to capture complex relationships within the data. The model includes multiple layers with varying numbers of neurons and employs the ReLU activation function.

After selecting a sequential neural network architecture with four hidden layers and ReLU activation functions, with 128, 256, 256, and 256 neurons respectively, and a linear activation for the output layer, the model is ready for training and testing with the dataset. Following this, the deep learning model is trained using the dataset to learn patterns and relationships within the data. Once trained, the model's performance is evaluated to ensure its effectiveness in making predictions.

3.2.4 Blockchain Architecture for Data Storage

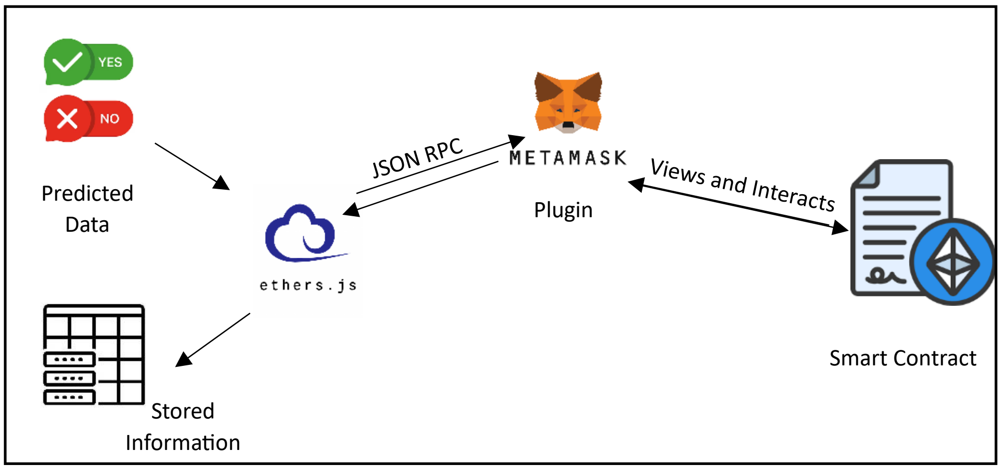


Figure 3.5: Blockchain Architecture

The proposed architecture in figure 3.5 comprises the following components: 1. Predicted data, which the deep learning model generates in response to an input prediction. 2. Ether.js is a popular JavaScript library for communicating with the Ethereum blockchain, which includes tools for developing decentralised apps and interfacing with smart contracts. It provides a simple JSON RPC for sending transactions, searching blockchain data, and managing Ethereum contracts, making it an effective tool for frontend and backend development. 3. MetaMask Plugin, MetaMask Is a browser extension and mobile app that functions as a digital wallet for managing Ethereum assets and engaging with decentralised apps. It provides smooth access to Ethereum-based websites and easily authorises transactions. 4. A smart contract is a self-executing contract whose terms are explicitly encoded into code on the blockchain. It automatically enforces and implements the terms of the agreement, eliminating the need for middlemen while ensuring transparency and immutability. 5. saved information: Following the execution of the smart contract, the information saved in the block is shown on the front end as a table.

A.Flow of Execution

When a user interacts with the application by clicking the "Save Data" button, their input data is sent to a specific function called addInformation() within the Ether.js library. This function serves as a bridge between the frontend and the Ethereum blockchain.

Inside addInformation() the input data is formatted and prepared to be stored on the blockchain. Utilizing JSON-RPC (Remote Procedure Call), a communication protocol, the function invokes the Metamask plugin. Metamask acts as the interface between the web application and the Ethereum network.

Upon invocation, Metamask prompts the user to confirm the transaction. This confirmation is crucial for security and ensures that the user authorizes the data storage operation on the blockchain. Once confirmed, Metamask creates a transaction and sends it to the Ethereum network.

Within the Ethereum network, the smart contract responsible for storing the data is deployed. This smart contract contains predefined rules and logic, which govern how the data is stored and accessed. Once the transaction is confirmed by the network, the smart contract executes, storing the provided information securely on the blockchain.

Subsequently, another function called getAllInformation() is automatically triggered. This function retrieves all the stored information from the blockchain. Once retrieved, the data is formatted and displayed on the frontend of the application in the form of a table. This ensures that users can easily view and access the stored information directly from the application interface, enhancing transparency and usability.

B. Smart contract design

<u>Pseudo Code:</u>

```
    Contract data_storage {
        uint blockCount
        string[] applicationNumbers
        // define struct to represent loan application
        Struct Block {
            string person_name

            string person_income

            string person_emp_length

            string loan_grade

            string application_no

            string loan_amnt

            string loan_int_rate
```

```
                string loan_percent_income

                string cb_person_cred_hist_length

                string approval
        }

        mapping (string => block) blocks
        // Function to add to loan application information

        addInformation (string _person_name, string
_person_income, string _person_emp_length, string _loan_grade,
string _application_no, string _loan_amnt, string _loan_int_rate,
string _loan_percent_income, string _cb_person_cred_hist_length,
string _approval):

Block memory newBlock = Block({
        person_name: _person_name,
        person_income: _person_income,

        person_emp_length: _person_emp_length,

        loan_grade: _loan_grade,

        application_no: _application_no,

        loan_amnt: _loan_amnt,

        loan_int_rate: _loan_int_rate,

        loan_percent_income: _loan_percent_income,

        cb_person_cred_hist_length: _cb_person_cred_hist_length,

        approval: _approval });

        blocks[_application_no] = newblock;

        blockCount++;

        applicationNumbers.push(_application_no);

// Function to retrieve all stored loan application information
```

```
    GetAllInformation():
        block[] allBlocks
        for(i=0; i < blockCount ;i++):
            allBlocks[i] := blocks[application_no[i]]
        return allBlocks
    // Function to retrieve loan application information by application number
    GetInformationByApplicationNumber(application_no):
        // Retrieve Block using provided application number
        return blocks[application_no];
    // Function to get the total number of stored loan applications
    GetNumberOfBlocks():
        ssreturn blockCount
}
```

C. Explanation of Smart contract

This smart contract, named “data_storage”, is designed as a decentralized data storage solution for managing loan application information on the Ethereum blockchain. It employs the solidity programming language and follows the SPDX-License-Identifier MIT.

a. <u>Data Structure</u>: The contract utilizes a structured data approach by defining a block “struct” to organize information related to loan applications. This “struct” encapsulates various attributes such as the applicant's details (name, income, employment length), loan specifics (grade, amount, interest rate), credit history length, and approval status. This structured approach enhances readability, organization, and ease of access to the stored data.

b. <u>Mapping and Array</u>: The contract employs a mapping named block to associate each loan application number with its corresponding block data. This mapping allows for efficient and direct retrieval of application data based on the application number.

Additionally, an array named is applicationNumber used to maintain a list of application numbers, enabling iteration and retrieval of all stored applications.

c. Data Addition and Counting: The addInformation() function facilitates the addition of new loan application data to the contract. Upon adding a new application, the blockCount is incremented, providing a dynamic count of the total number of stored applications. Simultaneously, the application number is appended to the applicationNumber array, ensuring comprehensive tracking of all applications.

d. Data Retrieval: The contract offers several functions for retrieving stored application data. The getAllInformation function returns an array containing all stored block data, allowing external parties to access comprehensive information about all loan applications. Additionally, the getInformationByApplicationNumber function enables the retrieval of specific application data by providing the application number as input. Moreover, the getNuberOfBlocks function offers a straightforward method to obtain the total count of stored applications.

3.2.5 Integrating Deep Learning model and Blockchain Environment

In this integration project, the initial step involves converting the trained deep learning model into a pickle file, a common serialization format in Python. Once the model is serialized, the next phase is to establish a Flask backend server. This backend server is crucial for loading the pickle file containing the model and serving it for inference purposes. Subsequently, a Flask backend is developed to act as the bridge between the React.js frontend and the deep learning model. This backend is responsible for handling API requests and routing communication between the frontend and the model.

On the frontend side, Ether.js, a library for interacting with the Ethereum blockchain, is integrated with React.js. This integration enables seamless communication with the blockchain network, facilitating the execution of transactions and interaction with smart contracts. Speaking of smart contracts, they are developed to manage transactions within the blockchain environment. These contracts are coded to handle specific functionalities and are subsequently deployed onto the Ethereum blockchain.

With the smart contracts in place, the React.js frontend is enhanced to incorporate features that trigger blockchain transactions using Ether.js. This enables users to interact with the blockchain directly from the frontend interface. Meanwhile, on the backend, REST commands are utilized to send or fetch data from the Flask server for model prediction. This ensures that data flows smoothly between the frontend, backend, and the deep learning model.

Finally, after development and testing phases are complete, the frontend, backend, and smart contracts are deployed to their respective environments. Rigorous testing is conducted to ensure functionality, security, and performance are up to standards. Once deployed, ongoing maintenance and monitoring are essential to address any issues that may arise and to ensure the smooth operation of the integrated system.

CHAPTER-4. DESGIN

4.1 UML Diagrams

A. Use Case Diagram:

Figure 4.1 represents a Use Case diagram with the User as the primary actor. The diagram outlines four key Use Cases: "Predict loan approval," "Saving data on blockchain," "Get all loan approval information on blockchain," and "Get information of particular loan approval." These represent various functionalities available to the user within the system, such as loan prediction, blockchain data storage, and retrieval of loan approval details.

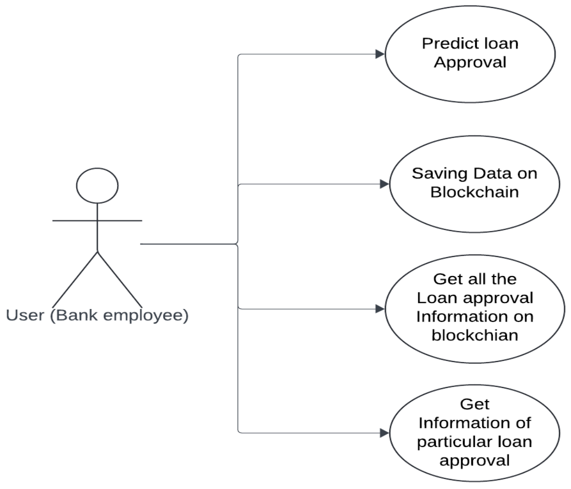

User (Bank employee)
Predict loan Approval
Saving Data on Blockchain
Get all the Loan approval Information on blockchian
Get Information of particular loan approval

Figure 4.1: Use Case Diagram

B. Sequence Diagram

The sequence diagram depicts message exchanges among objects in a proposed architecture, with the user as the main actor. Objects include React interface, Flash, deep learning model, and blockchain environment. It illustrates how messages flow between these entities, revealing the communication sequence within the system

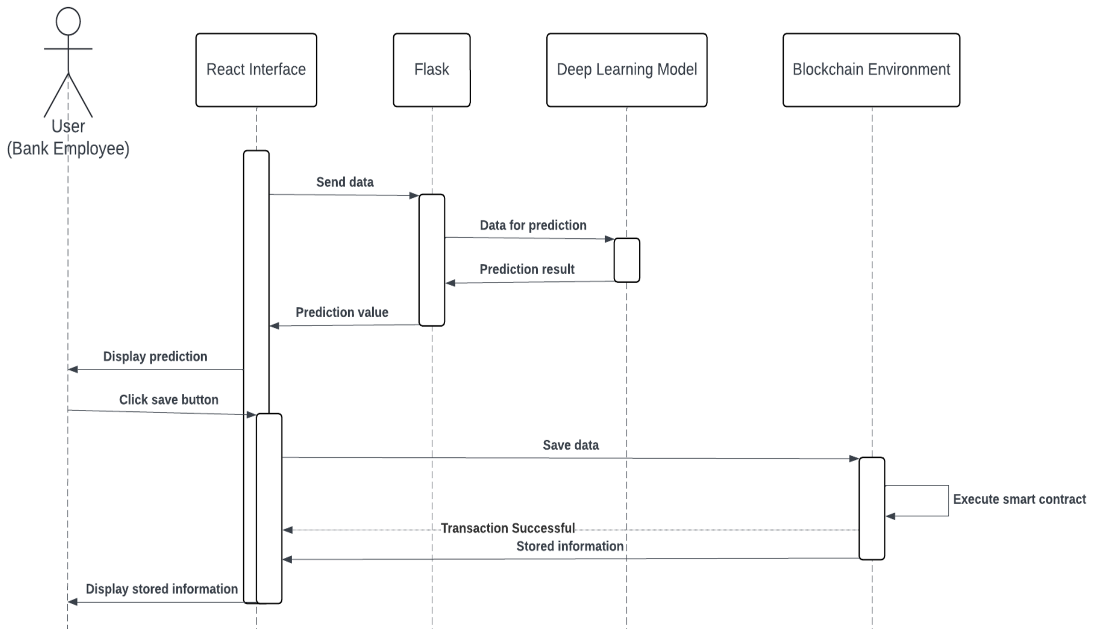


Figure 4.2: Sequence Diagram

C. Class Diagram

The class diagram illustrates several key classes: App, Flask, Resource, DLModel, Api, MinMaxScaler, BlockchainInterface, Web3Provider, Signer, and Contract. Each class possesses distinct attributes and functions tailored to its purpose. When functions are invoked, they execute operations based on their specific attributes. The functions within these classes are equipped with numerous attributes, each serving a designated role within the application's functionality.

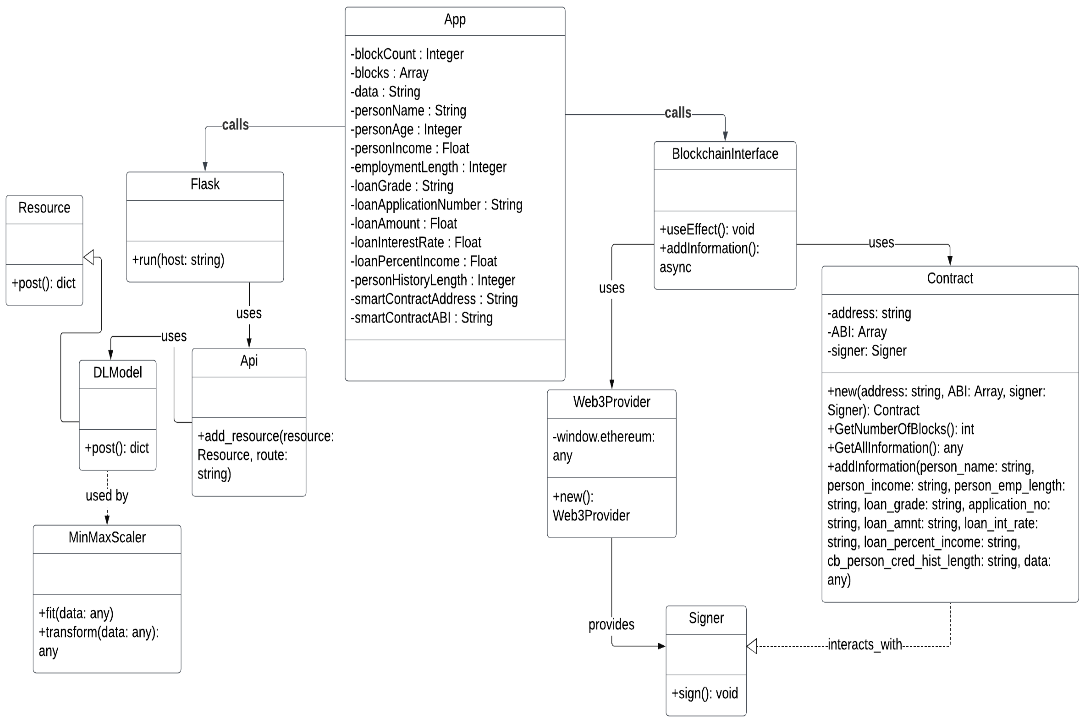

App
-blockCount : Integer
-blocks : Array
-data : String
-personName : String
-personAge : Integer
-personIncome : Float
-employmentLength : Integer
-loanGrade : String
-loanApplicationNumber : String
-loanAmount : Float
-loanInterestRate : Float
-loanPercentIncome : Float
-personHistoryLength : Integer
-smartContractAddress : String
-smartContractABI : String
calls
calls
Flask
+run(host: string)
Resource
+post(): dict
uses
uses
DLModel
+post(): dict
Api
+add_resource(resource: Resource, route: string)
used by
MinMaxScaler
+fit(data: any)
+transform(data: any): any
BlockchainInterface
+useEffect(): void
+addInformation(): async
uses
uses
Contract
-address: string
-ABI: Array
-signer: Signer
+new(address: string, ABI: Array, signer: Signer): Contract
+GetNumberOfBlocks(): int
+GetAllInformation(): any
+addInformation(person_name: string, person_income: string, person_emp_length: string, loan_grade: string, application_no: string, loan_amnt: string, loan_int_rate: string, loan_percent_income: string, cb_person_cred_hist_length: string, data: any)
Web3Provider
-window.ethereum: any
+new(): Web3Provider
provides
Signer
+sign(): void
interacts_with

Figure 4.3: Class Diagram

D. Component Diagram

The system comprises react.js for frontend, Flask backend for interfacing with a deep learning model (model.pkl), and Metamask for Ethereum interaction implementing smart contract. React.js interacts with Flask to utilize the model.pkl. and react.js verifies MetaMask presence, enabling interaction with a smart contract deployed via Remix IDE on Ethereum.

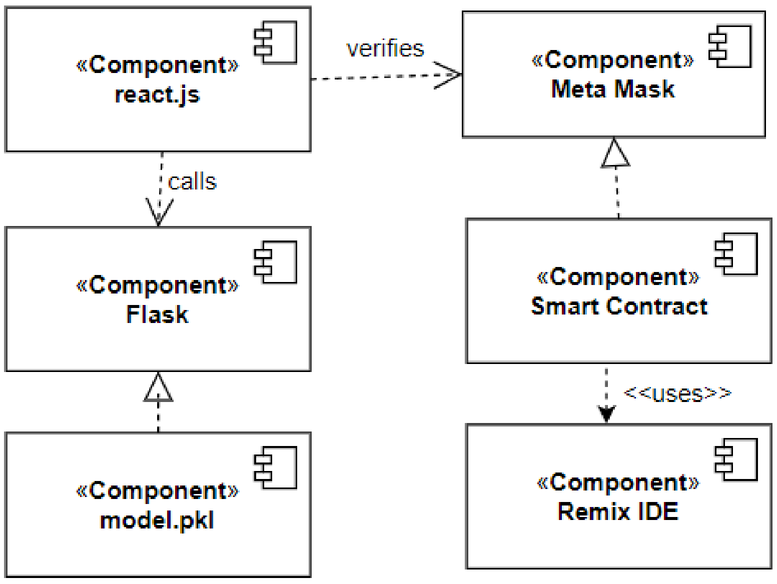


Figure 4.4: Component Diagram

E. Deployment Diagram

The deployment diagram illustrates the interaction between user devices represented by web browsers, and a web server. The server comprises a React.js front end, Flask backend utilizing a pickle file model.pkl, and integrates with a blockchain network for data storage and security.

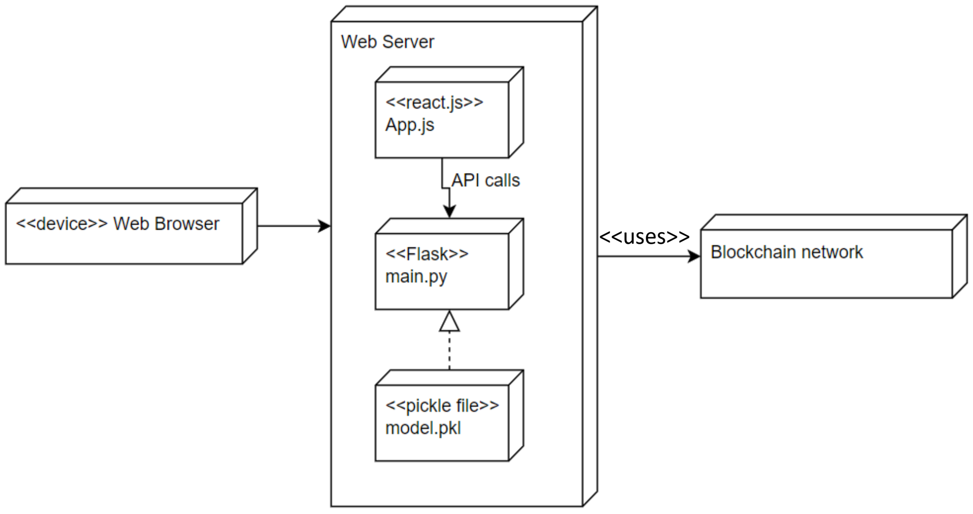


Figure 4.5: Deployment Diagram

CHAPTER-5. IMPLEMENTATION

5.1 List of Program Files

a. **main.py:** This is the main Python script that serves as the entry point for the virtual painter application. It imports necessary modules, initializes key variables. The python script is a configuration file for a Flask web application with RESTful API capabilities. Its primary purpose is to set up the Flask application environment and configure it for local development. The script initializes Flask, sets a secret key, and handles CORS (Cross-Origin Resource Sharing) to allow cross-origin requests. However, it seems some parts related to database configuration and API setup are commented out, indicating they might be under development or not necessary for the current functionality. Finally, the script runs the Flask application on the local server. Overall, it provides a foundation for building a Flask web application with RESTful API endpoints.

b. **model.py:** This Python script establishes a Flask web application with a single RESTful API endpoint designed to make predictions using a machine learning model.

   Imports:

   - The script imports necessary modules from Flask, Flask-RESTful, and scikit-learn (Flask, request, jsonify, Api, Resource, MinMaxScaler) for creating a web application, handling HTTP requests, and preprocessing data.
   - It also imports pickle, pandas, and numpy for loading a pre-trained machine learning model, manipulating data, and performing numerical computations.

   API Setup:

   - An instance of the Flask-RESTful Api class is created and associated with the Flask application (api = Api(app)).

   Resource Class DLModel

   - Defines a Flask-RESTful resource class named DLModel that inherits from Resource. This class represents the RESTful endpoint for making predictions.

Inside the post method:

- It attempts to load a pre-trained machine learning model (model1.pkl) and its associated training data from disk.
- Retrieves input data from the POST request in JSON format.
- Scales the input data using MinMaxScaler to ensure consistency with the model's training data.
- Makes predictions using the loaded model on the scaled input data.
- Formats the prediction output as either "YES" or "NO" based on a threshold of 0.5.
- Returns the prediction result as a JSON response.

Exception Handling:

- The execution inside the post method is wrapped in a try-except block to catch any potential exceptions.
- If an exception occurs during the prediction process, it returns an error message as a JSON response.

API Resource Registration:

- The DLModel resource is registered with the API at the /api/dlmodel endpoint.

Overall, this script provides a straightforward implementation for serving machine learning model predictions via a RESTful API using Flask. It demonstrates how to handle incoming data, preprocess it, and make predictions, while also handling potential errors gracefully.

c. **model1.pkl:** this file is a is a serialized Python object, used for storing deep learning model that we have trained. It preserves model architecture, parameters, and learned patterns, enabling easy loading and utilization without retraining. It contain relevant training data or metadata necessary for making predictions. Pickle serialization ensures cross-version compatibility, facilitating deployment in production environments.

d. **App.js:** This file is a React component (App.js) that serves as the front end for a loan approval prediction application.

Component Structure:

- The file begins with import statements for necessary libraries and components, including React, ethers (for Ethereum interaction), moment (for date formatting), and ToastContainer (for displaying notifications).

- The App() function defines the main React component, which renders the application's UI.

State Management:

- The component utilizes React's useState() hook to manage various state variables, such as input data (person_name, person_age, etc.), prediction results (data), and blockchain data (blockCount, blocks).

Data Fetching and Smart Contract Interaction:

- The fetchData() function sends input data to a Flask server (http://127.0.0.1:5000/api/dlmodel) for prediction using a deep learning model. It updates the data state with the prediction result.
- The component interacts with an Ethereum smart contract deployed at SC_address using the ethers.js library. It retrieves data from the blockchain and adds new information using functions like **GetAllInformation()**,**GetNumberOfBlocks()**, and **addInformation()**.

User Interface:

- The UI includes input fields for entering various loan application details.
- Users can predict loan approval status by clicking the "Predict" button and save loan application data to the blockchain by clicking the "Save Data" button.
- A search bar allows users to filter and search stored information based on application number or person's name.
- A table displays stored loan application information retrieved from the blockchain, including details like person's name, application number, income, employment length, etc.

Error Handling:

- The component includes basic error handling logic, such as validating input data and displaying warning messages using Toast notifications (react-toastify) for invalid inputs or failed interactions.

Overall, this file combines React components with Ethereum smart contract interactions and Flask server requests to create a web application for predicting loan approval status and managing loan application data.

5.2 Datasets

Our dataset Sourced from Kaggle dataset we used is "credit_risk_dataset".

The dataset, named "credit_risk_dataset," comprises various attributes related to individuals and their loan applications. These features include:

1. `person_age`: Age of the individual applying for the loan.
2. `person_income`: Income of the individual.
3. `person_home_ownership`: Home ownership status of the individual.
4. `person_emp_length`: Employment length of the individual.
5. `loan_intent`: Purpose of the loan.
6. `loan_grade`: Grade assigned to the loan.
7. `loan_amnt`: Amount of the loan requested.
8. `loan_int_rate`: Interest rate on the loan.
9. `loan_status`: Status of the loan application.
10. `loan_percent_income`: Percentage of income dedicated to the loan.
11. `cb_person_default_on_file`: Default status of the individual based on credit bureau information.
12. `cb_person_cred_hist_length`: Length of credit history of the individual.

Selected Features:

After careful consideration, the following features were selected for further analysis:

1. `loan_percent_income`
2. `loan_int_rate`
3. `loan_grade`
4. `loan_amnt`
5. `person_emp_length`
6. `person_age`
7. `cb_person_cred_hist_length`
8. `person_income`

These features were chosen based on their relevance and potential impact on assessing credit risk. They provide a comprehensive understanding of the applicant's financial situation, credit history, and loan characteristics, which are crucial factors in determining loan approval and assessing risk.

5.3 Other Support Files

a. Analysis.ipynb: The provided file appears to be the front-end component of a loan approval prediction application, focusing on the integration of a deep learning model for prediction. While it includes various functionalities for interacting with the model and displaying results, it lacks explicit documentation or code related to the analysis, data collection, preprocessing, model selection, training, testing, and result analysis stages typically associated with building and evaluating a deep learning model.

b. **data_storage.sol:** This Solidity smart contract, named "**data_storage**", serves as a decentralized storage solution for loan application data. It includes functionalities for adding new loan application information, retrieving information for all stored applications, and fetching details for a specific application by its unique application number. The contract maintains a count of the total number of stored applications through the "**blockCount**" variable and stores application numbers in an array named "**applicationNumbers**". Each loan application is represented by a struct named "**Block**", containing various fields (such as applicant's name, income, employment length, loan grade, and approval status). The "**addInformation**" function allows adding new application details to the contract's storage, while the "**GetAllInformation**" function retrieves all stored application data. Additionally, the contract provides methods to retrieve information for a specific application and obtain the total number of stored applications. Overall, this contract facilitates transparent and immutable storage of loan application records on the Ethereum blockchain, ensuring data integrity and accessibility while adhering to the principles of decentralization and transparency.

CHAPTER-6: EXPERIMENTAL RESULT/OBSERVATIONS

6.1 Experimental Setup

Application Stack

In our application development process, we utilized a diverse set of tools, languages, and frameworks to build a robust and functional application. Below are the components of our application stack:

Text Editors/IDEs:

- **Visual Studio Code:** Visual Studio Code is a popular and versatile code editor developed by Microsoft, offering features such as syntax highlighting, code completion, debugging, and version control integration.
- **RemixIDE:** RemixIDE is an online Integrated Development Environment (IDE) specifically designed for Solidity smart contract development. It provides a user-friendly interface for writing, debugging, and deploying Ethereum smart contracts.

Package Managers:

- **npm (Node Package Manager):** npm is the default package manager for Node.js and JavaScript, allowing developers to install, manage, and share packages of code easily.
- **pip (Python Package Installer):** pip is the package installer for Python, enabling users to install and manage Python packages from the Python Package Index (PyPI).

Web Extensions:

- **MetaMask:** MetaMask is a browser extension that serves as a cryptocurrency wallet and Ethereum gateway, allowing users to interact with decentralized applications (dApps) and the Ethereum blockchain directly from their web browser.

Frontend Libraries and Frameworks:

- **React.js:** React.js is a JavaScript library for building user interfaces, developed by Facebook. It enables the creation of interactive and reusable UI components, facilitating the development of dynamic and responsive web applications.

- **Ethers.js:** Ethers.js is a JavaScript library for interacting with the Ethereum blockchain and smart contracts. It provides a simple and intuitive API for sending transactions, querying contract data, and managing Ethereum wallets.

Backend Framework:

- **Flask:** Flask is a lightweight and flexible micro web framework for Python, suitable for building web applications and APIs. It provides tools and libraries for routing requests, handling HTTP responses, and integrating with databases and other services.

Development Languages:

- **JavaScript:** JavaScript is a versatile and widely-used programming language for web development, known for its compatibility with web browsers and its ability to create dynamic and interactive web pages.
- **Python:** Python is a high-level programming language known for its simplicity and readability. It is widely used for web development, data analysis, machine learning, and automation.
- **Solidity:** Solidity is a programming language specifically designed for writing smart contracts on the Ethereum blockchain. It is statically typed and supports inheritance, libraries, and complex user-defined types.

Testnet:

- **Sepolia Testnet:** Sepolia Testnet is a test network designed for Ethereum developers to deploy and test smart contracts and decentralized applications without using real Ether (ETH) on the Ethereum mainnet. It allows for experimentation and debugging in a simulated environment.

Libraries Used

In our analysis and model development process, we employed several libraries to facilitate data manipulation, visualization, machine learning, and model persistence. The following libraries were utilized:

1. NumPy: NumPy is a fundamental package for scientific computing in Python, providing support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays.

2. Pandas: Pandas is a powerful data analysis and manipulation library, offering data structures and operations for manipulating numerical tables and time series. It provides easy-to-use data structures and functions for data cleaning, exploration, and transformation.
3. Matplotlib: Matplotlib is a comprehensive library for creating static, interactive, and animated visualizations in Python. It allows users to create plots, histograms, scatterplots, and other types of visualizations to explore and communicate data insights effectively.
4. Scikit-learn: Scikit-learn is a widely-used machine learning library that provides simple and efficient tools for data mining and data analysis. It includes various algorithms for classification, regression, clustering, dimensionality reduction, and model selection, as well as utilities for preprocessing data and evaluating models.
5. TensorFlow: TensorFlow is an open-source machine learning framework developed by Google for building and training machine learning models, particularly deep learning models. It provides a flexible architecture for deploying computation across a variety of platforms (CPUs, GPUs, TPUs) and supports distributed computing.
6. Pickle: Pickle is a Python module used for serializing and deserializing Python objects. It allows objects to be serialized into a byte stream and saved to a file, which can then be loaded and deserialized back into memory. Pickle is commonly used for saving trained machine learning models to disk for later use without having to retrain the model.

These libraries played crucial roles in various stages of our analysis, from data preprocessing and exploration to model training, evaluation, and deployment.

6.2 Mathematical Formulas

Mean:

In preprocessing data, the mean formula is commonly used to calculate the average value of a set of numbers. The formula for the mean (often denoted by the symbol μ) is

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$\bar{x}$ = Mean

N = Number of values in the dataset

x_i = Input value

ReLU:

In deep learning models, Rectified Linear Unit (ReLU) is a commonly used activation function. Its formula is

$$\mathbf{f(x) = max(0, x)}$$

f(x) = output value after function is performed

x = input value

Normalization:

Normalization in mathematical formulas used in preprocessing typically refers to scaling data to a standard range, often between 0 and 1 or -1 and 1, to ensure consistency and improve the performance of machine learning algorithms.

$$\boldsymbol{x_{norm}} = \frac{\boldsymbol{x - x_{\min}}}{\boldsymbol{x_{\max} - x_{\min}}}$$

x = Original value you want to normalize

$x_{\min}$ = minimum value in the dataset

$x_{\max}$ = maximum value in the dataset

x_{norm} = normalised value

Linear function:

In deep learning models, the linear function is often used as the final layer in regression tasks or as part of other layers. The formula for a linear function is

$$\boldsymbol{f(x) = x}$$

f(x) = output value after function is performed

x = input value

Mean absolute error:

The mean absolute error (MAE) is a common loss function used in regression tasks in deep learning models. It measures the average absolute difference between the predicted values and the actual values. The formula for MAE is

$$\boldsymbol{MAE = \frac{1}{n}\sum_{i=1}^{n}|y_{i-}\hat{y}i|}$$

n = Total number of samples or data points

y_i = Actual value for the ith sample

$\hat{y}i$ = Predicted value for the ith sample

6.3 Experiments on Selecting Deep Learning Model

Feature Analysis and selection:

SNO	Number of features	Feature names	Accuracy
1.	4 features	loan_percent_income loan_int_rate loan_grade loan_amnt	0.8540
2.	5 features	loan_percent_income loan_int_rate loan_grade loan_amnt person_emp_length	0.8576
3.	6 features	loan_percent_income loan_int_rate loan_grade loan_amnt person_emp_length	0.8569

		cb_person_cred_hist_length	
4.	6 features	loan_percent_income loan_int_rate loan_grade loan_amnt person_emp_length person_age	0.8520
5.	7 features	loan_percent_income loan_int_rate loan_grade loan_amnt person_emp_length person_age cb_person_cred_hist_length	0.8523
6.	8 features	loan_percent_income loan_int_rate loan_grade loan_amnt person_emp_length person_age cb_person_cred_hist_length person_income	0.8575

Table 6.1: Analysis of feature selection

The model utilizing five features achieves the accuracy around 0.8576, the analysis reveals that the model employing eight features attains a commendable accuracy of 0.8575 while also incorporating all the requisite columns: loan_percent_income, loan_int_rate, loan_grade, loan_amnt, person_emp_length, person_age, cb_person_cred_hist_length, and person_income. Therefore, the model with eight features is selected as it balances accuracy with the inclusion of necessary features for a comprehensive analysis.

Experiment with change in neuron layers

Number of Layers	Number of neurons in each layer	Accuracy
2 layers	128,256	0.8437
2 layers	256,256	0.8500
3 layers	128, 256, 256	0.8519
3 layers	265, 256, 256	0.8500
3 layers	265, 128, 256	0.8503
4 layers	**128, 265, 256, 256**	**0.8553**
4 layers	256, 256, 256, 256	0.8533
5 layers	128, 256,256,256,256	0.8499
5 layers	256,256,256,256,256	0.8550

Table 6.2: Analysis with different neuron layers

Experiments with changing Epoches

Number of Epoch	**50**	100	200	1000
Accuracy	**0.8557**	0.8495	0.8460	0.8249
Precision	**0.6757**	0.6675	0.6512	0.5880
F1	**0.6012**	0.5637	0.5736	0.5543
recall	**0.6363**	0.6112	0.6099	0.5706

Table 6.3: Analysis with different epochs

Experiment with changing learning rate and Epoches

Learning rate	Epoch=50	Epoch=100
0.1	**0.8551**	0.8519
0.01	0.8515	0.8537
0.001	0.8539	0.8499
0.0001	0.8498	0.8522

Table 6.4: Analysis with different learning rate and epoches

Experiment using different optimizers:

Optimizers	Accuracy
Stochastic Gradient Descent (SGD):	0.7901
Adams: (Adaptive Moment Estimation):	**0.8550**
RMSprop (Root Mean Square Propagation):	0.8501
Adagrad (Adaptive Gradient Algorithm):	0.7901
Nadam	0.7901

Table 6.5: Analysis with different optimisers

This comprehensive approach aimed to develop a robust deep learning model capable of effectively handling the intricacies of the dataset and delivering improved predictive performance.

CHAPTER-7: DISCUSSION OF RESULTS

7.1 Model Evaluation

1.Confusion Matrix:

A confusion matrix was utilized to visualize the model's classification results, distinguishing between true positives, true negatives, false positives, and false negatives.

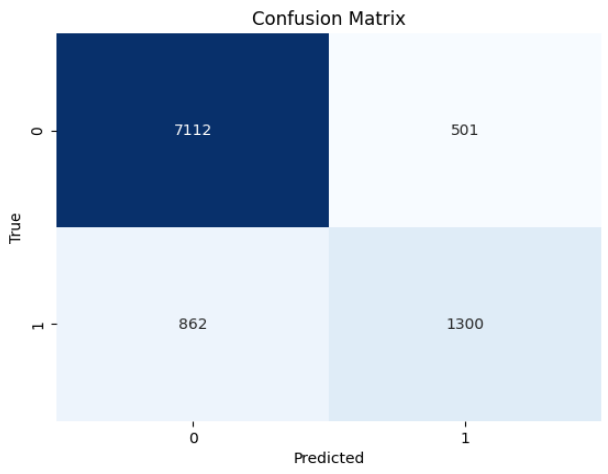


Figure 7.1 : Confusion matrix

The above confusion matrix states that:

- True Positives (TP): 7112
- False Negatives (FN): 862
- False Positives (FP): 501
- True Negatives (TN): 1300

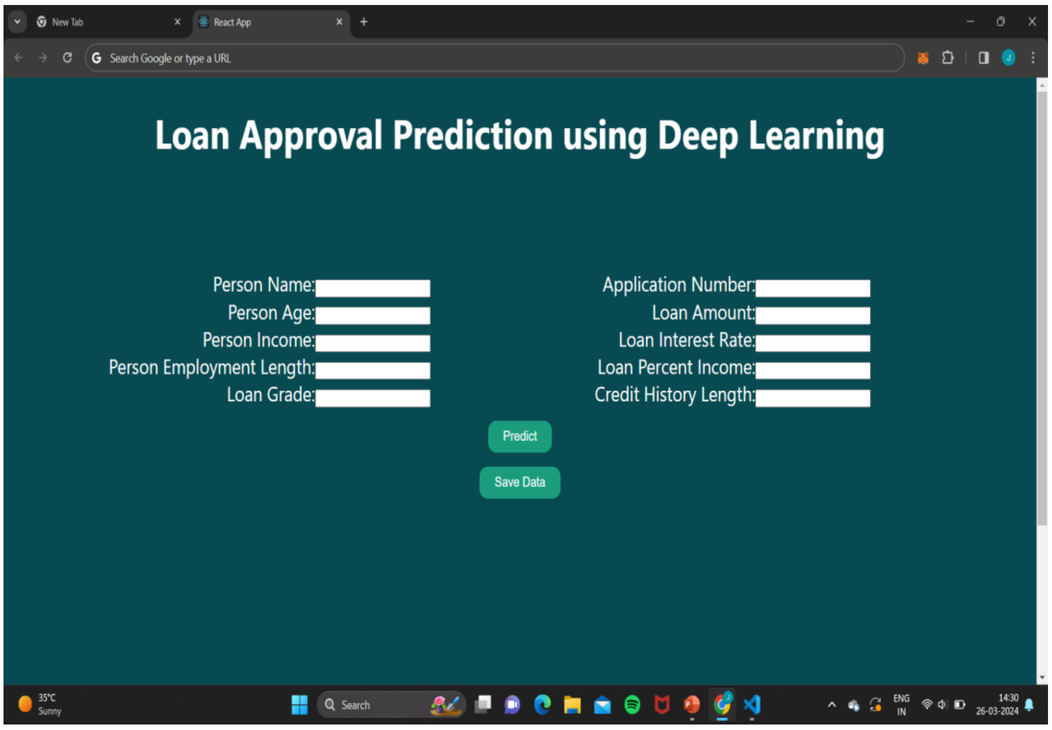
2. Performance Metrics:

The model's performance was evaluated using multiple metrics, including accuracy, precision, recall, F1-score, sensitivity, and specificity. These metrics provide a comprehensive assessment of the model's ability to classify loan applications accurately.

- Accuracy - 0.8605626598465473
- Precision - 0.7218212104386452
- Recall -0.6012950971322849
- F1 score -0.6560686348725713

7.2 User Interface and Application working

- **User Interface**

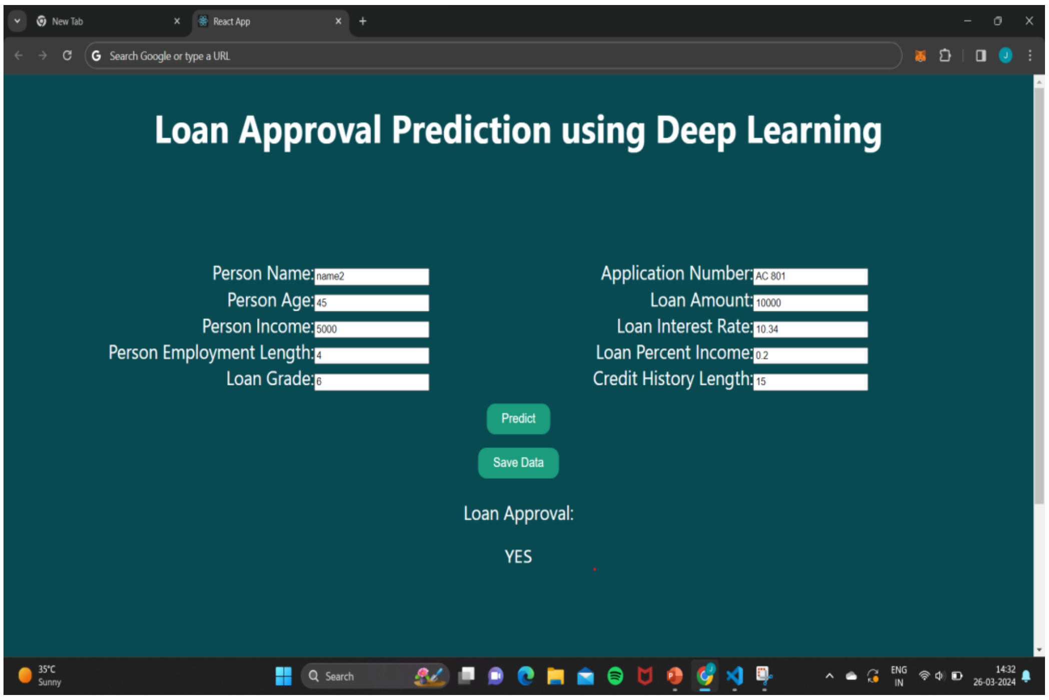


Loan Approval Prediction using Deep Learning

Person Name:		Application Number:	
Person Age:		Loan Amount:	
Person Income:		Loan Interest Rate:	
Person Employment Length:		Loan Percent Income:	
Loan Grade:		Credit History Length:	

Figure 7.2 : User Interface

- **Deep Learning Model Prediction Output Upon Clicking 'Predict' Button**



Loan Approval Prediction using Deep Learning

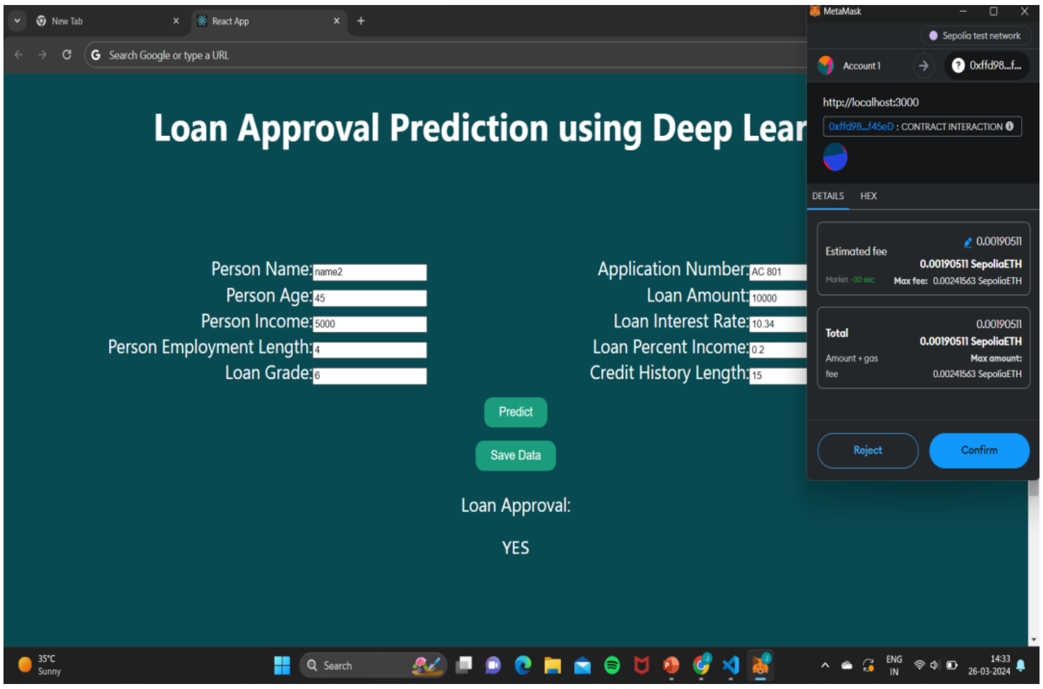
Person Name:	name2	Application Number:	AC 801
Person Age:	45	Loan Amount:	10000
Person Income:	5000	Loan Interest Rate:	10.34
Person Employment Length:	4	Loan Percent Income:	0.2
Loan Grade:	6	Credit History Length:	15

Loan Approval:

YES

Figure 7.3 : Model Prediction Output

- **Blockchain Initialization: Permission Request for Data Saving**



Loan Approval Prediction using Deep Lear

Person Name:	name2
Person Age:	45
Person Income:	5000
Person Employment Length:	4
Loan Grade:	6
Application Number:	AC 801
Loan Amount:	10000
Loan Interest Rate:	10.34
Loan Percent Income:	0.2
Credit History Length:	15

Loan Approval:

YES

MetaMask

Sepolia test network

Account 1

0xffd98...f...

http://localhost:3000

0xffd98...f45eD : CONTRACT INTERACTION

Estimated fee 0.00190511

0.00190511 SepoliaETH

Market ~30 sec Max fee: 0.00241563 SepoliaETH

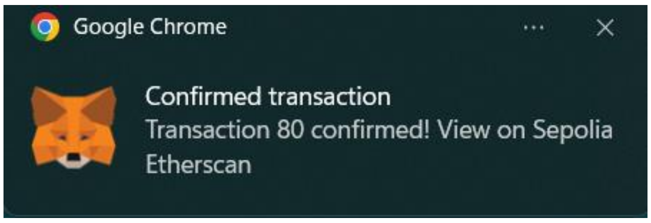
Total 0.00190511

0.00190511 SepoliaETH

Amount + gas fee Max amount: 0.00241563 SepoliaETH

Figure 7.4 : Saving upon Blockchian

- **Blockchain Transaction Confirmation via MetaMask**



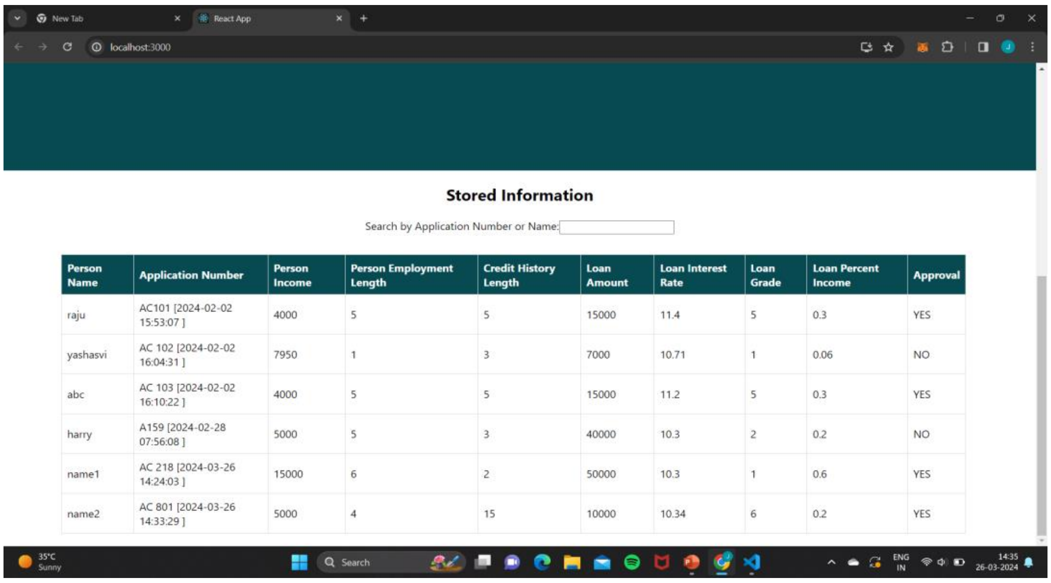
Google Chrome

Confirmed transaction

Transaction 80 confirmed! View on Sepolia Etherscan

Figure 7.5 : Confirmed Transaction

- **Displaying Stored Information via Smart Contract**



Stored Information

Search by Application Number or Name:

Person Name	Application Number	Person Income	Person Employment Length	Credit History Length	Loan Amount	Loan Interest Rate	Loan Grade	Loan Percent Income	Approval
raju	AC101 [2024-02-02 15:53:07]	4000	5	5	15000	11.4	5	0.3	YES
yashasvi	AC 102 [2024-02-02 16:04:31]	7950	1	3	7000	10.71	1	0.06	NO
abc	AC 103 [2024-02-02 16:10:22]	4000	5	5	15000	11.2	5	0.3	YES
harry	A159 [2024-02-28 07:56:08]	5000	5	3	40000	10.3	2	0.2	NO
name1	AC 218 [2024-03-26 14:24:03]	15000	6	2	50000	10.3	1	0.6	YES
name2	AC 801 [2024-03-26 14:33:29]	5000	4	15	10000	10.34	6	0.2	YES

Figure 7.6 : Saved Information on Blockchain

CHAPTER-8: SUMMARY, CONCLUSION, RECOMMENDATIONS

8.1 Summary

This project delves into a comprehensive study aimed at enhancing banking practices by predicting loan defaults, thereby managing risks effectively and optimizing lending strategies. It underscores the utilization of advanced machine learning techniques, leveraging datasets sourced from Kaggle, to predict potential defaulters. The study emphasizes the significance of considering a broad spectrum of personal attributes beyond basic financial data, including factors such as age, intent, credit history, loan amount, and duration. Moreover, it advocates for the integration of cutting-edge technologies like Deep Learning and Blockchain to modernize the loan approval process, ensuring heightened accuracy, security, and transparency.

The credit risk assessment model uses a neural network to predict loan approvals. Data is sourced from Kaggle, preprocessed to handle missing values and outliers, and converted to numerical format. The model's performance is evaluated using metrics like accuracy and precision. Blockchain technology is proposed for secure data storage, with components like smart contracts and MetaMask plugin ensuring transparency. The flow involves user interaction, data formatting, transaction confirmation, and data retrieval for display on the frontend.

8.2 Findings

Integration of Deep Learning and Blockchain: The project combines deep learning algorithms for loan default prediction with blockchain technology for secure data storage, offering a robust solution for banking processes.

Enhanced Loan Approval Efficiency: By leveraging advanced predictive analytics and machine learning models, the study aims to streamline loan approval processes, reducing processing times and enhancing overall efficiency in assessing credit risk.

Model Performance and Selection: Through extensive experimentation, a neural network model with multiple layers is selected for credit risk prediction, achieving an accuracy of 86% and demonstrating balanced performance in predicting loan outcomes.

Blockchain-Based Data Storage: The utilization of blockchain technology ensures the integrity and immutability of stored data, providing a transparent and secure framework for recording transactions and managing financial records.

User-Friendly Application: The developed application offers an intuitive user interface for inputting loan details, predicting loan status, and retrieving information stored on the blockchain. Dynamic output tables present retrieved data in a structured format for user clarity and ease of understanding.

8.3 Conclusion

The study introduces a groundbreaking decentralized application poised to transform loan approval processes by harnessing the synergies between Deep Learning models and blockchain technology for data storage. This innovative program ensures data integrity through the precise predictions of the Deep Learning model and the inherent immutability of blockchain storage. Looking forward, the project envisions significant expansions, including the integration of advanced encryption protocols to fortify data protection. Additionally, there are plans to enhance the model's adaptability to diverse datasets, thereby increasing its effectiveness across various domains. This forward-thinking approach addresses the urgent need for trustworthy loan approval systems, paving the way for heightened model accuracy and robust data storage solutions in the future.

8.4 Justification of Findings

Characteristics achieved by our application

Accuracy: Our application achieves an impressive accuracy rate of 86%, surpassing traditional methods in predicting loan approval outcomes. This heightened accuracy is crucial for financial institutions as it reduces the risk of approving loans to individuals who may default, thereby minimizing non-performing assets and potential losses.

Transparency: By integrating blockchain technology, our application ensures that all outputs, including loan approval decisions, are securely stored in an immutable and tamper-proof manner. This transparency enhances trust in the decision-making process, as stakeholders can verify the integrity of the data and transactions recorded on the blockchain. This feature is particularly vital in the finance sector, where transparency is essential for regulatory compliance and customer trust.

Efficiency: The end-to-end user-friendly website of our application streamlines the loan approval process, making it more efficient and accessible to users. With a seamless interface, users can input loan details effortlessly, receive predictions promptly, and access stored information with ease. This efficiency not only saves time for both users and financial institutions but also enhances the overall user experience, leading to increased satisfaction and improved operational efficiency.

Risk Mitigation: Through the integration of advanced predictive analytics and machine learning algorithms, our application effectively mitigates the risk associated with loan approvals. By accurately identifying potential defaulters, financial institutions can make more informed decisions, leading to a reduction in non-performing assets (NPAs) and improved financial stability. This proactive approach to risk management enhances the overall resilience of the banking sector and fosters a more sustainable lending environment.

CHAPTER-9: FUTURE ENCHANCEMENT

Enhance Model Inclusivity:

The future scope of the project involves extending the predictive capabilities to encompass non-employed individuals, thereby broadening the applicability of the model to a wider range of loan applicants.

Strengthen Blockchain Security Measures:

As a recommendation for future development, the project can explore integrating additional blockchain security algorithms and introducing consensus mechanisms to enhance the overall security and reliability of the system.

Implement File Upload Functionality:

A potential area for future expansion includes incorporating file uploading functionality, which would enable users to upload relevant documents or data for loan assessment. This enhancement could further streamline the loan approval process and improve user experience.

REFERENCES

[1] Gogia, S., and U. Sharma. "Blockchain and machine learning based peer-to-peer lending for the post-pandemic economy." (2021).

[2] Deborah, R. Nancy, et al. "An Efficient Loan Approval Status Prediction Using Machine Learning." *2023 International Conference on Advanced Computing Technologies and Applications (ICACTA)*. IEEE, 2023.

[3]. Arun, Kumar, Garg Ishan, and Kaur Sanmeet. "Loan approval prediction based on machine learning approach." *IOSR J. Comput. Eng* 18.3 (2016): 18-21.

[4]. Gupta, Kanishk, et al. "Loanification-loan approval classification using machine learning algorithms." *Proceedings of the International Conference on Innovative Computing & Communication (ICICC)*. 2021.

[5]. Kadam, Ashwini S., et al. "Prediction for loan approval using machine learning algorithm." *International Research Journal of Engineering and Technology (IRJET)* 8.04 (2021).

[6]. Pramod, Ms Kathe Rutika, et al. "Prediction of Loan Approval using Machine Learning Algorithm: A Review Paper." (2021).

[7]. Chowdhury, Minhaj Uddin, et al. "Blockchain application in banking system." *Journal of Software Engineering and Applications* 14.7 (2021): 298-311.

[8]. Fang, Weiwei, et al. "Deep learning anti-fraud model for internet loan: Where we are going." *IEEE Access* 9 (2021): 9777-9784

[9]. Amarnadh, Vadipina, and Nageswara Rao Moparthi. "Prediction and assessment of credit risk using an adaptive Binarized spiking marine predators' neural network in financial sector." *Multimedia Tools and Applications* (2023): 1-37.

[10]. AC, Ramachandra, and Vishwas KN. "Prediction of Loan Approval in Banks Using Machine Learning Approach." (2023).

[11]. Anurima Majumdar, Romik Banerjee, Rounak Ghosh, Sagar Ghosh, Rudradip Bhattacharjee5, Sayak Saha6, Subhrajit Pallob7, Antara Ghosal8, Palasri Dhar9, Nabaneeta Banerjee1. "Loan Prediction by using Machine Learning".

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN TECHNOLOGY June 2022 | IJIRT | Volume 9 Issue 1 | ISSN: 2349-6002

[12]. Mamatha, Madhura, Mahalakshmi Mukri. "PREDICTING LOAN APPROVAL USING MACHINE LEARNING". International Research Journal of Modernization in Engineering Technology and Science (Peer-Reviewed, Open Access, Fully Refereed International Journal) Volume:05/Issue:05/May-2023

[13]. Loan Prediction System Using Machine Learning Anant Shinde1 , Yash Patil2 , Ishan Kotian3 , Abhinav Shinde4 and Reshma Gulwani5 1,2,3,4Department of Information Technology, RAIT, Nerul, India 5 D.Y. Patil Deemed to be University, Ramrao Adik Institute of Technology, Nerul, Navi Mumbai, India

[14]. Jena, Soumya Ranjan, and S. Vasantha. "PREDICTION OF MODERNIZED LOAN APPROVAL SYSTEM BASED ON MACHINE LEARNING APPROACH."

[15]. Suryadevara, Chaitanya Krishna. "A NEW WAY OF PREDICTING THE LOAN APPROVAL PROCESS USING ML TECHNIQUES." International Journal of Innovations in Engineering Research and Technology 6.12 (2019): 38-48.

[16]. Abakarim, Youness, Mohamed Lahby, and Abdelbaki Attioui. "Towards an efficient real-time approach to loan credit approval using deep learning." 2018 9th International Symposium on Signal, Image, Video and Communications (ISIVC). IEEE, 2018.

[17]. Melese, Tamiru, et al. "Credit-Risk Prediction Model Using Hybrid Deep—Machine-Learning Based Algorithms." Scientific Programming 2023 (2023).

[18] Vishal Singh "Prediction of Modernized Loan Approval System Based on Machine Learning Approach",2021