# Predicting Personal Loan Approval Using Machine Learning

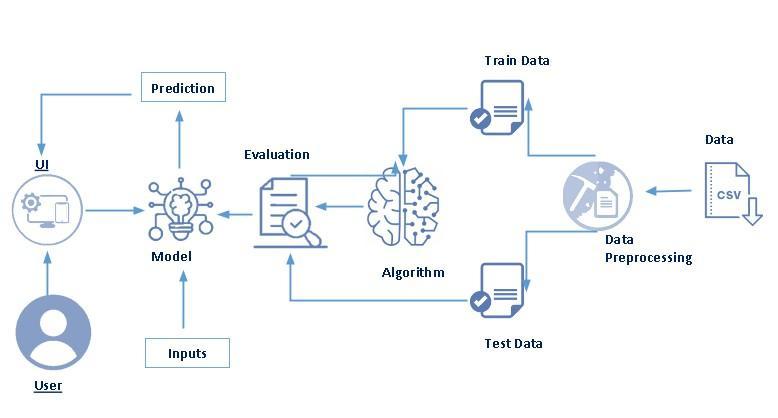
**Project Description:**

A loan is a sum of money that is borrowed and repaid over a period of time, typically with interest. There are various types of loans available to individuals and businesses, such as personal loans, mortgages, auto loans, student loans, business loans and many more.They are offered by banks, credit unions, and other financial institutions, and the terms of the loan, such as interest rate, repayment period, and fees, vary depending on the lender and the type of loan.

A personal loan is a type of unsecured loan that can be used for a variety of expenses such as home repairs, medical expenses, debt consolidation, and more. The loan amount, interest rate, and repayment period vary depending on the lender and the borrower's creditworthiness.To qualify for a personal loan, borrowers typically need to provide proof of income and have a good credit score.

Predicting personal loan approval using machine learning analyses a borrower's financial data and credit history to determine the likelihood of loan approval. This can help financial institutions to make more informed decisions about which loan applications to approve and which to deny.

**Technical Architecture:**



# Project Flow:

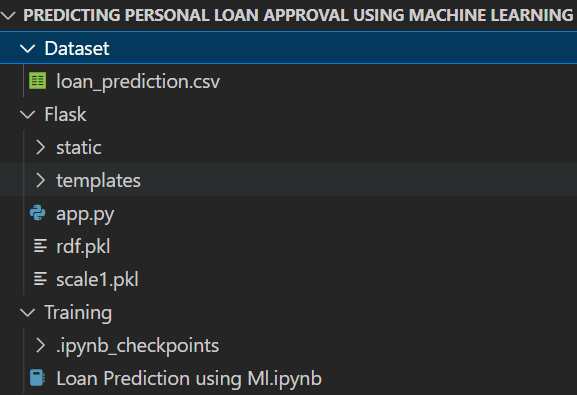
* User interacts with the UI to enter the input.
* Entered input is analysed by the model which is integrated.
* Once model analyses the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

* Define Problem / Problem Understanding
  + Specify the business problem
  + Business requirements
  + Literature Survey
  + Social or Business Impact.
* Data Collection & Preparation
  + Collect the dataset
  + Data Preparation
* Exploratory Data Analysis
  + Descriptive statistical
  + Visual Analysis
* Model Building
  + Training the model in multiple algorithms
  + Testing the model
* Performance Testing & Hyperparameter Tuning
  + Testing model with multiple evaluation metrics
  + Comparing model accuracy before & after applying hyperparameter tuning
* Model Deployment
  + Save the best model
  + Integrate with Web Framework
* Project Demonstration & Documentation
  + Record explanation Video for project end to end solution
  + Project Documentation-Step by step project development procedure

# Project Structure:

Create the Project folder which contains files as shown below



* We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.
* rdf.pkl is our saved model. Further we will use this model for flask integration.
* Training folder contains a model training file.

### **Define Problem / Problem Understanding:**

In this milestone, you will see the problem understanding.

### **Specify The Business Problem**

A loan is a sum of money that is borrowed and repaid over a period of time, typically with interest. There are various types of loans available to individuals and businesses, such as personal loans, mortgages, auto loans, student loans, business loans and many more.They are offered by banks, credit unions, and other financial institutions, and the terms of the loan, such as interest rate, repayment period, and fees, vary depending on the lender and the type of loan.

A personal loan is a type of unsecured loan that can be used for a variety of expenses such as home repairs, medical expenses, debt consolidation, and more. The loan amount, interest rate, and repayment period vary depending on the lender and the borrower's creditworthiness.To qualify for a personal loan, borrowers typically need to provide proof of income and have a good credit score.

Predicting personal loan approval using machine learning analyses a borrower's financial data and credit history to determine the likelihood of loan approval. This can help financial institutions to make more informed decisions about which loan applications to approve and which to deny.

**Business Requirements**

The business requirements for a machine learning model to predict personal loan approval include the ability to accurately predict loan approval based on applicant information, Minimise the number of false positives (approved loans that default) and false negatives (rejected loans that would have been successful).Provide an explanation for the model’s decision, to comply with regulations and improve transparency.

### **Literature Survey**

As the data is increasing daily due to digitization in the banking sector, people want to apply for loans through the internet. Machine Learning (ML), as a typical method for information investigation, has gotten more consideration increasingly. Individuals of various businesses are utilising ML calculations to take care of the issues dependent on their industry information. Banks are facing a significant problem in the approval of the loan. Daily there are so many applications that are challenging to manage by the bank employees, and also the chances of some mistakes are high.Most banks earn profit from the loan, but it is risky to choose deserving customers from the number of applications.There are various algorithms that have been used with varying levels of success. Logistic regression, decision tree, random forest, and neural networks have all been used and have been able to accurately predict loan defaults. Commonly used features in these studies include credit score, income, and employment history, sometimes also other features like age, occupation, and education level.

### **Social Or Business Impact.**

* Social Impact:- Personal loans can stimulate economic growth by providing individuals with the funds they need to make major purchases, start businesses, or invest in their education.
* Business Model/Impact:- Personal loan providers may charge fees for services such as loan origination, processing, and late payments.Advertising the brand awareness and marketing to reach out to potential borrowers to generate revenue.

### **Data Collection & Preparation**

ML depends heavily on data. It is the most crucial aspect that makes algorithm training possible. So this section allows you to download the required dataset.

### **Collect The Dataset**

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

In this project we have used .csv data. This data is downloaded from kaggle.com. Please refer to the link given below to download the dataset.

Link: <https://www.kaggle.com/datasets/altruistdelhite04/loan-prediction-problem-dataset>

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualisation techniques and some analysing techniques.

**Note:** There are a number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

### Importing The Libraries

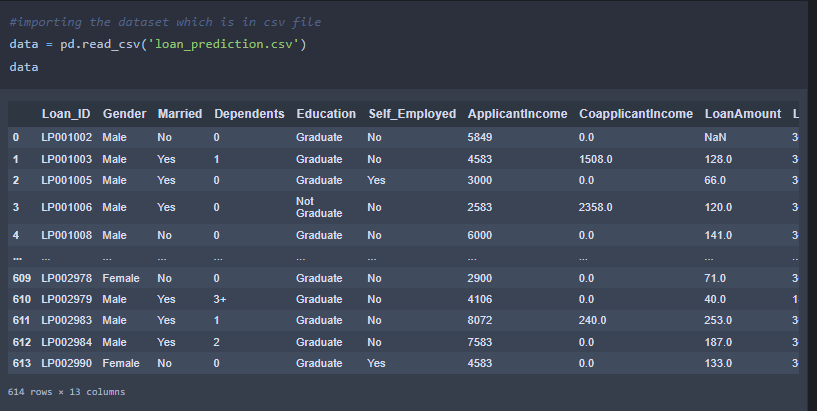
Import the necessary libraries as shown in the image. (optional) Here we have used visualisation style as fivethirtyeight.



**Read The Dataset**

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas.

In pandas we have a function called read\_csv() to read the dataset. As a parameter we have to give the directory of the csv file.



### **Data Preparation**

As we have understood how the data is, let's pre-process the collected data.

The download data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

* Handling missing values
* Handling categorical data
* Handling Imbalance Data

**Note**: These are the general steps of pre-processing the data before using it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

### **Handling Missing Values**

* Let’s find the shape of our dataset first. To find the shape of our data, the df.shape method is used. To find the data type, df.info() function is used.
* For checking the null values, df.isnull() function is used. To sum those null values we use .sum() function. From the below image we found that there are no null values present in our dataset. So we can skip handling the missing values step.
* From the above code of analysis, we can infer that columns such as gender ,married,dependents,self employed ,loan amount, loan amount term and credit history are having the missing values, we need to treat them in a required way.
* We will fill in the missing values in the numeric data type using the mean value of that particular column and categorical data type using the most repeated value.

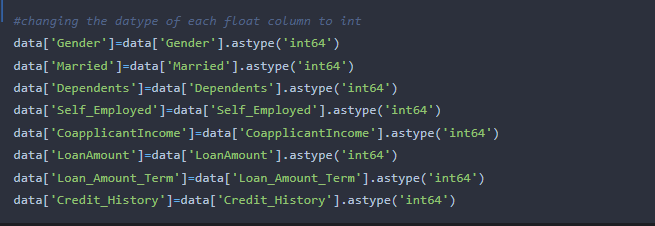
### **Handling Categorical Values**

As we can see our dataset has categorical data we must convert the categorical data to integer encoding or binary encoding.

To convert the categorical features into numerical features we use encoding techniques.

There are several techniques but in our project we are using manual encoding with the help of list comprehension.

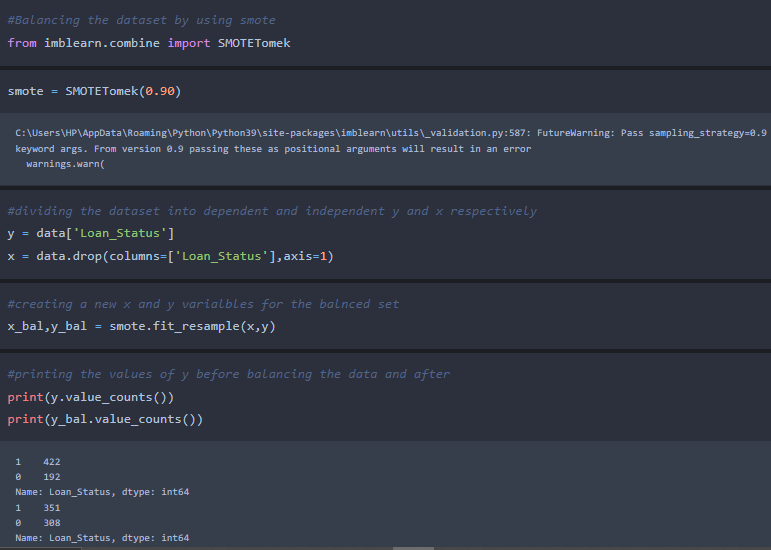
In our project, Gender ,married,dependents,self-employed,co-applicants income,loan amount ,loan amount term, credit history With list comprehension encoding is done.



### **Handling Imbalance Data**

Data Balancing is one of the most important step, which need to be performed for classification models, because when we train our model on imbalanced dataset ,we will get biassed results, which means our model is able to predict only one class element

For Balancing the data we are using the SMOTE Method.  
SMOTE: Synthetic minority over sampling technique, which will create new synthetic data points for under class as per the requirements given by us using KNN method.



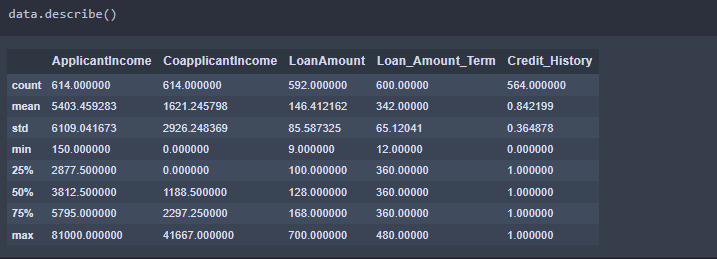
From the above picture, we can infer that ,previously our dataset had 492 class 1, and 192 class  items, after applying smote technique on the dataset the size has been changed for minority class.

**Exploratory Data Analysis**

In this milestone, we will see the exploratory data analysis

### **Descriptive Statistical**

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.



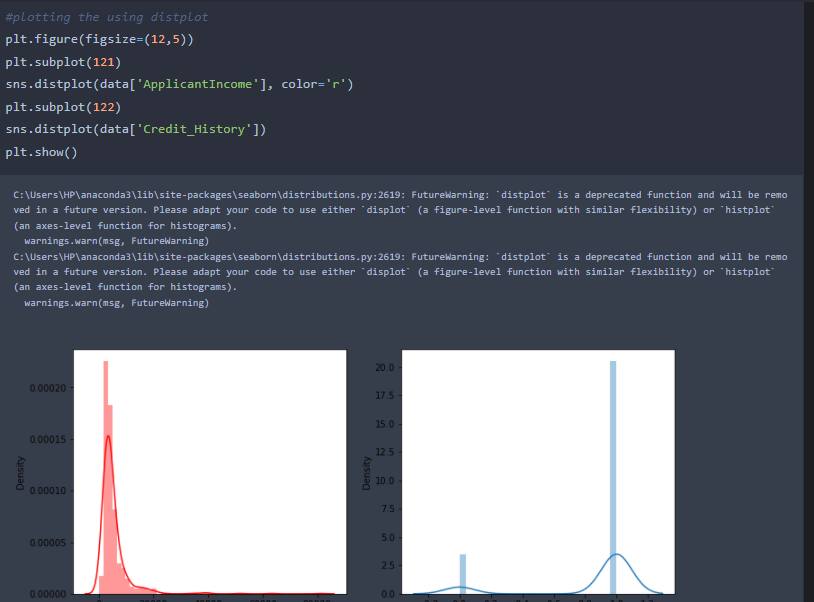
### **Visual Analysis**

Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions.

### **Univariate Analysis**

In simple words, univariate analysis is understanding the data with a single feature. Here we have displayed two different graphs such as distplot and countplot.

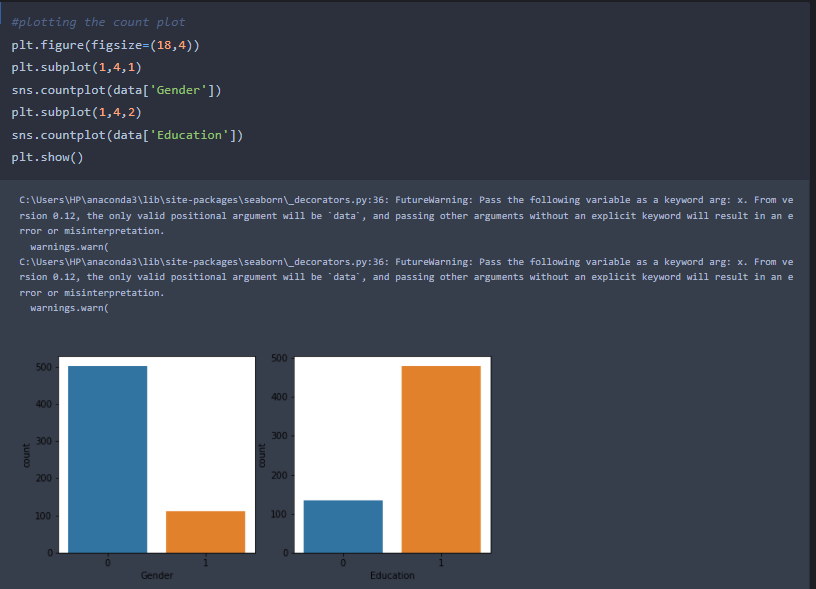
* The Seaborn package provides a wonderful function distplot. With the help of distplot, we can find the distribution of the feature. To make multiple graphs in a single plot, we use subplot.



* In our dataset we have some categorical features. With the count plot function, we are going to count the unique category in those features. We have created a dummy data frame with categorical features. With for loop and subplot we have plotted this below graph.
* From the plot we came to know, Applicants income is skewed towards left side, where as credit history is categorical with 1.0 and 0.0

**Countplot:-**A count plot can be thought of as a histogram across a categorical, instead of quantitative, variable. The basic API and options are identical to those for barplot() , so you can compare counts across nested variables.From the graph we can infer that , gender and education is a categorical variables with 2 categories , from gender column we can infer that 0-category is having more weightage than category-1,while education with 0,it means no education is a underclass when compared with category -1, which means educated .

### **Bivariate Analysis**





From the above graph we can infer the analysis such as

* Segmenting the gender column and married column based on bar graphs
* Segmenting the Education and Self-employed based on bar graphs ,for drawing insights such as educated people are employed.
* Loan amount term based on the property area of a person holding.

### **Multivariate Analysis**

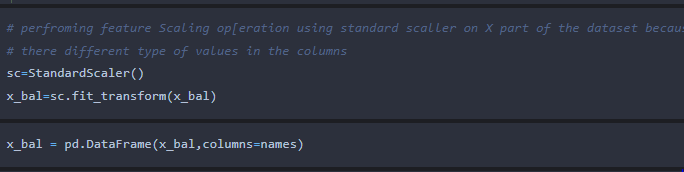
In simple words, multivariate analysis is to find the relation between multiple features. Here we have used a swarm plot from the seaborn package.



From the above graph we are plotting the relationship between the Gender, applicants income and loan status of the person.

Now, the code would be normalising the data by scaling it to have a similar range of values, and then splitting that data into a training set and a test set for training the model and testing its performance, respectively.

**Scaling the Data**  
Scaling is one the important process, we have to perform on the dataset, because of data measures in different ranges can leads to mislead in prediction.Models such as KNN, Logistic regression need scaled data, as they follow distance based method and Gradient Descent concept.

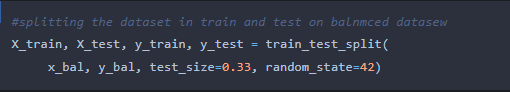


We will perform scaling only on the input values.Once the dataset is scaled, it will be converted into an array and we need to convert it back to a dataframe.

**Splitting data into train and test**Now let’s split the Dataset into train and test sets

Changes: first split the dataset into x and y and then split the data set.

Here x and y variables are created. On x variable, df is passed with dropping the target variable. And on y target variable is passed. For splitting training and testing data we are using the train\_test\_split() function from sklearn. As parameters, we are passing x, y, test\_size, random\_state.

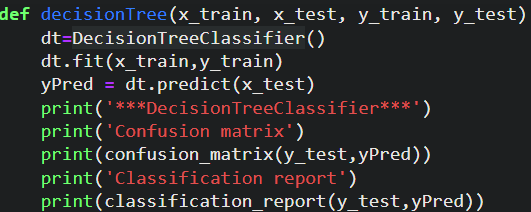


### **Training The Model In Multiple Algorithms**

Now our data is cleaned and it’s time to build the model. We can train our data on different algorithms. For this project we are applying four  classification algorithms. The best model is saved based on its performance.

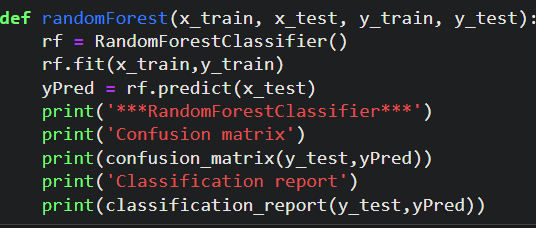
### **Decision Tree Model**

A function named decisionTree is created and train and test data are passed as the parameters. Inside the function, DecisionTreeClassifier algorithm is initialised and training data is passed to the model with the .fit() function. Test data is predicted with .predict() function and saved in a new variable. For evaluating the model, a confusion matrix and classification report is done.



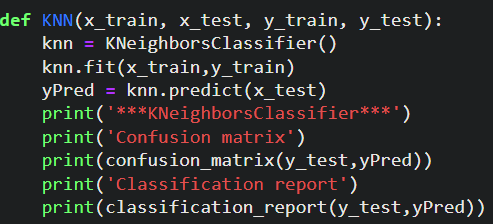
### **Random Forest Model**

A function named randomForest is created and train and test data are passed as the parameters. Inside the function, RandomForestClassifier algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in a new variable. For evaluating the model, a confusion matrix and classification report is done.



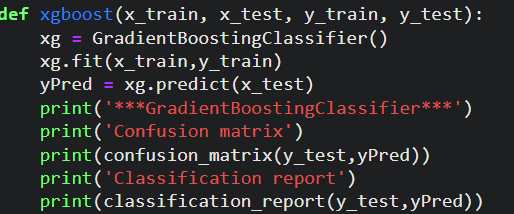
### **KNN Model**

A function named KNN is created and train and test data are passed as the parameters. Inside the function, KNeighborsClassifier algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done.



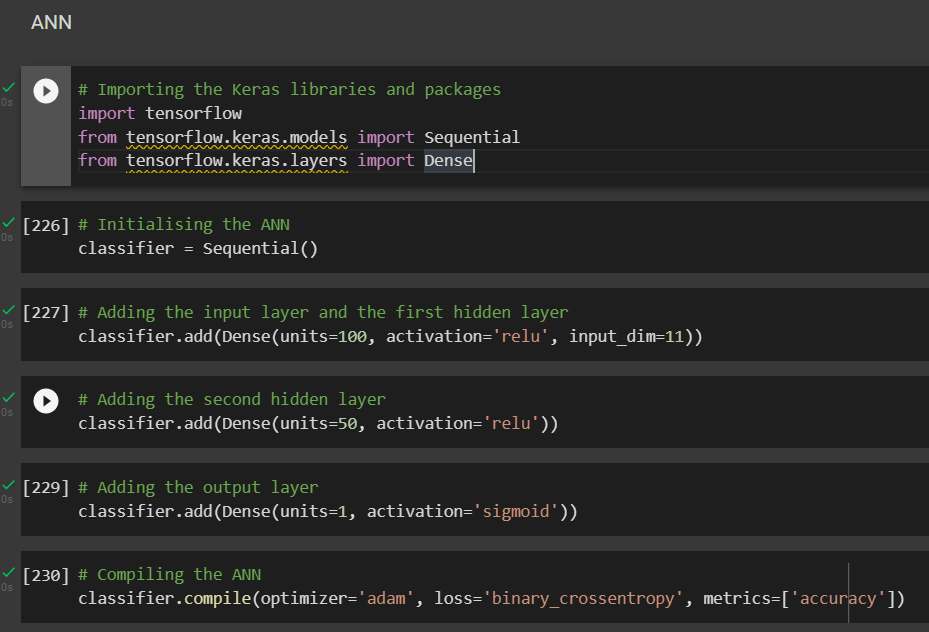
### **Xgboost Model**

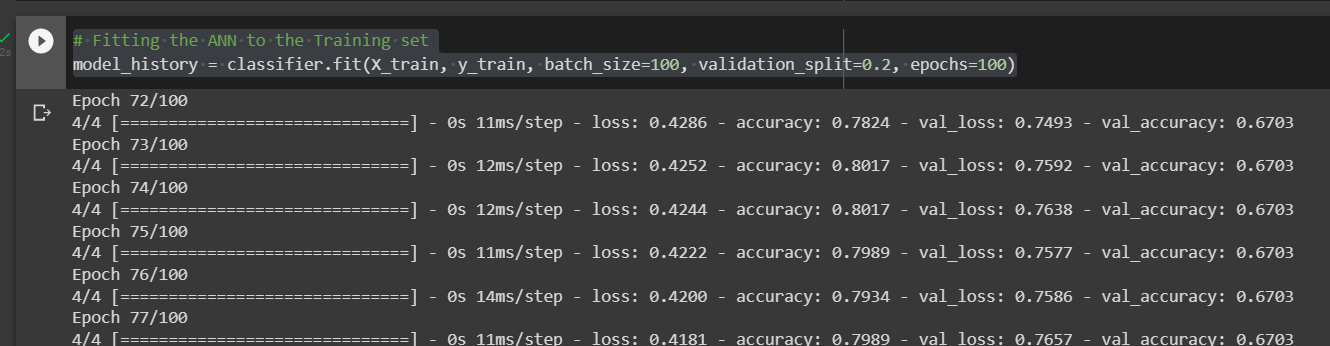
A function named xgboost is created and train and test data are passed as the parameters. Inside the function, GradientBoostingClassifier algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done.

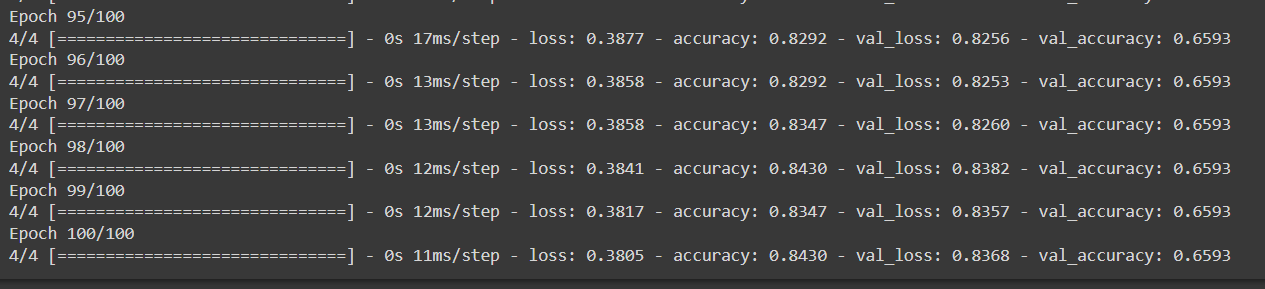


### **ANN Model**

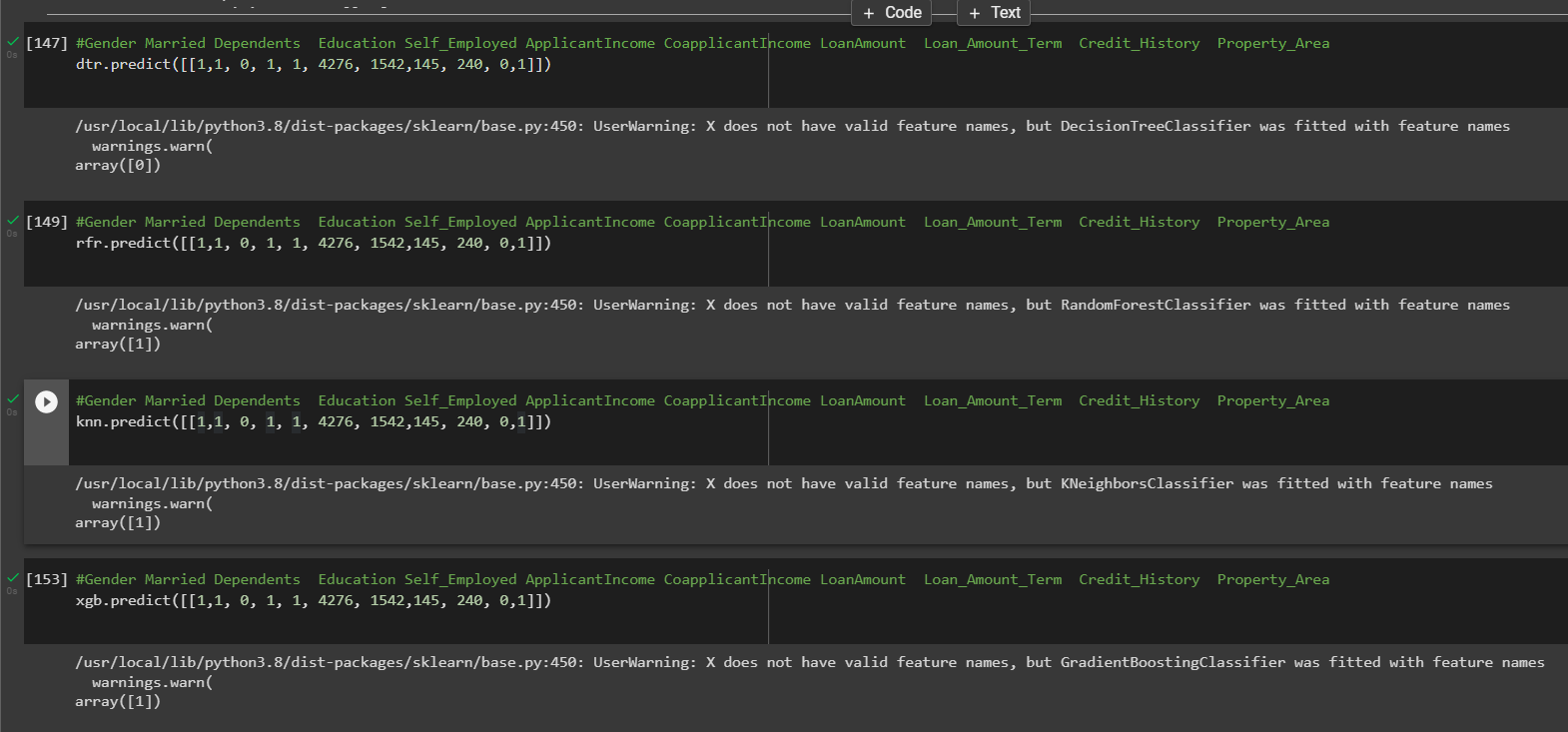
Building and training an Artificial Neural Network (ANN) using the Keras library with TensorFlow as the backend. The ANN is initialised as an instance of the Sequential class, which is a linear stack of layers. Then, the input layer and two hidden layers are added to the model using the Dense class, where the number of units and activation function are specified. The output layer is also added using the Dense class with a sigmoid activation function. The model is then compiled with the Adam optimizer, binary cross-entropy loss function, and accuracy metric. Finally, the model is fit to the training data with a batch size of 100, 20% validation split, and 100 epochs.



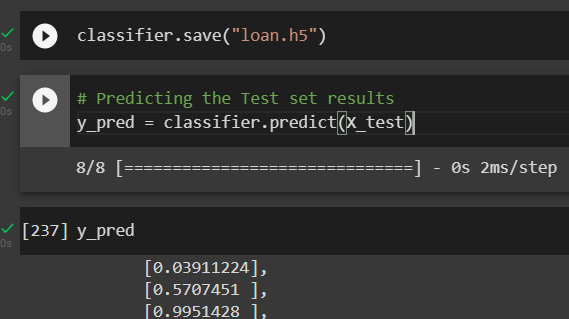


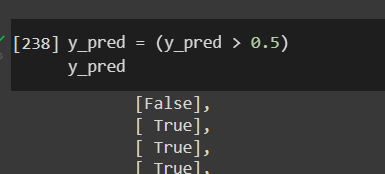


### **Testing The Model**

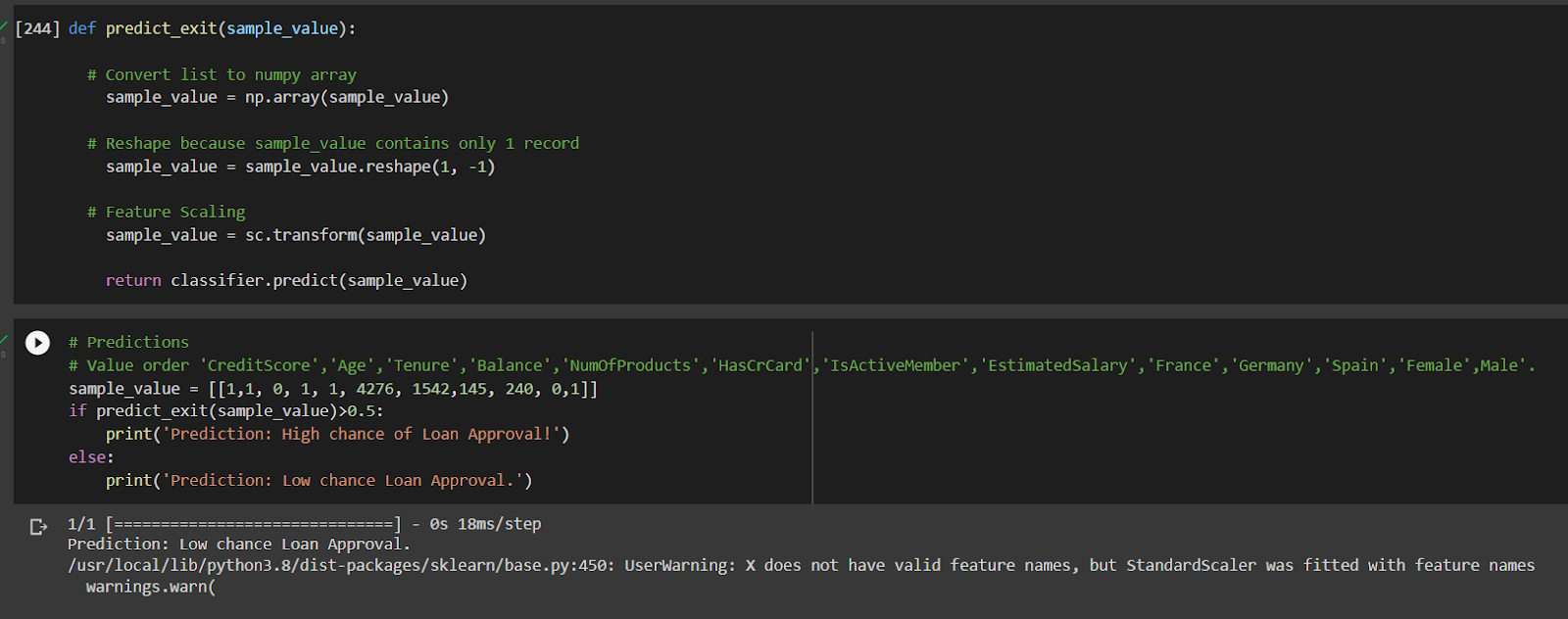
****

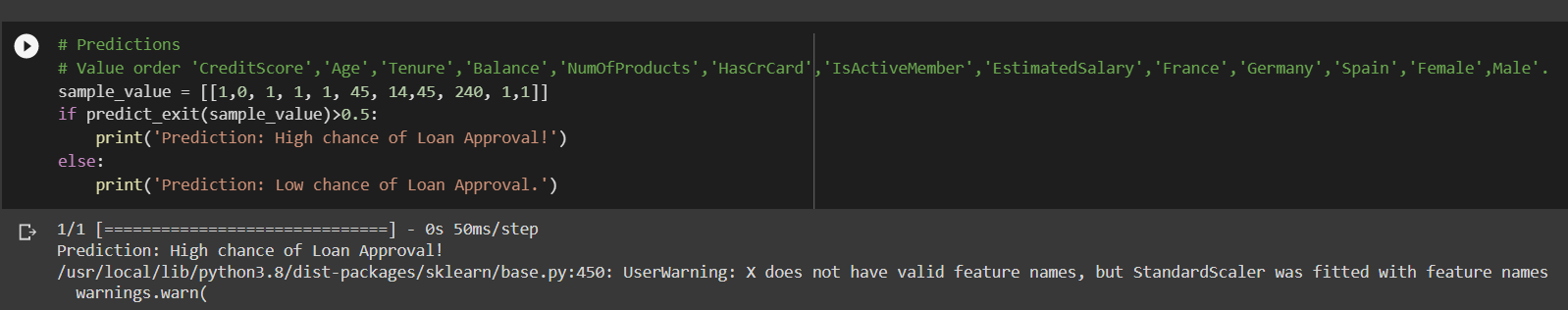
In ANN we first have to save the model to the test the inputs

****

****

This code defines a function named "predict\_exit" which takes in a sample\_value as an input. The function then converts the input sample\_value from a list to a numpy array. It reshapes the sample\_value array as it contains only one record. Then, it applies feature scaling to the reshaped sample\_value array using a scaler object 'sc' that should have been previously defined and fitted. Finally, the function returns the prediction of the classifier on the scaled sample\_value.

****

****

### **Model Building**

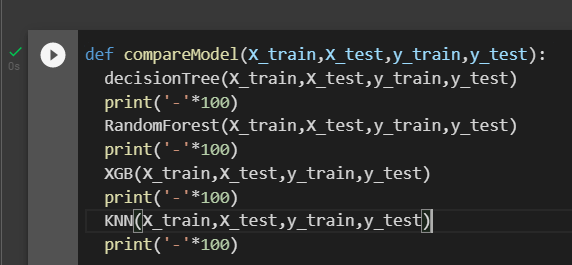
In this milestone, We will see the model building.

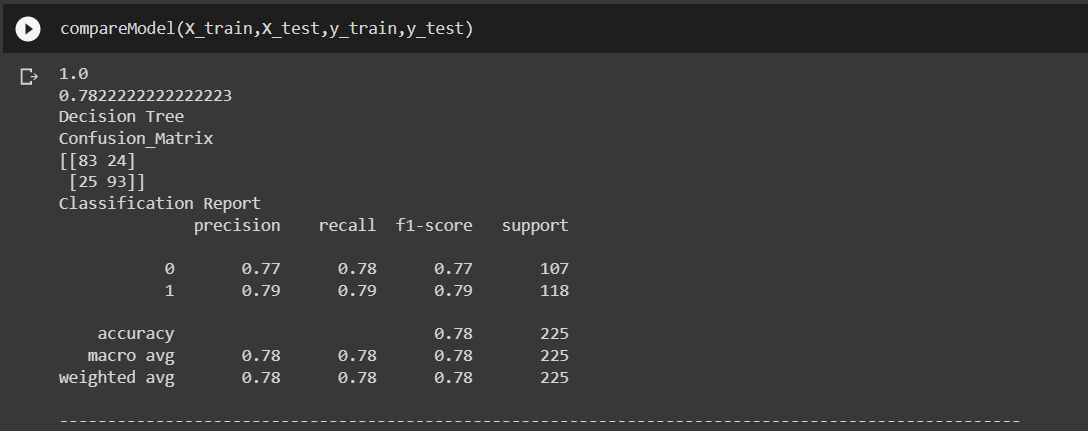
### **Testing Model With Multiple Evaluation Metrics**

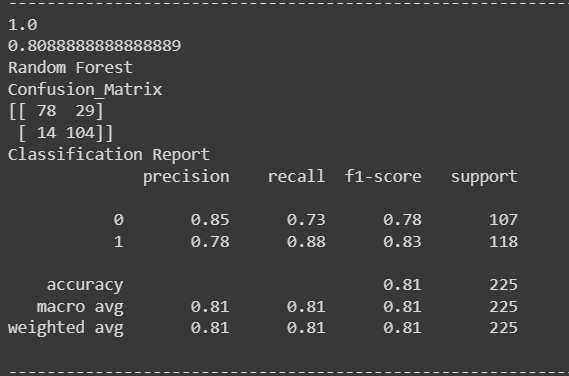
Multiple evaluation metrics means evaluating the model's performance on a test set using different performance measures. This can provide a more comprehensive understanding of the model's strengths and weaknesses. We are using evaluation metrics for classification tasks including accuracy, precision, recall, support and F1-score.

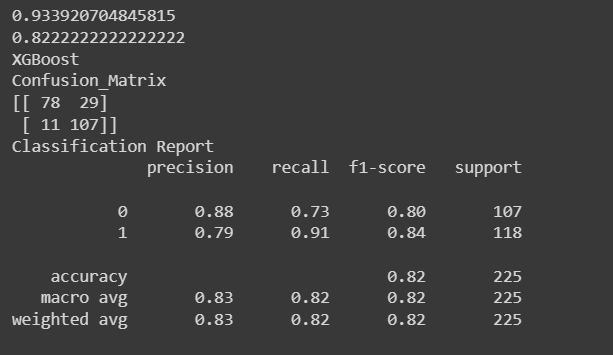
### **Compare The Model**

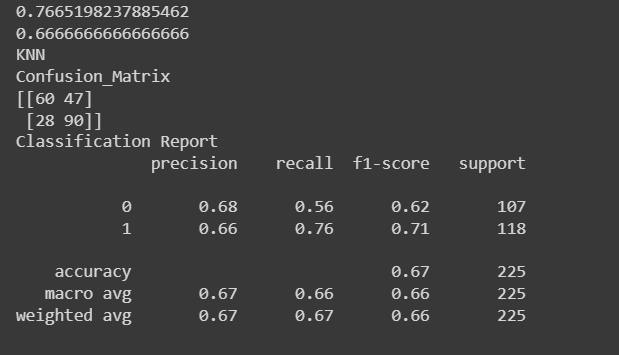
For comparing the above four models, the compareModel function is defined.

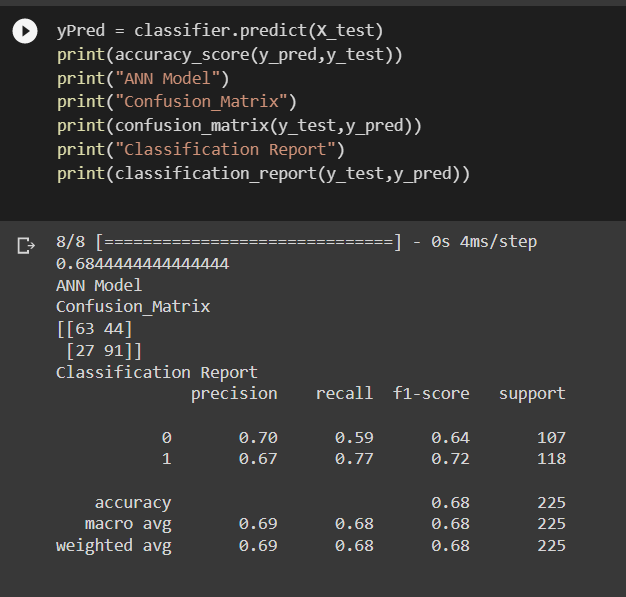










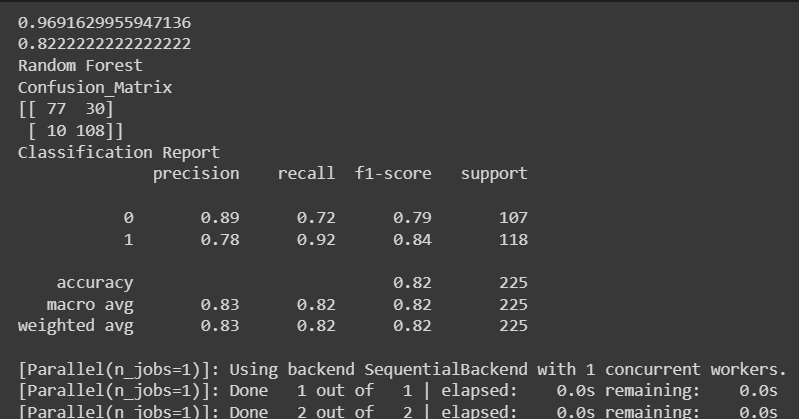


After calling the function, the results of models are displayed as output. From the five models Xgboost is performing well. From the below image, We can see the accuracy of the model. Xgboost is giving the accuracy of 93.39% with training data , 82.2% accuracy for the testing data.

### **Comparing Model Accuracy Before & After Applying Hyperparameter Tuning**

Evaluating performance of the modelFrom sklearn, cross\_val\_score is used to evaluate the score of the model. On the parameters, we have given rf (model name), x, y, cv (as 5 folds). Our model is performing well. So, we are saving the model by pickle.dump().



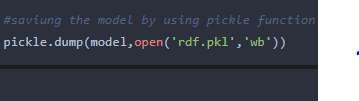


### **Performance Testing & Hyperparameter Tuning**

In this milestone, we will see the performance testing and hyperparameter Tuning

### **Save The Best Model**

Saving the best model after comparing its performance using different evaluation metrics means selecting the model with the highest performance and saving its weights and configuration. This can be useful in avoiding the need to retrain the model every time it is needed and also to be able to use it in the future.



### **Integrate With Web Framework**

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

This section has the following tasks

* Building HTML Pages
* Building server side script
* Run the web application

### **Building Html Pages**

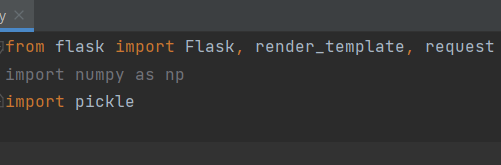
For this project create two HTML files namely

* home.html
* predict.html

and save them in the templates folder.

### **Build Python Code**

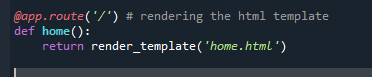
Import the libraries

****

Load the saved model. Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (\_\_name\_\_) as argument.



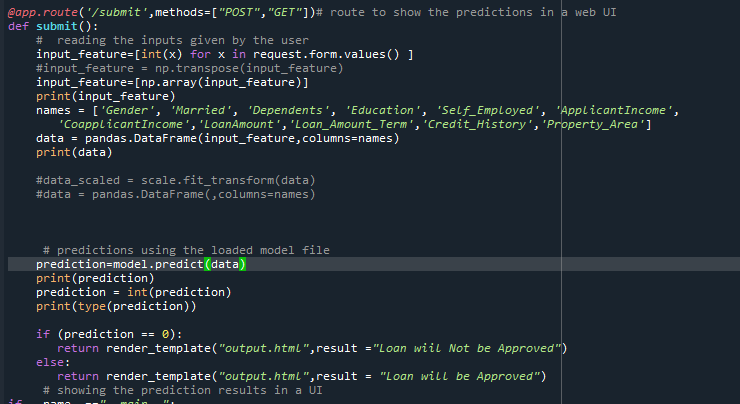
**Render HTML page:**



**Here we will be using a declared constructor to route to the HTML page which we have created earlier.**

In the above example, ‘/’ URL is bound with the home.html function. Hence, when the home page of the web server is opened in the browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrieves the value from UI:



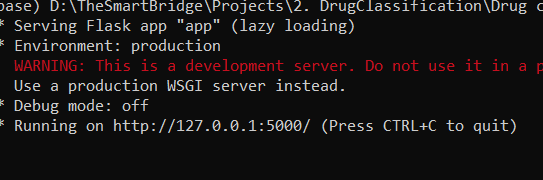
Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the submit.html page earlier.

**Main Function:**

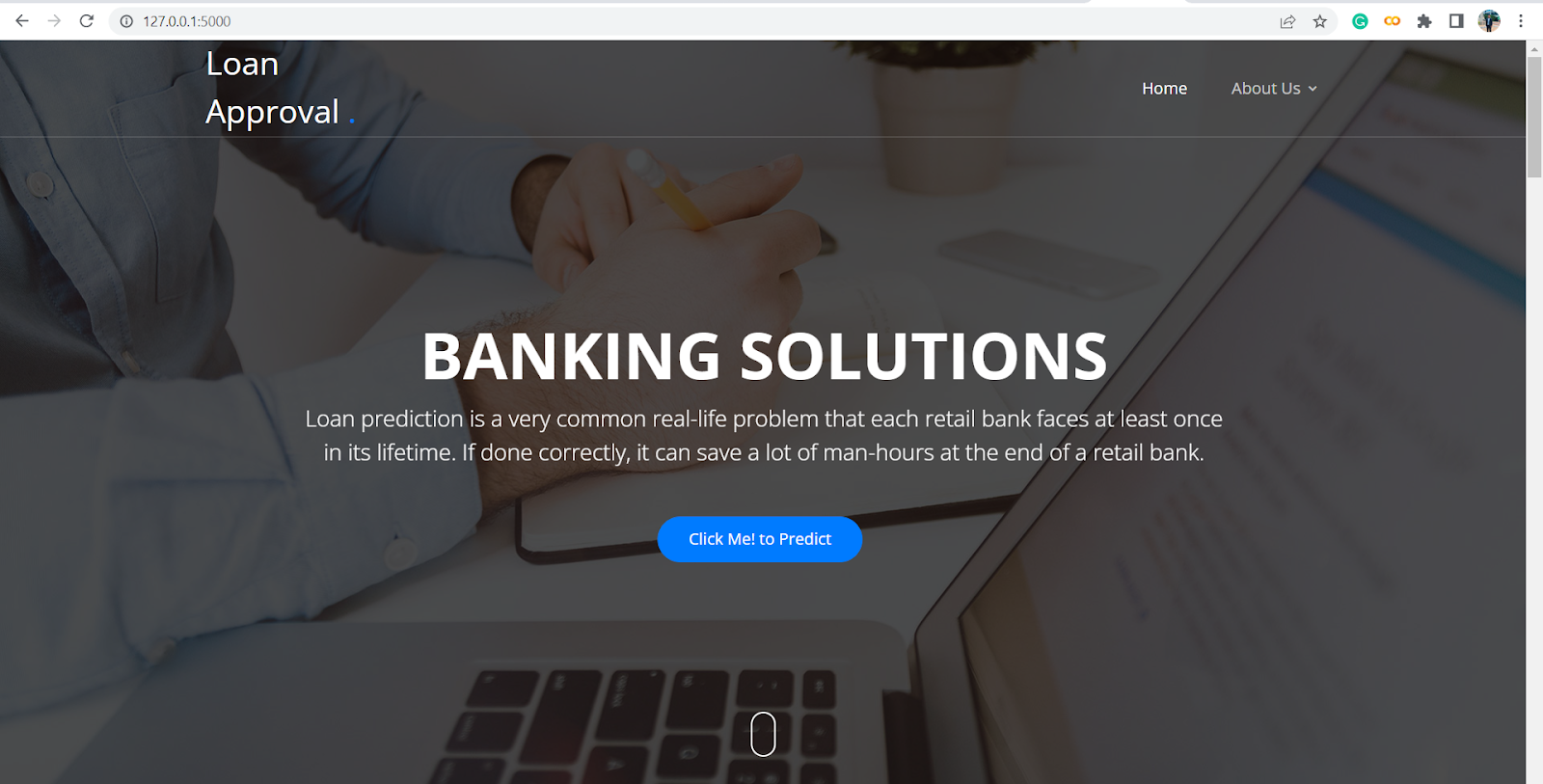


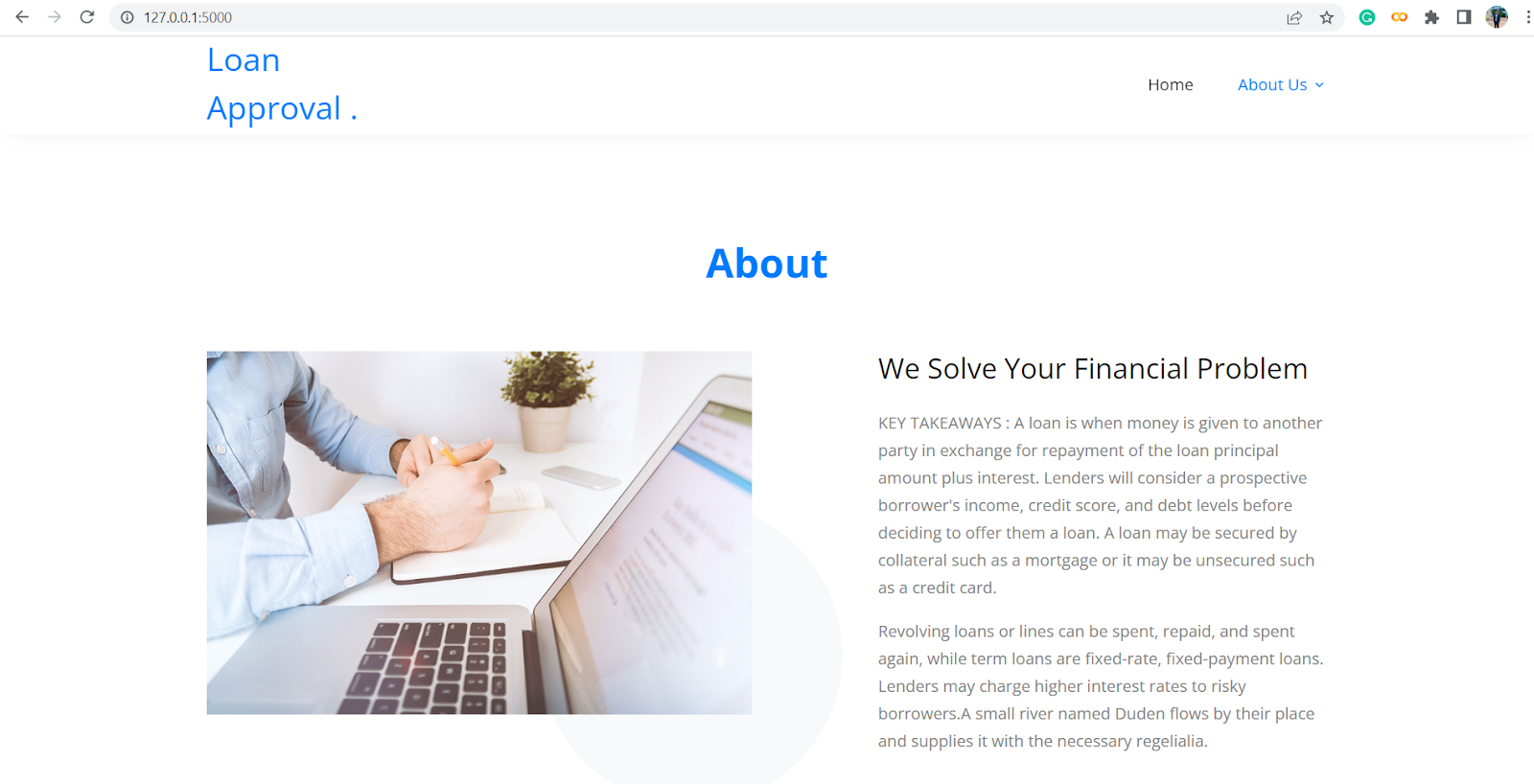
### **Run The Web Application**

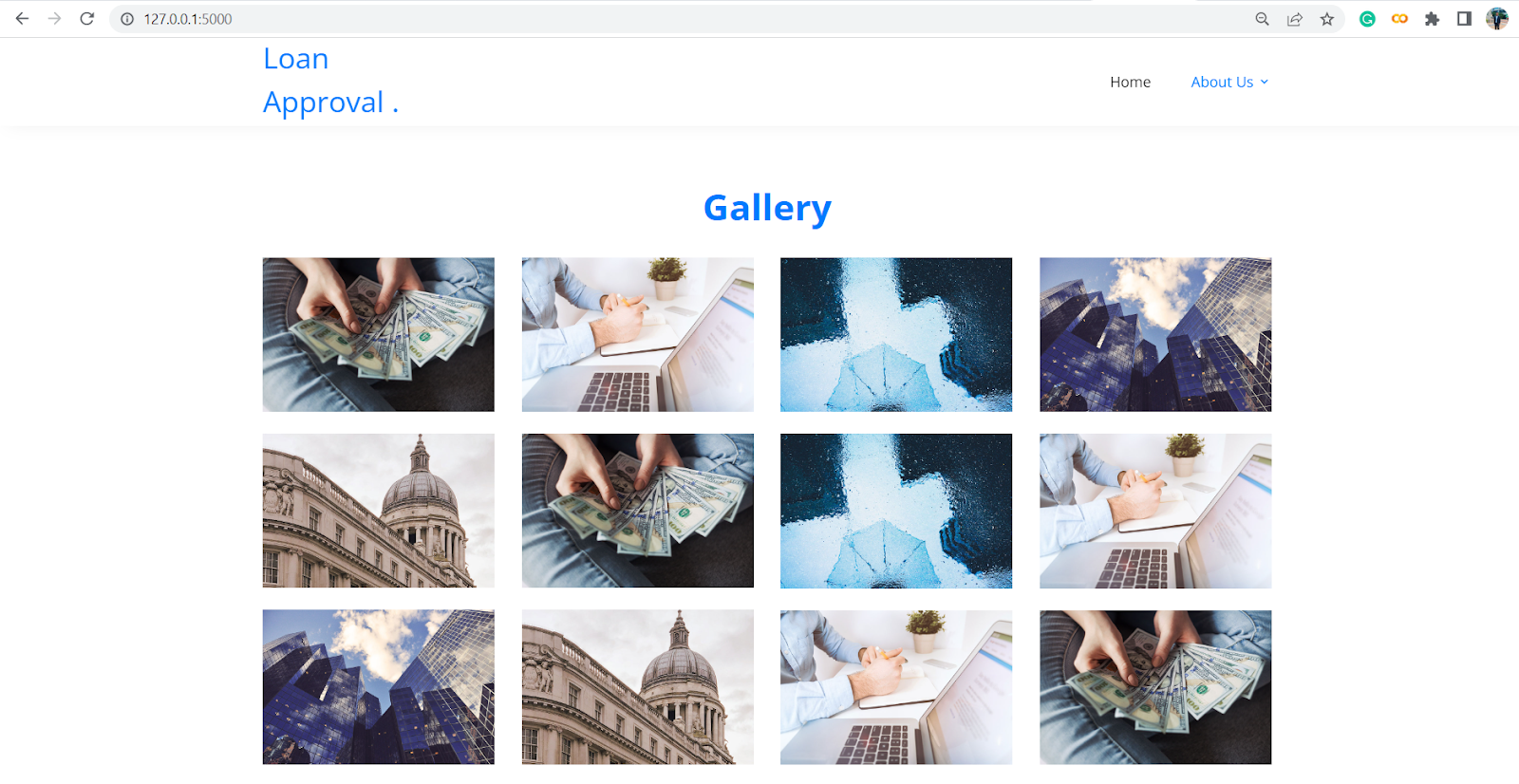
* Open anaconda prompt from the start menu
* avigate to the folder where your python script is.
* Now type “python app.py” command
* Navigate to the localhost where you can view your web page.
* Click on the predict button from the top left corner, enter the inputs, click on the submit button, and see the result/prediction on the web.

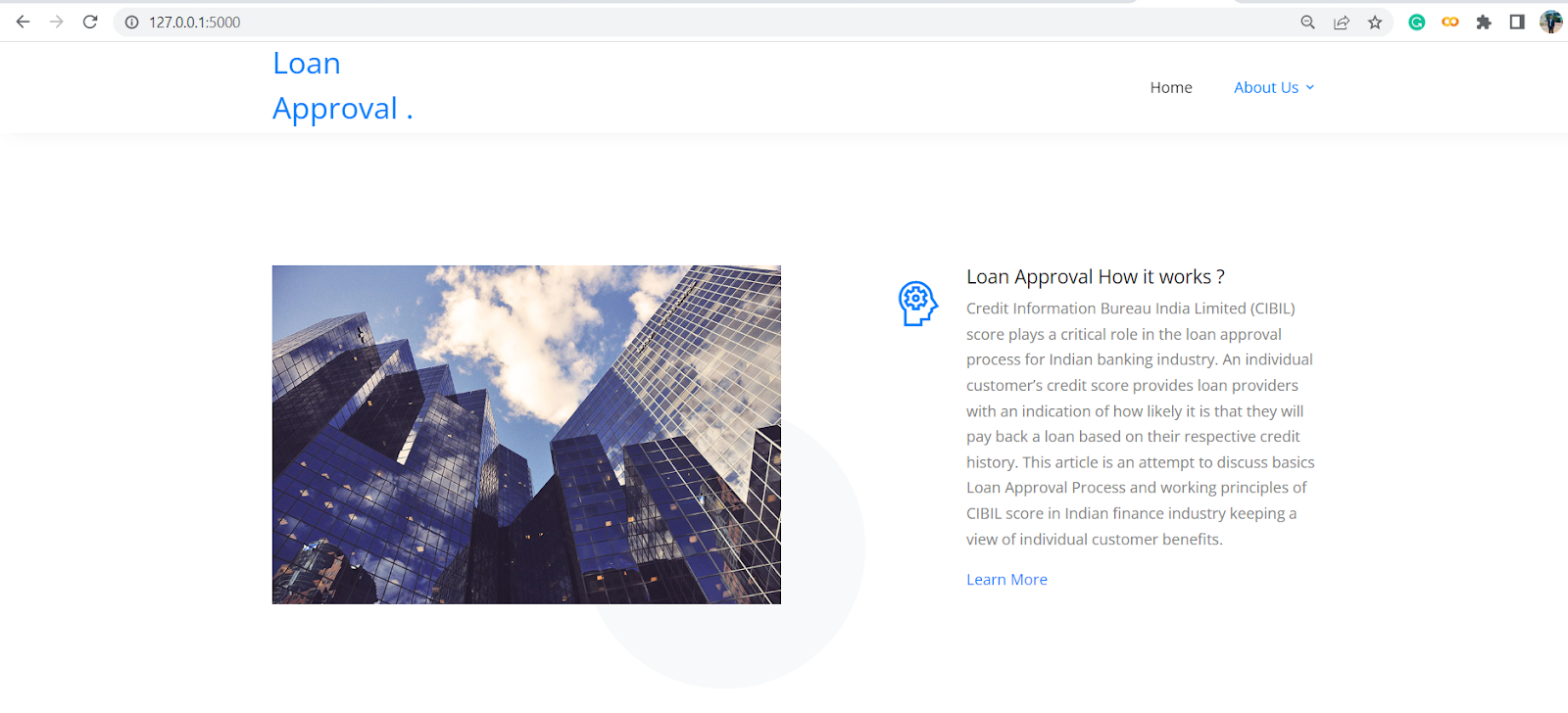


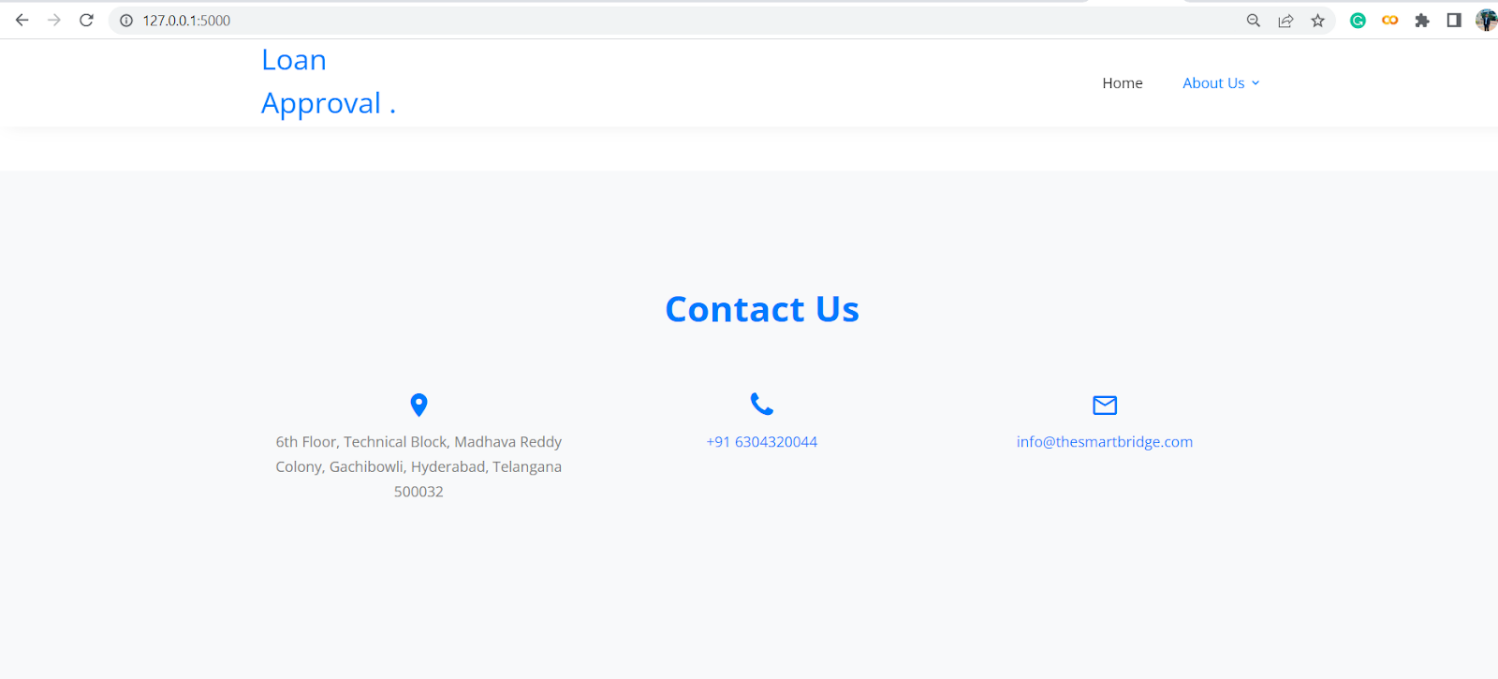
Now,Go the web browser and write the localhost url (http://127.0.0.1:5000) to get the below result

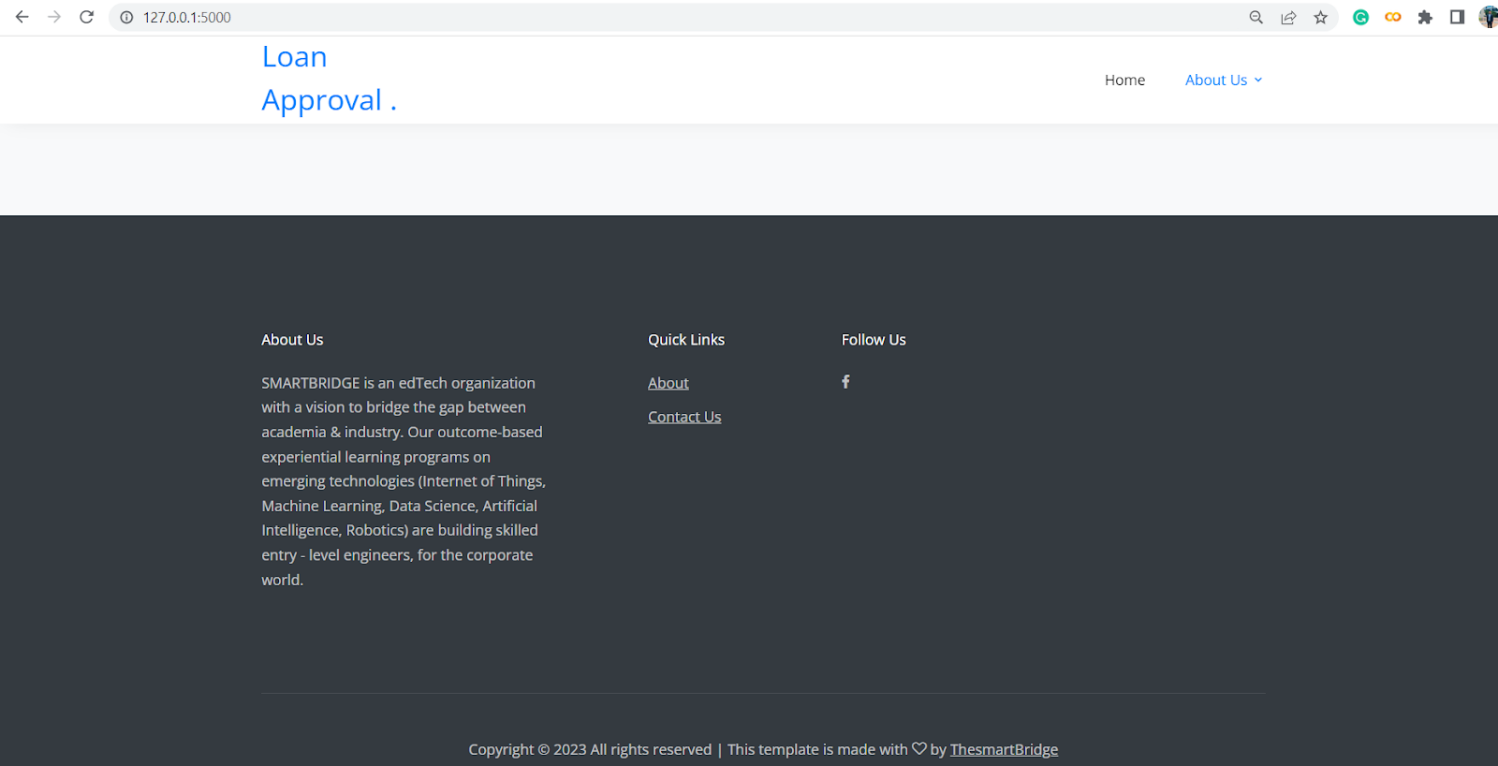




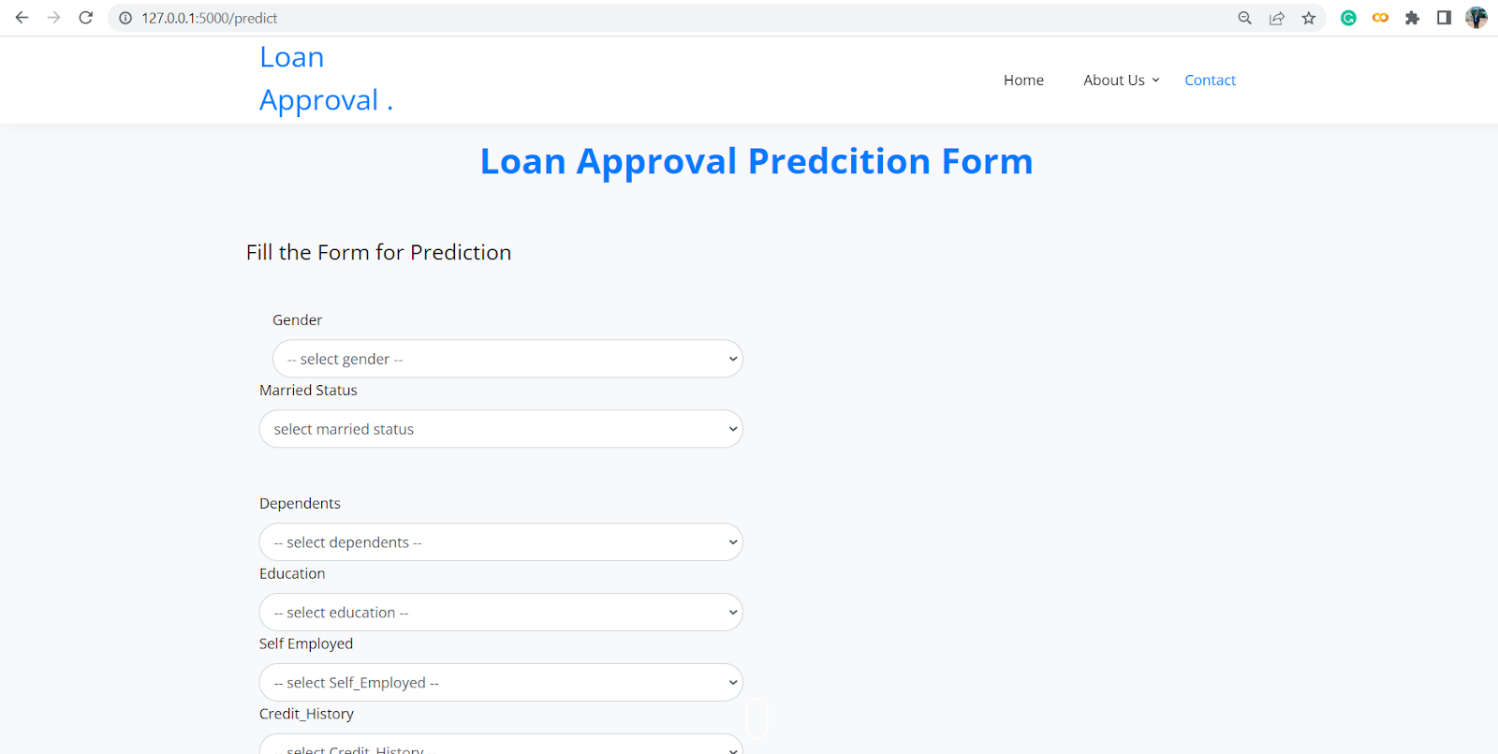


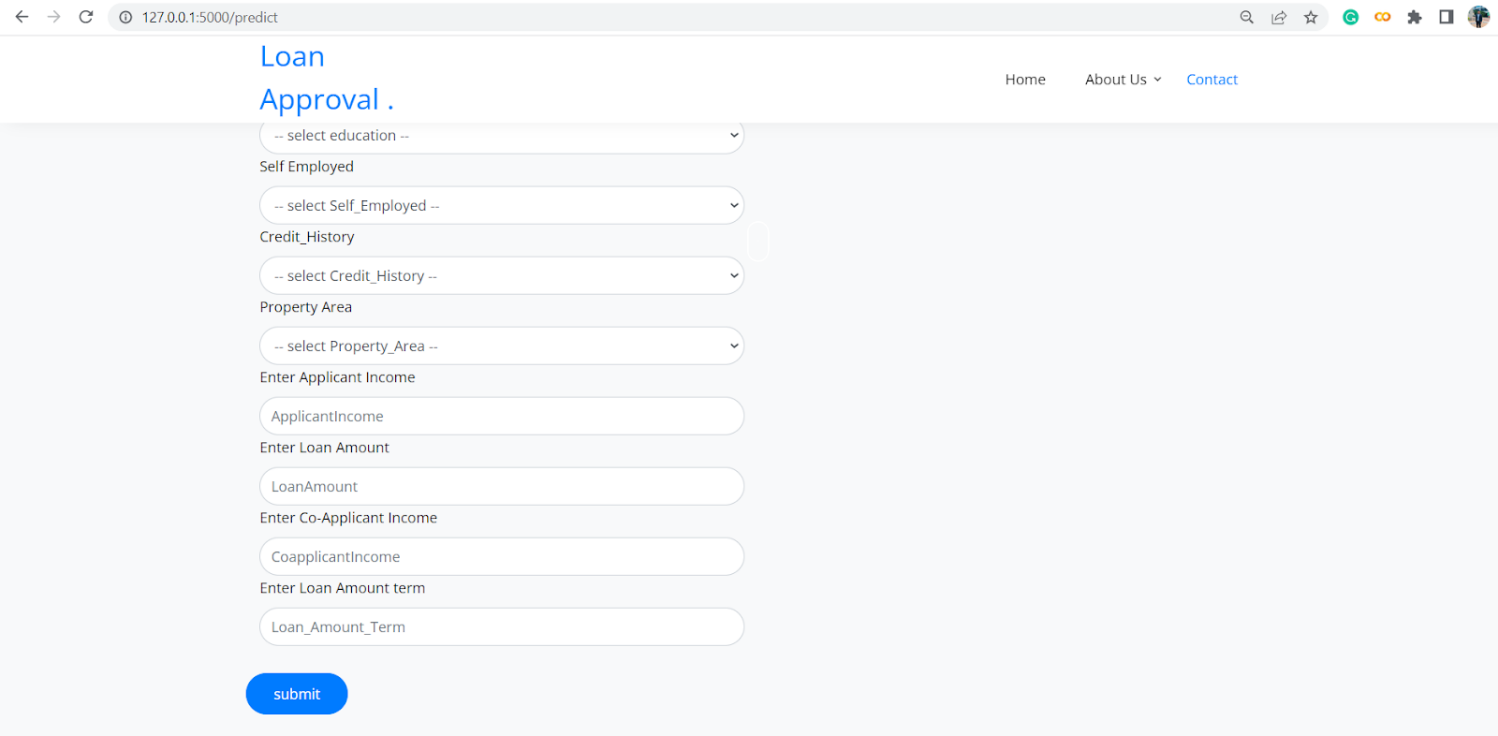




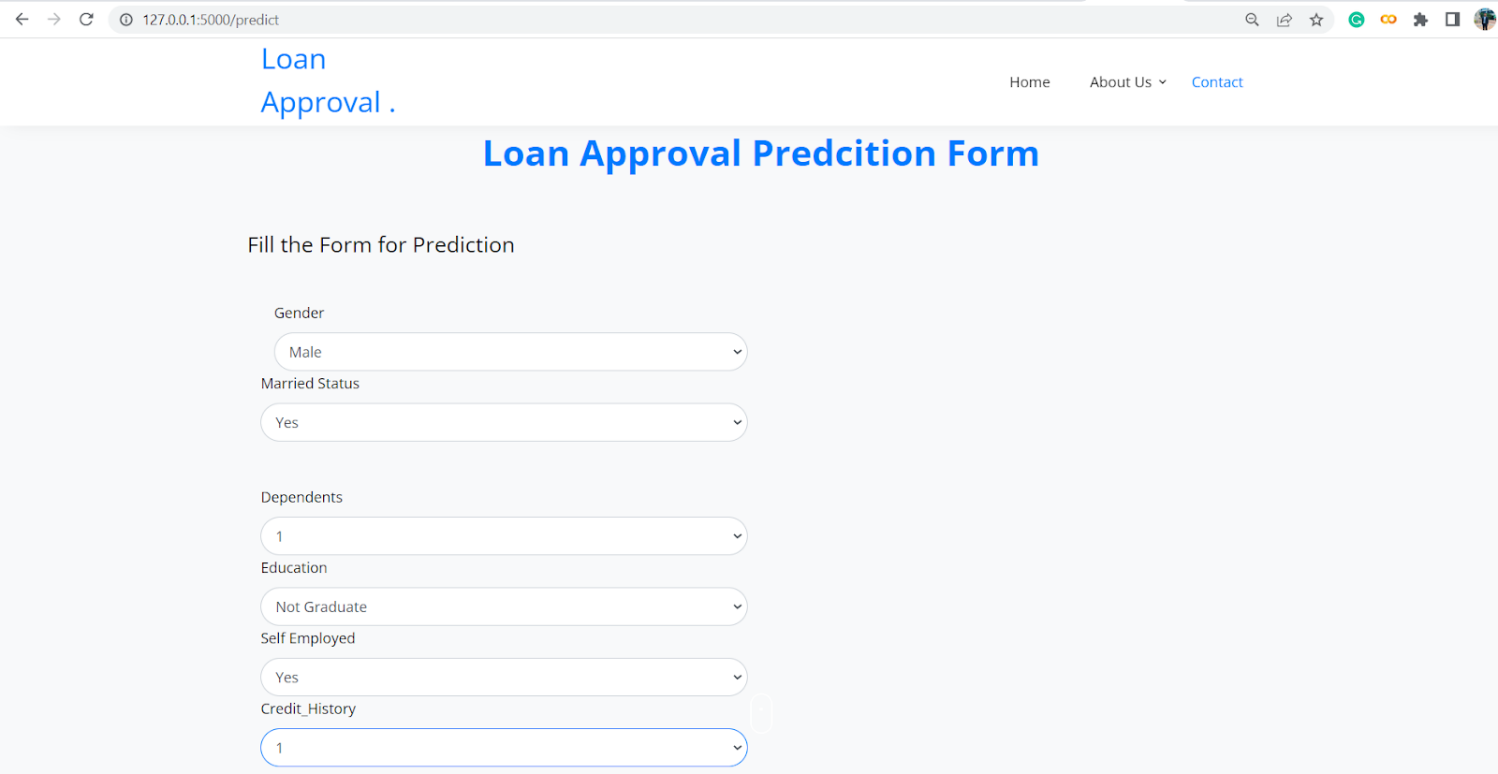


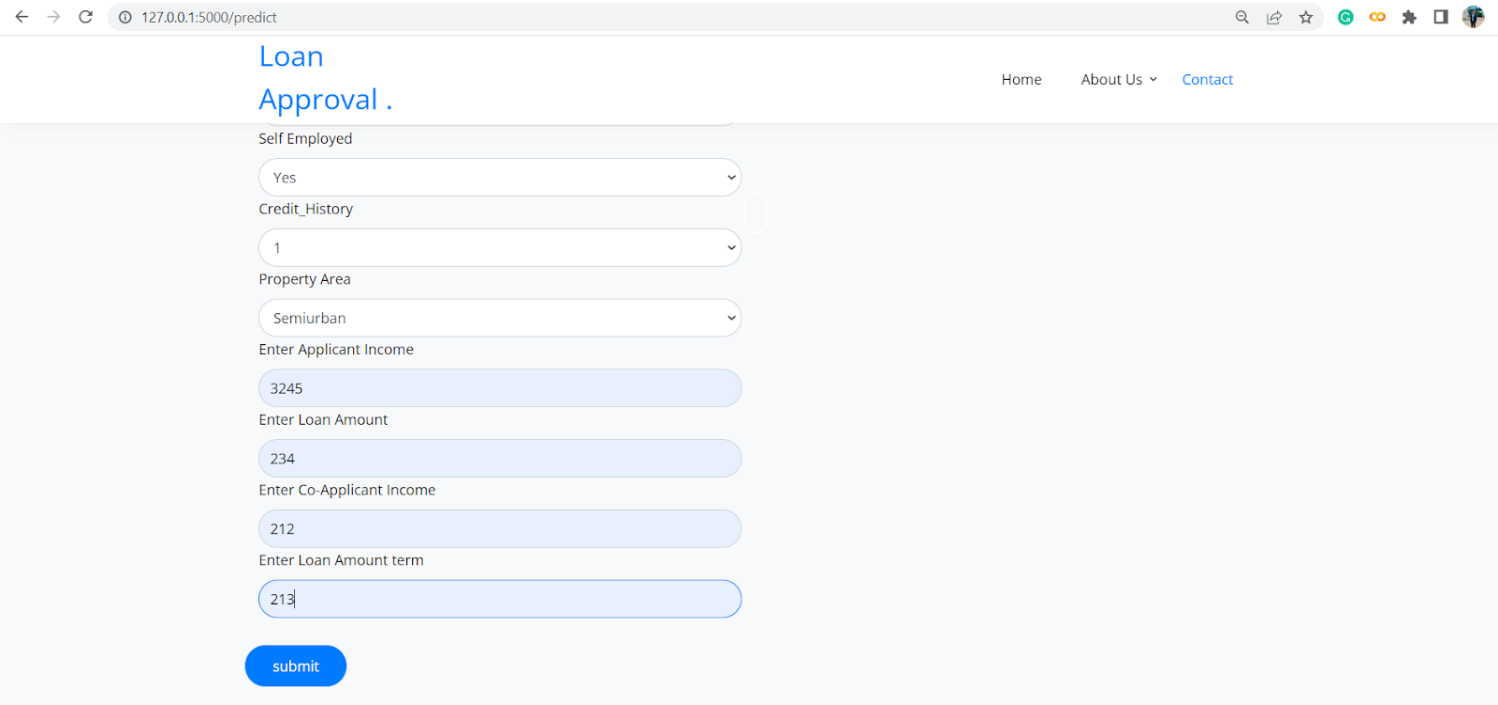
Now,when you click on click me to predict the button from the banner you will get redirected to the prediction page.



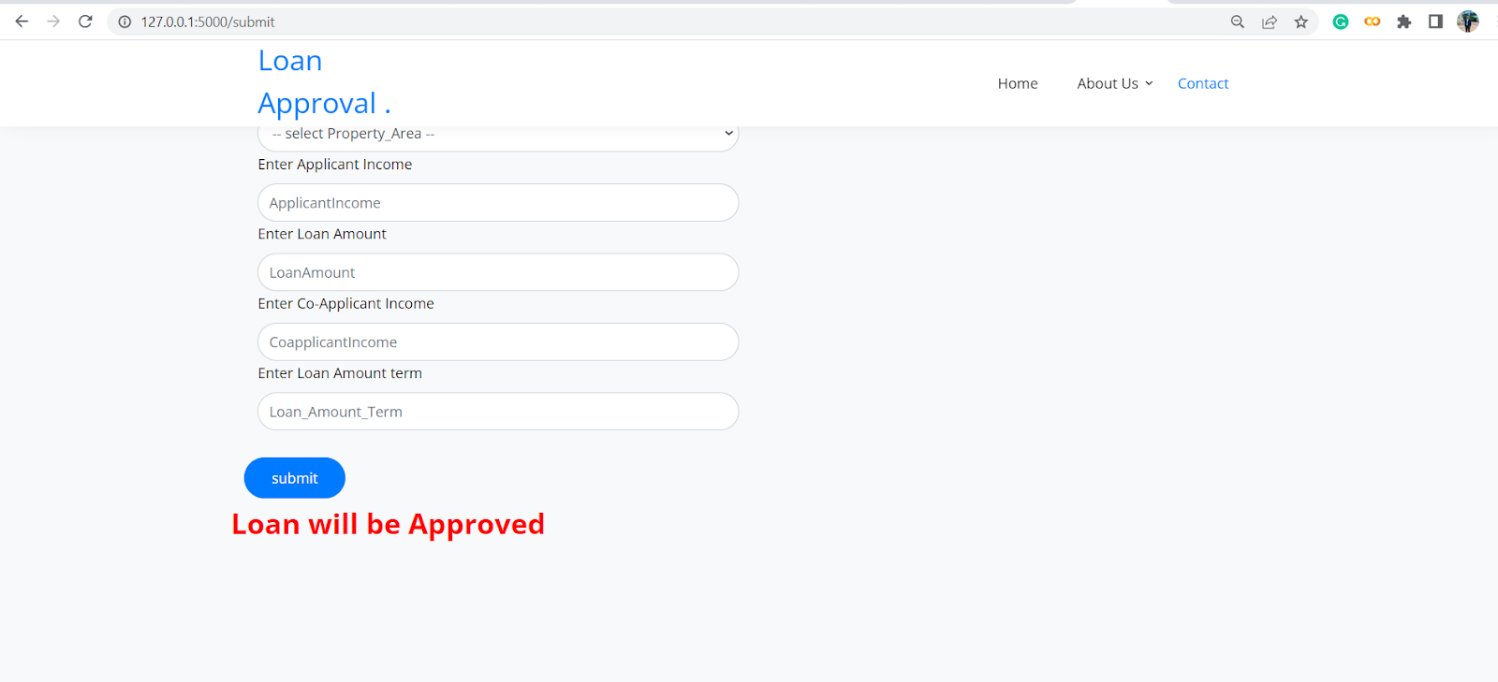


Input 1- Now, the user will give inputs to get the predicted result after clicking onto the submit button





Now when you click on the submit button you will get the result in the same page.



### **Model Deployment**

In this Milestone, We will see the Model Deployment.

### **Record Explanation Video For Project End To End Solution**

Record explanation Video for project end to end solution

### **Project Documentation-Step By Step Project Development Procedure**

Create document as per the template provided

### **Project Demonstration & Documentation**

In this milestone, we will see the project demonstration and documentation