1. ABSTRACT

The loan approval prediction system using machine learning is a project aimed at developing an automated system to predict the loan approval status of a loan application based on various factors. The system is designed to provide accurate and reliable loan approval predictions, which can be used by banks, financial institutions and other lending organizations.

The loan approval prediction system uses machine learning algorithms to analyze historical loan data such as loan amount, income, credit score, employment history and other relevant factors to predict the likelihood of loan approval. The system is trained on a large dataset of historical loan applications, which is used to build a predictive model that can accurately predict the loan approval status of new loan applications.

2. SYSTEM REQUIREMENTS

2.1 Hardware Requirements:

- Processor – Intel Core i5
- RAM – 4GB
- Storage – 512GB
- Graphics – 4GB
- Network

2.2 Software Requirements:

- Operating System – Windows 10
- Tool – Python IDLE
- Packages – Flask, Random Forest Classifier, NumPy, Pandas

3. ABOUT THE SOFTWARE

3.1 Front End (HTML and CSS):

HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets) are two fundamental technologies used to create web pages and web applications.

HTML is a markup language that defines the structure and content of web pages. It uses a series of tags to create elements such as headings, paragraphs, images, links, and forms. These tags provide structure to the content and allow web browsers to display the page correctly.

CSS is a style sheet language that describes the visual presentation of HTML documents. It allows developers to control the layout, typography, colors, and other visual aspects of a web page. CSS separates the presentation of the content from the content itself, making it easier to maintain and update the design of a website.

Together, HTML and CSS enable developers to create visually appealing and user-friendly websites that are accessible to users on any device with an internet connection. They are essential technologies for building the user interface of web applications and are used extensively in front-end web development.

HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets) are two essential technologies used for building web pages and web applications. HTML provides the structure and content of a web page, while CSS is used to style and layout the content.

In the context of a loan approval prediction system using machine learning, HTML and CSS would be used to build the user interface for the system. The HTML would be used to define the structure and content of the web page, including the input fields for the user to enter their information. The CSS would be used to style and layout the page, making it visually appealing and easy to use.

The loan approval prediction system would use machine learning algorithms to analyze the user's input data and predict whether or not they would be approved for a loan. The HTML and CSS would provide the user interface for entering their information and viewing the results of the prediction.

Overall, HTML and CSS play a critical role in building the user interface for a loan approval prediction system using machine learning, making it easy for users to enter their information and view the results of the prediction in a visually appealing and user-friendly way.

HTML is a markup language that uses a series of tags to define the structure and content of web pages. Tags are used to create elements such as headings, paragraphs, images, links, and forms. HTML

is a versatile language that can be used to create a wide range of web content, from simple static web pages to complex web applications.

CSS is a style sheet language that allows developers to control the visual presentation of HTML documents. CSS can be used to control the layout, typography, colors, and other visual aspects of a web page. CSS is designed to be modular and reusable, making it easier to maintain and update the design of a website.

HTML and CSS are often used together in front-end web development. HTML provides the structure and content of a web page, while CSS is used to style and layout the content. By separating the presentation of the content from the content itself, CSS makes it easier to create responsive and accessible websites that work well on a variety of devices.

In addition to HTML and CSS, there are many other technologies and frameworks used in front-end web development, such as JavaScript, Bootstrap, and React. Together, these technologies allow developers to create highly interactive and dynamic web applications that provide a rich user experience.

3.2 Tool (python IDLE):

Python is a popular high-level programming language that was first released in 1991 by Guido van Rossum. It is a general-purpose language that can be used for a wide range of tasks, from scripting to web development to scientific computing and data analysis.

One of the main features of Python is its simplicity and readability, making it an ideal language for beginners to learn. Python code is often described as being “executable pseudocode” due to its readability and expressiveness. It also has a large standard library and a wide range of third-party libraries, making it easy to get started with and very versatile.

Python is an interpreted language, which means that it is not compiled like languages such as C or Java. Instead, the code is executed directly by the Python interpreter. This can make development faster and more flexible, but it also means that Python can be slower than compiled languages for some tasks.

Python is open-source and has a large and active community of developers. This means that there are many resources available for learning and getting help with Python, including online tutorials, forums, and documentation

- **Syntax:** Python has a clean and simple syntax that is easy to read and write. It uses indentation to indicate blocks of code, rather than curly braces or other symbols.

- **Object-oriented**: Python is an object-oriented language, which means that it allows you to define classes and objects that encapsulate data and behavior.
- **Dynamically typed**: Python is a dynamically typed language, which means that you don't have to specify variable types when you declare them. This can make code easier to write and read, but can also lead to more errors at runtime.
- **Cross-platform:** Python can run on a variety of operating systems, including Windows, Mac OS, and Linux, making it a very versatile language.
- **Large community**: Python has a large and active community of developers, which means that there are many resources available for learning and getting help with Python. This includes online forums, documentation, and tutorials.
- **Libraries and frameworks**: Python has a large number of libraries and frameworks available for a wide range of tasks, including web development, scientific computing, machine learning, data analysis, and more.
- **High-level:** Python is a high-level language, which means that it abstracts away many low-level details of computer programming. This can make it easier to write code quickly and to understand code written by others.

Overall, Python is a powerful and versatile programming language that is widely used in a variety of fields. Its ease of use, large community, and extensive library of tools make it an ideal language for beginners and experienced developers alike.

Dataset:

In machine learning, a dataset is a collection of data used to train, test, and validate a machine learning model. The dataset consists of one or more samples, where each sample is a set of input features and the corresponding output label (for supervised learning) or target variable (for unsupervised learning). The dataset is used to train a machine learning model, which learns to map the input features to the output labels or target variables.

Datasets can come in many forms, depending on the type of problem being solved and the nature of the data. Some common types of datasets in machine learning include:

- **Structured datasets:** These datasets consist of tabular data, where each row represents a sample and each column represents a feature or attribute.
- **Text datasets:** These datasets consist of text data, such as emails, news articles, or social media posts, and are often used in natural language processing (NLP) tasks.

- **Image datasets:** These datasets consist of images, such as photographs, digital artwork, or medical scans, and are often used in computer vision tasks.
- **Audio datasets:** These datasets consist of audio recordings, such as music, speech, or sound effects, and are often used in speech recognition or audio processing tasks.
- **Time-series datasets:** These datasets consist of data collected over time, such as stock prices, weather data, or sensor readings, and are often used in forecasting or anomaly detection tasks.

Before using a dataset to train a machine learning model, it is important to preprocess and clean the data to ensure that it is consistent and accurate. Once the data is preprocessed, it can be split into training, validation, and testing sets, and used to train and evaluate a machine learning model.

- **Size:** The size of a dataset can vary widely, from a few hundred samples to millions or even billions of samples. The size of the dataset can have a significant impact on the performance and accuracy of a machine learning model.
- **Bias**: Datasets can be biased, meaning they may not represent the full range of variation in the real world. For example, a dataset used to train a facial recognition system may be biased towards a particular demographic, leading to inaccurate predictions for other demographics.
- **Overfitting:** If a machine learning model is trained on a small or biased dataset, it may become overfitted, meaning it performs well on the training data but poorly on new, unseen data. To avoid overfitting, it is important to use a diverse and representative dataset.
- **Imbalanced datasets:** Datasets can also be imbalanced, meaning that one class or label is overrepresented compared to others. This can lead to a model that performs well for the overrepresented class but poorly for the others. Techniques such as oversampling, under sampling, or using weighted loss functions can be used to address this issue.
- **Public datasets:** There are many publicly available datasets that can be used for machine learning tasks. These datasets can be a great resource for researchers and practitioners, as they often provide a standardized benchmark for comparing different models and algorithms.
- **Data augmentation**: Data augmentation techniques can be used to artificially increase the size of a dataset by creating new samples through transformations such as rotation, scaling, or adding noise. This can improve the robustness and generalization of a machine learning model.

Overall, datasets are a crucial component of machine learning, as they provide the training data that allows a model to learn and make predictions. It is important to carefully select, pre-process, and clean datasets to ensure that they are representative, diverse, and unbiased, in order to build accurate and robust machine learning models.

4. SYSTEM ANALYSIS

4.1 Existing System:

There are many existing loan approval prediction systems that use machine learning techniques. Here are some examples

- **Lending Club:** Lending Club is a peer-to-peer lending platform that uses machine learning algorithms to analyze credit risk and approve loans. The system considers factors such as credit score, debt-to-income ratio, employment history, and loan amount to make lending decisions.
- **Zest Finance:** Zest Finance uses machine learning to analyze credit risk for loans. The system uses non-traditional data sources such as social media and mobile phone usage to predict loan defaults. Zest Finance claims to have improved credit scores for millions of borrowers and reduced default rates for lenders.
- **Upstart:** Upstart is a lending platform that uses machine learning to assess creditworthiness. The system analyze factors such as education, employment history, and income to make lending decisions. Upstart claims to have reduced default rates by 75% compared to traditional lending models.
- **Kabbage:** Kabbage is a lending platform that uses machine learning to analyze credit risk for small business loans. The system considers factors such as business revenue, cash flow, and social media presence to make lending decisions. Kabbage claims to have funded over $14 billion in loans to small businesses.
- **Sift Science:** Sift Science is a fraud detection platform that uses machine learning to analyze user behaviour and detect fraudulent activity. The system analyze factors such as IP address, device type, and transaction history to identify suspicious behaviour. Sift Science claims to have reduced fraud rates by 80% for some clients.

These are just a few examples of loan approval prediction systems that use machine learning techniques. There are many other systems available, each with their own unique approach to credit risk analysis and lending decision-making.

4.2 Proposed System:

Building a loan approval prediction system using machine learning techniques can be a complex project that requires careful planning and execution. Here are some steps you can follow to create such a system:

- **Gather data:** The first step in building a loan approval prediction system is to gather relevant data. You will need data about loan applicants, their credit history, income, employment status, and other relevant information. This data can be obtained from various sources such as credit bureaus, financial institutions, and government agencies.
- **Clean and pre-process data:** Once you have collected the data, you need to clean and pre-process it. This involves removing any duplicates, filling in missing values, and transforming the data into a format that can be used by machine learning algorithms.
- **Define the problem:** You need to define the problem you are trying to solve. In this case, the problem is to predict whether a loan application should be approved or rejected based on the applicant's information.
- **Choose a machine learning algorithm:** There are several machine learning algorithms that can be used for loan approval prediction, such as logistic regression, decision trees, random forests, and neural networks. You will need to choose the algorithm that best suits your needs based on the size of the data set, the complexity of the problem, and other factors.
- **Train the model:** Once you have chosen an algorithm, you will need to train the model using the data you have collected. This involves splitting the data into a training set and a test set, and then using the training set to train the model.
- **Evaluate the model**: After training the model, you need to evaluate its performance on the test set. This will give you an idea of how well the model is able to predict loan approvals.
- **Fine-tune the model**: If the model is not performing well, you may need to fine-tune it by adjusting its parameters or using a different algorithm. This process may involve iterating several times until you get a model that performs well.
- **Deploy the model:** Once you have a model that performs well, you can deploy it in a production environment. This may involve integrating it with other systems or creating a user interface for it.
- **Monitor the model:** Finally, you need to monitor the model to ensure that it continues to perform well over time. This may involve retraining the model periodically or updating it with new data.

Overall, building a loan approval prediction system using machine learning techniques can be a challenging project, but by following these steps, you can create an accurate and reliable system that can help financial institutions make better lending decisions

.

5. SYSTEM DESIGN

5.1 Dataset Description:

The loan approval prediction dataset is a collection of data that is commonly used in machine learning to predict whether a loan application is likely to be approved or not. The dataset contains a variety of information about loan applicants, including their age, gender, marital status, education, income, loan amount, loan term, credit history, and other relevant factors.

The dataset may be structured in a tabular format, where each row corresponds to a loan applicant and each column corresponds to a feature of the applicant. The dataset may also contain labels indicating whether each loan application was approved or denied, which is the target variable that machine learning models are trained to predict.

Some features that may be included in the dataset are:

1. Applicant Income: the income of the applicant
2. Coapplicant Income: the income of the co-applicant (if any)
3. Loan Amount: the amount of the loan requested
4. Loan_Amount_Term: the term of the loan (in months)
5. Credit_History: whether the applicant has a credit history (1 = yes, 0 = no)
6. Property_Area: the area where the applicant's property is located
7. Gender: the gender of the applicant
8. Education: the level of education of the applicant
9. Marital_Status: the marital status of the applicant
10. Dependents: the number of dependents of the applicant

The loan approval prediction dataset is a popular dataset used in many machine learning applications, and it can be used to train and test various machine learning models such as logistic regression, decision trees, and random forests to predict the likelihood of a loan application being approved or denied.

5.2. Methodology:

Working of Random Forest Algorithm:

Random Forest works in 2 phases. First is to create the random forest by combining N decision tree and second is to make predictions for each tree created in first phase.

Step 1: Select random k data points from the training set.

Step 2: Build the decision trees associated with the selected data points.

Step 3: Choose the number N for decision trees you want to build.

Step 4: Repeat Step1 & Step2.

Step 5: For new data points, find the predictions of each decision tree, assign the new data points to the category that wins the majority votes.

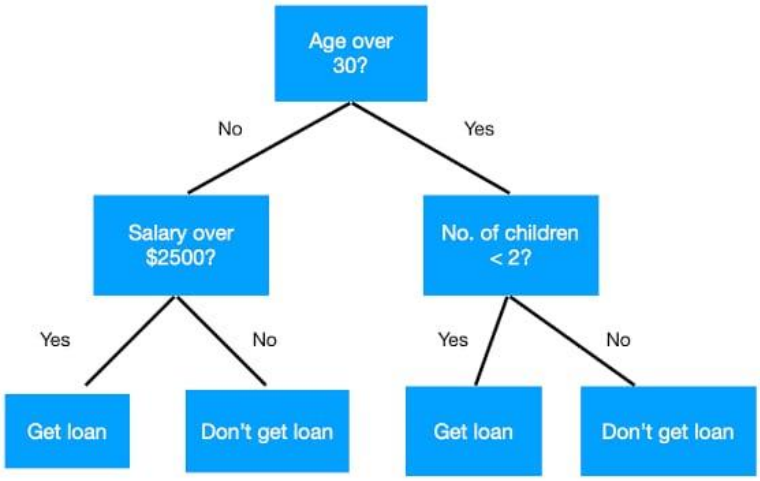

Fig 5.2.1 Work flow of Random Forest Algorithm

6. SYSTEM IMPLEMENTATION

6.1 Source Code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.naive_bayes import GaussianNB
data = pd.read_csv("LoanApprovalPrediction.csv")
##Data Preprocessing and Visualization
##Get the number of columns of object datatype.
obj = (data.dtypes == 'object')
##print("Categorical variables:",len(list(obj[obj].index)))
# Dropping Loan_ID column
data.drop(['Loan_ID'],axis=1,inplace=True)


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

```
#missing values
for col in data.columns:
    data[col] = data[col].fillna(data[col].mean())
    data.isna().sum()



#splitting data set
from sklearn.model_selection import train_test_split
X = data.drop(['Loan_Status'],axis=1)
Y = data['Loan_Status']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,test_size=0.4,random_state=1)

#accuracy
models = {"random_forest": RandomForestClassifier(n_estimators=100)}
model = models['random_forest']
model.fit(X_train,Y_train)
Y_pred = model.predict(X_test)
print("Accuracy score of ",model._class.name_,"=",100*metrics.accuracy_score(Y_test,Y_pred))



# single data prediction
new_gender=input("your gender:?")
new_married_sts=input("married or unmarried")
dependent_count=int(input("your dependency"))
education_sts=input("graduate or not graduate")
self_emp_sts=input("are you self_employed")
applicant=int(input("your income"))
co_applicant=int(input("your co applicant income"))
loan_amount=int(input("loan amount"))
loan_amount_term=int(input("loan amount term"))
```

```
credit=int(input("credit history"))
property_area=input("property location")
new_gender_transform=label_encoder.fit_transform([new_gender])
new_married_sts_transform=label_encoder.fit_transform([new_married_sts])
education_sts_transform=label_encoder.fit_transform([education_sts])
self_emp_sts_transform=label_encoder.fit_transform([self_emp_sts])
property_area_transform=label_encoder.fit_transform([property_area])
prediction=model.predict([[new_gender_transform[0],new_married_sts_transform[0],dependent_count
,education_sts_transform[0],self_emp_sts_transform[0],applicant,co_applicant,loan_amount,loan_amo
unt_term,credit,property_area_transform[0]]])
print(prediction)
```

Flask source code:

```
from flask import Flask, render_template, request
import pickle
from sklearn.preprocessing import LabelEncoder
app = Flask(_name_)
@app.route('/')
def homepage():
    return render_template('login.html')
@app.route("/loan", methods=['POST', 'GET'])
def register():
    if request.method == 'POST':
        n = request.form.get('name')
        a = request.form.get('mobilenumber')
        return  render_template('confirm.html',name=n,mobilenumber=a)
@app.route("/predict", methods=['POST', 'GET'])
def completed():
    if request.method == 'POST':
```

```
        label_encoder = LabelEncoder()
        new_gender=request.form.get('Gender')
        new_married_sts=request.form.get('Married')
        dependent_count=int(request.form.get('Dependents'))
        education_sts=request.form.get('Education')
        self_emp_sts=request.form.get('Self_Employed')
        applicant=int(request.form.get('ApplicantIncome'))
        co_applicant=int(request.form.get('CoapplicantIncome'))
        loan_amount=int(request.form.get('LoanAmount'))
        loan_amount_term=int(request.form.get('Loan_Amount_Term'))
        credit=int(request.form.get('Credit_History'))
        property_area=request.form.get('Property_Area')

        new_gender_transform=label_encoder.fit_transform([new_gender])
        new_married_sts_transform=label_encoder.fit_transform([new_married_sts])
        education_sts_transform=label_encoder.fit_transform([education_sts])
        self_emp_sts_transform=label_encoder.fit_transform([self_emp_sts])
        property_area_transform=label_encoder.fit_transform([property_area])

# Prediction Process:-
        filename='loan_prediction_model.sav'
        loaded_model = pickle.load(open(filename, 'rb'))

prediction=(loaded_model.predict([[new_gender_transform[0],new_married_sts_transform[0],dependent_count,education_sts_transform[0],self_emp_sts_transform[0],applicant,co_applicant,loan_amount,loan_amount_term,credit,property_area_transform[0]]]))
        output=label_encoder.inverse_transform(prediction)

    return str(output)

if _name_ == '_main_':
```

```
#app.run(debug = True)
app.run()
```

HTML and CSS Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="stylesheet" href="https://codepen.io/gymratpacks/pen/VKzBEp#0">
<link href='https://fonts.googleapis.com/css?family=Nunito:400,300' rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="static/app.css">
</head>
<body>
<div>
<form action="/predict"method="post">
<div class="row">
<div class="col-md-12">
<h1><center>Loan Prediction Machine Learning</center> </h1>
<title><center>Loan Prediction Machine Learning </center></title>
<legend><span class="number"></span><center>Basic Info</center></legend>
<label>Gender:</label>
<input type="radio" id="Male" value="Male" name="Gender"><label for="Male"
class="light">Male</label>
<input type="radio" id="Female" value="Female" name="Gender"><label for="Female"
class="light">Female</label><br>
<br>
<br>
<label for="Married history">Married:</label>
```

```
<select id="Married history" name="Married">
<option value="Yes">Yes</option>
<option value="No">No</option>
</select>
<br>
<br>
<label for="dependents">Dependent:</label>
<select id="dependents" name="Dependents">
<option value="0">Not applicable</option>
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3 or more</option>
</select>
<br>
<br>
<label for="Education">Education level:</label>
<select id="education" name="Education">
<option value="1">Graduate</option>
<option value="0">Not graduate</option>
</select>
<br>
<br>
<label for="SelfEmployed">Employment status:</label>
<select id="selfemployed" name="Self_Employed">
<option value="0">Not self employed</option>
<option value="1">Self employed</option>
</select>
<br>
<br>
<label for="ApplicantIncome">Current monthly income:</label>
```

```
<input type="number" id="ApplicantIncome" name="ApplicantIncome">
<br>
<br>
<label for="CoapplicantIncome">Spouse's income (if not married put 0):</label>
<input type="number" id="CoapplicantIncome" name="CoapplicantIncome">
<br>
<br>
<label for="LoanAmount">Desired loan amount:</label>
<input type="number" id="LoanAmount" name="LoanAmount">
<br>
<br>
<label for="LoanTerm">Chosen loan term:</label>
<select id="LoanTerm" name="Loan_Amount_Term">
<option value="0">Loan term</option>
<option value="12">12 days</option>
<option value="36">36 days</option>
<option value="60">2 months</option>
<option value="84">84 days</option>
<option value="120">4 months</option>
<option value="180">6 months</option>
<option value="240">8 months</option>
<option value="300">10 months</option>
<option value="360">1 year</option>
<option value="480">16 months</option>
</select>
<br>
<br>
<label for="Credit history">Credit history:</label>
<select id="credit history" name="Credit_History">
<option value="1">Yes</option>
```

```
<option value="0">No</option>

</select>

<br>

<br>

<label>Geographical area:</label>

<input type="checkbox" id="Semiurban" value="0" name="Property_Area"><label class="light"
for="semiurban">Semiurban</label>

<input type="checkbox" id="urban" value="1" name="Property_Area"><label class="light"
for="urban">Urban</label>

<input type="checkbox" id="rural" value="2" name="Property_Area"><label class="light"
for="rural">Rural</label>

<br>

<br>

<label>Choose Alogrithm</label>

<br>

<input type="radio" id="KNN" value="0" name="group"><label class="light"
for="semiurban">KNN</label>

<input type="radio" id="RFC" value="1" name="group"><label class="light"
for="urban">RFC</label>

<input type="radio" id="SVM" value="2" name="group"><label class="light"
for="rural">SVM</label>

<br>

<br>

<button type="submit">SUMBIT</button>

</form>

<br>

<br>

{{ prediction_text }}

</div>

</div>

</body>

</html>
```

```
@import url(https://fonts.googleapis.com/css?family=Open+Sans);

.btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }

.btn-large { padding: 9px 14px; font-size: 15px; line-height: normal; -webkit-border-radius: 5px; -moz-border-radius: 5px; border-radius: 5px; }

.btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position: 0 -15px; -webkit-transition: background-position 0.1s linear; -moz-transition: background-position 0.1s linear; -ms-transition: background-position 0.1s linear; -o-transition: background-position 0.1s linear; transition: background-position 0.1s linear; }

.btn-primary, .btn-primary:hover { text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25); color: #ffffff; }

.btn-primary.active { color: rgba(255, 255, 255, 0.75); }

.btn-primary {

background-color: #4a77d4; background-image: -moz-linear-gradient(top, #6eb6de, #4a77d4); background-image: -ms-linear-gradient(top, #6eb6de, #4a77d4); background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#6eb6de), to(#4a77d4)); background-image: -webkit-linear-gradient(top, #6eb6de, #4a77d4); background-image: -o-linear-gradient(top, #6eb6de, #4a77d4); background-image: linear-gradient(top, #6eb6de, #4a77d4); background-repeat: repeat-x; filter: progid:dximagetransform.microsoft.gradient(startColorstr=#6eb6de, endColorstr=#4a77d4, GradientType=0);  border: 1px solid #3762bc; text-shadow: 1px 1px 1px rgba(0,0,0,0.4); box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.5); }

.btn-primary:hover, .btn-primary:active, .btn-primary.active, .btn-primary.disabled, .btn-primary[disabled] { filter: none; background-color: #4a77d4; }

.btn-block { width: 100%; display:block; }


* { -webkit-box-sizing:border-box; -moz-box-sizing:border-box; -ms-box-sizing:border-box; -o-box-sizing:border-box; box-sizing:border-box; }


html { width: 100%; height:auto; overflow:auto; }

body {

width: 100%;

height:100%;

font-family: 'Open Sans', sans-serif;

background: #092756;

color: #fff;

font-size: 18px;
```

```
text-align:center;

letter-spacing:1.2px;

background: -moz-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4)
10%,rgba(138,114,76,0) 40%),-moz-linear-gradient(top,  rgba(57,173,219,.25) 0%, rgba(42,60,87,.4)
100%), -moz-linear-gradient(-45deg,  #670d10 0%, #092756 100%);

background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4)
10%,rgba(138,114,76,0) 40%), -ms-linear-gradient(top,  rgba(57,173,219,.25) 0%,rgba(42,60,87,.4)
100%), -ms-linear-gradient(-45deg,  #670d10 0%,#092756 100%);

filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#3E1D6D',
endColorstr='#092756',GradientType=1 );

}

.login {

position: absolute;

top: 30%;

left: 47%;

margin: -160px 0 0 -140px;

width:400px;

height:1200px;

}

.login h1 { color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-spacing:1px; text-align:center; }

input {

width: 100%;

margin-bottom: 10px;

background: rgba(0,0,0,0.3);

border: none;

outline: none;

padding: 10px;

font-size: 13px;

color: #fff;

text-shadow: 1px 1px 1px rgba(0,0,0,0.3);

border: 1px solid rgba(0,0,0,0.3);

border-radius: 4px;
```

```
box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
-webkit-transition: box-shadow .5s ease;
-moz-transition: box-shadow .5s ease;
-o-transition: box-shadow .5s ease;
-ms-transition: box-shadow .5s ease;
transition: box-shadow .5s ease;
}
select{
width: 100%;
margin-bottom: 10px;
background: rgba(0,0,0,0.3);
border: none;
outline: none;
padding: 10px;
font-size: 13px;
color: #fff;
text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
border: 1px solid rgba(0,0,0,0.3);
border-radius: 4px;
box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
-webkit-transition: box-shadow .5s ease;
-moz-transition: box-shadow .5s ease;
-o-transition: box-shadow .5s ease;
-ms-transition: box-shadow .5s ease;
transition: box-shadow .5s ease;
}
input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px rgba(255,255,255,0.2);
}
#credits {
font-size: small;}
```

6.2: Screen Shots

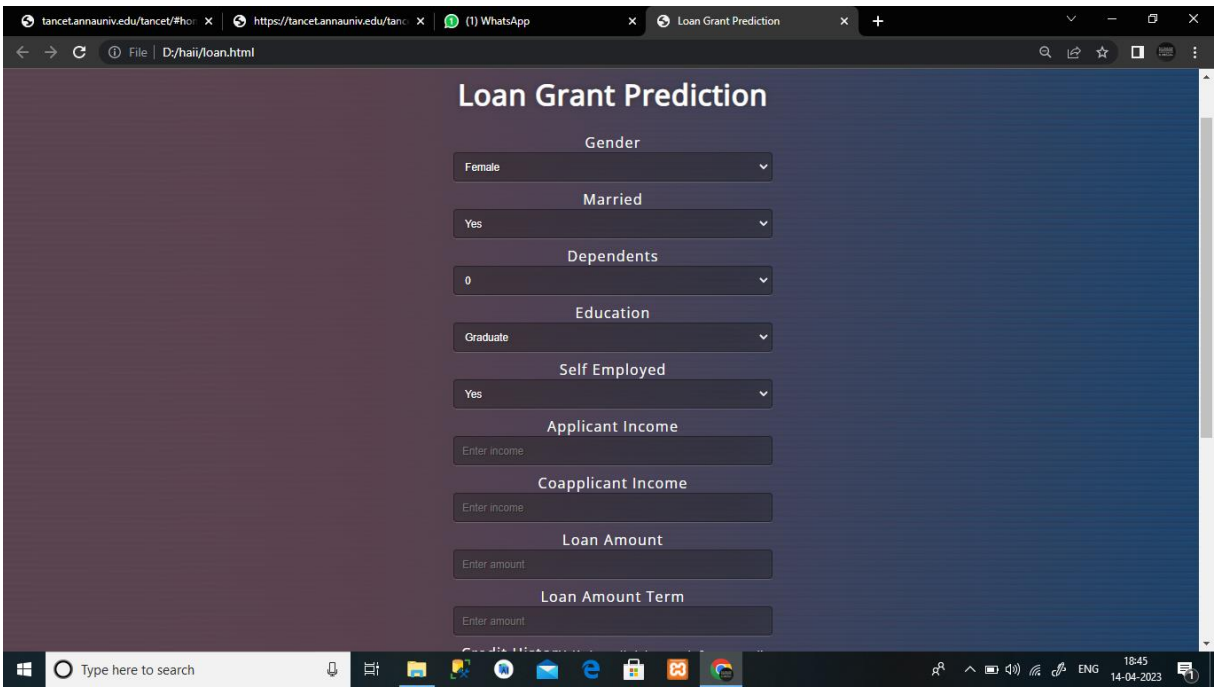

Fig 6.2.1 Data Fetching Screen

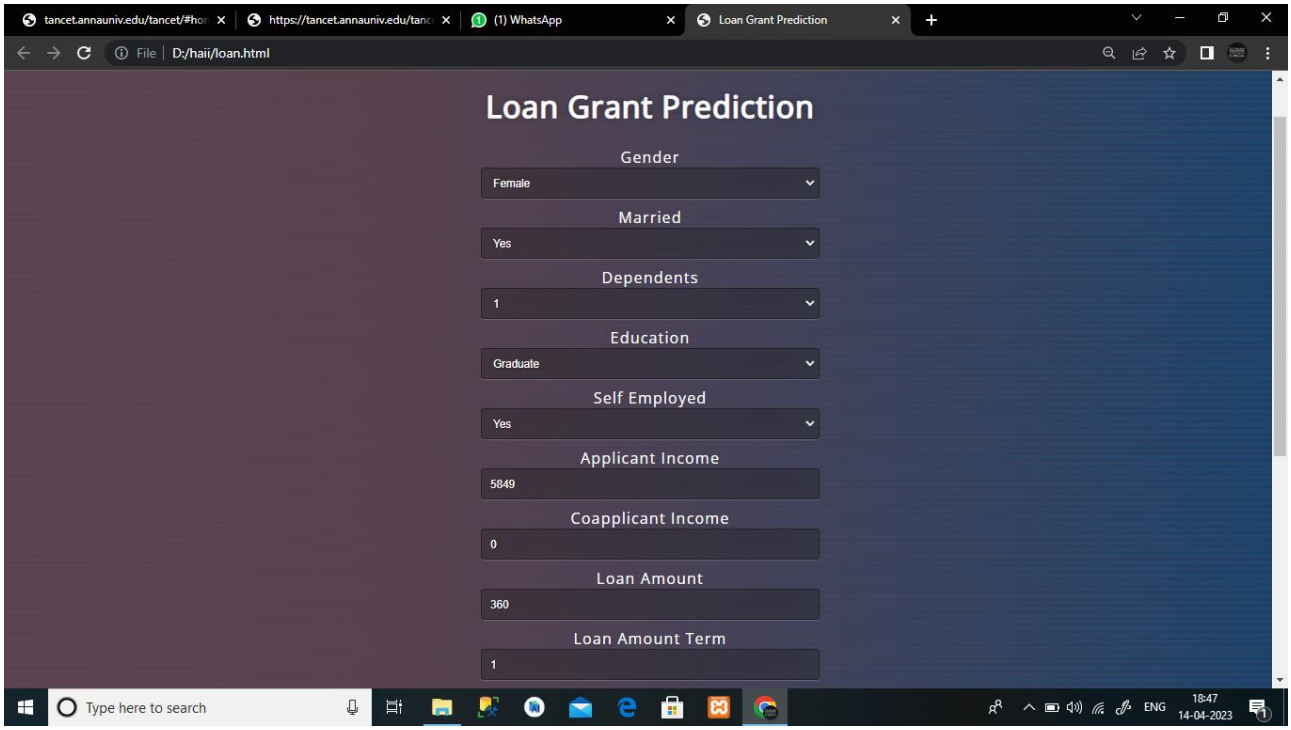

Fig 6.2.2 After Filling the Data

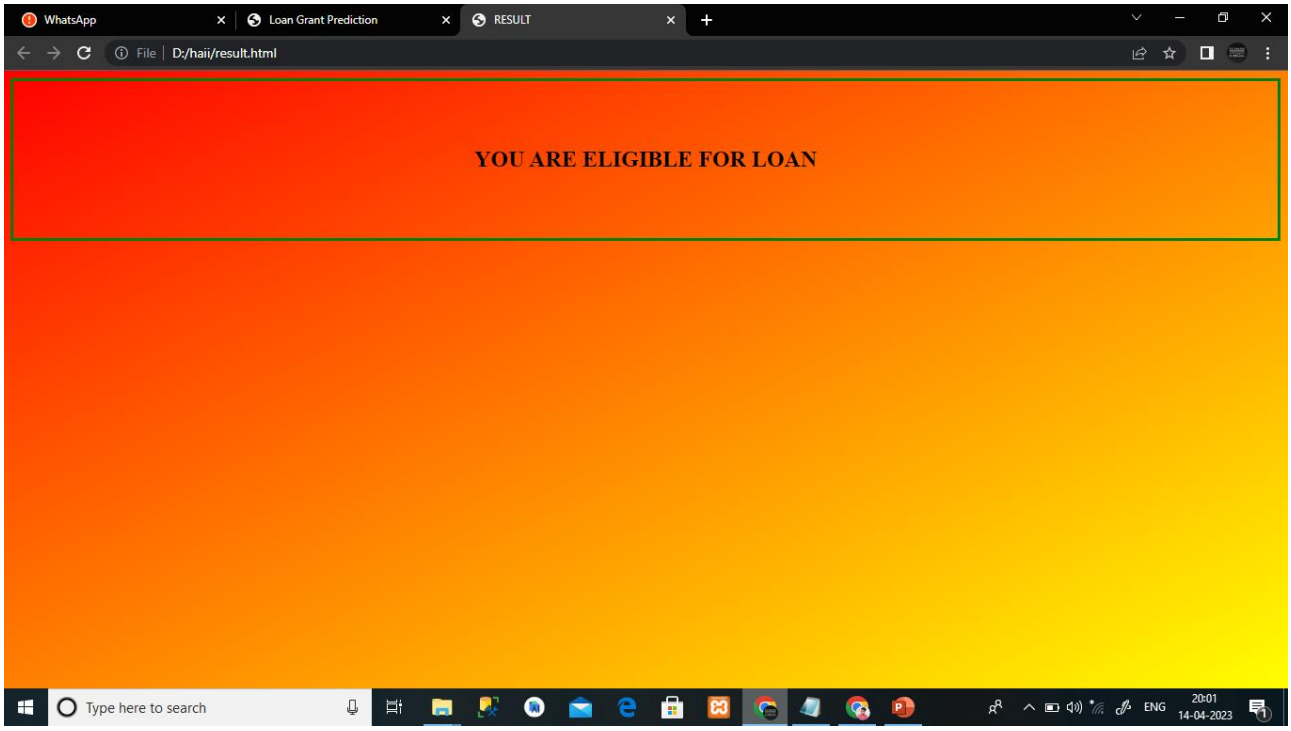

Fig 6.2.3 Output Screen

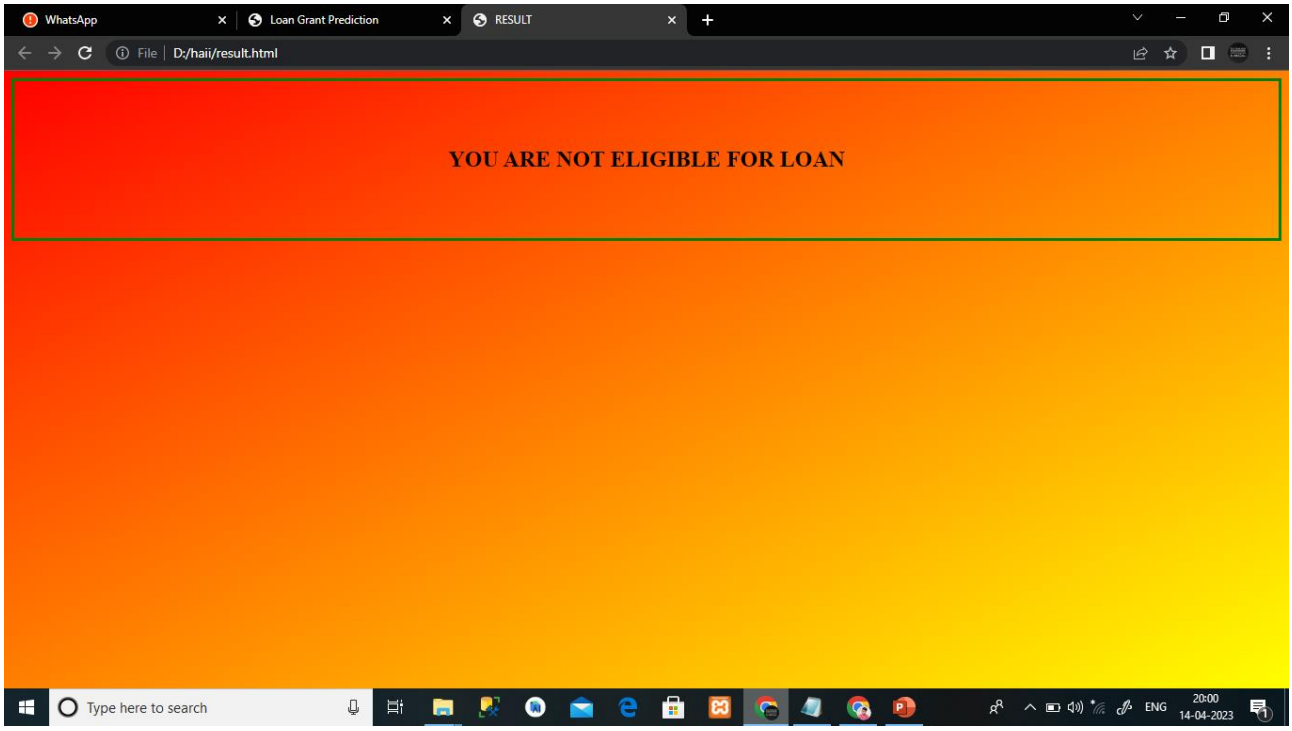

Fig 6.2.4 Output Screen

7. CONCLUSION

In conclusion, developing a loan approval prediction system using machine learning techniques is a complex but important project for financial institutions. By leveraging historical loan data and machine learning algorithms, a system can help to reduce the risk of loan defaults and make more informed lending decisions.

The project typically involves collecting and pre-processing data, performing feature engineering, training a machine learning algorithm, and evaluating the performance of the model. The final system can be deployed in a production environment, integrated with other software tools, and used to process large volumes of loan applications in real-time.

Overall, developing a loan approval prediction system using machine learning techniques requires expertise in data science, machine learning, and software development. However, the benefits of such a system can be significant, including reducing the risk of loan defaults, improving lending decisions, and ultimately increasing profitability for financial institutions.

BIBLIOGRAPHY

Book Reference:

- "Credit Risk Analytics: Measurement Techniques, Applications, and Examples in SAS" by Bart Baesens, Daniel Roesch, and Harald Scheule, 2nd edition, Wiley, United States, 2016.
- "Applied Predictive Modeling" by Max Kuhn and Kjell Johnson 1st edition, Springer, United States, 2013.
- "Data Mining: Practical Machine Learning Tools and Techniques" by Ian H. Witten, Eibe Frank, and Mark A. Hall 4th edition, Morgan Kaufmann Publishers, United States, 2016.
- "Machine Learning for Dummies" by John Paul Mueller and Luca Massaron 1st edition, Wiley, United States, 2016.
- "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems" by Aurélien Géron (2nd edition, O'Reilly Media, United States, 2019.

Web Reference:

- www.analyticsvidhya.com/blog/01/complete-tutorial-learn-data-science-python-scratch-2
- www.datacamp.com/community/tutorials/loan-approval-prediction-machine-learning
- www.towardsdatascience.com/loan-approval-prediction-using-random-forest-classifiera
- www.medium.com/loan-prediction-with-machine-learning-techniques-7d2c524b82ab