BEL TRAINING PROJECT

“**Credit Card Fraud Detection Using Machine Learning”**



**Project by:**

**Chaithra N**

**Monika Sai S**

**Syed Junaid Y**

**Preethi V**

**INDEX**

Table of contents:

* Introduction
* Objectives
* System Architecture
* Programming Codes
* Results
* Conclusion

## Introduction

## Credit card fraud has become a significant challenge in the financial industry, resulting in substantial financial losses annually. Traditional fraud detection systems often struggle to adapt to the evolving nature of fraudulent activities, leading to a high rate of false positives and undetected fraudulent transactions. This project aims to develop an effective and efficient credit card fraud detection model using machine learning techniques.

We utilise a publicly available dataset that contains transactions made by credit cardholders, with features that have been transformed to preserve privacy. The dataset is highly imbalanced, with fraudulent transactions representing a small fraction of the total. To address this, we apply various data preprocessing techniques and Several machine learning algorithms, including Logistic Regression, Decision Trees, and Neural Networks, are employed to classify transactions as fraudulent or legitimate. The performance of these models is evaluated using metrics .The findings demonstrate the effectiveness of machine learning in enhancing the accuracy and reliability of credit card fraud detection systems.

* Objectives

The primary objective of this project is to develop an accurate and efficient machine learning model for detecting fraudulent credit card transactions. Specifically, the project aims to:

* **Identify Fraudulent Transactions**:

Accurately classify transactions as either fraudulent or legitimate by leveraging machine learning algorithms, minimizing the occurrence of false negatives (fraud that goes undetected) and false positives (legitimate transactions incorrectly flagged as fraud).

* **Handle Imbalanced Data**:

Address the challenge of imbalanced datasets, where fraudulent transactions are significantly outnumbered by legitimate ones, by employing appropriate data preprocessing techniques and model evaluation metrics.

* **Optimize Model Performance:**

Evaluate and optimize various machine learning models to achieve the best balance between detection accuracy and computational efficiency, ensuring the model is suitable for deployment in real-time fraud detection systems.

* **Enhance Security**:

Contribute to the overall security of financial transactions by providing a robust tool that financial institutions can integrate into their existing systems to prevent and mitigate the impact of credit card fraud.

* **Scalability and Real-Time Detection:**

Design the model with scalability in mind, ensuring it can handle large volumes of transactions and be implemented in real-time detection environments.

* **System Architecture**
* **Data Collection**

**Source**: Transaction data is collected from various sources such as credit card networks, financial institutions, and payment gateways.

**Components**: Data includes transaction details like time, amount, merchant ID, anonymized features, and labels indicating whether the transaction is fraudulent or legitimate.

* **Data Storage**

**Database**: A robust, scalable database system is used to store raw transaction data and historical records.

* **Data Preprocessing**

**Data Cleaning**: Raw data is cleaned to remove duplicates, handle missing values, and correct inconsistencies.

**Feature Engineering**: New features are created, and existing features are transformed to improve model performance.

* **Model Training and Development**

We use **logistic regression** to classify transactions as either legitimate or fraudulent based on their features. Logistic regression is a widely used classification algorithm that models the probability of an

event occurring based on input features. The logistic regression model is trained on the training data using the LogisticRegression function from scikit-learn. The trained model is then used to predict the

target variable for the testing data.

* **Model Evaluation**

The performance of the model is evaluated using the accuracy metric, which is the fraction of correctly classified transactions. The accuracy on the training and testing data is calculated using the **accuracy\_score()** function from scikit-learn.

* **Model Deployment**

This project is implemented using **HTML** and **CSS** to provide dynamic and user interface. Then the final model is deployed in a productive environment.

* **Monitoring and Maintenance**

Model Monitoring: The performance of the deployed model is continuously monitored using metrics such as Precision, Recall, and the number of false positives/negatives.

* **User Interface Dashboard**

A user-friendly dashboard is provided for stakeholders to monitor the system’s performance, view transaction statistics, and review flagged transactions.

* **Code**
* **fraud.py**

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

import streamlit as st

import joblib

st.set\_page\_config(page\_title="Credit Card Fraud Detection", layout="wide")

@st.cache(allow\_output\_mutation=True)

def load\_data(file):

"""Load data from a CSV file."""

return pd.read\_csv(file)

def train\_model(data):

"""Train a logistic regression model on the given data."""

legit = data[data.Class == 0]

fraud = data[data.Class == 1]

legit\_sample = legit.sample(n=len(fraud), random\_state=2)

data = pd.concat([legit\_sample, fraud], axis=0)

X = data.drop(columns="Class")

y = data["Class"]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.2, stratify=y, random\_state=2

)

model = LogisticRegression(max\_iter=1000)

model.fit(X\_train, y\_train)

save\_model(model, 'fraud\_detection\_model.pkl') # Save the model

train\_acc = accuracy\_score(y\_train, model.predict(X\_train))

test\_acc = accuracy\_score(y\_test, model.predict(X\_test))

return model, train\_acc, test\_acc

def save\_model(model, filename):

"""Save the trained model to a file."""

joblib.dump(model, filename)

def load\_model(filename):

"""Load the trained model from a file."""

return joblib.load(filename)

st.title("Credit Card Fraud Detection")

file = st.file\_uploader("Upload your credit card transaction CSV file:")

if file is not None:

data = load\_data(file)

st.write("Data shape:", data.shape)

# Check if model file exists

try:

model = load\_model('fraud\_detection\_model.pkl')

except FileNotFoundError:

model, train\_acc, test\_acc = train\_model(data)

st.write("Training accuracy:", train\_acc)

st.write("Test accuracy:", test\_acc)

else:

st.write("Model loaded successfully")

st.subheader("Check a transaction")

feature\_names = data.drop(columns="Class").columns.tolist()

# Create input fields for each feature

features = {}

for feature in feature\_names:

# Create input widgets for each feature

features[feature] = st.number\_input(f"Enter value for {feature}", value=0.0)

if st.button("Predict"):

try:

# Convert input values to a DataFrame

transaction\_df = pd.DataFrame([features])

prediction = model.predict(transaction\_df)

st.write("Prediction:", "Fraudulent" if prediction[0] == 1 else "Legitimate")

except Exception as e:

st.error(f"Error processing input: {e}")

* **HTML Code**

<!DOCTYPE html>

<html lang="en">

<head>

<style>

{

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Arial', sans-serif;

background-color: #f0f4f8;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

}

.login-container {

background-color: white;

padding: 40px;

border-radius: 8px;

box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);

width: 350px;

text-align: center;

}

h1 {

margin-bottom: 20px;

font-size: 24px;

color: #333;

}

.input-group {

margin-bottom: 15px;

text-align: left;

}

label {

display: block;

margin-bottom: 5px;

font-weight: bold;

}

input {

width: 100%;

padding: 10px;

border: 1px solid #ccc;

border-radius: 4px;

font-size: 16px;

}

button {

width: 100%;

padding: 10px;

background-color: #007bff;

color: white;

border: none;

border-radius: 4px;

font-size: 18px;

cursor: pointer;

}

button:hover {

background-color: #0056b3;

}

#login-result {

margin-top: 20px;

font-size: 16px;

color: red;

}

</style>

<meta charset="UTF-8">

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

<title>Credit Card Fraud Detection - Login</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<img src="C:\Users\CHAITHRA\Downloads\credit card fraud detection\fraud.png">

<div class="login-container">

<h1>Login to Credit Card Fraud Detection</h1>

<form id="loginForm">

<div class="input-group">

<label for="username">Username</label>

<input type="text" id="username" name="username" placeholder="Enter your username" required>

</div>

<div class="input-group">

<label for="password">Password</label>

<input type="password" id="password" name="password" placeholder="Enter your password" required>

</div>

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

</form>

<div id="login-result"></div>

</div>

</body>

<script>

document.getElementById("loginForm").addEventListener("submit", function(event) {

event.preventDefault();

const username = document.getElementById("username").value;

const password = document.getElementById("password").value;

// Simple login validation (this should be replaced by real authentication)

const validUsername = "user123";

const validPassword = "password123";

const resultElement = document.getElementById("login-result");

if (username === validUsername && password === validPassword) {

// Redirect to the main fraud detection page

window.location.href = "http://localhost:8501/";

} else {

resultElement.innerHTML = "Invalid username or password.";

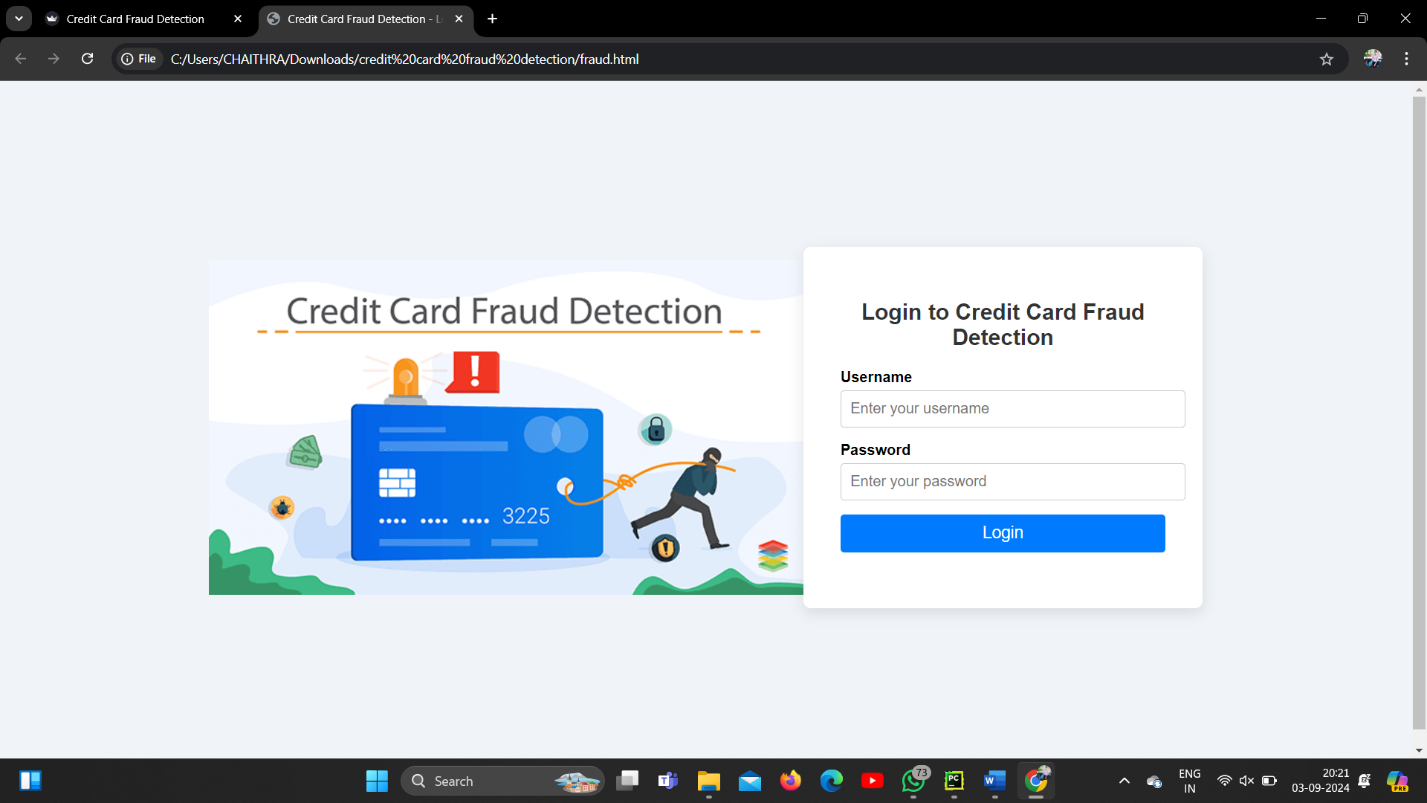
}

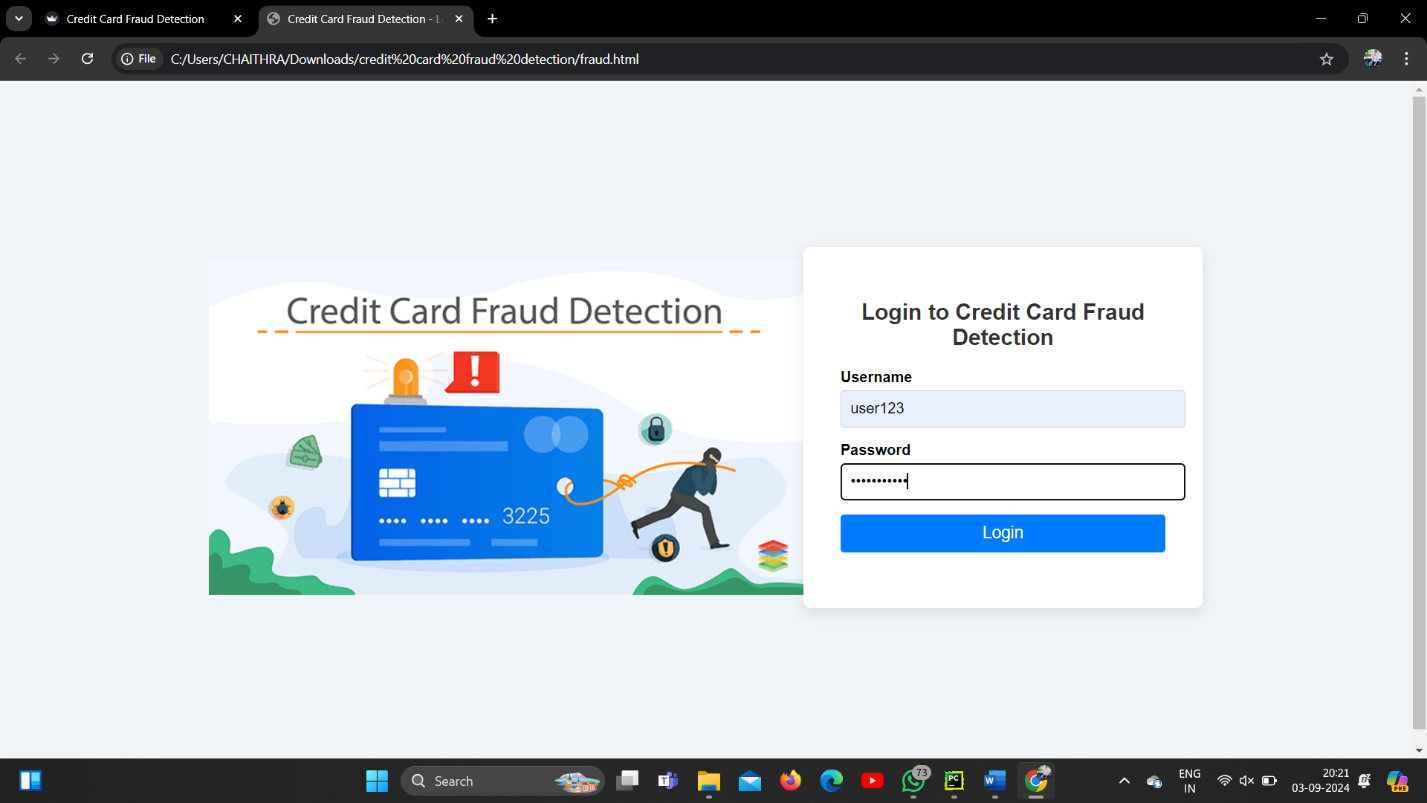
});

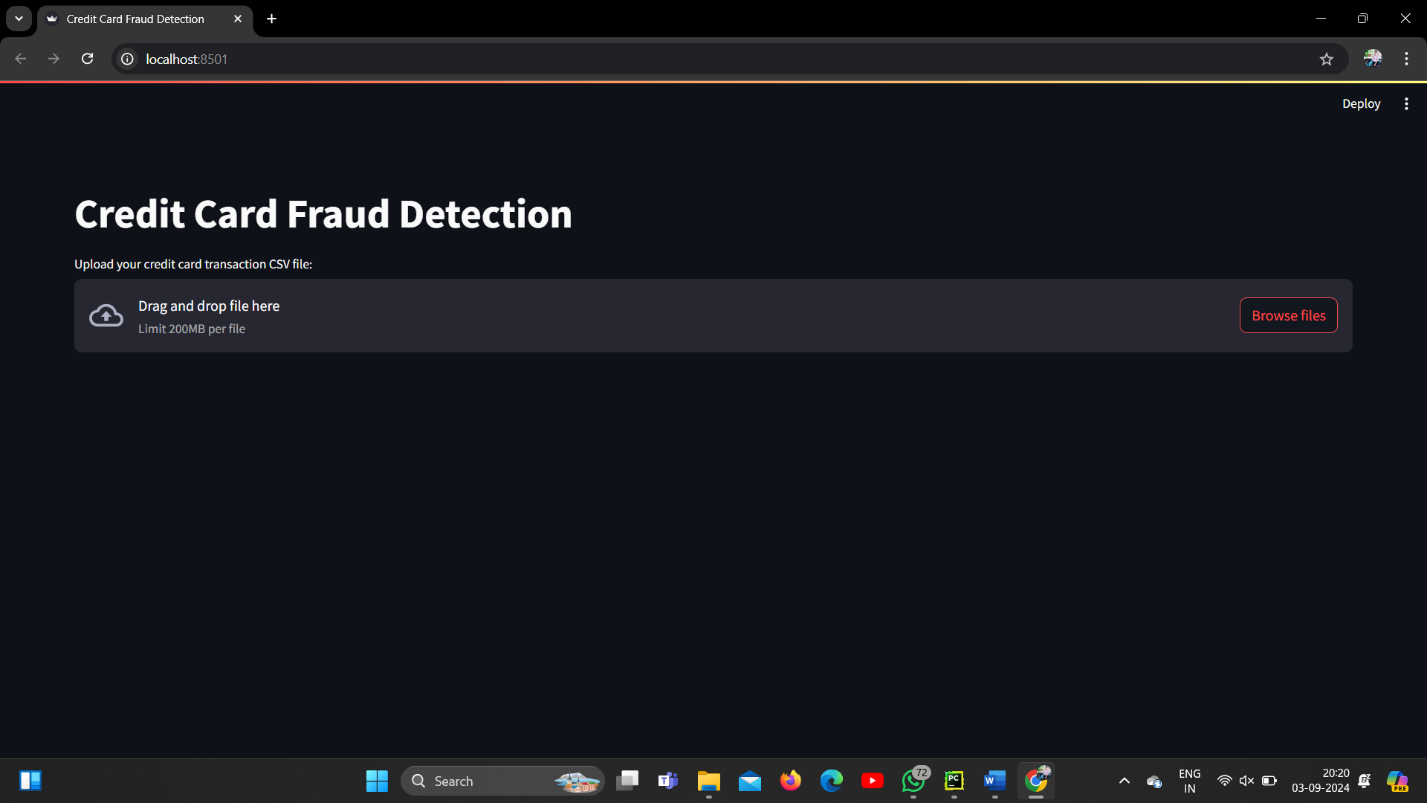
</script>

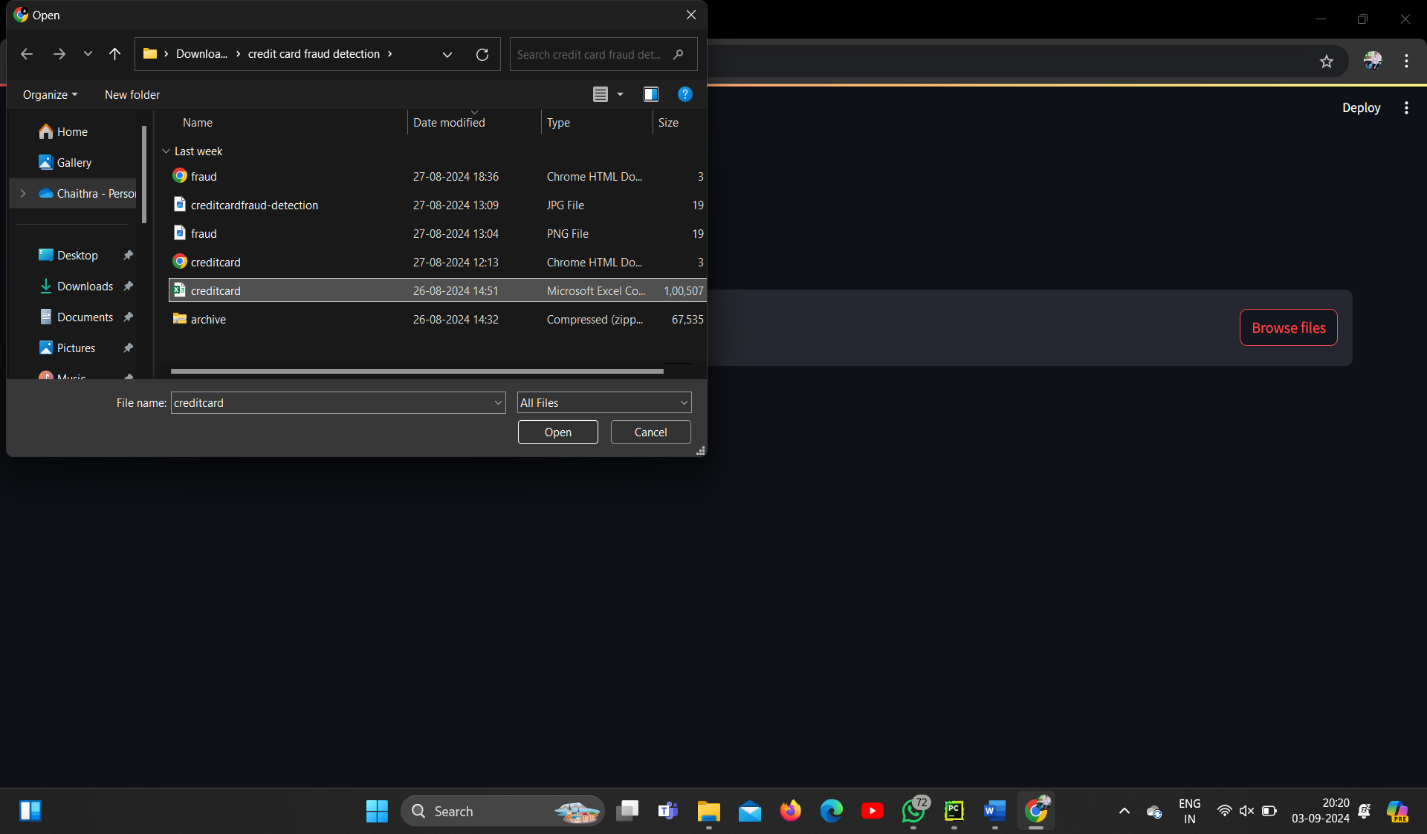
</html>

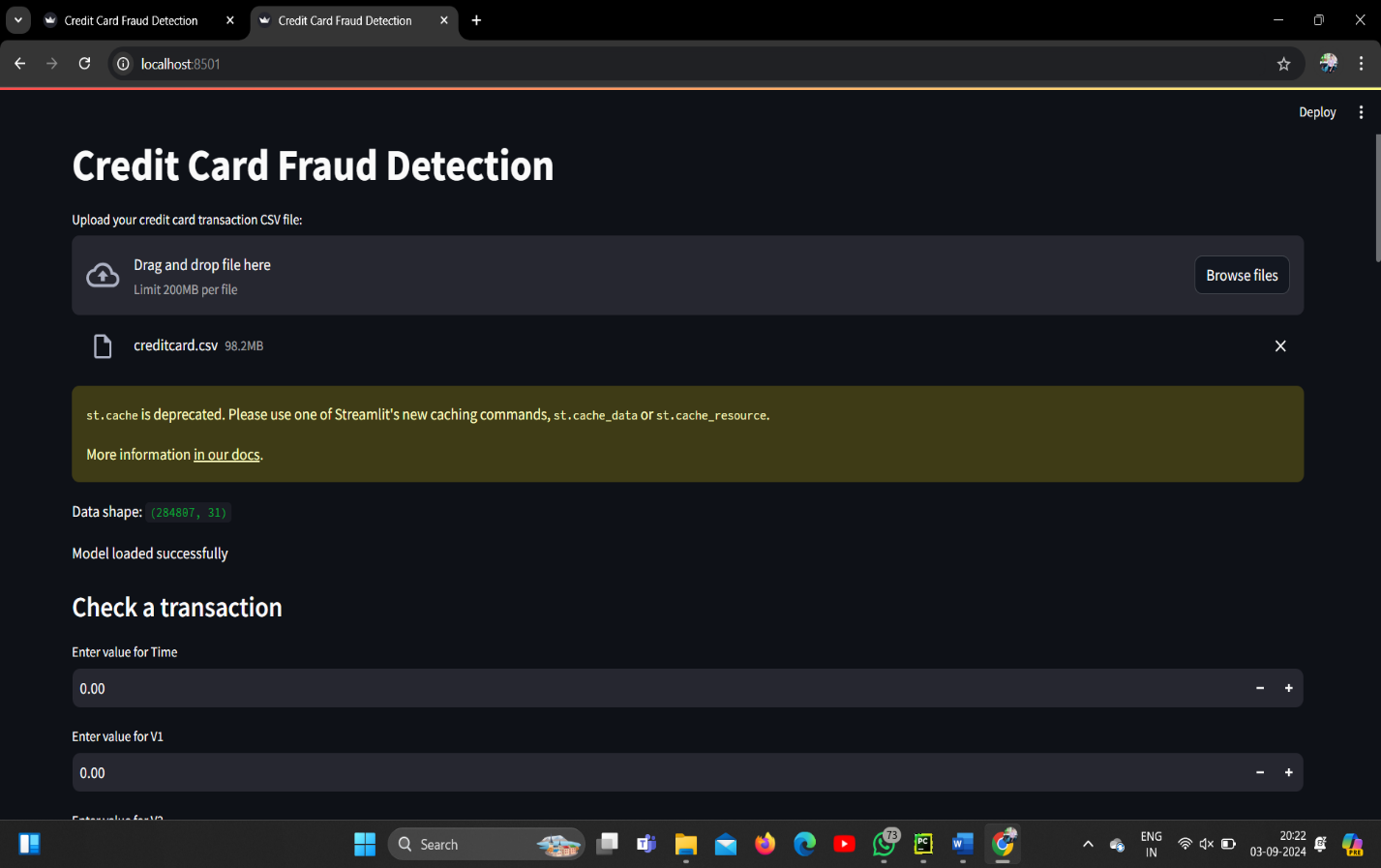
* **Results**

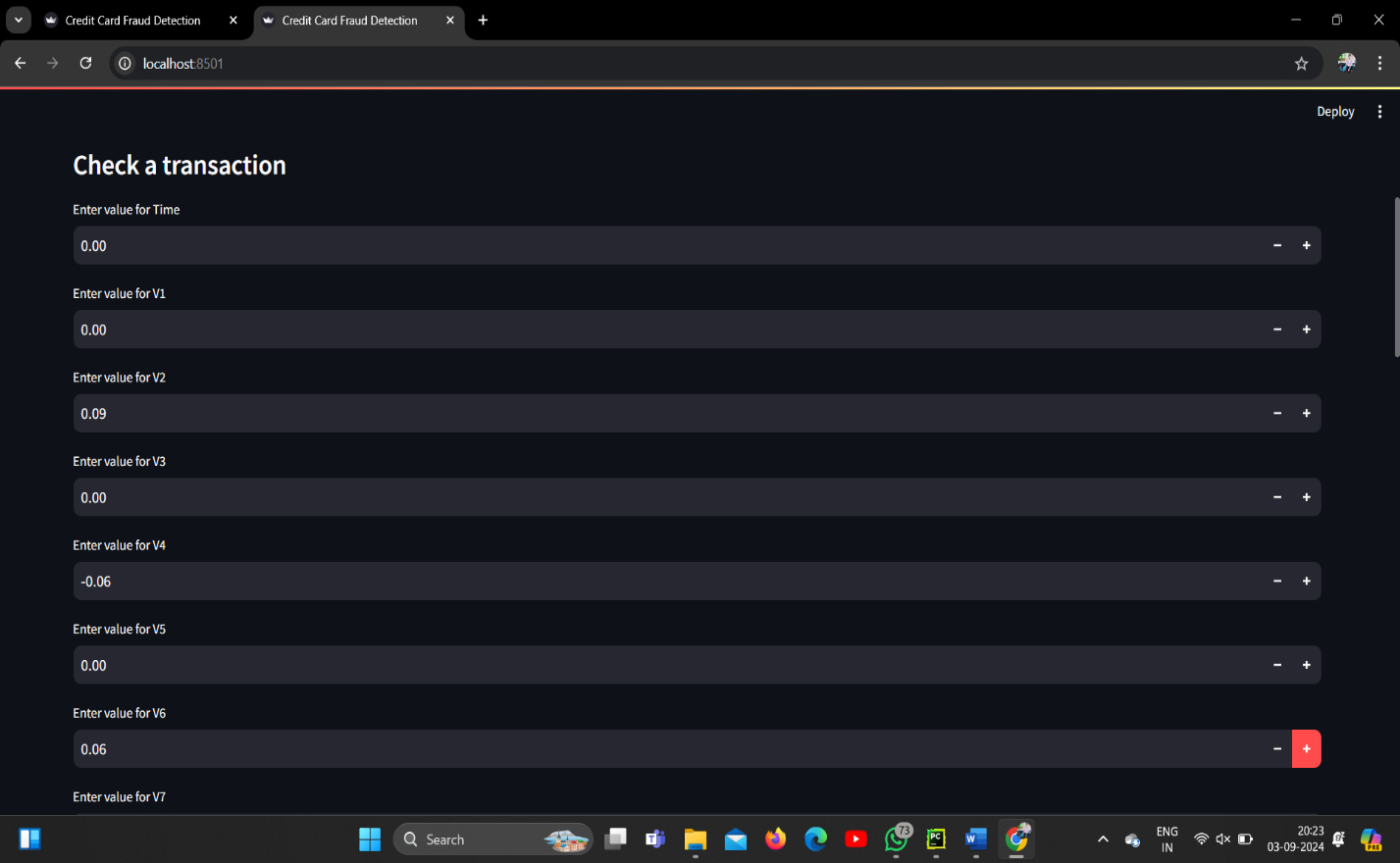


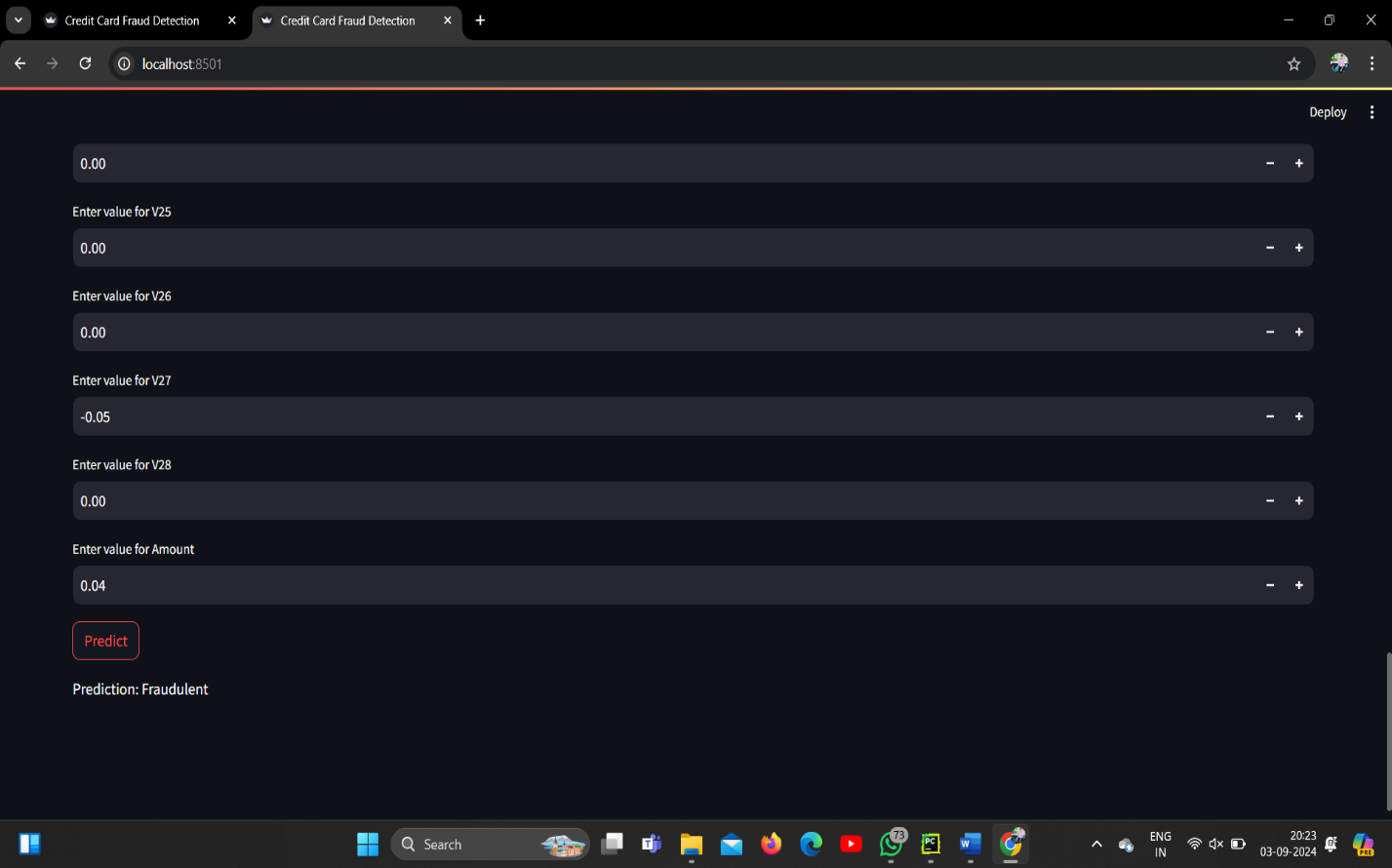


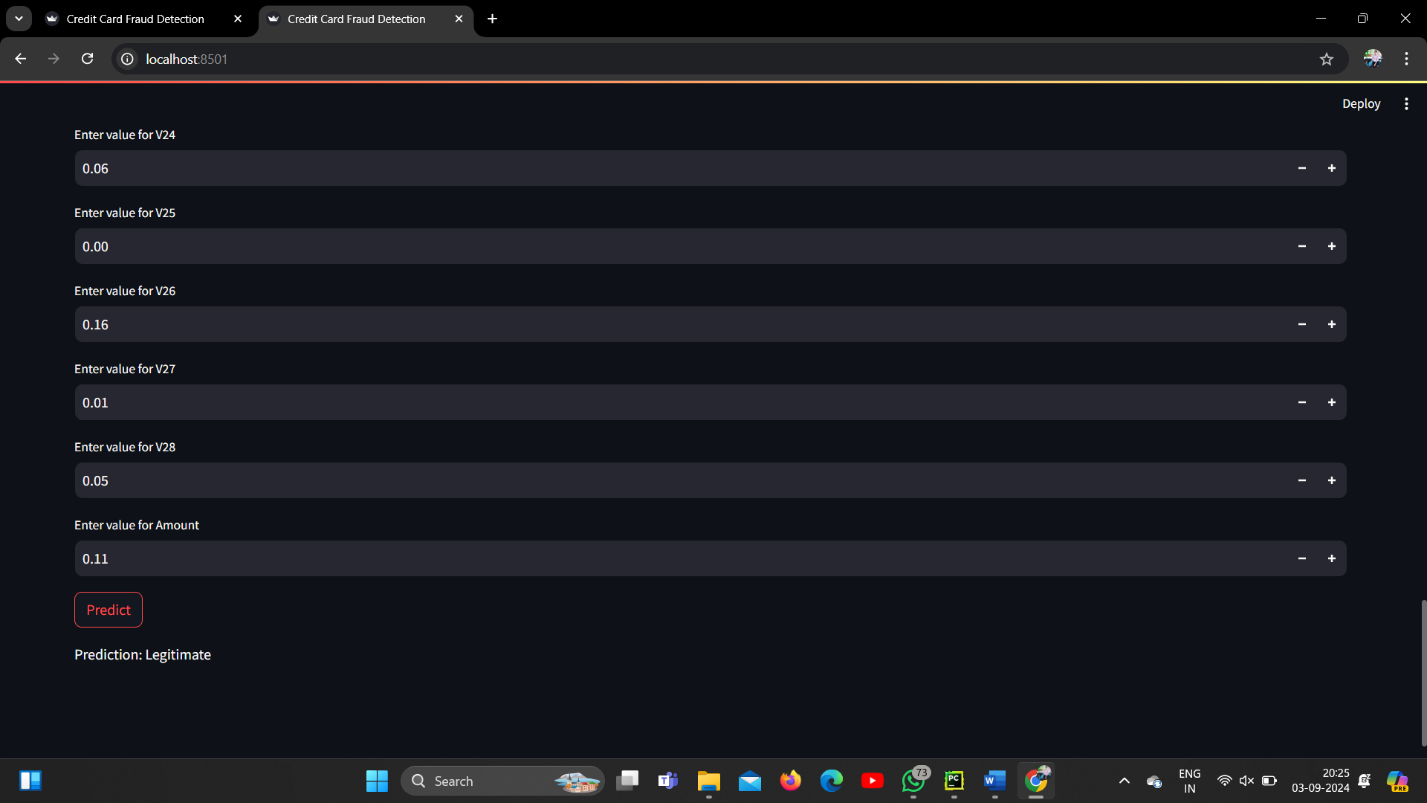












* **Conclusion**

The successful detection of credit card fraud is crucial in safeguarding financial transactions and maintaining trust in the financial industry. This project demonstrates the effectiveness of machine learning techniques in identifying fraudulent transactions with high accuracy and efficiency. By employing various machine learning algorithms and addressing the inherent challenges of imbalanced data, we were able to develop a robust model that minimizes both false positives and false negatives.

In conclusion, machine learning provides a powerful tool for enhancing the security of credit card transactions. Future work could focus on incorporating additional data sources, exploring advanced machine learning techniques like deep learning, and refining the model’s ability to adapt to new fraud patterns. By doing so, we can further improve the accuracy and reliability of fraud detection systems, ensuring greater protection for consumers and financial institutions alike.