

INDEX

1. INTRODUCTION

- 1.1 Project Overview
- 1.2 Purpose

2. LITERATURE SURVEY

- 2.1 Existing problem
- 2.2 References
- 2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

- 3.1 Empathy Map Canvas
- 3.2 Ideation & Brainstorming
- 3.3 Proposed Solution
- 3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

- 4.1 Functional requirement
- 4.2 Non-Functional requirements

5. PROJECT DESIGN

- 5.1 Data Flow Diagrams
- 5.2 Solution & Technical Architecture
- 5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

- 6.1 Sprint Planning & Estimation
- 6.2 Sprint Delivery Schedule
- 6.3 Reports from JIRA

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

- 7.1 Feature 1
- 7.2 Feature 2
- 7.3 Database Schema (if Applicable)

8. TESTING

- 8.1 Test Cases
- 8.2 User Acceptance Testing

9. RESULTS

- 9.1 Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

- Source Code
- GitHub & Project Demo Link

INTRODUCTION

1.1 PROJECT OVERVIEW:

One of the most critical factors which affect our country's economy and financial condition is the credit system governed by the banks. Banks across the globe recognize the process of credit risk evaluation.

The prediction of credit defaulters is one of the complex tasks for any bank. But by forecasting the loan defaulters, the banks may reduce their loss by lowering their non-profit assets so that recovery of approved loans can occur without any loss, and it can play as the contributing parameter of the bank statement. This makes the study of this loan approval prediction important. Machine Learning techniques are crucial and valuable in predicting these types of data.

Loan sanctioning and credit scoring forms a multi-billion dollar industry -- in the US alone. With everyone from young students, entrepreneurs, and multi-million dollar companies turning to banks to seek financial support for their ventures, processing these applications creates a complex and cumbersome task for any banking institution. As of 2022, more than 20 million people in the US have active loans owing a collective debt of 178 billion dollars. Despite that, more than 20% of all applicants were denied loans. The loan approval or rejection has enormous ramifications for both the applicant and the bank, causing possible opportunity costs for both parties. Banks like Wells Fargo and Morgan Stanley have looked at the use of AI in determining lending risk and developing a loan prediction system in recent years to overcome human bias and delays in the application processing time.

Traditional processes determine the risk by manually looking at the applicant's income, credit history, and several other dynamic parameters and creating a data-driven risk model. Despite using data science in this process, there is still a large amount of manual work involved. Researchers have recently explored the possibility of using deep learning in this process. For example, credit score and credit history are essential parameters for assessing the applicant's lending risk. DL- based approaches such as Embedding Transactional Recurrent Neural Networks (E.T.-RNN) compute the credit scores of applicants by looking at the history of their credit and debit card transactions. Such an approach eliminates the high dependency on manual intervention, extensive domain knowledge, and human bias in loan approval prediction.

1.2 PURPOSE:

A loan is the core business part of banks. The main portion of the bank's profit directly comes from the profit earned from the loans. Though the bank approves a loan after a regress process of verification and testimonial, still there's no surety whether the chosen hopeful is the right hopeful or not. This process takes new time while doing it manually. We can prophesy whether that particular hopeful is safe, and the whole testimonial process is automated by machine literacy style. Loan Prognostic is helpful for the retainer of banks as well as for the hopeful also.

However, developing such a model is challenging due to the increased demand for loans. The organizations can use prototypes of the model to make the correct decision to approve or reject the request for the loan customers. This work includes constructing an ensemble model by combining different machine-learning models. Banks struggle to get the upper hand over each other to enhance overall business due to tight competition. Credit Risk assessment is a crucial issue faced by Banks nowadays, which helps them to evaluate if a loan application can be a defaulter at a later stage so that they can go ahead and grant the loan or not. This helps the banks minimize possible losses and increase the volume of credits.

LITERATURE SURVEY

2.1 EXISTING PROBLEM:

Bank employees manually check applicants' details and give an eligible applicant the loan. Checking the details of all applicants takes a lot of time.

Checking the details of all applicants consumes a lot of time and effort. There are chances that human error may occur due to checking all details manually. There is a possibility of assigning loans to an ineligible applicant

2.2 REFERENCES:

References were made from the articles and papers and the following has been derived,

SNO	TITLE	PROPOSED WORK	TOOLS USED / ALGORITHM	TECHNOLOGY	ADVANTAGES / DISADVANTAGES
1	A federated learning-based approach for loan defaults prediction	The prediction system find difficult to learn the patterns of defaulters . In Previous training models we predict on the basic of same organization. It result in an inappropriate prediction.	Synthetic Minority Over-sampling Technique(SMOTE)	A machine learning algorithm	The legislative rules have already begun limiting the centralized approaches for private sensitive data and the idea of using centralized approaches

SNO	TITLE	PROPOSED WORK	TOOLS USED / ALGORITHM	TECHNOLOGY	ADVANTAGES / DISADVANTAGES
2	Loan default prediction using Diversified sensitivity Under sampling	The loan default prediction is to predict rather the borrower will delay the repayment or not.	<ul style="list-style-type: none"> Hybrid under sampling K-means Clustering 	Neural networks.	It can be used effectively in practice by reducing the misclassification costs of loan default prediction model.

SNO	TITLE	PROPOSED WORK	TOOLS USED / ALGORITHM	TECHNOLOGY	ADVANTAGES / DISADVANTAGES
3	An Approach for Prediction of Loan Approval using Machine Learning Algorithm	<p>The models are compared on the basis of the performance measures such as sensitivity and specificity.</p> <p>It focus on the credit score rather than gender and marital status</p>	The Logistic regression model	Machine Learning	Its deal with the choosing of the right parameter .

SNO	TITLE	PROPOSED WORK	TOOLS USED / ALGORITHM	TECHNOLOGY	ADVANTAGES / DISADVANTAGES
4	Bank Loan Prediction System using Machine Learning	primary objective is to predict whether the loan approval to a specific individual is safe or not	Logistic Regression Random Forest Django	Machine Learning	it is very clear that it reduces all the frauds done at the time of loan approval. it will not deal with some special cases when only one parameter is enough for the decision, but it is quite efficient and reliable in some instant

2.3 PROBLEM STATEMENT DEFINITION:

With the enhancement in the banking sector, many people apply for bank loans, but the bank has limited assets, which it grants to only limited people, so finding out to whom the loan can be given is a typical process for the banks. We tried to reduce this risk by selecting a credible person to save many bank efforts and assets. It was done by mining the previous records of the people to whom the loan was granted before, and based on these records, the machine was trained using the machine learning model, which gave the most accurate result. The main goal of this paper is to predict if a loan assignment to a specific person will be safe or not.

Finance companies and banks deal with different kinds of loans, such as education loans, shop loans, home loans, personal loans, etc., all are part of our country's loan types. All the companies and banks are in villages, towns, and cities. After the customer has applied for a loan, these banks/companies want to validate the customer details for that candidate's eligibility. The system's main purpose is to get applicant loans approved based on train models.

IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes.

It is a valuable tool to help teams better understand their users.

Creating an effective solution requires understanding the problem and the person experiencing it.

Creating the map helps participants consider things from the user's perspective, along with their goals and challenges.

1. What have we heard say? What can we imagine them saying?

- i. Hard to find the credit defaulters
- ii. Not able to verify identity
- iii. Not able to service

2. What are them Wants, needs, hopes and dreams? what other thoughts might influence their behavior?

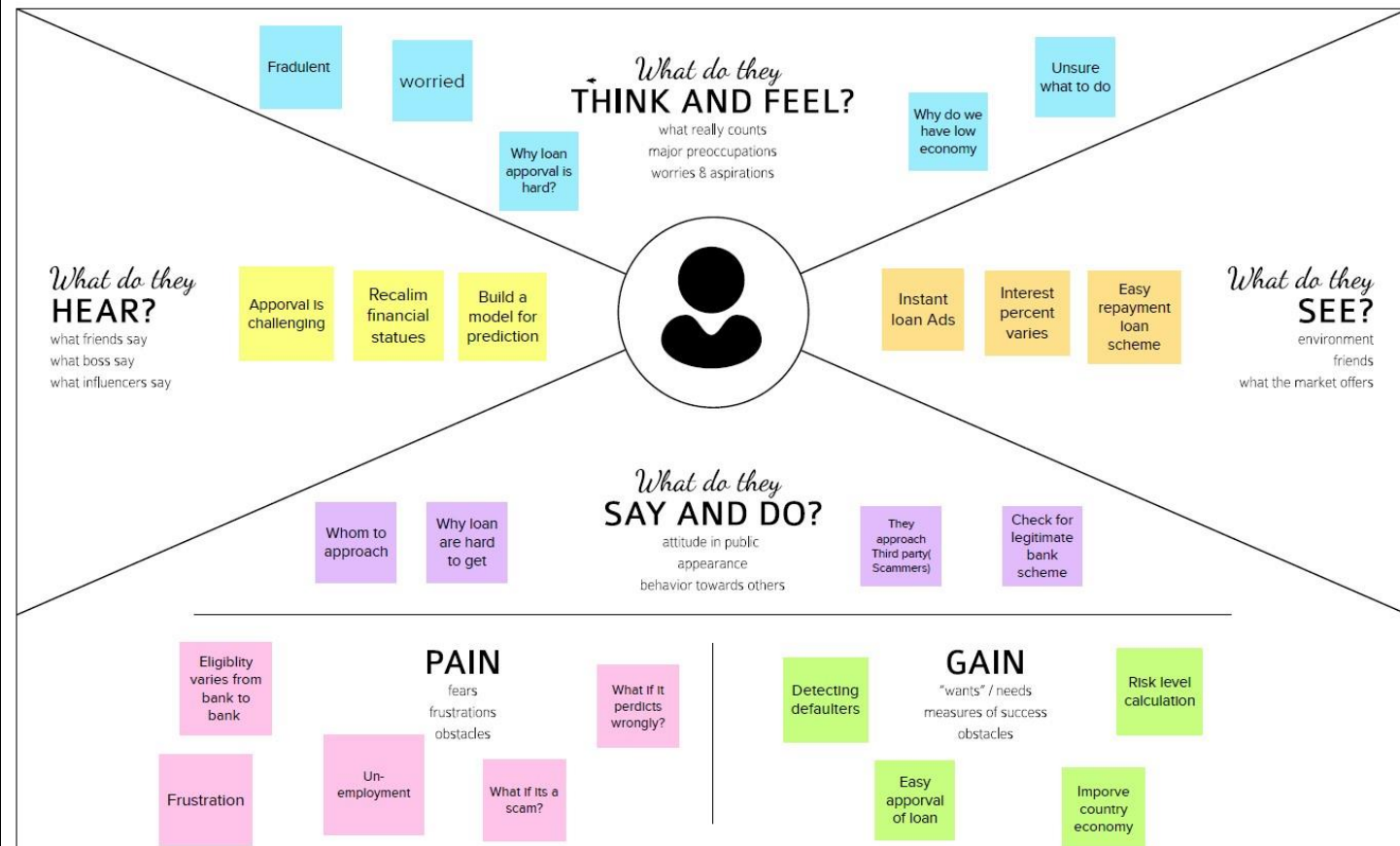
- i. Some methods/models to identify defaulters
- ii. Ability to minimize loss

3. What behavior have we observed ? what can we imagine them doing?

- i. Manual background checks
- ii. Requirement of an assert for loan
- iii. Manual verification

4. what are their fears, frustration and anxieties? What other feelings might influence their behavior?


- i. Are the given document genuine?
- ii. Will the customer be able to repay?
- iii. Can we service all types of customers?



3.2 IDEATION & BRAINSTORMING:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem-solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon. All participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Template




Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- 10 minutes to prepare
- 1 hour to collaborate
- 2-8 people recommended

[Share template feedback](#)



Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A

Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B

Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.

C

Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) →

1

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.


5 minutes

PROBLEM

Applicant Credibility


Prediction For Loan


Approval





Key rules of brainstorming


To run a smooth and productive session


 Stay in topic.

 Encourage wild ideas.

 Defer judgment.

 Listen to others.

 Go for volume.

 If possible, be visual.

2

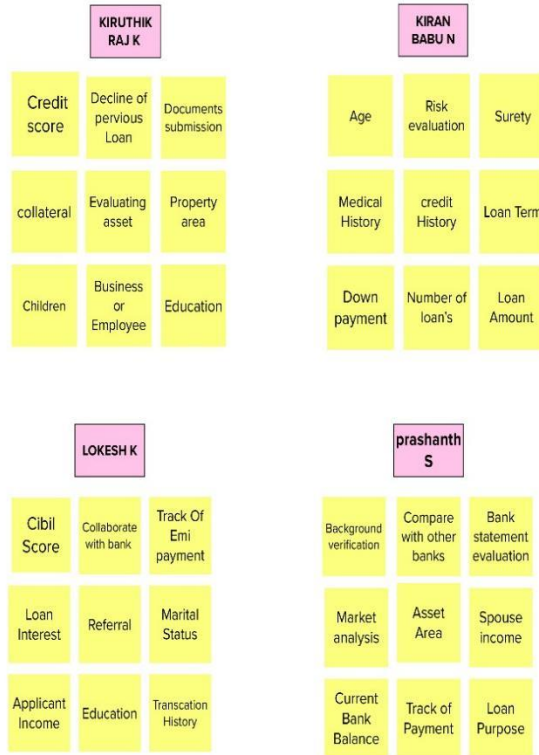
Brainstorm

Write down any ideas that come to mind that address your problem statement.

🕒 10 minutes

TIP

You can select a sticky note and hit the pencil (write) to switch to edit or the eraser (delete) to delete a sticky note.

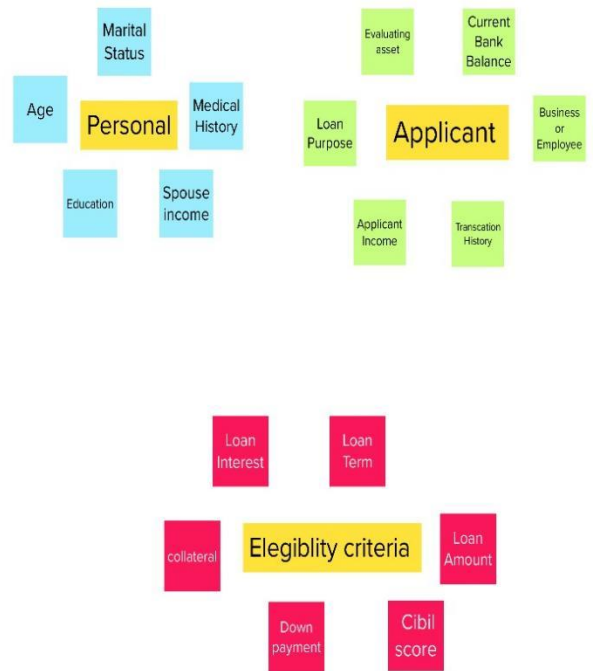


3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

🕒 20 minutes



4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

🕒 20 minutes



THE PROBLEM STATEMENT:

One of the most critical factors which affect our country's economy and financial condition is the credit system governed by the banks. The process of a bank's credit risk evaluation is recognized at banks across the globe. "As we know, credit risk evaluation is very crucial. There is a variety of techniques are used for risk level calculation. In addition, credit risk is one of the main functions of the banking community and should have some automation

Model:

- Model must have high accuracy
- Must be scalable
- Website can be made to access
- Future proof

DATA:

- ✓ Clean data works well
- ✓ Preprocessing must be done properly
- ✓ Need large data for generalization
- ✓ Privacy must be guarded
- ✓ Proper dataset should be used

WORKING OF MODEL:

- ❖ Uptime of model should be good
- ❖ Flask can be used for integration
- ❖ Model must be able to predict with great accuracy

MINDSET OF USER:

- i. UI must feel good
- ii. Ease to use

3.3 PROPOSED SOLUTION:

In our proposed system, we combine datasets from different sources to form a generalized dataset and use four machine learning algorithms, such as Random Forest, Logistic regression, Decision tree, and Naive Bayes algorithm, on the same dataset. The dataset we collected for predicting given data is split into the training set and a test set in the ratio of 8:2. The data model, which was created using Machine learning algorithms, applied to the training set and based on a maximum test result from the four algorithms, the test set prediction is made using the algorithm that has maximum performance. After that, we deploy the model using Flask Framework.

We developed automatic loan prediction using machine learning techniques to deal with the problem. We will train the machine with the previous dataset. So machines can analyze and understand the process. Then the machine will check for eligible applicants and give us the result. Advantages: The time period for loan sanctioning will be reduced. The whole process will be automated so that human error will be avoided. Eligible applicants will be sanctioned loans without any delay.

S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	The prediction of credit defaulters is one of the difficult tasks for any bank
2.	Idea / Solution description	We will be using classification algorithms such as Decision tree, Random Forest, KNN, and xgboost. We will train and test the data with these algorithms. From this best model is selected and saved in pkl format
3.	Novelty / Uniqueness	As soon as the dataset is provided, the model will predict whether to lend the loan or not.

4.	Social Impact / Customer Satisfaction	One of the most important factors which affect our country's economy and financial condition is the credit system governed by the banks. As we know credit risk evaluation is very crucial, there is a variety of techniques are used for risk level calculation. In addition, credit risk is one of the main functions of the banking community.
5.	Business Model (Revenue Model)	By using : can predict loan defaulters before lending the loan Without using : banks will have to face huge losses.
6.	Scalability of the Solution	Banks need not to go through the background verification process of the applicant by using this model. The model will predict the defaulter.

3.4 PROBLEM SOLUTION FIT:

Define CS, fit	1. CUSTOMER SEGMENT(S) CS <ul style="list-style-type: none"> Bankers Loan Officers Organization Account holders Type of card user 	6. CUSTOMER CONSTRAINTS CC <ul style="list-style-type: none"> Delay in approval of loan Unable to find the credit score Error in credibility of asset Financial in stable of customer 	5. AVAILABLE SOLUTIONS AS <ul style="list-style-type: none"> Manually done on the based on credit score. Existing machine learning models that are not reliable and fail in abnormal condition of user.
	2. JOBS-TO-BE-DONE / PROBLEMS J&P <ul style="list-style-type: none"> Prediction of loan Classification of yes or no Why declined of loan Arrangement of security No proper guidance Reference in lending loan delay in sanctioning loan 	9. PROBLEM ROOT CAUSE RC <p>Its various from person to person and their financial condition. When the person is new to the bank and as no history of loan and credit score . When the defaulter increases risk it affect the evolutions of other user.</p>	7. BEHAVIOUR BE <p>The user can select the type of loan and term. High interest rate affect the user and their living condition.</p>
Identify strong	3. TRIGGERS TR <ul style="list-style-type: none"> Be visible to client Highlight required scheme Make loan interest as low Keep process short 	10. YOUR SOLUTION SL <p>The proposed solution is the prediction of credit defaulters using classification algorithms and detect the credit risk evaluation .We use classification algorithms such as KNN and XGBOOST algorithms that forecast the loan defaulters and predict loan approval.</p>	8.CHANNELS of BEHAVIOUR CH <p>ONLINE</p> <ul style="list-style-type: none"> Prediction of loan approval easily know. Credit score is visible Bank statement .
	4. EMOTIONS: BEFORE / AFTER EM <p>When the loan is approved you feel joyed while its rejected we feel sad. It is easily attacked by the hackers.</p>		<p>OFFLINE</p> <ul style="list-style-type: none"> Submission of documents No proper treatment of customer Credit score and history affect the loan approval.

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT:

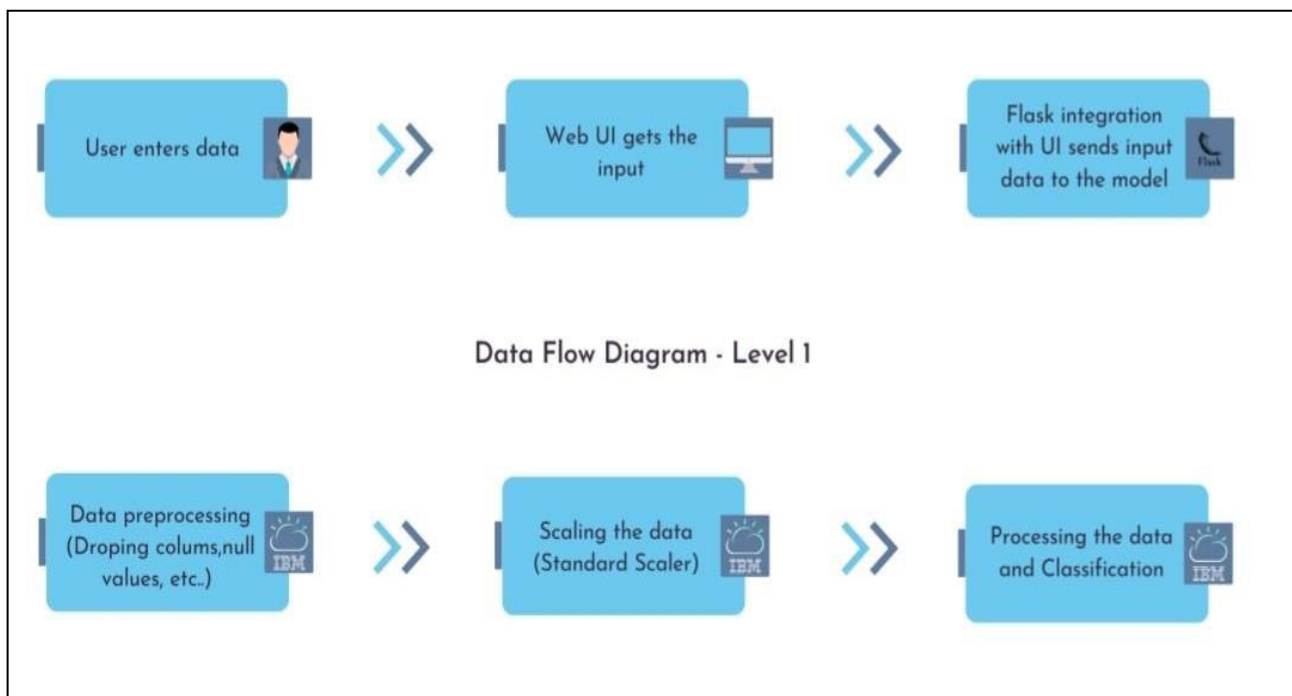
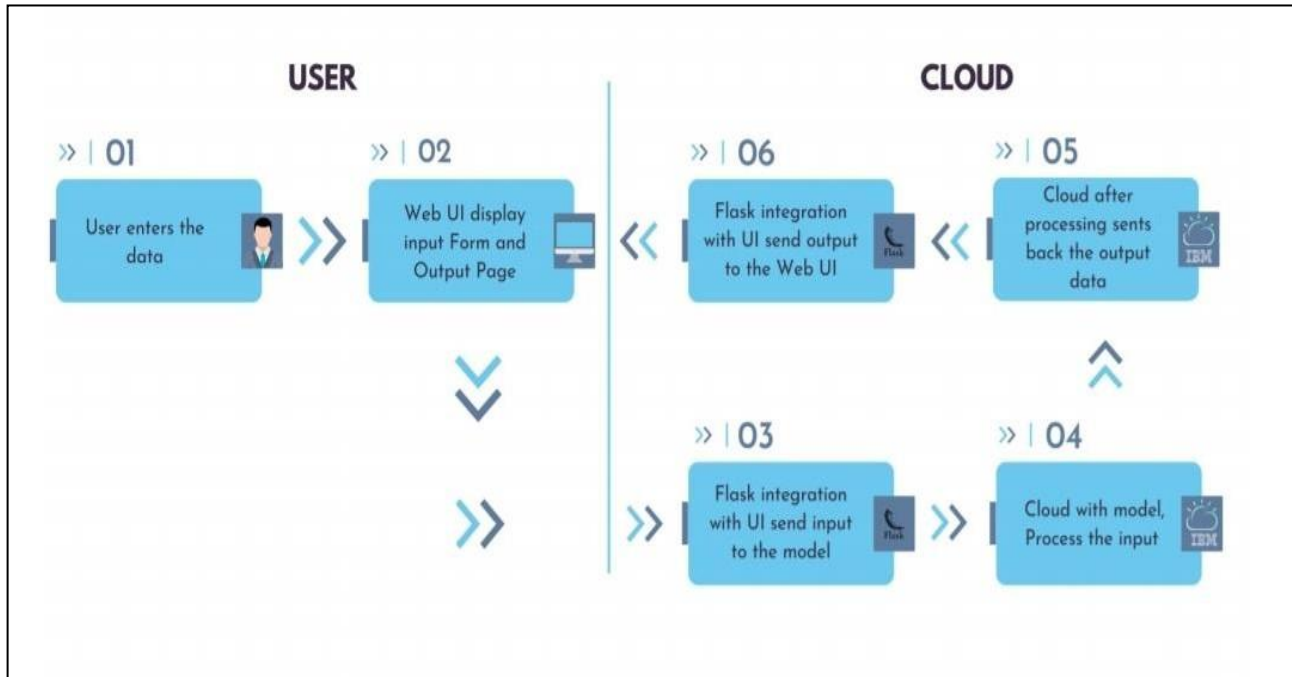
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User input	Fill the necessary details
FR-2	Eligibility of loan	Loan approval Loan rejection
FR-3	Chat bot	Clarifying user's doubts

4.2 NON-FUNCTIONAL REQUIREMENT:

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Simple and easy to use UI. User just need to fill the details.
NFR-2	Reliability	Ensemble the output of various ML models.
NFR-3	Performance	Web based application Ability to indicate error in user's input
NFR-4	Availability	Application is available for the users 24/7 as it is hosted in IBM cloud. Users can access the site through any web browser.
NFR-5	Scalability	Can be extended for other types of loans.

PROJECT DESIGN

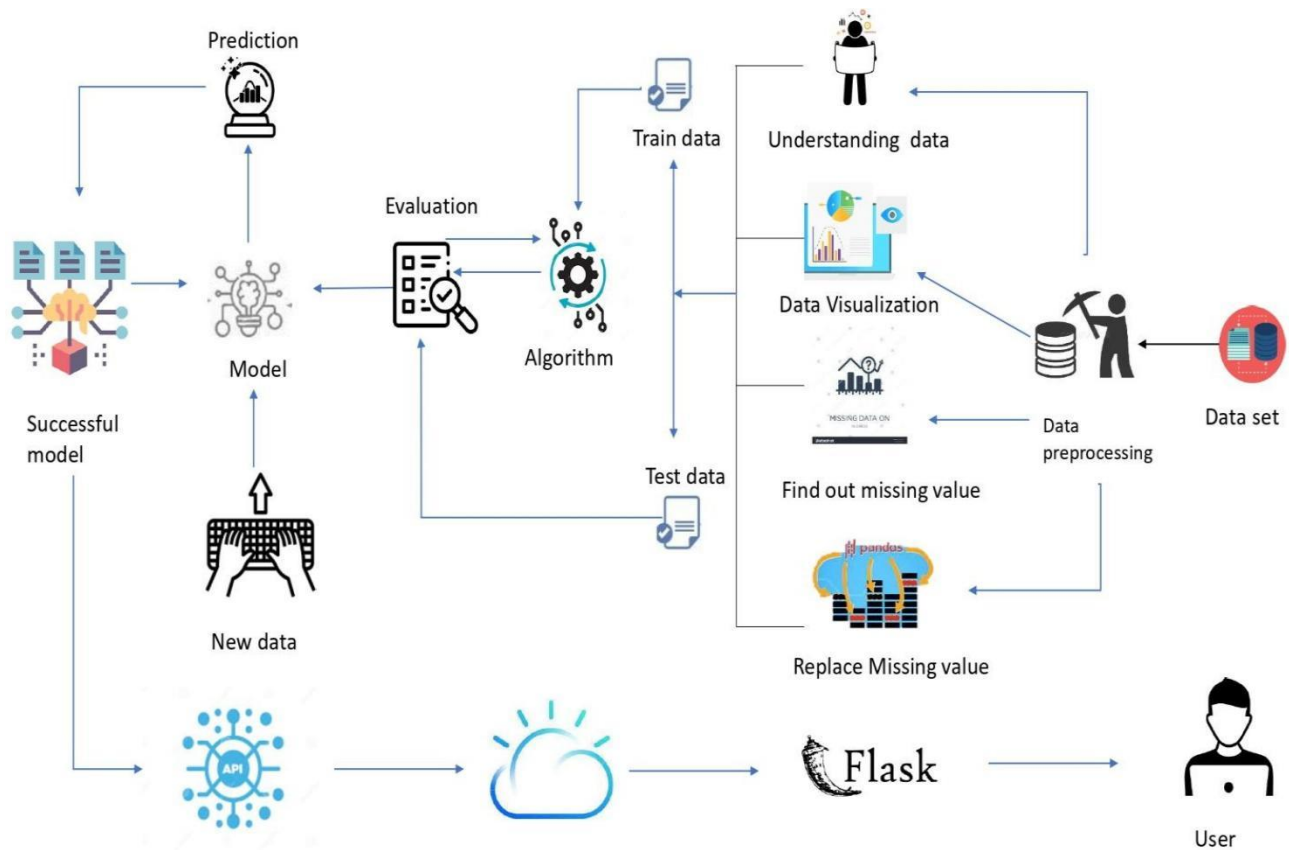
5.1 Data Flow Diagrams:



5.2 SOLUTION AND TECHNICAL ARCHITECTURE:

Solution Architecture: Solution architecture is a complex process – with many sub- processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the software's structure, characteristics, behavior, and other aspects to project stakeholders.
- Define features development phases and solution requirements.
- Provide specifications for the solution's definition, management, and delivery.



Explanation for the Architecture Diagram:

1. First, the Model is trained with the obtained dataset. IBM gives the data set
2. Next, the dataset will be pre-processed, and then the data will be split to train and test data.
3. Then the model would be saved as a PKL file
4. A website would be created for the interaction, and Flask would be used to integrate the model and website
5. The User would give the input; the inputs would be processed, and then the prediction would be made.
6. After the prediction is made, the output would be given as “Eligible” or “Not Eligible.”
7. This can be scaled even more as an API and integrated into the Mobile banking application, making it even more convenient for the customer to know the eligibility

5.3 USER STORIES:

- Need for the data to be clean enough for Model Prediction
- As a user, I would need a place to enter my data to predict my results.
- As the data is clean now, the data can be used to Train and Evaluate the results.
- Using Flask, we can now integrate the Model.
- with the input given by the user
- After Complete integration, now the Model should be deployed in IBM Cloud and put to use

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION:

SPRINT 1:

A sprint is an Agile methodology that helps you to complete a set amount of work in a time- boxed period. In this, we have done our data preprocessing and loading a dataset, leading to splitting datasets into train sets and test sets. This process is explained as follows,

Dataset

A set of data used to train the model is known as a machine learning dataset. A dataset is used as an example to educate the machine learning algorithm on how to generate predictions.

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
1	LP001002	Male	No	0	Graduate	No	5849	0		360	1	Urban	Y
2	LP001003	Male	Yes	1	Graduate	No	4583	1508	128	360	1	Rural	N
3	LP001005	Male	Yes	0	Graduate	Yes	3000	0	66	360	1	Urban	Y
4	LP001006	Male	Yes	0	Not Graduate	No	2583	2358	120	360	1	Urban	Y
5	LP001008	Male	No	0	Graduate	No	6000	0	141	360	1	Urban	Y
6	LP001011	Male	Yes	2	Graduate	Yes	5417	4196	267	360	1	Urban	Y
7	LP001013	Male	Yes	0	Not Graduate	No	2333	1516	95	360	1	Urban	Y
8	LP001014	Male	Yes	3+	Graduate	No	3036	2504	158	360	0	Semiurban	N
9	LP001018	Male	Yes	2	Graduate	No	4006	1526	168	360	1	Urban	Y
10	LP001020	Male	Yes	1	Graduate	No	12841	10968	349	360	1	Semiurban	N
11	LP001024	Male	Yes	2	Graduate	No	3200	700	70	360	1	Urban	Y
12	LP001027	Male	Yes	2	Graduate		2500	1840	109	360	1	Urban	Y
13	LP001028	Male	Yes	2	Graduate	No	3073	8106	200	360	1	Urban	Y
14	LP001029	Male	No	0	Graduate	No	1853	2840	114	360	1	Rural	N
15	LP001030	Male	Yes	2	Graduate	No	1299	1086	17	120	1	Urban	Y
16	LP001032	Male	No	0	Graduate	No	4950	0	125	360	1	Urban	Y
17	LP001034	Male	No	1	Not Graduate	No	3596	0	100	240		Urban	Y
18	LP001036	Female	No	0	Graduate	No	3510	0	76	360	0	Urban	N
19	LP001038	Male	Yes	0	Not Graduate	No	4887	0	133	360	1	Rural	N
20	LP001041	Male	Yes	0	Graduate		2600	3500	115		1	Urban	Y
21	LP001043	Male	Yes	0	Not Graduate	No	7660	0	104	360	0	Urban	N
22	LP001046	Male	Yes	1	Graduate	No	5955	5625	315	360	1	Urban	Y
23	LP001047	Male	Yes	0	Not Graduate	No	2600	1911	116	360	0	Semiurban	N

This is how an original dataset would look and will be provided in a CSV file format which is then preprocessed by evaluating the data set.

Preprocessing

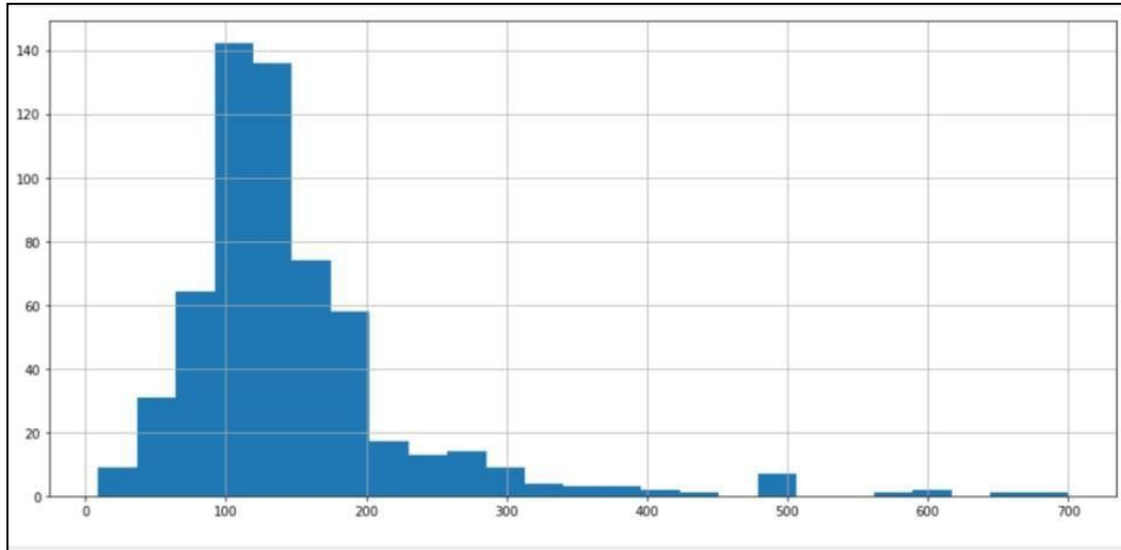
Data mining methods are used in preprocessing to normalize the data collected from Kaggle. There is a need to convert because the dataset may have missing values and noisy data. So, we are using a data mining method for the cleaning method. Before using the model selection process, we used preprocessing method to reduce the null values and then recovered the data with the help of train/test split with the help of MinMaxScaler. *Minmax Scalar*, for each value in every feature, Minmax Scalar cipher the minimum value within the feature and then divided by the vary. The range is the distinction between the first most and the original minimum. It preserves the shapes of the first original distribution.

Data preprocessing is a stage in data analysis that converts raw data into a format that computers and machine learning software can understand and evaluate. Machines like to process neat and orderly information; they interpret data as 1s and 0s. Therefore, it is simple to calculate structured data like whole numbers and percentages. Unstructured data must be organized and cleaned up before analysis, though.

Preprocessing involves specific methods that have to be performed to check whether the given dataset is valid for use. Those methods are explained step by step in the following.

i. **UNDERSTANDING THE FEATURES OF DATASET:**

We will begin at a basic level by identifying the structure of, and then the types of data in front of us. Once we can understand what the data is, we can start to fix problems with the data. For example, we must know how much of our information is missing and what to do when we have missing data.



ii. ANALYSIS OF CATEGORICAL DATA:

Categorical data classifies an observation as belonging to one or more categories. For example, an item might be judged as good or bad, or a response to a survey might include categories

such as agree, disagree, or no opinion. Stat graphics have many procedures for dealing with such data, including modeling procedures in the Analysis of Variance, Regression Analysis, and Statistical Process Control sections.

```
df['Loan_Status'].value_counts()['Y']
```

422

```
pd.crosstab(df ['Credit_History'], df ['Loan_Status'], margins=True)
```

Loan_Status	N	Y	All
Credit_History			
0.0	82	7	89
1.0	97	375	475
All	179	383	564

```
def percentageConvert(ser):
    return ser/float(ser[-1])
```

```
tabs = pd.crosstab(df ["Credit_History"], df ["Loan_Status"], margins=True).apply(percentageConvert,
    tabs
```

Loan_Status	N	Y	All
Credit_History			
0.0	0.921345	0.078652	1.0
1.0	0.204211	0.795789	1.0
All	0.317376	0.682624	1.0

```
app_loan = tabs['Y'][1]
print(f'{app_loan*100:.2f} % applicants got their loans approved')
```

79.58 % applicants got their loans approved

iii. **THE CHOICE OF AN BETTER DATASET TO TRAIN:**

Machine learning is a subset of AI that trains machines with vast volumes of data to think and act like humans without being explicitly programmed.

The models are available in python open-source software. There are various machine learning methods to perform an algorithm, but we have to choose the best algorithm for the enhancement of the model and for better accuracy.

Decision Trees

The basic algorithmic rule of the call tree needs all attributes or options to be discredited. Feature choice relies on the most significant info gain of possibilities. The data pictured in the call tree will delineate the IF-THEN rules. This model is an associated degree extension of C4.5 classification algorithms represented by Quinlan.

Random Forest

Random forests are a classifying learning framework for characterization (and backslide) that works by building a very large number of Decision trees at planning time and yielding the class that's the mode of the classes surrendered by individual trees.

Support Vector Machine

Used SVM to build and train a model, prepare a demonstration utilizing human cell records, and classify cells to whether the tests are benign (mild state) or dangerous (evil state).

Support vector machines are managed learning models that utilize affiliation R-learning calculation, which analyzes attributes and distinguished design information for application classification. SVM can beneficially perform a replacement using the kernel trick, verifiably mapping their inputs into high dimensional attribute spaces.

K-nearest neighbor (KNN)

The KNN algorithm is a simple supervised machine learning algorithm that can unravel classification and replace issues. It is easy to implement and understand but significantly slows as the size of that data on use grows.

In all of these, RANDOM FOREST has given the best accuracy alongside a better result. Random forest is an ensemble learning method for both classification and replacement issues. The advantage of random decision forest is reduced fitting and helps to improve accuracy and runs efficiently on large

datasets and works on both continuous and categorical values, and predicts the analysis of data with the help of test data. We used data mining methodology to predict the likely default from a dataset that contains information about home loan applications, thereby helping the banks for making better decisions in the future. On this basis, the prediction methodology based on the LSTM and SVM methods is proposed, the prediction results are compared with the traditional algorithm, and the feasibility of the model is confirmed. And this random forest has helped to select the best dataset to train, and the further process is carried out.

iv. **HANDLING THE DATASET WITH NULL VALUES:**

There are many techniques for handling null values. Which techniques are appropriate for a given variable can depend strongly on the algorithms you intend to use, as well as statistical patterns in the raw data—in particular, the missing values' **missingness** and the randomness of the locations of the missing values. Moreover, different techniques may be appropriate for different variables in a given data set. Sometimes it is helpful to apply several techniques to a single variable. Finally, note that corrupt values are generally treated as nulls.

How to handle null values

1. Deleting Rows

2. Replacing with

Mean/Median/Mode

3. Assigning
A Unique Category

4. Predicting the Missing Values

5. Using Algorithms Which Support Missing Values

➤ Deleting Rows

This method is commonly used to handle null values. Here, we either delete a particular row if it has a null value for a particular feature and a particular column

if it has more than 70-75% missing values.

This method is advised only when enough samples are in the data set.

➤ Replacing with Mean/Media/Mode:

This strategy can be applied to a feature that has numeric data like the age of a person or the ticket fare. We can calculate the mean, median, or mode of the feature and replace it with the missing values.

```
df['Gender'].fillna(df['Gender'].mode()[0], inplace=True)
df['Married'].fillna(df['Married'].mode()[0], inplace=True)
df['Dependents'].fillna(df['Dependents'].mode()[0], inplace=True)

df['LoanAmount'].fillna(df['LoanAmount'].mean(), inplace=True)
df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mean(), inplace=True)
df['ApplicantIncome'].fillna(df['ApplicantIncome'].mean(), inplace=True)
df['CoapplicantIncome'].fillna(df['CoapplicantIncome'].mean(), inplace=True)

df['Gender'].fillna(df['Gender'].mode()[0], inplace=True)
df['Married'].fillna(df['Married'].mode()[0], inplace=True)
df['Dependents'].fillna(df['Dependents'].mode()[0], inplace=True)
df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0], inplace=True)
df['Credit_History'].fillna(df['Credit_History'].mode()[0], inplace=True)
```

➤ Assigning A Unique Category

A categorical feature will have a definite number of possibilities, such as gender. Since they have a definite number of classes, we can assign another class for the missing values.

➤ Predicting The Missing Values

Using the features which do not have missing values, we can predict the nulls with the help of a machine learning algorithm. This method may result in better accuracy unless a missing value is expected to have a high variance.

➤ Using Algorithms Which Support Missing Values

KNN is a machine learning algorithm that works on the distance measure principle. This algorithm can be used when there are nulls present in the dataset. While the algorithm is applied, KNN considers the missing values by taking the majority of the K nearest value.

df.isnull().any()

Gender

False

Married

False

Dependents

False

Education

False

Self_Employed

False

ApplicantIncome

False

CoapplicantIncome

False

LoanAmount

False

Loan_Amount_Term

False

Credit_History

False

Property_Area

False

Loan_Status

False

dtype: bool

df.head()

Gender

Married

Dependents

Education

Self_Employed

ApplicantIncome

CoapplicantIncome

LoanAmount

Loan_Amount_Term

Credit_History

Property_Area

0

Male

No

0

Graduate

No

5.674026

0.0

4.537444

360.0

1.0

1

Male

Yes

1

Graduate

No

5.430109

1505.0

4.532030

360.0

1.0

2

Male

Yes

0

Graduate

Yes

5.006365

0.0

4.159655

360.0

1.0

3

Male

Yes

0

Not Graduate

No

7.556707

2355.0

4.757492

360.0

1.0

4

Male

No

0

Graduate

No

5.699315

0.0

4.945760

360.0

1.0

vi TRAINING DATASET:

Training data (or a training dataset) is the initial data used to train machine learning models. Training datasets are fed to machine learning algorithms to teach them how to make predictions or perform a desired task.

Training a dataset depends upon the method we train our dataset; on this basis, we have two divisions supervised and unsupervised learning.

Random forest is a supervised learning algorithm. A random forest is an ensemble of decision trees combined with a bagging technique.

In bagging, decision trees are used as parallel estimators. Combining many decision trees in parallel greatly reduces the risk of overfitting and results in a much more accurate model.

The success of a random forest highly depends on using uncorrelated decision trees. If we use the same or similar trees, the overall result will not be much different than that of a single decision tree. Random forests achieve to have uncorrelated decision trees by bootstrapping and feature randomness.

Bootstrapping is randomly selecting samples from training data with replacement. They are called bootstrap samples. Feature randomness is achieved by selecting features randomly for each decision tree in a random forest. The number of features used for each tree in a random forest can be controlled with the `max_features` parameter.

1	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
2	1	1	0	0	0	8.6993147463210191	2230.0	3.370720823966222	360.0	1	1	0
3	1	1	0	0	0	7.992265643270743	2900.0	4.573107333201131	360.0	1	1	1
4	1	1	2	0	0	8.740336742730447	1695.0	3.3471073307174683	360.0	1	1	1
5	1	1	0	0	0	7.641364441260972	3130.0	4.592030263994017	360.0	1	1	1
6	1	0	0	0	0	8.334711621820917	0.0	4.384967473670372	360.0	0	1	0
7	1	1	0	0	0	9.559895399016661	3266.0	6.343636360525306	360.0	1	0	0
8	1	1	3	1	1	8.111632807830774	2166.0	4.367334430433382	360.0	1	1	1
9	1	1	2	1	0	8.121153242078325	1917.0	4.718498571293094	360.0	0	0	0
10	0	1	0	0	0	8.1936009367288773	0.0	4.68213122712422	360.0	1	1	1
11	1	0	0	0	0	9.130558325450053	0.0	5.23110616354357	360.0	1	0	1

vi. TESTING THE DATASET:

Once we train the model with the training dataset, it's time to test the model with the test dataset. This dataset evaluates the model's performance and ensures that the model can generalize well with the new or unseen dataset. ***The test dataset is another subset of the original data independent of the training dataset.*** However, it has some similar features and class probability distribution and uses it as a benchmark for model evaluation once the model training is completed. Test data is a well-organized dataset containing data for each type of scenario for a given problem that the model would face when used in the real world. Usually, the test dataset is approximately 20-25% of the total original data for an ML project.

At this stage, we can also check and compare the testing accuracy with the training accuracy, which means how accurate our model is with the test dataset against the training dataset. If the model's accuracy on training data is more significant than that on testing data, then the model is said to have overfitting.

The testing data should:

- Represent or part of the original dataset.
- It should be large enough to give meaningful predictions.

1	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
2	1	1	0	0	0	9.114139513302214	0.0	3.429345628954441	360.0	1	1	1
3	1	1	0	0	0	8.368693183097793	0.0	4.567334430453352	360.0	1	1	1
4	1	1	2	0	0	8.334931631422434	1447.0	3.062393033026967	360.0	1	0	1
5	0	0	0	0	0	7.9724660139745633	0.0	4.2626798770413133	360.0	1	0	1
6	1	0	0	0	0	7.907631394711039	0.0	4.248493242049339	360.0	1	1	1
7	1	1	1	0	0	7.453491605030754	2232.0	4.67252534461906	360.0	1	0	1
8	1	1	2	0	0	8.220672170297232	0.0	4.757491742782046	360.0	1	1	1
9	1	1	0	0	1	8.006367367630246	0.0	4.159634742026423	360.0	1	2	1
10	1	1	3	1	0	7.581152202227102	1357.0	3.133291394497779	360.0	1	0	0
11	1	1	1	0	1	6.907733278982137	3022.0	4.700480363792417	360.0	1	2	0
12	1	1	2	0	0	8.317193191416238	0.0	4.276666119016033	360.0	0	1	0
13	1	1	0	0	0	8.011333109161256	2153.0	3.030437921392433	360.0	1	0	1
14	1	1	3	1	0	7.63373561309333	734.0	4.343294782270004	480.0	1	1	1
15	1	1	2	1	0	8.158966563645876	1390.0	4.537444175729332	360.0	1	0	1
16	1	1	2	0	1	8.10614916233133	3300.0	4.477336314475207	360.0	1	2	1
17	1	1	0	0	0	8.0106913391303	3033.0	4.333576591600541	300.0	1	2	1
18	1	0	0	0	0	7.8136103320335903	0.0	4.077337443390372	360.0	1	2	1

SPRINT 2:

► MODEL BUILDING:

Model Building

Setting up methods for data collection, understanding and paying attention to what is significant in the data to address the questions you are posing, and finding a simulation, statistical, or mathematical model to gain understanding and make predictions are all part of the model-building process.

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MaxAbsScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.model_selection import cross_val_score
from sklearn.metrics import f1_score
import pickle
```

```
scaler = MaxAbsScaler()
```

```
train = pd.read_csv('train.csv')
```

```
test = pd.read_csv('test.csv')
```

```
train.head()
```

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area
0	1	1	0	0	0	8.699515	2250.0	5.579730	360.0	1	1
1	1	1	0	0	0	7.992269	2900.0	4.575197	360.0	1	1
2	1	1	2	0	0	8.740337	1695.0	5.347105	360.0	1	1
3	1	1	0	0	0	7.641564	3150.0	4.532030	360.0	1	1
4	1	0	0	0	0	8.334712	0.0	4.584967	360.0	0	1

Pickle is primarily used in Python to serialize and deserialize Python object structures. To put it another way, it is the procedure of converting a Python object into a byte stream so that it can be stored in a file or database, have its state preserved across sessions, or be used to transfer data over a network.

► Testing & Training

The train/test approach is a way to gauge how accurate your model is.

Because the data set is divided into two sets—a training set and a testing set—this technique is known as train/test.

SPRINT 3:

- **Web UI**

- ▶ **HTML**

The most fundamental component of the Web is HTML (HyperText Markup Language). It describes the purpose and organization of web content. Links that join online pages together, either inside a single website or between websites, are referred to as "hypertext." An essential component of the Web are links. You can participate actively in the World Wide Web by publishing content online and linking it to other people's web pages.

- ▶ **CSS**

A stylesheet language called Cascading Style Sheets (CSS) is used to specify how a document written in HTML or XML is presented (including XML dialects such as SVG, MathML or XHTML). CSS specifies how items should be shown in various media, including speech, paper, screens, and other media. According to W3C guidelines, CSS is one of the basic languages of the open web and is standardized across all Web browsers.

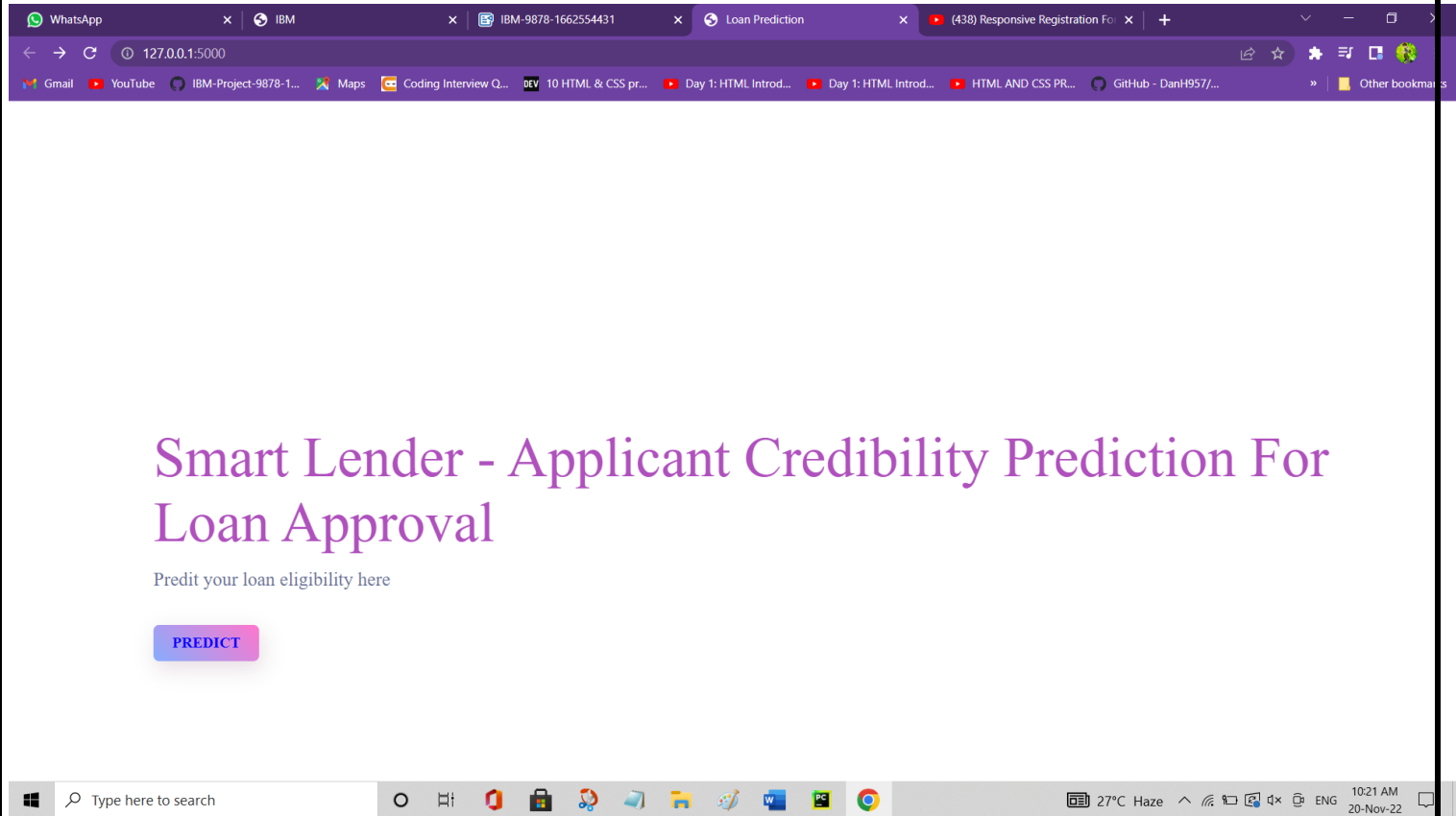
- ▶ **JS**

Although it is most famous for being the scripting language for Web pages, JavaScript (commonly abbreviated to JS) is a lightweight, interpreted, object-oriented language with first-class functions that is also utilized in other non-browser applications. It is a dynamic, prototype-based multi- paradigm scripting language that supports imperative, functional, and object-oriented programming paradigms.

JavaScript, which operates on the client side of the web, may be used to plan or programme how web pages should act in response to an event. JavaScript is a popular scripting language for managing the behavior of web pages because it is simple to learn and extremely effective.

We have integrated HTML, CSS & JS to create a website UI

I. HOME PAGE:



II. PREDICT PAGE:

LOAN ELIGIBILITY PREDICTION

Fill the form for prediction

Back

Name

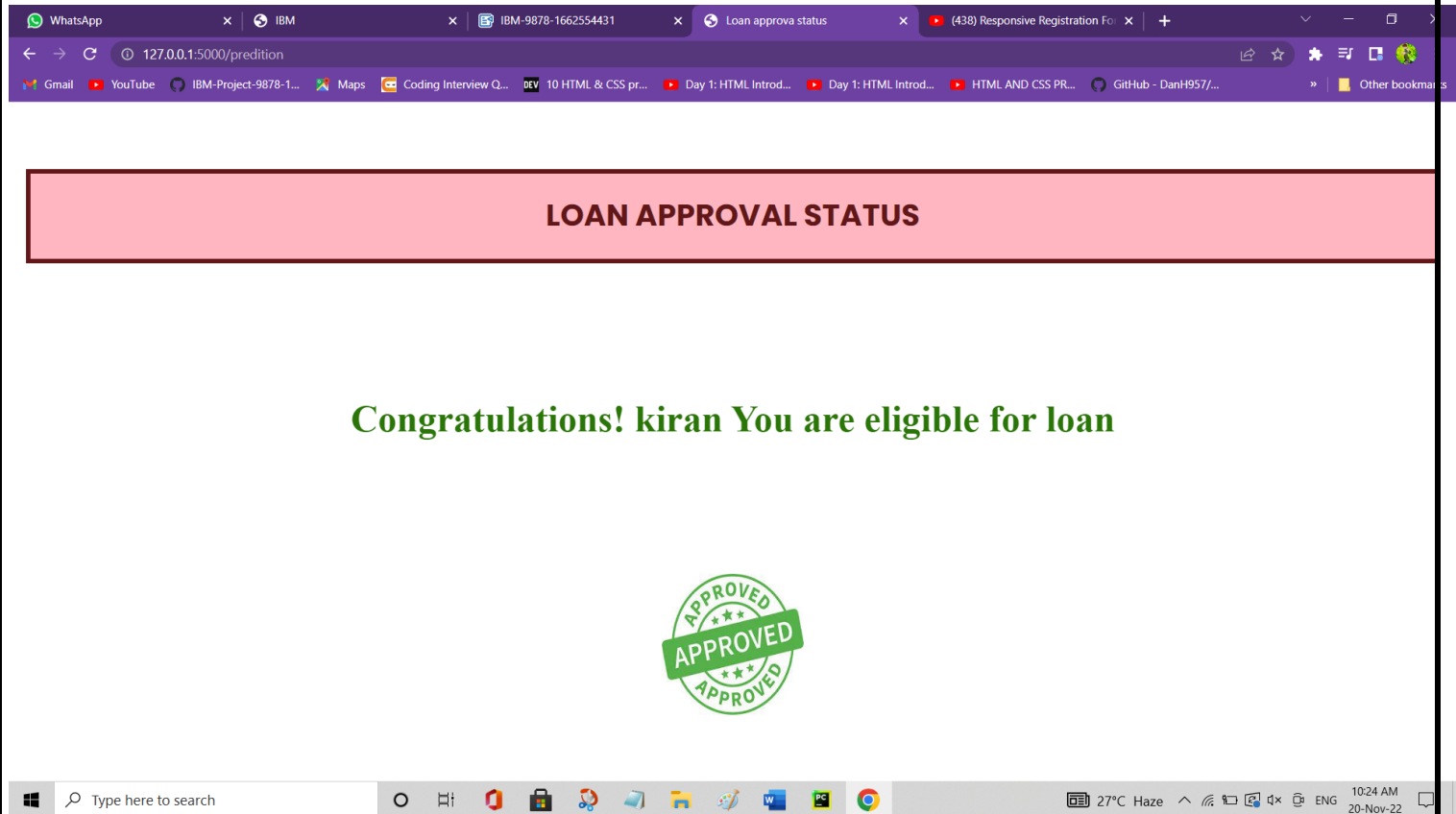
Email ID

Mobile Number

Type here to search

27°C Haze 10:22 AM 20-Nov-22

III. SUBMIT PAGE:



▪ Web Framework

Web application developers can write applications without having to be concerned about low-level details like protocol, thread management, and other issues thanks to a web framework, also known as a web application framework.

▪ Flask

Python is used to create the Flask web application framework. It was created by Armin Ronacher, who served as the team leader of Pocco, a worldwide group of Python aficionados. The Werkzeug WSGI toolkit and the Jinja2 template engine serve as the foundation for Flask. They're both Pocco projects. Its core is compact and simple to expand.

```

2  from flask import render_template, Flask, request
3  import numpy as np
4  import pickle
5
6  app= Flask(__name__, template_folder='templates')
7
8  model = pickle.load(open("rdf.pkl", 'rb'))
9  scale = pickle.load(open('scale.pkl', 'rb'))
10
11 @app.route('/')
12 def home():
13     return render_template('index.html')
14 @app.route('/predict.html')
15 def formpg():
16     return render_template('predict.html')
17 @app.route('/submit', methods = ['POST'])
18 def predict():
19     loan_num, gender, married, depend, education, self_emp, applicant_income, co_income, loan_amount, loan_term, credit_history, pro
20     if gender == 'Male':
21         gender = 1
22     else:
23         gender = 0
24
25     if married == 'Yes':
26         married = 1
27     else:
28         married = 0
29
30     if education == 'Graduate':
31         education = 0
32     else:
33         education = 1
34
35     if self_emp == 'Yes':
36         self_emp = 1
37     else:
38         self_emp = 0
39
40     if depend == '3+':
41         depend = 3
42
43     applicant_income = int(applicant_income)
44     applicant_income = np.log(applicant_income)
45     loan_amount = int(loan_amount)
46     loan_amount = np.log(loan_amount)

```

This is how the flask is used for developing web application, and There is a built-in development server and a fast debugger provided which gives a better result.

▪ Pickle

Pickle is primarily used in Python to serialize and deserialize Python object structures. To put it another way, it is the procedure of converting a Python object into a byte stream so that it can be stored in a file or database, have its state preserved across sessions, or be used to transfer data over a network.

SPRINT 4:

Cloud deployment is the process of deploying an application through one or more hosting models—software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS)—that leverage the cloud. This includes architecting, planning, implementing, and operating workloads on the cloud.

▪ Cloud web

Cloud integration connects disparate cloud-based systems into a single platform. By breaking down software silos, cloud integration platforms let you access and manage applications and data from different software systems in one place.

Streamline business operations. Improve the efficiency of data management. Reduce costs. Improve customer interactions.

```
2 from flask import render_template, Flask, request
3 import numpy as np
4 import pickle
5
6 app = Flask(__name__, template_folder='templates')
7
8 model = pickle.load(open("rdf.pkl", 'rb'))
9 scale = pickle.load(open('scale.pkl', 'rb'))
10
11 @app.route('/')
12 def home():
13     return render_template('index.html')
14 @app.route('/predict.html')
15 def formpg():
16     return render_template('predict.html')
17 @app.route('/submit', methods = ['POST'])
18 def predict():
19     loan_num, gender, married, depend, education, self_emp, applicant_income, co_income, loan_amount, loan_term, credit_history, pro
20     if gender == 'Male':
21         gender = 1
22     else:
23         gender = 0
24
25     if married == 'Yes':
26         married = 1
27     else:
28         married = 0
29
30     if education == 'Graduate':
31         education = 0
32     else:
33         education = 1
34
35     if self_emp == 'Yes':
36         self_emp = 1
37     else:
38         self_emp = 0
39
40     if depend == '3+':
41         depend = 3
42
43     applicant_income = int(applicant_income)
44     applicant_income = np.log(applicant_income)
45     loan_amount = int(loan_amount)
46     loan_amount = np.log(loan_amount)
```

■ IBM Cloud:

```

1
2 from flask import render_template, Flask, request
3 import numpy as np
4 import pickle
5 import requests
6
7 # NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
8 API_KEY = "hmIOFhnjuvRGrJaKtFnyvNKEQTINuL4eRrcnbp6K7c8R"
9 token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":API_KEY, "grant_type": 'urn:ibm
10 mltoken = token_response.json()["access_token"]
11
12 header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
13
14
15 app= Flask(__name__, template_folder='templates')
16
17 scale = pickle.load(open('scale.pkl','rb'))
18
19 @app.route('/')
20 def home():
21     return render_template('index.html')
22 @app.route('/predict.html')
23 def formpg():
24     return render_template('predict.html')
25 @app.route('/submit', methods = ['POST'])
26 def predict():
27     loan_num,gender,married,depend,education,self_emp,applicant_income,co_income,loan_amount,loan_term,credit_history,pro
28     if gender == 'Male':

```

■ Cloud integration:

```

import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
client_5158bfd5065b40c4b6cf7e02a60cf879 = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='Rob46tTNo970_Wdw9cPUe7whW_akOBfAud9qWugyZBTB',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

body = client_5158bfd5065b40c4b6cf7e02a60cf879.get_object(Bucket='ibmsmartlender-donotdelete-pr-fn1gc
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

test = pd.read_csv(body)
test.head()

```

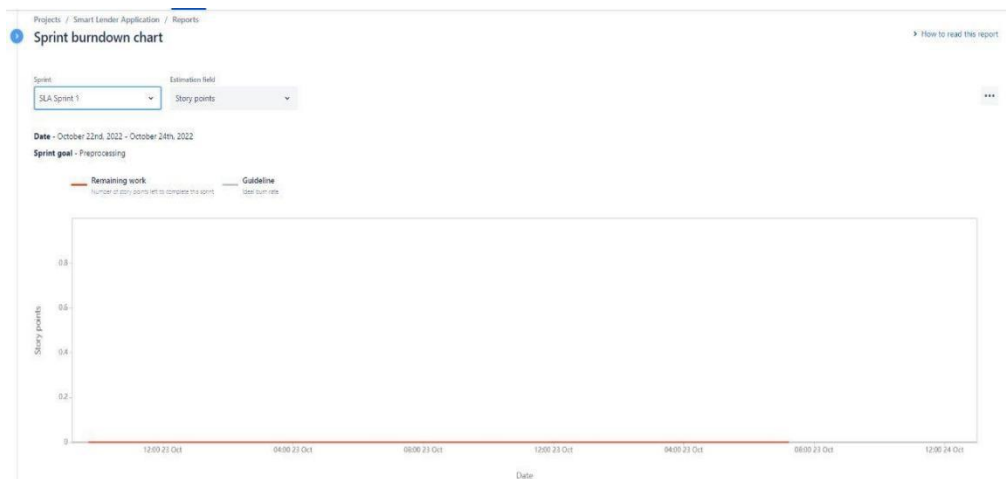
	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Cr
0	1	1	0	0	0	9.114160	0.0	5.429346	360.0	
1	1	1	0	0	0	8.368693	0.0	4.867534	360.0	
2	1	1	2	0	0	8.334952	1447.0	5.062595	360.0	
3	0	0	0	0	0	7.972466	0.0	4.262680	360.0	
4	1	0	0	0	0	7.907652	0.0	4.248495	360.0	

6.2 Sprint Delivery Schedule:

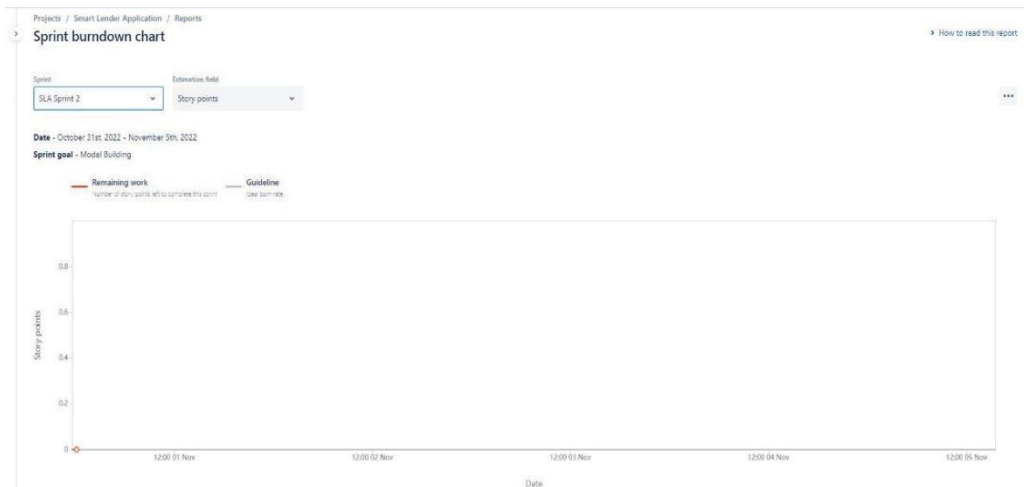
SPRINTS	CREATED	RESOLVED
SPRINT 1	October 22	October 24
SPRINT 2	October 22	November 3
SPRINT 3	October 22	November 3
SPRINT 4	October 22	November 4

6.3 Reports from JIRA:

SPRINT 1:



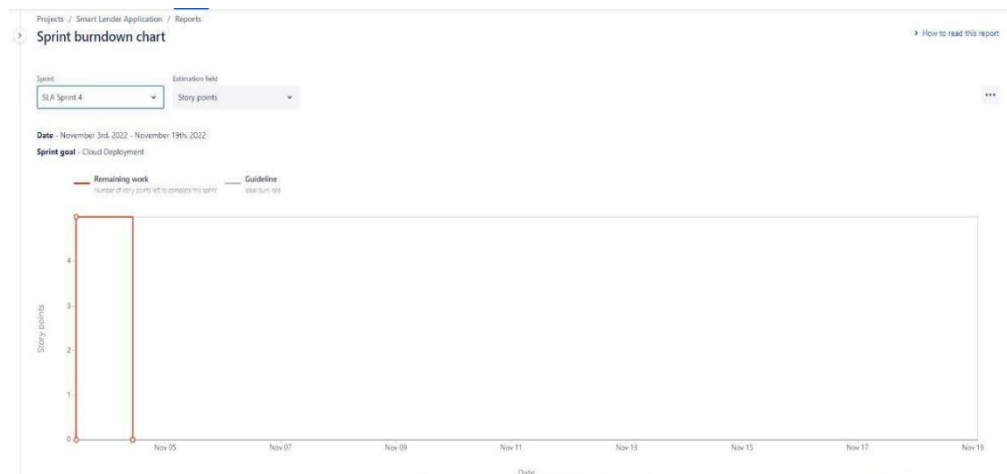
SPRINT 2:



SPRINT 3:



SPRINT 4:



ISSUES COMPLETED OUTSIDE OF SPRINT:

Completed issues

[View in issue navigator](#)

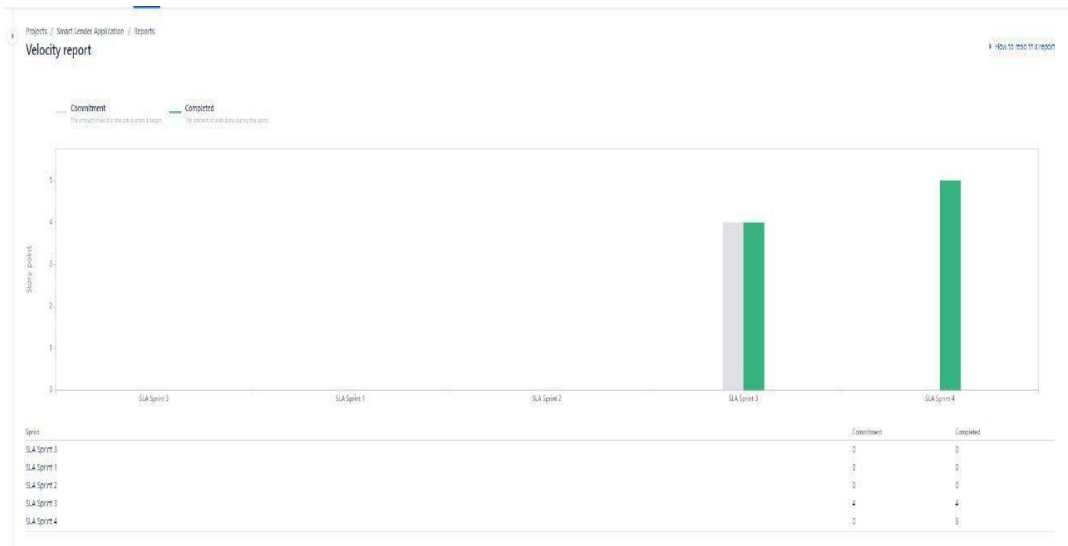
Key	Summary	Issue type	Epic	Status	Assignee	Story points
SLA-10	Complete Flask integration with the website	Story		DONE	DR	1
SLA-12	Flask Integration with website	Story		DONE	DR	1
SLA-13	Testing the work of Flask	Story		DONE	S	1
SLA-9	Trying to Deploy in cloud	Story		DONE	S	1

Issues completed outside of sprint

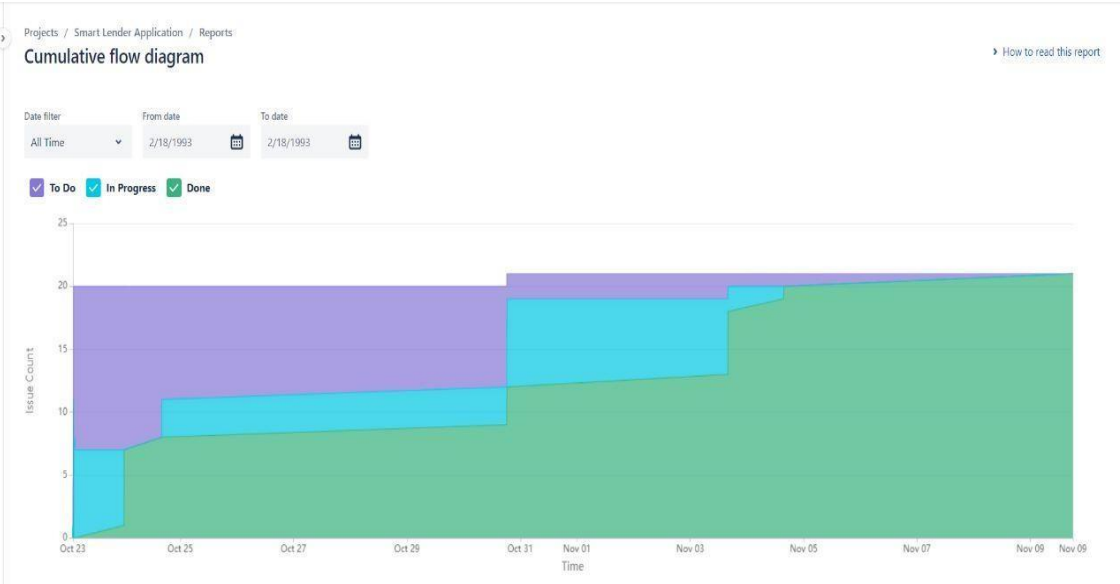
[View in issue navigator](#)

Key	Summary	Issue type	Epic	Status	Assignee	Story points
SLA-11	Learning of Flask	Story		DONE	DR	1

VELOCITY REPORT:



CUMULATIVE FLOW DIAGRAM:



COMPLETED ISSUES:

Completed issues

View in issue navigator

Key	Summary	Issue type	Epic	Status	Assignee	Story points
SLA-22	Work with ibm cloud	Story		DONE	S	5

OVERALL PREPROCESSING:



CODING & SOLUTIONING

7.1 FEATURE 1:

```
Import pandas as pd
import
numpy as np
import matplotlib.pyplot as plt
import
seaborn as sns
%matplotlib inline

df = pd.read_csv('loan_prediction.csv')
df.head(10)
df.describe()
df.isnull().any()
df.drop('Loan_ID',axis=1,inplace=True)
df.Property_Area.unique()
plt.figure(figsize=(15,7))
df['ApplicantIncome'].hist(bins=25)
plt.show()
df.boxplot(column='ApplicantIncome',figsize=(15,7))
df.boxplot(column='ApplicantIncome', by = 'Education',figsize=(15,7))
df['Property_Area'].value_counts()
```

Analysis of Categorical Values

```
df['Loan_Status'].value_counts()['Y']
pd.crosstab(df ['Credit_History'], df ['Loan_Status'], margins=True)
plt.figure(figsize=(15,7))
df['ApplicantIncome'].hist(bins=20)
plt.show()
df['ApplicantIncome'] = np.log(df['ApplicantIncome'])

plt.figure
(figsize=(15,7))
df['ApplicantIncome'].hist(bins=25)
plt.show()
```

```

def percentageConvert(ser):
    return ser/float(ser[-1])
tabs = pd.crosstab(df ["Credit_History"], df ["Loan_Status"],
    margins=True).apply(percentageConvert, axis=1)
tabs
app_loan = tabs['Y'][1]
print(f'{app_loan*100:.2f} % applicants got their loans approved')
So this is a good data set to train with

```

```

df['Self_Employed'].fillna('No',inplace=True)
#df['TotalIncome'] = df['ApplicantIncome']
+ df['CoapplicantIncome']
#df['TotalIncome_log'] =
np.log(df['TotalIncome'])

```

```

#plt.figure(figsize=(15,
))
#df['TotalIncome'].hist(b
ins=25) #plt.show()

```

```

#plt.figure(figsize=(17)
)
#df['TotalIncome_log'].h
ist(bins=25) #plt.show()

```

```

plt.figure(figsize=
(15,7))
df['LoanAmount'].h
ist(bins=20)
plt.show()

```

```
df['LoanAmount'] =  
np.log(df['LoanAmount'])  
plt.figure(figsize=(15,7))  
df['LoanAmount'].hist(bins  
=25)  
plt.show()
```

Now to Handle with null values

```
df['Gender'].fillna(df['Gender'].mode()[0],inplace=True)  
df['Married'].fillna(df['Married'].mode()[0],inplace=True)  
df['Dependents'].fillna(df['Dependents'].mode()[0],inplace=True)  
  
df['LoanAmount'].fillna(df['LoanAmount'].mean(),  
inplace=True)  
df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mean(),  
inplace=True)  
df['ApplicantIncome'].fillna(df['ApplicantIncome'].mean(),  
inplace=True)  
df['CoapplicantIncome'].fillna(df['CoapplicantIncome'].mean(),  
inplace=True) df['Gender'].fillna(df['Gender'].mode()[0],  
inplace=True) df['Married'].fillna(df['Married'].mode()[0],  
inplace=True)  
df['Dependents'].fillna(df['Dependents'].mode()[0],  
inplace=True)  
df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0],  
inplace=True)  
df['Credit_History'].fillna(df['Credit_History'].mode()[0],  
inplace=True)  
  
df.isnull().any()  
  
df.head()
```

```
cat=['Gender','Married','Dependents','Education','Self_Employed','Credit_History','Property_Area'] target = ['Loan_Status']
all_cols = ['Gender', 'Married',
'Dependents', 'Education', 'Self_Employed',
'ApplicantIncome', 'CoapplicantIncome',
'Loan_Amount_Term', 'Credit_History',
'Property_Area', 'Loan_Status', 'TotalIncome_log',
'LoanAmount_log']
```

```
from sklearn.preprocessing import LabelEncoder,OneHotEncoder
for var in cat:
    le = LabelEncoder()
    df[var]=le.fit_transform(df[var].astype('str'))
```

```
print('Done encoding Categorical Values')
for tar in target:
    oe = OneHotEncoder()
    df[tar]=le.fit_transform(df[tar].astype('str'))
print('Done encoding Target Value')
```

```
df.head(5)
```

```
from sklearn.model_selection import train_test_split
```

```
train, test = train_test_split(df,test_size=0.2,random_state=42)
test.to_csv('test.csv',encoding='utf-8',index=False)
train.to_csv('train.csv',encoding='utf-8',index=False)
```

7.1 FEATURE 2:

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MaxAbsScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import
GradientBoostingClassifierfrom sklearn.metrics import
confusion_matrix
from sklearn.metrics import classification_report
from sklearn.model_selection import
cross_val_scorefrom sklearn.metrics import
f1_score
import pickle
scaler = MaxAbsScaler()
train = pd.read_csv('train.csv')test =
pd.read_csv('test.csv')

train.head()
train_y = train.iloc[:,-1]

train_x = train.drop('Loan_Status',axis=1)
test_y = test.iloc[:,-1]
test_x = test.drop('Loan_Status',axis=1)

x = pd.concat([train_x,test_x],axis=0)y =
pd.concat([train_y,test_y],axis=0)

train_x = scaler.fit_transform(train_x)
test_x = scaler.transform(test_x)

def decisionTree(train_x,test_x,train_y,test_y):dt
= DecisionTreeClassifier() dt.fit(train_x,train_y)
y_pred = dt.predict(test_x)
print("**** Decision Tree Classifier ****")
print('Confusion Matrix')
print(confusion_matrix(test_y,y_pred))
```

```
print('Classification Report')
print(classification_report(test_y,y_pred))
```

def

```
randomForest(train_x,test_x,train_
y,test_y):rf =
RandomForestClassifier()
rf.fit(train_x,train_y)
y_pred = rf.predict(test_x)
print("**** Random Forest
Classifier ****")print('Confusion
Matrix')
print(confusion_matrix(test_y,y_p
red)) print('Classification Report')
print(classification_report(test_y,
y_pred))
```

def

```
knn(train_x,test_x,train_y,t
est_y):knn =
KNeighborsClassifier()
knn.fit(train_x,train_y)
y_pred = knn.predict(test_x)
print("**** KNeighbour
Classifier ****")
print('Confusion Matrix')
print(confusion_matrix(test_y,y
_pred)) print('Classification
Report')
print(classification_report(test_
y,y_pred))
```

def

```
xgboost(train_x,test_x,train_y
,test_y):xg =
GradientBoostingClassifier()
xg.fit(train_x,train_y)
y_pred = xg.predict(test_x)
print("**** Gradient Boosting
Classifier ****")print('Confusion
Matrix')
print(confusion_matrix(test_y,y pre
```

```
d)) print('Classification Report')
print(classification_report(test_y,y_
pred))
decisionTree(train_x,test_x,train_y,test_y)
```

def

```
randomForest(train_x,test_x,train_
y,test_y):rf =
RandomForestClassifier()
rf.fit(train_x,train_y)
y_pred = rf.predict(test_x)
print("**** Random Forest
Classifier ****")print('Confusion
Matrix')
print(confusion_matrix(test_y,y_p
red)) print('Classification Report')
print(classification_report(test_y,
y_pred))
```

def

```
knn(train_x,test_x,train_y,t
est_y):knn =
KNeighborsClassifier()
knn.fit(train_x,train_y)
y_pred = knn.predict(test_x)
print("**** KNeighbour
Classifier ****")
print('Confusion Matrix')
print(confusion_matrix(test_y,y
_pred)) print('Classification
Report')
print(classification_report(test_
y,y_pred))
```

def

```
xgboost(train_x,test_x,train_y,test
_y):xg =
GradientBoostingClassifier()
xg.fit(train_x,train_y)
y_pred = xg.predict(test_x)
print("**** Gradient Boosting Classifier
```



```
****")print('Confusion Matrix')
print(confusion_matrix(test_y,y_pred))
print('Classification Report')
print(classification_report(test_y,y_pred)
)
```

```
decisionTree(train_x,test_x,train_y,test_y)
randomForest(train_x,test_x,train_y,test_y)
knn(train_x,test_x,train_y,test_y)
xgboost(train_x,test_x,train_y,test_y)
rf =
RandomForestClassifier()
rf.fit(train_x,train_y)
ypred = rf.predict(test_x)
f1_score(ypred,test_y,average='weig
hted') cv =
cross_val_score(rf,x,y,cv=5)
```

```
np.mean(cv)
pickle.dump(rf,open('rdf.pkl','wb'))
pickle.dump(scaler,open('scale.pkl','wb'))
```

TESTING

8.1 TEST CASES

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status
Home Page_TC_001	UI	Home Page	Verify the user is able to see the button to load for the next page	1.Enter URL and click go 2.Wait for loading and watch	Null	Button for predict should display	Working as expected	Pass
Home Page_TC_002	Functional	Home Page	Verify the button redirects to another page	1.Enter URL and click go 2.Check whether the button is visible or not 3.Click on the button to verify that redirects into another page	Null	On clicking, the button should redirect to another page	Working as expected	Pass
Form Page_TC_003	UI	Form Page	Verify the user is able to see input fields to enter the details required	1.Enter URL and click go 2.Go to the form page by clicking the button in the home page 3.Wait until the form page loads 4.Check all the input fields are visible	gender:'male',married:'yes',dependents:'0',education:'graduate',self_employed:'no',applicant_income:'5849',coapplicant_income:'0',loan_amount:'128000',loan_amount_term:'360',credit_history:'1',property_area:'urban'	Input fields should be visible	Working as expected	Pass

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status
Form Page_TC_004	Functional	Form Page	Verify the user is able to enter the details without any issues in the form	1.Enter URL and click go 2.Go to the form page by clicking the button in the home page 3.Wait until the form page loads 4.Check all the input fields are visible 5.Click on the input fields and verify whether they can enter the details	gender:'male',married:'yes',dependents:'0',education:'graduate',self_employed:'no',applicant_income:'5849',coapplicant_income:'0',loan_amount:'128000',loan_amount_term:'360',credit_history:'1',property_area:'urban'	Input fields should be working fine	Working as expected	Pass
Form Page_TC_005	Functional	Form Page	Verify the user is able to click on the submit button to redirect to the other page	1.Enter URL and click go 2.Go to the form page by clicking the button in the home page 3.Wait until the form page loads 4.Check all the input fields are visible 5.Click on the input fields and verify whether they can enter the details 6.Click on the submit button to redirect to the result page	gender:'male',married:'yes',dependents:'0',education:'graduate',self_employed:'no',applicant_income:'5849',coapplicant_income:'0',loan_amount:'128000',loan_amount_term:'360',credit_history:'1',property_area:'urban'	The submit button should redirect to another page	Working as expected	Pass

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status
Result Page_TC_006	UI	Result Page	Verify the user is able to see the prediction text	1.Enter URL and click go 2.Go to the form page by clicking the button in the home page 3.Wait until the form page loads 4.Check all the input fields are visible 5.Click on the input fields and verify whether they can enter the details 6.Click on the submit button to redirect to the result page 7. Check whether the prediction text is visible	Null	The submit button should redirect to another page	Working as expected	Pass

8.2 USER ACCEPTANCE TESTING

DEFECT ANALYSIS

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
UI	10	4	2	3	20
Model	10	8	5	2	25
Integration	9	6	0	0	15
Cloud	11	7	3	0	22
Totals	40	25	10	5	82

RESULTS

9.1 PERFORMANCE METRICS

S.No.	Parameter	Values																														
1.	Metrics	<div><div>Classification Model:</div><div>Confusion Matrix – [[18 25] [2 75]], Accuracy Score – 79% & Classification Report –</div><table><tr><td></td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr><tr><td>0</td><td>0.90</td><td>0.42</td><td>0.57</td><td>43</td></tr><tr><td>1</td><td>0.76</td><td>0.97</td><td>0.85</td><td>80</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.78</td><td>123</td></tr><tr><td>macro avg</td><td>0.83</td><td>0.70</td><td>0.71</td><td>123</td></tr><tr><td>weighted avg</td><td>0.81</td><td>0.78</td><td>0.75</td><td>123</td></tr></table></div>		precision	recall	f1-score	support	0	0.90	0.42	0.57	43	1	0.76	0.97	0.85	80	accuracy			0.78	123	macro avg	0.83	0.70	0.71	123	weighted avg	0.81	0.78	0.75	123
	precision	recall	f1-score	support																												
0	0.90	0.42	0.57	43																												
1	0.76	0.97	0.85	80																												
accuracy			0.78	123																												
macro avg	0.83	0.70	0.71	123																												
weighted avg	0.81	0.78	0.75	123																												

```
In [18]: randomForest(train_x,test_x,train_y,test_y)
```

**** Random Forest Classifier ****

Confusion Matrix

```
[[18 25]
 [ 2 78]]
```

Classification Report

	precision	recall	f1-score	support
0	0.90	0.42	0.57	43
1	0.76	0.97	0.85	80
accuracy			0.78	123
macro avg	0.83	0.70	0.71	123
weighted avg	0.81	0.78	0.75	123

```
In [22]: f1_score(ypred,test_y,average='weighted')
```

```
Out[22]: 0.7977251407129455
```

```
In [23]: cv = cross_val_score(rf,x,y,cv=5)
```

```
In [24]: np.mean(cv)
```

```
Out[24]: 0.7915367186458749
```

ADVANTAGES & DISADVANTAGES

ADVANTAGES	DISADVANTAGES
<ul style="list-style-type: none">✓ Optimization of Loan Life Cycle.✓ Digital Lending technology thrives on process speed✓ Easy capture of Applicant information✓ Quicker Decision Making✓ Consistency✓ Comfort across Devices✓ Perfect for first time borrowers✓ Compliance with Rules and Regulations✓ Power of Analytics	<ul style="list-style-type: none">✓ Strict eligibility criteria✓ One of the major disadvantages of a bank loan is that banks can be cautious about lending to small businesses✓ Lengthy application process✓ Not suitable for ongoing expenses✓ Secured loans carry risk

CONCLUSION

In this report, we have proposed customer loan prediction using supervised learning techniques for loan candidates as valid or failed to pay customers. Various algorithms were implemented to predict customer loans. Optimum results were obtained using Random Forest, KNN, and SVM, Decision Tree Classifier. On comparing these four algorithms, random forest is the high accuracy. From a correct analysis of positive points and constraints on the part, it can be safely ended that the merchandise could be an extremely efficient part. This application is functioning correctly and meeting all or any Banker necessities. This part is often obstructed in several different systems. There are several cases of computer glitches and errors in content, and the most significant weight of option is mounted in a machine-driven prediction system. Therefore, the so-called software system might be created in the future with more secure, reliable, and dynamic weight adjustment. In close to future, this module of prediction can be integrated with the module of machine-driven processing systems. The analysis starts with data cleaning and processing missing values, exploratory research, and finally, model building and evaluation of the model. The best accuracy on the public test set is when we get a higher accuracy score and other performance metrics which will be found out. This model can help to predict the approval of a bank loan or not for a candidate.

FUTURE SCOPE

The future of smart lender platforms is bright; there is a massive untapped loan market in the range of 20 lakh crores. Because they are technology-based, these lending platforms have a significant potential for providing novel solutions and disruptive technology for the borrowing and lending business. Today's generation is considerably more knowledgeable and technologically aware and believes in spending money to get varied life experiences. As the habit of consumerization grows, platforms that provide instant approval and paperless loans with minimal time and personal interaction are expected to grow significantly by providing the masses with the opportunity for financial inclusion while also generating higher returns for lenders to encourage them.

APPENDIX

SOURCE CODE:

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Loan Prediction</title>
  <style>
    @import
url("https://fonts.googleapis.com/css2?family=Aref+Ruqaa+Ink:wght@700&display=swap");
    @import url("https://fonts.googleapis.com/css2?family=EB+Garamond&display=swap");
    @import url("https://fonts.googleapis.com/css2?family=Antic+Slab&display=swap");

    html {
      user-select: none;
    }

    body {
      margin-top: 5%;
      color: white;
    }

    html {
      background: linear-gradient(rgba(0, 0, 0, 0.5), rgba(0, 0, 0, 0.5)), url(static/loanapp.png);
      height: 100%;
      background-position: center;
      background-repeat: no-repeat;
      background-size: cover;
      object-fit: cover;
    }

    h1 {
      font-size: 45px;
      font-family: "Aref Ruqaa Ink", serif;
    }

    h3 {
      font-size: 20px;
      font-family: "Antic Slab", serif;
    }

    h6 {
      font-size: 20px;
      font-family: "Antic Slab", serif;
    }

  }

  /* ~~~~~ BUTTON ~~~~~ */
```

```

.container,
.container:before,
.container:after {
    box-sizing: border-box;
    padding: 0;
    margin: 0;
    font: 300 1em/1.5 "Open Sans", "Helvetica Neue", Arial, sans-serif;
    text-decoration: none;
    color:#4a4f4d;
}

.btn {
    background:#383a3b;
}

.container {
    min-width: 500px;
    margin: 5% auto;
    text-align: center;
}

button:hover {
    cursor: pointer;
}

button {
    background: transparent;
    outline: none;
    position: relative;
    border: 3px solid #FCDDB0;
    padding: 15px 50px;
    overflow: hidden;
}

/*button:before (attr data-hover)*/
button:hover:before {
    opacity: 1;
    transform: translate(0, 0);
}

button:before {
    content: attr(data-hover);
    position: absolute;
    top: 1.1em;
    left: 0;
    width: 100%;
    text-transform: uppercase;
    letter-spacing: 3px;
    font-weight: 800;
    font-size: 0.8em;
    opacity: 0;
    transform: translate(-100%, 0);
    transition: all 0.3s ease-in-out;
}

/*button div (button text before hover)*/
button:hover div {
    opacity: 0;
    transform: translate(100%, 0);
}

```

```
button div {
  text-transform: uppercase;
  letter-spacing: 3px;
  font-weight: 800;
  font-size: 0.8em;
  transition: all 0.3s ease-in-out;
}

/*--- Footer ---*/

.footer {
  margin-top: 10px;
}

.nav-link {
  font-weight: bold;
  font-size: 14px;
  text-transform: uppercase;
  text-decoration: none;
  color: #59c99c;
  padding: 20px 0px;

  display: inline-block;
  position: relative;
  opacity: 0.75;
}

#d {
  margin-top: -40px;
  font-family: "EB Garamond", serif;
  letter-spacing: 0.5px;
}

#p {
  margin-top: -50px;
  font-family: "EB Garamond", serif;
  letter-spacing: 0.5px;
}

.nav-link:hover {
  opacity: 1;
}

.nav-link::before {
  transition: 300ms;
  height: 3px;
  content: "";
  position: absolute;
  background-color: #4a4f4d;
}

.nav-link-fade-up::before {
  width: 100%;
  bottom: 5px;
  opacity: 0;
}
```

```

.nav-link-fade-up:hover::before {
  bottom: 10px;
  opacity: 1;
}

p {
  color: white;
  font-family: "Aref Ruqaa Ink", serif;
  letter-spacing: 0.5px;
}

.tooltip {
  position: relative;
  display: inline-block;
  /* If you want dots under the hoverable text */
}

/* Tooltip text */
.tooltip .tooltiptext {
  border-radius: 10px;
  visibility: hidden;
  width: 100px;
  color: #4a4f4d;
  right: 28vh;
  /* Position the tooltip text - see examples below! */
  position: absolute;
  z-index: 1;
}

/* Show the tooltip text when you mouse over the tooltip container */
.tooltip:hover .tooltiptext {
  visibility: visible;
}

.tooltip1 {
  position: relative;
  display: inline-block;
  /* If you want dots under the hoverable text */
}

/* Tooltip text */
.tooltip1 .tooltiptext1 {
  border-radius: 10px;
  visibility: hidden;
  width: 100px;
  color: #59c99c;
  text-align: center;
  left: 28vh;
  /* Position the tooltip text - see examples below! */
  position: absolute;
  z-index: 1;
}

```

```
/* Show the tooltip text when you mouse over the tooltip container */
.tooltip1:hover .tooltiptext1 {
  visibility: visible;
}

@media only screen and (max-width: 600px) {
  html {
    width: 100% !important;
  }
  body {
    margin-top: 110px;
  }
  h1 {
    font-size: 40px;
  }
  h3 {
    font-size: 15px;
  }
  .container {
    min-width: 200px;
  }
  .btn {
    margin-right: 2vh;
  }
  #d {
    letter-spacing: 0px;
    font-size: 14px;
  }
  #p {
    letter-spacing: 0px;
    font-size: 14px;
  }
  .footer {
    margin-top: 15vh;
  }
  .tooltip .tooltiptext {
    display: none;
  }
  .tooltip1 .tooltiptext1 {
    display: none;
  }
}
</style>
</head>
```

```
<body>
  <main>
    <div>

      </div>
    <center>
      <br><br><br><br><br><br>
      <h1>Loan Prediction using Random forest</h1><br>
      <h2>Get to know your applicant application will get accepted or not</h2>

      <div class="container">
        <a href="predict">
          <button style="color: #ffffff; " class="btn" data-hover="Loan Predictor"
onclick="predict">
            <div>Predict</div>
          </button>
        </a>
      </div>
    </center>
  </main>
</body>
</html>
```

PREDICT PAGE:

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8" />
```

```
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

```
<title>Loan predictor</title>
```

```
<style>
```

```
  @import
```

```
url("https://fonts.googleapis.com/css2?family=Aref+Ruqaa+Ink:wght@700&display=swap");
```

```
  @import url("https://fonts.googleapis.com/css2?family=Albert+Sans&display=swap");
```

```
  @import url("https://fonts.googleapis.com/css2?family=EB+Garamond&display=swap");
```

```
html {
```

```
  height: 100%;
```

```
}
```

```
body {
```

```
  margin: 0;
```

```
  margin-bottom: 50%;
```

```
  padding: 0;
```

```
  font-family: sans-serif;
```

```
  /* background: linear-gradient(#141e30, #243b55); */
```

```
  background-image: linear-gradient(rgba(0, 0, 0, 0.5),
```

```
    rgba(0, 0, 0, 0.5)),
```

```
    url(static/loanapp.png);
```

```
  height: 100%;
```

```
  background-position: center;
```

```
  background-repeat: no-repeat;
```

```
  background-size: cover;
```

```
  background-attachment: fixed;
```

```
  object-fit: fill;
```

```
}
```

```
.login-box {  
  position: absolute;  
  top: 100%;  
  left: 50%;  
  width: 400px;  
  padding: 40px;  
  transform: translate(-50%, -50%);  
  background: rgba(0, 0, 0, 0.5);  
  box-sizing: border-box;  
  box-shadow: 0 15px 25px rgba(0, 0, 0, 0.6);  
  border-radius: 10px;  
}
```

```
::placeholder {  
  color: aliceblue;  
}
```

```
.login-box h2 {  
  margin: 0 0 30px;  
  padding: 0;  
  color: #fff;  
  text-align: center;  
}
```

```
.fon {  
  color: #fff;  
  text-align: center;  
  font-family: "Albert Sans", sans-serif;  
}
```

```
.login-box .user-box {  
  position: relative;  
}
```

```
.login-box .user-box input {  
  width: 100%;  
  padding: 10px 0;  
  font-size: 16px;  
  color: #fff;  
  margin-bottom: 30px;
```



```
border: none;
border-bottom: 1px solid #fff;
outline: none;
background: transparent;
}
```

```
.login-box .user-box label {
  position: absolute;
  top: 0;
  left: 0;
  padding: 10px 0;
  font-size: 16px;
  color: #fff;
  pointer-events: none;
  transition: 0.5s;
}
```

```
.login-box .user-box input:focus~label,
.login-box .user-box input:valid~label {
  top: -20px;
  left: 0;
  color: #FCDDDB0;
  font-size: 12px;
}
```

```
/*--- Button */
```

```
.container,
.container:before,
.container:after {
  box-sizing: border-box;
  padding: 0;
  margin: 0;
  font: 300 1em/1.5 "Open Sans", "Helvetica Neue", Arial, sans-serif;
  text-decoration: none;
  color: #111;
}
```

```
.btn {
  background: rgba(236, 240, 241, 0.425);
}
```

```
.container {
  min-width: 500px;
  margin: 5% auto;
  text-align: center;
}

button:hover {
  cursor: pointer;
}

button {
  background: transparent;
  outline: none;
  position: relative;
  border: 3px solid #FCDDB0;
  padding: 15px 50px;
  overflow: hidden;
}

/*button:before (attr data-hover)*/
button:hover:before {
  opacity: 1;
  transform: translate(0, 0);
}

button:before {
  content: attr(data-hover);
  position: absolute;
  top: 1.1em;
  left: 0;
  width: 100%;
  text-transform: uppercase;
  letter-spacing: 3px;
  font-weight: 800;
  font-size: 0.8em;
  opacity: 0;
  transform: translate(-100%, 0);
  transition: all 0.3s ease-in-out;
}
```

```
/*button div (button text before hover)*/
```

```
button:hover div {  
  opacity: 0;  
  transform: translate(100%, 0);  
}
```

```
button div {  
  text-transform: uppercase;  
  letter-spacing: 3px;  
  font-weight: 800;  
  font-size: 0.8em;  
  transition: all 0.3s ease-in-out;  
}
```

```
/*--- Footer ---*/
```

```
.footer {  
  margin-top: 200vh;  
  margin-bottom: 10px;  
}
```

```
.nav-link {  
  font-weight: bold;  
  font-size: 14px;  
  text-transform: uppercase;  
  text-decoration: none;  
  color: #ffffff;  
  padding: 20px 0px;  
  /* margin: 0px 20px;*/  
  
  display: inline-block;  
  position: relative;  
  opacity: 0.75;  
}
```

```
#d {  
  margin-top: -40px;  
  font-family: "EB Garamond", serif;  
  letter-spacing: 0.5px;  
}
```

```
#p {
  margin-top: -50px;
  font-family: "EB Garamond", serif;
  letter-spacing: 0.5px;
}

.nav-link:hover {
  opacity: 1;
}

.nav-link::before {
  transition: 300ms;
  height: 3px;
  content: "";
  position: absolute;
  background-color: #FCDDDB0;
}

.nav-link-fade-up::before {
  width: 100%;
  bottom: 5px;
  opacity: 0;
}

.nav-link-fade-up:hover::before {
  bottom: 10px;
  opacity: 1;
}

p {
  color: white;
  font-family: "Aref Ruqaa Ink", serif;
  letter-spacing: 0.5px;
}

.tooltip {
  position: relative;
  display: inline-block;
  /* If you want dots under the hoverable text */
}
```

```
/* Tooltip text */
.tooltip .tooltiptext {
    border-radius: 10px;
    visibility: hidden;
    width: 100px;
    right: 28vh;
    /* Position the tooltip text - see examples below! */
    position: absolute;
    z-index: 1;
}

/* Show the tooltip text when you mouse over the tooltip container */
.tooltip:hover .tooltiptext {
    visibility: visible;
}

.tooltip1 {
    position: relative;
    display: inline-block;
    /* If you want dots under the hoverable text */
}

/* Tooltip text */
.tooltip1 .tooltiptext1 {
    border-radius: 10px;
    visibility: hidden;
    width: 100px;
    left: 28vh;
    /* Position the tooltip text - see examples below! */
    position: absolute;
    z-index: 1;
}

/* Show the tooltip text when you mouse over the tooltip container */
.tooltip1:hover .tooltiptext1 {
    visibility: visible;
}
```

```
@media only screen and (max-width: 600px) {  
  .login-box {  
    width: 300px;  
  }  
  
  .container {  
    min-width: 200px;  
  }  
  
  .footer {  
    position: sticky;  
    margin-top: 198vh;  
    font-size: 20px;  
  }  
  
  #d {  
    letter-spacing: 0px;  
    font-size: 14px;  
  }  
  
  #p {  
    letter-spacing: 0px;  
    font-size: 14px;  
  }  
  
  .fon {  
    font-size: 15px;  
  }  
  
  .tooltip .tooltiptext {  
    display: none;  
  }  
  
  .tooltip1 .tooltiptext1 {  
    display: none;  
  }  
}  
</style>  
</head>
```

```

<body>
  <div class="login-box">
    <h2 style="text-transform: uppercase; font-family: 'Aref Ruqaa Ink', serif">
      Loan Prediction using Random forest - <br />
      <span style="font-size: 14px; color: azure">Know your Loan eligibility</span>
    </h2>
    <p class="fon" style="font-size: 14px">
      Let's begin by entering your deatils below
    </p>
    <br />
    <form action="/submit" method="post">
      <div class="user-box">
        <input type="text" name="name" required="" onfocus="this.placeholder='Enter your
name'"
          onblur="this.placeholder=''" />
        <label>Name</label>
      </div>

      <div class="user-box">
        <input list="gender" type="data-list" name="Gender" required=""
onchange="resetIfInvalid(this);"
          onfocus="this.placeholder='Enter your Gender'" onblur="this.placeholder=''" />
        <label>Gender</label>
        <datalist id="gender" name="gender">
          <option value="Male"></option>
          <option value="female"></option>
        </datalist>
      </div>

      <div class="user-box">
        <input list="married" type="text" name="Married" required=""
onchange="resetIfInvalid(this);"
          onfocus="this.placeholder='Enter your Marital Status'" onblur="this.placeholder=''" />
        <label>Married</label>
        <datalist id="married" name="married">
          <option value="yes"></option>
          <option value="no"></option>
        </datalist>
      </div>
    </form>
  </div>

```

```
<div class="user-box">
  <input list="dep" type="text" name="Dependents" required=""
onchange="resetIfInvalid(this);"
  onfocus="this.placeholder='Enter your Dependents'" onblur="this.placeholder="" />
  <label>Dependents</label>
  <datalist id="dep" name="dep">
    <option value="0"></option>
    <option value="1"></option>
    <option value="2"></option>
    <option value="3+"></option>
  </datalist>
</div>

<div class="user-box">
  <input list="edu" type="text" name="Education" required=""
onchange="resetIfInvalid(this);"
  onfocus="this.placeholder='Enter your Educational Qualification'"
onblur="this.placeholder="" />
  <label>Education</label>
  <datalist name="edu" id="edu">
    <option value="Graduate"></option>
    <option value="Non-Graduate"></option>
  </datalist>
</div>

<div class="user-box">
  <input list="emp" type="text" name="Self_Employes" required=""
onchange="resetIfInvalid(this);"
  onfocus="this.placeholder='Are you self employed?'" onblur="this.placeholder="" />
  <label>Self Employed</label>
  <datalist name="emp" id="emp">
    <option value="yes"></option>
    <option value="no"></option>
  </datalist>
  <div class="user-box">
    <input list="credit" type="text" name="Credit_History" required=""
onchange="resetIfInvalid(this);"
    onfocus="this.placeholder='Enter your Credit History'" onblur="this.placeholder="" />
    <label>Credit History</label>
    <datalist name="credit" id="credit">
      <option value="yes"></option>
      <option value="no"></option>
    </datalist>
  </div>
</div>

<div class="user-box">
```



```

        <input list="prop" type="text" name="Property_Area" required=""
onchange="resetIfInvalid(this);"
        onfocus="this.placeholder='Enter your area of the property'"
onblur="this.placeholder=''" />
        <label>Property Area</label>
        <datalist name="prop" id="prop">
            <option value="Urban"></option>
            <option value="Rural"></option>
            <option value="Semi-Rural"></option>
        </datalist>
    </div>

</div>
<div class="user-box">
    <input type="number" name="ApplicantIncome" required=""
        onfocus="this.placeholder='Enter your Income in Dollars'" onblur="this.placeholder=''"
/>
    <label>Applicant Income</label>
</div>
<div class="user-box">
    <input type="number" name="Co_Applicant_Income" required=""
        onfocus="this.placeholder='Enter your CO Applicant Income in Dollars'"
        onblur="this.placeholder=''" />
    <label>CO Applicant Income</label>
</div>
<div class="user-box">
    <input type="number" name="LoanAmount" required=""
        onfocus="this.placeholder='Enter your Loan Amount in Dollars'"
onblur="this.placeholder=''" />
    <label>Loan Amount</label>
</div>
<div class="user-box">
    <input list="term" type="text" name="Loan_Amount_Term" required=""
onchange="resetIfInvalid(this);"
        onfocus="this.placeholder='Enter the loan amount term'" onblur="this.placeholder=''"
/>

```

```

    <label>Loan Amount Term</label>
    <datalist name="term" id="term">
      <option value="480"></option>
      <option value="360"></option>
      <option value="300"></option>
      <option value="240"></option>
      <option value="180"></option>
      <option value="120"></option>
      <option value="84"></option>
      <option value="60"></option>
      <option value="36"></option>
      <option value="12"></option>
    </datalist>
  </div>

  <div class="container">
    <a href="submit.html">
      <button style="color: #ffffff ;" class="btn" data-hover="PREDICT"
onclick="submit.html">
        <div>SUBMIT</div>
      </button>
    </a>
  </div>
</form>
</div>

</body>
<script>
  function resetIfInvalid(el) {
    //just for beeing sure that nothing is done if no value selected
    if (el.value == "") return;
    var options = el.list.options;
    for (var i = 0; i < options.length; i++) {
      if (el.value == options[i].value)
        //option matches: work is done
        return;
    }
    //no match was found: reset the value
    el.value = "";
  }
</script>

</html>

```

SUBMIT PAGE:

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
  <meta charset="UTF-8">
```

```
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
  <title>Loan Prediction</title>
```

```
  <style>
```

```
    @import
```

```
url('https://fonts.googleapis.com/css2?family=Aref+Ruqaa+Ink:wght@700&display=swap');
```

```
    @import url('https://fonts.googleapis.com/css2?family=EB+Garamond&display=swap');
```

```
  body {
```

```
    color: white;
```

```
    font-family: 'Aref Ruqaa Ink', serif;
```

```
    background-image: linear-gradient(rgba(0, 0, 0, 0.5),
```

```
      rgba(0, 0, 0, 0.5)),
```

```
    url(static/loanapp.png);
```

```
    height: 10%;
```

```
    background-position: center;
```

```
    background-repeat: no-repeat;
```

```
    background-size: cover;
```

```
    background-attachment: fixed;
```

```
    object-fit: fill;
```

```
  }
```

```
  .output {
```

```
    margin-top: 15%;
```

```
  }
```

```
/*--- Footer ---*/
```

```
.footer {
```

```
  margin-top: 21vh;
```

```
}
```

```
.nav-link {
  font-weight: bold;
  font-size: 14px;
  text-transform: uppercase;
  text-decoration: none;
  color: #ffffff;
  padding: 20px 0px;
  /* margin: 0px 20px;*/

  display: inline-block;
  position: relative;
  opacity: 0.75;
}

#d {
  margin-top: -40px;
  font-family: 'EB Garamond', serif;
  letter-spacing: 0.5px;
}

#p {
  /* margin-top: -50px;*/
  font-family: 'EB Garamond', serif;
  letter-spacing: 0.5px;
}

.nav-link:hover {
  opacity: 1;
}

.nav-link::before {
  transition: 300ms;
  height: 3px;
  content: "";
  position: absolute;
  background-color: #FCDDDB;
}

.nav-link-fade-up::before {
  width: 100%;
  bottom: 5px;
  opacity: 0;
}
```

```
.nav-link-fade-up:hover::before {
  bottom: 10px;
  opacity: 1;
}

p {
  color: white;
  font-family: 'Aref Ruqaa Ink', serif;
  letter-spacing: 0.5px;
}

.tooltip {
  position: relative;
  display: inline-block;
  /* If you want dots under the hoverable text */
}

/* Tooltip text */
.tooltip .tooltiptext {
  border-radius: 10px;
  visibility: hidden;
  width: 100px;
  right: 28vh;
  /* Position the tooltip text - see examples below! */
  position: absolute;
  z-index: 1;
}

/* Show the tooltip text when you mouse over the tooltip container */
.tooltip:hover .tooltiptext {
  visibility: visible;
}

.tooltip1 {
  position: relative;
  display: inline-block;
  /* If you want dots under the hoverable text */
}
```

```
/* Tooltip text */
.tooltip1 .tooltiptext1 {
  border-radius: 10px;
  visibility: hidden;
  width: 100px;
  top: 3vh;
  left: 28vh;
  /* Position the tooltip text - see examples below! */
  position: absolute;
  z-index: 1;
}

/* Show the tooltip text when you mouse over the tooltip container */
.tooltip1:hover .tooltiptext1 {
  visibility: visible;
}

@media only screen and (max-width: 600px) {

  body {
    margin-top: 30vh;
  }

  .footer {
    margin-top: 30vh;
  }

  .tooltip .tooltiptext {
    display: none;
  }

  .tooltip1 .tooltiptext1 {
    display: none;
  }
}
</style>
</head>
```

```
<body>
  <main class="output">
    <center>
<!--      <h1>SMART LENDER</h1>-->
<!--      <h3>{% if prediction == 0 % }-->
<!--      -->

<!--  {% endif % }-->
<!--      {% if prediction == 1 % }-->
<!--      -->
<!--  {% endif % }-->

<!--      </h3>-->

      <h1>LOAN APPROVAL STATUS</h1>
      <h2>{{ prediction_text }}</h2>

    </center>
  </main>
</body>

</html>
```

IBM APP:

```
from flask import render_template, Flask, request
import numpy as np
import pickle
import requests
```

```
app = Flask(__name__, template_folder='templates')
```

```
model = pickle.load(open("rdf.pkl", 'rb'))
scale = pickle.load(open('scale.pkl', 'rb'))
```

```
@app.route('/')
def home():
    return render_template('home.html')
```

```
@app.route('/home.html')
def home1():
    return render_template('home.html')
```

```
@app.route('/predict')
def formpg():
    return render_template('predict.html')
```

```
@app.route('/submit', methods=['POST', 'GET'])
def predict():
    if request.method == 'POST':
        gender = request.form['Gender']
        married = request.form['Married']
        depend = request.form['Dependents']
        education = request.form['Education']
        self_emp = request.form['Self_Employes']
        credit_history = request.form['Credit_History']
        property_area = request.form['Property_Area']
        applicant_income = float(request.form['ApplicantIncome'])
        coapplicantIncome = float(request.form['Co_Applicant_Income'])
        loan_amount = float(request.form['LoanAmount'])
        loan_term = float(request.form['Loan_Amount_Term'])
```

```
    if gender == 'Male':
```

```
        gender = 1
```



```
else:
    gender = 0

if married == 'Yes':
    married = 1
else:
    married = 0

if education == 'Graduate':
    education = 0
else:
    education = 1

if self_emp == 'Yes':
    self_emp = 1
else:
    self_emp = 0

if depend == '3+':
    depend = 3

if credit_history == 'Yes':
    credit_history = 1
else:
    credit_history = 0

if property_area == 'Urban':
    property_area = 2

elif property_area == 'Rural':
    property_area = 0
else:
    property_area = 1

features = [gender, married, depend, education, self_emp, applicant_income, coapplicantIncome,
loan_amount,
            loan_term, credit_history, property_area]

con_features = [np.array(features)]

scale_features = scale.fit_transform(con_features)
prediction = model.predict(scale_features)
print(prediction)
```

```
# prediction = model.predict(scale_features)
# if prediction == 1:
#     return render_template('submit.html', prediction=1 )
# else:
#     return render_template('submit.html', prediction=0)
if prediction == 1:
    return render_template('approve.html',
                           prediction_text='Congratulations! ' + ' You are eligible for loan')
else:
    return render_template('reject.html', prediction_text='Sorry ' +' You are not eligible for loan')

if __name__ == "__main__":
    app.run(debug=True)
```