FRAUD TRANSACTION DETECTION SYSTEM USING MACHINE LEARNING

A Project report submitted in partial fulfillment
for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING (CYBER SECURITY)

Submitted by

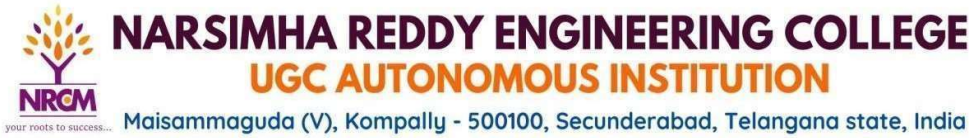
ADUSUMALLI KIRAN KUMAR	**21X05A6202**
MASKU SAI PRIYA REDDY	**20X01A6207**
HOTARKAR RAVI TEJA	**20X01A6202**
KEMIDI VISHAL	**20X01A6203**

Under the esteemed guidance of

Dr. N. Srinivasa Rao Ph.D

Associate Professor

Department of Computer Science and Engineering (Cyber Security)



www.nrcmec.org
(2020 – 2024)

NARSIMHA REDDY ENGINEERING COLLEGE
UGC AUTONOMOUS INSTITUTION
Maisammaguda (V), Kompally - 500100, Secunderabad, Telangana state, India

Department of Computer Science and Engineering (Cyber Security)

your roots to success...

CERTIFICATE

This is to certify that the project report entitled **" Fraud Transaction Detection System using Machine Learning"** is the bonafide work done by **Adusumalli Kiran Kumar, 21X05A6202, Masku Sai Priya Reddy, 20X01A6207, Hotarkar Ravi Teja, 20X01A6202, Kemidi Vishal, 20X01A6203,** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Department of Computer Science and Engineering (Cyber Security)**, from Jawaharlal Nehru Technological University Hyderabad, during the year 2020-2024.

Project Guide

Dr. N. Srinivasa Rao Ph.D,

Associate Professor,

Department of CSE (Cyber Security),

NARSIMHA REDDY ENGINEERING COLLEGE,

Maisammaguda, Kompally, Secunderabad.

Head of the Department

Dr. M. Parthasaradhi Ph.D,

Associate Professor & Head,

Department of CSE (Cyber Security),

NARSIMHA REDDY ENGINEERING COLLEGE,

Maisammaguda, Kompally, Secunderabad.

Submitted for the viva-voice examination held on:

External Examiner

NARSIMHA REDDY ENGINEERING COLLEGE
UGC AUTONOMOUS INSTITUTION
Maisammaguda (V), Kompally - 500100, Secunderabad, Telangana state, India

Department of Computer Science and Engineering (Cyber Security)

your roots to success...

DECLARATION

We **Adusumalli Kiran Kumar, 21X05A6202, Masku Sai Priya Reddy, 20X01A6207, Hotarkar Ravi Teja, 20X01A6202, Kemidi Vishal, 20X01A6203,** hereby declare that the Project Work entitled **"Fraud Transaction Detection System using Machine Learning"** done under the esteemed guidance of **Dr. N. Srinivasa Rao Ph.D,** Associate Professor, Department of Computer Science and Engineering (Cyber Security) and is submitted in partial fulfillment of the requirements for the award of the Bachelor degree in Computer Science and Engineering (Cyber Security).

Date:

Place: Maisammaguda, Kompally, Hyderabad.

Adusumalli Kiran Kumar (21X05A6202)
Masku Sai Priya Reddy (20X01A6207)
Hotarkar Ravi Teja (20X01A6202)
Kemidi Vishal (20X01A6203)

ACKNOWLEDGEMENT

First, we are thankful to our guide **Dr. N. Srinivasa Rao** for his valuable guidance and encouragement. His helping attitude and suggestions have helped us in the successful completion of the project. Successful completion of any project cannot be done without proper support and encouragement.

We sincerely thank the **Sri. J. Narsimha Reddy- Chairman, Mr. J. Trishul Reddy - Secretary, Mr. J. Thrilok Reddy – Treasurer, Dr. A. Mohan Babu, Director, NRCM** for providing all the necessary facilities during the course of study.

We highly indebted to Principal **Dr. R. Lokanadham** for the facilities provided to accomplish this project.

We would like to thank my Dean-CSE **Dr. B. Rama Subba Reddy** for his constructive criticism throughout my project.

We wish to convey my special thanks to **Dr. M. Parthasaradhi**, Head of Computer Science and Engineering in Narsimha Reddy Engineering College, for giving the required information in doing my project.

We would like to thank Project coordinator, **Mr. P. Kishore Kumar (Ph.D)** for their support and advices to get and complete project work for his guidance and regular schedules. We extremely great full to my department staff members who helped me in successful completion of this project.

We would like to thank our parents and friends, who have the greatest contributions in all our achievements, for the great care and blessings in making us successful in all our endeavors.

Adusumalli Kiran Kumar	**21X05A6202**
Masku Sai Priya Reddy	**20X01A6207**
Hotarkar Ravi Teja	**20X01A6202**
Kemidi Vishal	**20X01A6203**

ABSTRACT

In the fast-paced world of financial transactions, real-time fraud detection is paramount. This project explores how Finance Max can leverage machine learning to develop a robust fraud detection system. From data collection and preprocessing to model training and deployment, this end-to-end project outlines the key steps and considerations in building an effective fraud detection system that safeguards financial operations by identifying and mitigating fraudulent transactions in real-time.

KEYWORDS: Fraudulent transaction; real time detection; finance max; anomaly detection

Contents

LIST OF FIGURES

ABBREVIATIONS

AUC	Area Under the Receiver Operating Characteristic
CSS	Cascading Style Sheets
CSV	Comma Separated Value
DFD	Data Flow Diagram
GUI	Graphical User Interface
GDI	Get Domain Info
HTTP	Hyper Text Transfer Protocol
HTML	Hyper Text Markup language
JS	JavaScript
JSON	JavaScript Object Notation
ML	Machine Learning
RDBMS	Relational Database Management System
RNN	Recurrent Neural Networks
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine
UML	Unified Modelling Language
URL	Uniform Resource Locater
WSRM	Web Services Reliable Messaging

CHAPTER 1 – INTRODUCTION

1.1 INTRODUCTION

In the fast-paced landscape of financial transactions, real-time fraud detection is not just a necessity; it's a critical imperative. Finance Max, a forward-thinking financial institution, recognizes the urgency of safeguarding its operations against fraudulent activities. This project aims to explore how Finance Max can leverage machine learning (ML) to construct a robust fraud detection system that operates seamlessly in real time.

1.1.1 KEY STEPS IN BUILDING AN EFFECTIVE FRAUD DETECTION SYSTEM

1. **Data Collection and Preprocessing**:
    - Finance Max begins by accumulating historical transaction data. This dataset includes both legitimate transactions and fraudulent ones.
    - The data undergoes rigorous preprocessing, including handling missing values, normalizing features, and encoding categorical variables.
2. **Feature Engineering**:
    - Relevant features are extracted from the transaction data. These features might include transaction amount, location, time of day, user behavior patterns, and more.
    - Additionally, new features are created to capture unusual behavior or anomalies.
3. **Model Selection and Training**:
    - Finance Max explores various ML algorithms suitable for fraud detection. Some popular choices include:
        - **Support Vector Machine (SVM)**: Effective for binary classification tasks.
        - **Random Forests**: An ensemble of decision trees.
        - **Logistic Regression: It** is a valuable tool in fraud detection**.**
    - The choice of model depends on factors such as dataset size, complexity, and interpretability requirements.
    - The selected model is trained using historical data.
4. **Real-Time Scoring and Deployment**:
    - The trained ML model is deployed in a real-time scoring environment.
    - As new transactions arrive, the model instantly evaluates them and assigns a fraud probability score.

5. **Threshold Setting and Alerting**:
    - Finance Max defines a threshold for fraud probability scores.
    - Transactions with scores above this threshold are flagged as potentially fraudulent.
    - Alerts are triggered for further investigation.
6. **Feedback Loop and Model Updates**:
    - Continuous monitoring is crucial. Finance Max collects feedback on flagged transactions (both true positives and false positives).
    - This feedback informs periodic retraining and fine-tuning of the ML model.
7. **Evaluation Metrics**:
    - The system's effectiveness is assessed using metrics such as:
        - **Precision**: The proportion of correctly identified fraud cases among all flagged cases.
        - **Recall**: The proportion of actual fraud cases detected by the system.
        - **F1-score**: The harmonic mean of precision and recall.
        - **Area Under the Receiver Operating Characteristic (ROC-AUC)**: Measures overall performance.
8. **Adaptive Learning and Ongoing Vigilance**:
    - Finance Max implements adaptive learning mechanisms.
    - As fraudsters evolve their tactics, the ML model adapts by learning from new patterns.

CHAPTER 2 – LITERATURE SURVEY

Fraud detection in financial transactions is a critical concern for institutions like Finance Max. With the advancement of machine learning techniques, real-time fraud detection has become more achievable. This literature survey aims to explore the existing research and methodologies related to using machine learning for building effective fraud detection systems.

1. Machine Learning Techniques for Fraud Detection

Numerous studies have demonstrated the efficacy of machine learning algorithms in detecting fraudulent transactions. Supervised learning algorithms such as logistic regression, ensemble methods like random forests and gradient boosting have shown promising results in binary classification tasks. Unsupervised learning techniques such as clustering and anomaly detection methods like One-Class SVM have been effective in identifying unusual patterns indicative of fraud.

2. Feature Engineering and Selection

Feature engineering plays a crucial role in developing accurate fraud detection models. Relevant features such as transaction amount, frequency, location, device information, and user behavior need to be carefully selected and engineered. Techniques like PCA and feature scaling help in extracting meaningful information from raw data, enhancing the performance of machine learning models.

3. Handling Imbalanced Data

Fraudulent transactions are often rare compared to legitimate ones, leading to imbalanced datasets. Various approaches, including oversampling, undersampling, and synthetic data generation techniques like SMOTE, have been proposed to address class imbalance issues. Ensemble methods like XGBoost are also effective in handling imbalanced data and improving model performance.

4. Model Evaluation and Validation

Proper evaluation and validation are essential for assessing the performance of fraud detection models. Metrics such as precision, recall, F1-score, and AUC-ROC are commonly used to evaluate model effectiveness. Cross-validation, stratified sampling, and temporal validation techniques ensure the robustness and generalization capability of the model across different datasets and time periods.

5. Real-Time Implementation and Scalability

Deploying a real-time fraud detection system requires efficient model inference and scalability. Technologies such as stream processing frameworks enable the processing of high-velocity transaction data in real time. Model deployment frameworks like TensorFlow Serving and Kubernetes facilitate the deployment and monitoring of machine learning models in production environments, ensuring scalability and reliability.

6. Challenges and Future Directions

Despite the advancements in machine learning-based fraud detection, several challenges remain, including the need for interpretable models, handling evolving fraud patterns, and ensuring regulatory compliance. Future research directions include exploring deep learning approaches, integrating external data sources, and developing hybrid models combining supervised and unsupervised techniques for enhanced fraud detection capabilities.

CHAPTER 3 – SYSTEM ANALYSIS

3.1 SYSTEM STUDY

In this critical phase, we embark on a thorough exploration of the existing system, aiming to understand its intricacies, limitations, and the compelling need for an innovative solution. Here are the key steps involved:

1. Identifying Stakeholders

We recognize the various stakeholders who play pivotal roles in the fraud detection ecosystem:

- **Finance Max:** The financial institution seeking to safeguard its assets and maintain customer trust.
- **Customers:** The end-users whose transactions are at the heart of this system.
- **Fraud Analysts:** The vigilant experts responsible for investigating suspicious activities.
- **Regulatory Bodies:** Entities enforcing compliance and ensuring adherence to legal standards.

2. Gathering Requirements

Our primary objective is to detect fraudulent transactions in real time. We must understand the specific needs and expectations of Finance Max:

- **Real-Time Detection:** Finance Max requires immediate alerts when potentially fraudulent transactions occur.
- **Accuracy:** Minimizing false positives (flagging legitimate transactions as fraudulent) is crucial.
- **Scalability:** The system must handle a large volume of transactions efficiently.
- **Adaptability:** It should evolve alongside emerging fraud tactics.

3. Analyzing Existing Processes

We scrutinize the current state of affairs within Finance Max's fraud detection framework:

- **Manual Intervention:** Human analysts review transactions, which introduces latency.
- **Threshold-Based Rules:** Fixed thresholds trigger alerts, but they lack flexibility.
- **Batch Processing:** The existing system operates in batch mode, delaying detection.

3.2 EXISTING SYSTEM

The current approach relies on traditional methods, which have their limitations:

Neural Networks: Neural networks play a significant role in fraud detection, leveraging their ability to learn complex patterns from data.

Decision Trees: A decision tree is a hierarchical structure that recursively splits the dataset into smaller subsets based on the most relevant features

Navie Bayes: Naive Bayes is based on Bayes' Theorem and assumes that features are conditionally independent given the class label.

3.3 DISADVANTAGES OF EXISTING SYSTEM

The drawbacks of the current system are evident:

- **Latency:** Real-time detection remains elusive due to manual processes and batch processing.
- **False Alarms:** Rule-based triggers generate false positives, burdening analysts.
- **Missed Patterns:** Subtle fraud patterns often slip through the cracks.
- **Scalability Challenges:** Manual review doesn't scale well with transaction volumes.

3.4 PROPOSED SYSTEM

Our proposed system leverages machine learning (ML) to revolutionize fraud detection:

Random Forest: Random Forest is an ensemble learning method that constructs multiple decision trees during training.

Linear Regression: Linear Regression is a statistical method used for binary classification. It predicts the probability of an event occurring based on input features.

SVM: Support Vector Machine (SVM) is a powerful machine learning algorithm widely used for fraud detection, including credit card fraud

3.5 ADVANTAGES OF PROPOSED SYSTEM

The benefits of our proposed solution are compelling:

- **Real-Time Detection:** Immediate alerts for suspicious transactions.
- **Reduced False Positives:** ML models learn from data, minimizing unnecessary alarms.
- **Scalability:** Efficiently handle high transaction volumes.

3.6 SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS:

- System : Intel i3
- Hard Disk : 1 TB Min or 256GB SSD
- Monitor : 14’ Color Monitor.
- Ram : 8GB Min

SOFTWARE REQUIREMENTS:

- Operating System : Windows 11.
- Coding Language : Python with Machine Learning
- Front-End : HTML, CSS, JS
- Data Base : MongoDB.

CHAPTER 4 - IMPLEMENTATION

4.1 MODULES

- Users
- Data Preprocessing
- SVM (Support Vector Machine)

MODULES DESCRIPTION:

- **User:**

The User can register first. While registering he required a valid username and email and for further communications. Once the user register then admin can activate the user. Once admin activated the user then user can login into our system. User can upload the dataset based on our dataset column matched. For algorithm execution data must be in int or float format. Here we took Transaction Data dataset for testing purpose. User can also add the new data for existing dataset based on our application. User can click the Data Preparations in the web page so that the data cleaning process will be starts. The cleaned data and its required graph will be displayed in the Application.

- **Data Preprocessing:**

A dataset can be viewed as a collection of data objects, which are often also called as a Sentiment tweet like positive, negative and neutral. Data objects are described by a number of features that capture the basic characteristics of an object, such as the mass of a physical object or the time at which an event occurred, etc. Features are often called as variables, characteristics, fields, attributes, or dimensions.

- **SVM (Support Vector Machine):**

Based on the split criterion, the cleaned data is split into 80% training and 20% test, then the dataset is subjected to one machine learning algorithm such as Support Vector Machine (SVM) by fine tuning auto encoding models. Thus, we have analyzed the results of our experiment and methodology using the contextual information.

4.2 SOURCE CODE

4.2.1 Frontend

- **Python Code:**
- **File Name:** app.py

```
import csv
from flask import Flask, redirect, request, jsonify, render_template, send_file, url_for
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from pymongo import MongoClient
import pymongo
app = Flask(__name__, static_url_path='/static')
# Load the dataset
df = pd.read_csv("C:/Users/KIRAN KUMAR/Downloads/SNO_23-20240324T030541Z-001/SNO_23/23finance/transactiondata.csv")
# Preprocessing
# Drop unnecessary columns
df = df.drop('TransactionID', axis=1)
# Split the 'Timestamp' column into minutes and seconds
df[['Minutes', 'Seconds']] = "57:31.1".split(':')
# Convert minutes and seconds to float
df['Minutes'] = df['Minutes'].astype(float)
df['Seconds'] = df['Seconds'].astype(float)
# Convert the time to seconds
df['Timestamp_seconds'] = df['Minutes'] * 60 + df['Seconds']
# Drop the original 'Timestamp' column and the intermediate columns
df = df.drop(['Timestamp', 'Minutes', 'Seconds'], axis=1)
# Encode categorical variables using one-hot encoding
encoder = OneHotEncoder()
df_encoded = pd.DataFrame(encoder.fit_transform(df[['TransactionLocation']]).toarray(), columns=encoder.categories_[0])
```

```
# Concatenate the encoded features with the original dataframe
df = pd.concat([df.drop(['TransactionLocation'], axis=1), df_encoded], axis=1)
# Splitting the data into features (X) and target variable (y)
X = df.drop('IsFraud', axis=1)
y = df['IsFraud']
# Splitting the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Initialize the Random Forest classifier
random_forest = RandomForestClassifier(random_state=42)
# Train the Random Forest classifier
random_forest.fit(X_train, y_train)
@app.route("/")
def index():
    return render_template('login.html')
@app.route("/menu" , methods=['GET'])
def menu():
    return render_template('menus.html')
@app.route("/apps" , methods=['GET'])
def apps():
    return render_template('index.html')
@app.route('/display', methods=['GET'])
def display():
    return render_template('dataset.html')
@app.route('/report', methods=['GET'])
def report():
    return send_file('prdeclaration_merged.pdf')
@app.route('/welcome', methods=['GET'])
def welcome():
    return render_template('welcome.html')
@app.route('/uses', methods=['GET'])
def uses():
    return render_template('uses.html')
@app.route('/about', methods=['GET'])
def about():
```

```
    return render_template('about.html')
@app.route('/backend', methods=['GET'])
def backend():
    return render_template('Finance.html')
@app.route('/login', methods=['POST'])
def login():
    username=request.form.get("username")
    password=request.form.get("password")
    uri = "mongodb://localhost:27017"
    db_name = "USER_DETAILS"
    collection = "SIGNUP"
    client = MongoClient(uri)
    db = client[db_name]
    collection = db[collection]
    query={"username": username, "password" : password}
    result=collection.find_one(query)
    print(query, result)
    if result:
        return jsonify({"stat": True})
    else:
       return jsonify({"stat" : False})
@app.route('/signup', methods=['POST'])
def signup():
    username=request.form.get("susername")
    password=request.form.get("spassword")
    email=request.form.get("semail")
    uri = "mongodb://localhost:27017"
    db_name = "USER_DETAILS"
    collection_name = "SIGNUP"
    data=[{"username": username, "password" : password, "Email" : email}]
    client = MongoClient(uri)
    db = client[db_name]
    collection = db[collection_name]
    result = collection.insert_many(data)
```

```
    client.close()
    return jsonify({"stat": True})
@app.route("/predict", methods=["POST"])
def predict():
    try:
        amount = float(request.form.get("amount"))
        location = request.form.get("location")
        timestamp = request.form.get("timestamp")
            data = pd.DataFrame({"Amount": [amount], "TransactionLocation": [location], "Timestamp": [timestamp]})
        data[['Minutes', 'Seconds']] = "57:31.1".split(':')
        data['Minutes'] = data['Minutes'].astype(float)
        data['Seconds'] = data['Seconds'].astype(float)
        data['Timestamp_seconds'] = data['Minutes'] * 60 + data['Seconds']
        data = data.drop(['Minutes', 'Seconds', 'Timestamp'], axis=1)
                encoded_data = pd.DataFrame(encoder.transform(data[['TransactionLocation']]).toarray(), columns=encoder.categories_[0])
        data = pd.concat([data.drop(['TransactionLocation'], axis=1), encoded_data], axis=1)
        prediction = random_forest.predict(data)
        return jsonify({"prediction": int(prediction[0])})
    except Exception as e:
        print(e)
        return jsonify({"error": str(e)})
if __name__ == "__main__":
    app.run(debug=True)
```

- **HTML & CSS Code:**
- **File Name:** menus.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>MAJOR PROJECT</title>
   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.1.3/css/bootstrap.min.css">
   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-social/5.1.1/bootstrap-social.min.css">
   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.11.2/css/all.min.css">
   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.1.3/css/bootstrap.min.css">
      /*-----------------------------------------------
         Welcome Page
      -----------------------------------------------*/
      .author-image-large {
      position: absolute;
      right: 0;
      top: 0; }
      .author-image-large img {
         height: -webkit-calc(100vh - 4px);
         height: -moz-calc(100vh - 4px);
```

```
    height: calc(100vh - 4px); }
@media (min-width: 1200px)
{
.col-lg-offset-2 {
    margin-left: 16.66666667%;
}
}
@media (min-width: 1200px)
{
.col-lg-8 {
    width: 66.66666667%;
}
}
.hexagon-item:first-child {
    margin-left: 0;
}
.pt-table.desktop-768 .pt-tablecell {
    padding-bottom: 110px;
    padding-top: 60px;
}
.hexagon-item:hover .icon i
{
color:#ff0037;
transition:0.6s;
```

```
}
.hexagon-item:hover .title
{
-webkit-animation: focus-in-contract 0.5s cubic-bezier(0.250, 0.460, 0.450, 0.940) both;
        animation: focus-in-contract 0.5s cubic-bezier(0.250, 0.460, 0.450, 0.940) both;
}
/**************************/
@-webkit-keyframes focus-in-contract {
0% {
   letter-spacing: 1em;
   -webkit-filter: blur(12px);
        filter: blur(12px);
   opacity: 0;
}
100% {
   -webkit-filter: blur(0px);
        filter: blur(0px);
   opacity: 1;
}
}
@keyframes focus-in-contract {
0% {
   letter-spacing: 1em;
   -webkit-filter: blur(12px);
```

```
            filter: blur(12px);
        opacity: 0;
    }
    100% {
        -webkit-filter: blur(0px);
            filter: blur(0px);
        opacity: 1;
    }
    }
    @media only screen and (max-width: 767px)
    {
    .hexagon-item {
        float: none;
        margin: 0 auto 50px;
    }
    .hexagon-item:first-child {
        margin-left: auto;
    }
    .page-home .hexagon-item:nth-last-child(1), .page-home .hexagon-item:nth-last-child(2), .page-home .hexagon-item:nth-last-child(3) {
        -webkit-transform: rotate(30deg) translate(0px, 0px);
        -moz-transform: rotate(30deg) translate (0px, 0px);
        -ms-transform: rotate(30deg) translate (0px, 0px);
        -o-transform: rotate(30deg) translate (0px, 0px);
        transform: rotate(30deg) translate (0px, 0px);
```

```
        }
        }
    </style>
</head>
<body background="/static/person.avif">
    <main class="site-wrapper">
  <div class="pt-table desktop-768">
    <div class="pt-tablecell page-home relative";
    background-position: center;
    background-size: cover;>
                <div class="overlay"></div>
                <div class="container">
                    <div class="row">
                        <div class="col-xs-12 col-md-offset-1 col-md-10 col-lg-offset-2 col-lg-8">
                            <div class="page-title home text-center">
                                <h4 align="center" style="color: white;"></h4>
                                <p style="font-size:22px"><b>FinanceMax needs to detect fraudulent
transactions in real-time. How can they use machine learning to build an effective fraud detection system
                                </b></p>
                                <p><h4 align="center" style="color: white;">
                            </div>
                            <div class="hexagon-menu clear">
                                <div class="hexagon-item">
                                    <div class="hex-item">
                                        <div></div>
```

```
                    <div></div>
                    <div></div>
                </div>
                <div class="hex-item">
                    <div></div>
                    <div></div>
                    <div></div>
                </div>
                <a href="/welcome" class="hex-content">
                    <span class="hex-content-inner">
                        <span class="icon">
                            <i class="fa fa-universal-access"></i>
                        </span>
                        <span class="title">Team Details</span>
                    </span>
                    <svg viewBox="0 0 173.20508075688772 200" height="200" width="174"
version="1.1" xmlns="http://www.w3.org/2000/svg"><path d="M86.60254037844386
0L173.20508075688772 50L173.20508075688772 150L86.60254037844386 200L0 150L0 50Z"
fill="#1e2530"></path></svg>
                </a>
            </div>
            <div class="hexagon-item">
                <div class="hex-item">
                    <div></div>
                    <div></div>
```

```
                <div></div>
            </div>
            <div class="hex-item">
                <div></div>
                <div></div>
                <div></div>
            </div>
            <a  class="hex-content" href="/about">
                <span class="hex-content-inner">
                    <span class="icon">
                        <i class="fa fa-bullseye"></i>
                    </span>
                    <span class="title">Introduction</span>
                </span>
                <svg viewBox="0 0 173.20508075688772 200" height="200" width="174"
version="1.1" xmlns="http://www.w3.org/2000/svg"><path d="M86.60254037844386
0L173.20508075688772 50L173.20508075688772 150L86.60254037844386 200L0 150L0 50Z"
fill="#1e2530"></path></svg>
            </a>
        </div>
        <div class="hexagon-item">
            <div class="hex-item">
                <div></div>
                <div></div>
                <div></div>
```

```
                </div>
                <div class="hex-item">
                    <div></div>
                    <div></div>
                    <div></div>
                </div>
                <a  class="hex-content" href="/uses">
                    <span class="hex-content-inner">
                        <span class="icon">
                            <i class="fa fa-braille"></i>
                        </span>
                        <span class="title">Existing System</span>
                    </span>
                    <svg viewBox="0 0 173.20508075688772 200" height="200" width="174"
version="1.1" xmlns="http://www.w3.org/2000/svg"><path d="M86.60254037844386
0L173.20508075688772 50L173.20508075688772 150L86.60254037844386 200L0 150L0 50Z"
fill="#1e2530"></path></svg>
                </a>
            </div>
            <div class="hexagon-item">
                <div class="hex-item">
                    <div></div>
                    <div></div>
                    <div></div>
                </div>
```

```
                    <div class="hex-item">
                        <div></div>
                        <div></div>
                        <div></div>
                    </div>
                    <a class="hex-content" href="/display">
                        <span class="hex-content-inner">
                            <span class="icon">
                                <i class="fa fa-id-badge"></i>
                            </span>
                            <span class="title">Dataset</span>
                        </span>
                        <svg viewBox="0 0 173.20508075688772 200" height="200" width="174"
version="1.1" xmlns="http://www.w3.org/2000/svg"><path d="M86.60254037844386
0L173.20508075688772 50L173.20508075688772 150L86.60254037844386 200L0 150L0 50Z"
fill="#1e2530"></path></svg>
                    </a>
                </div>
                <div class="hexagon-item">
                    <div class="hex-item">
                        <div></div>
                        <div></div>
                        <div></div>
                    </div>
                    <div class="hex-item">
```

```
                    <div></div>
                    <div></div>
                    <div></div>
                </div>
                <a class="hex-content" href="/apps">
                    <span class="hex-content-inner">
                        <span class="icon">
                            <i class="fa fa-database"></i>
                        </span>
                        <span class="title">Frontend</span>
                    </span>
                    <svg viewBox="0 0 173.20508075688772 200" height="200" width="174"
version="1.1" xmlns="http://www.w3.org/2000/svg"><path d="M86.60254037844386
0L173.20508075688772 50L173.20508075688772 150L86.60254037844386 200L0 150L0 50Z"
fill="#1e2530"></path></svg>
                </a>
            </div>
            <div class="hexagon-item">
                <div class="hex-item">
                    <div></div>
                    <div></div>
                    <div></div>
                </div>
                <div class="hex-item">
                    <div></div>
```

```
                    <div></div>
                    <div></div>
                </div>
                <a class="hex-content" href="/backend">
                    <span class="hex-content-inner">
                        <span class="icon">
                            <i class="fa fa-file-code"></i>
                        </span>
                        <span class="title">CODE</span>
                    </span>
                    <svg viewBox="0 0 173.20508075688772 200" height="200" width="174" version="1.1" xmlns="http://www.w3.org/2000/svg"><path d="M86.60254037844386 0L173.20508075688772 50L173.20508075688772 150L86.60254037844386 200L0 150L0 50Z" fill="#1e2530"></path></svg>
                </a>
            </div>
            <div class="hexagon-item">
                <div class="hex-item">
                    <div></div>
                    <div></div>
                    <div></div>
                </div>
                <div class="hex-item">
                    <div></div>
                    <div></div>
```

```
                    <div></div>
                </div>
                <a  class="hex-content" href="report">
                    <span class="hex-content-inner">
                        <span class="icon">
                            <i class="fa fa-file-pdf"></i>
                        </span>
                        <span class="title">ProjectReport</span>
                    </span>
                    <svg viewBox="0 0 173.20508075688772 200" height="200" width="174"
version="1.1" xmlns="http://www.w3.org/2000/svg"><path d="M86.60254037844386
0L173.20508075688772 50L173.20508075688772 150L86.60254037844386 200L0 150L0 50Z"
fill="#1e2530"></path></svg>
                </a>
            </div>
        </div>
    </div>
</div>
</main>
</body>
</html>
```

4.2.2 Logic

- **Logic Code:**
- **File Name:** Finance Max needs to detect.ipynb

Libraries:

```
import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder

from sklearn.linear_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVCimport pandas as pd

from sklearn.metrics import accuracy_score, classification_report
```

Data:

```
df = pd.read_csv("transactiondata.csv")

df.head()
```

	TransactionID	Amount	Timestamp	TransactionLocation	IsFraud
0	8036	2763.365182	57:31.1	Chicago	1
1	2443	762.139911	57:31.1	Chennai	0
2	3227	242.295305	57:31.1	Chennai	0
3	2843	918.508006	57:31.1	Chennai	0
4	2237	892.410196	57:31.1	Mumbai	0

PreProcessing:

```
# Display basic statistics
```

print(df.describe())

	TransactionID	Amount	IsFraud
count	8200.000000	8200.000000	8200.000000
mean	4100.500000	546.415848	0.024390
std	2367.280437	480.744322	0.154267
min	1.000000	1.011623	0.000000
25%	2050.750000	249.438239	0.000000
50%	4100.500000	503.990179	0.000000
75%	6150.250000	761.233571	0.000000
max	8200.000000	4978.482686	1.000000

Distribution of legitimate and fraudulent transactions:

sns.countplot(x='IsFraud', data=df)

plt.title('Distribution of Legitimate and Fraudulent Transactions')

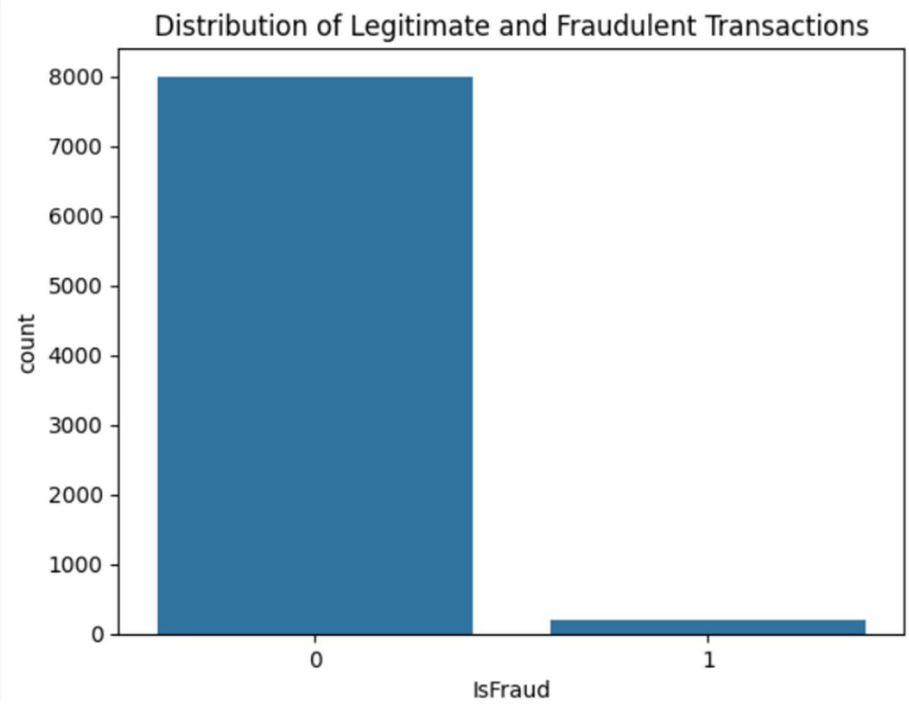


Fig.4.1 Legitimate and Fraud Transactions

Distribution of transaction amounts:

plt.figure(figsize=(10, 6))

sns.histplot(data=df, x='Amount', bins=50, kde=True, hue='IsFraud', multiple='stack')

plt.title('Distribution of Transaction Amounts')

plt.xlabel('Amount')

plt.ylabel('Frequency')

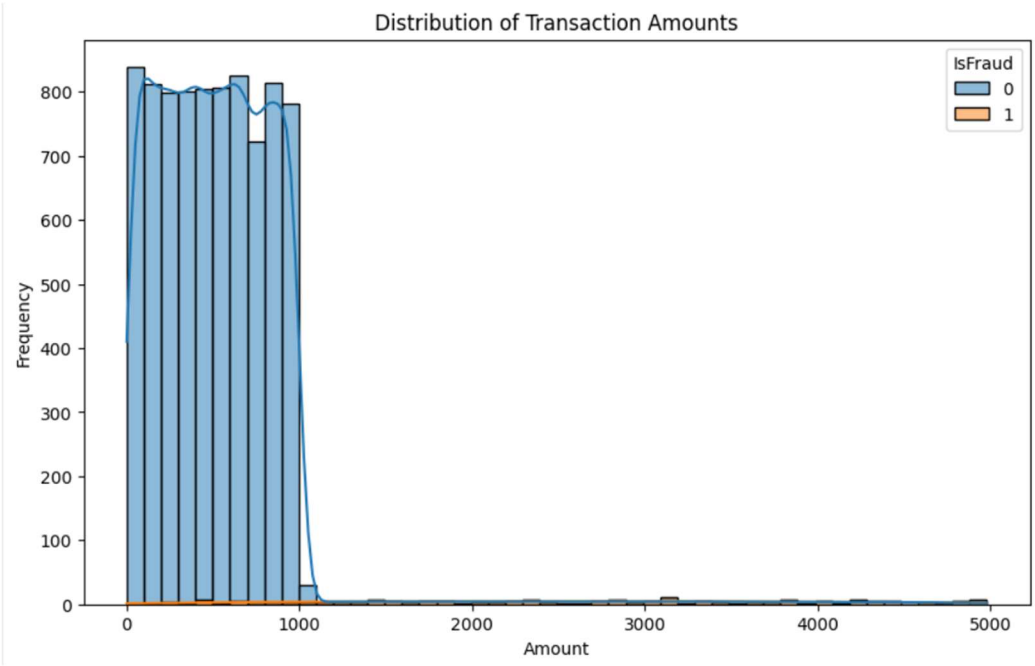


Fig.4.2 Transaction Amounts

Transaction locations for legitimate and fraudulent transactions:

plt.figure(figsize=(12, 6))

sns.countplot(x='TransactionLocation', hue='IsFraud', data=df)

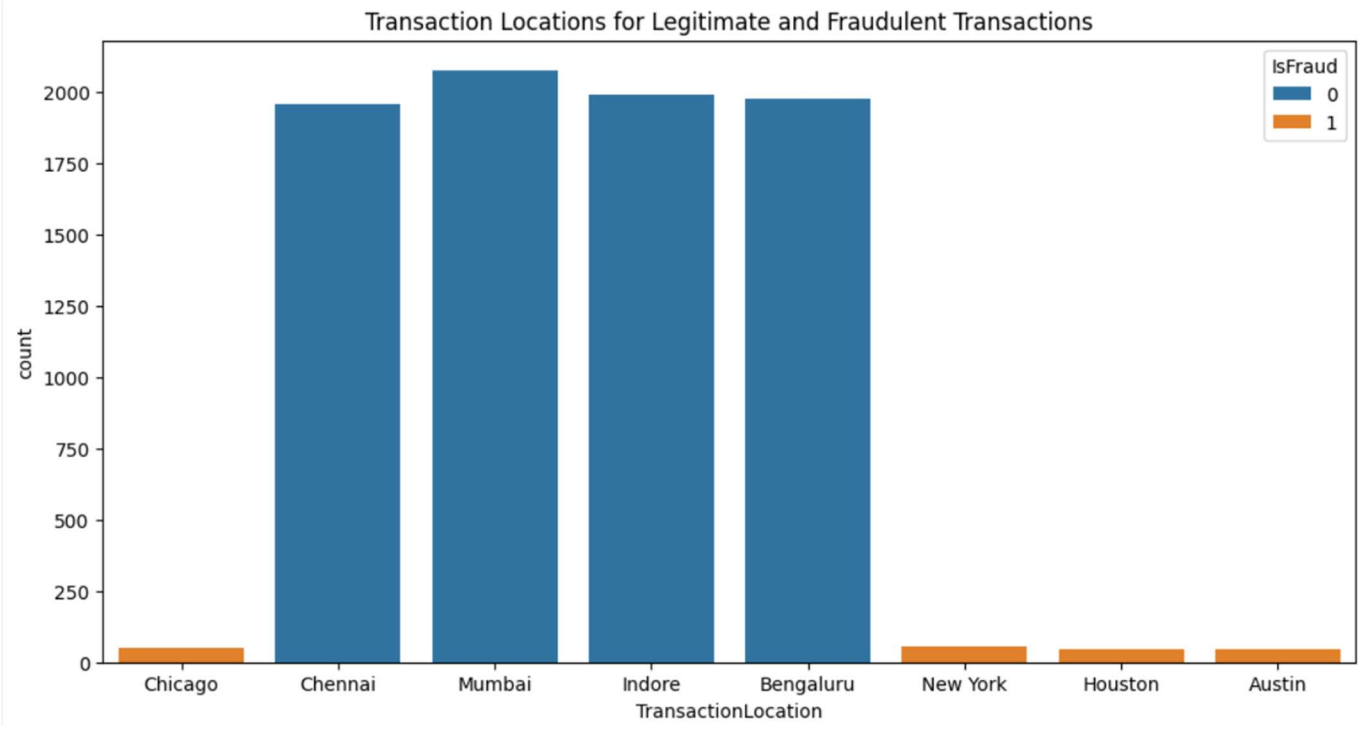


Fig.4.3 Transaction Locations for Legitimate and Fraud

Modal Comparision:

```
# Load the dataset

df = pd.read_csv("transactiondata.csv")

# Preprocessing

# Drop unnecessary columns

df = df.drop('TransactionID', axis=1)

# Split the 'Timestamp' column into minutes and seconds

df[['Minutes', 'Seconds']] = df['Timestamp'].str.split(':', expand=True)

# Convert minutes and seconds to float

df['Minutes'] = df['Minutes'].astype(float)

df['Seconds'] = df['Seconds'].astype(float)

# Convert the time to seconds

df['Timestamp_seconds'] = df['Minutes'] * 60 + df['Seconds']

# Drop the original 'Timestamp' column and the intermediate columns

df = df.drop(['Timestamp', 'Minutes', 'Seconds'], axis=1)

# Now 'Timestamp_seconds' column contains the time in seconds

# Continue with the rest of the preprocessing steps and model training

# Encode categorical variables

label_encoder = LabelEncoder()

df['TransactionLocation'] = label_encoder.fit_transform(df['TransactionLocation'])

# Splitting the data into features (X) and target variable (y)

X = df.drop('IsFraud', axis=1)

y = df['IsFraud']

# Splitting the dataset into training and testing sets
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the classifiers

logistic_regression = LogisticRegression(random_state=42)

random_forest = RandomForestClassifier(random_state=42)

svm_classifier = SVC(random_state=42)

# Train the models

logistic_regression.fit(X_train, y_train)

random_forest.fit(X_train, y_train)

svm_classifier.fit(X_train, y_train)

# Make predictions

y_pred_lr = logistic_regression.predict(X_test)

y_pred_rf = random_forest.predict(X_test)

y_pred_svm = svm_classifier.predict(X_test)

# Evaluate the models

print("Logistic Regression:")

print(classification_report(y_test, y_pred_lr))

print("Accuracy:", accuracy_score(y_test, y_pred_lr))

print("\nRandom Forest:")

print(classification_report(y_test, y_pred_rf))

print("Accuracy:", accuracy_score(y_test, y_pred_rf))

print("\nSupport Vector Machine:")

print(classification_report(y_test, y_pred_svm))

print("Accuracy:", accuracy_score(y_test, y_pred_svm))
```

Logistic Regression:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1607
1	1.00	0.79	0.88	33
accuracy			1.00	1640
macro avg	1.00	0.89	0.94	1640
weighted avg	1.00	1.00	1.00	1640

Accuracy: 0.9957317073170732

Random Forest:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1607
1	1.00	1.00	1.00	33
accuracy			1.00	1640
macro avg	1.00	1.00	1.00	1640
weighted avg	1.00	1.00	1.00	1640

Accuracy: 1.0

Support Vector Machine:

...

macro avg	1.00	0.88	0.93	1640
weighted avg	1.00	1.00	0.99	1640

Accuracy: 0.9951219512195122

Final Modal:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score
from sklearn.preprocessing import OneHotEncoder
# Load the dataset
df = pd.read_csv("transactiondata.csv")
# Preprocessing
# Drop unnecessary columns
df = df.drop('TransactionID', axis=1)
# Split the 'Timestamp' column into minutes and seconds
df[['Minutes', 'Seconds']] = df['Timestamp'].str.split(':', expand=True)
# Convert minutes and seconds to float
df['Minutes'] = df['Minutes'].astype(float)
df['Seconds'] = df['Seconds'].astype(float)
# Convert the time to seconds
df['Timestamp_seconds'] = df['Minutes'] * 60 + df['Seconds']
```

```
# Drop the original 'Timestamp' column and the intermediate columns

df = df.drop(['Timestamp', 'Minutes', 'Seconds'], axis=1)

# Encode categorical variables using one-hot encoding

encoder = OneHotEncoder()

df_encoded = pd.DataFrame(encoder.fit_transform(df[['TransactionLocation']]).toarray(),
columns=encoder.categories_[0])

# Concatenate the encoded features with the original dataframe

df = pd.concat([df.drop(['TransactionLocation'], axis=1), df_encoded], axis=1)

# Splitting the data into features (X) and target variable (y)

X = df.drop('IsFraud', axis=1)

y = df['IsFraud']

# Splitting the dataset into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the Random Forest classifier

random_forest = RandomForestClassifier(random_state=42)

# Train the Random Forest classifier

random_forest.fit(X_train, y_train)

# Make predictions

y_pred = random_forest.predict(X_test)

# Evaluate the model

print("Random Forest Classifier:")

print(classification_report(y_test, y_pred))

print("Accuracy:", accuracy_score(y_test, y_pred))
```

Random Forest Classifier:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1607
1	1.00	1.00	1.00	33
accuracy			1.00	1640
macro avg	1.00	1.00	1.00	1640
weighted avg	1.00	1.00	1.00	1640

Accuracy: 1.0

Final Modal with User Given Input:

```
# Assuming the features for the user are provided in a dictionary
user_features = {
    'Amount': 918.5080057,
    'TransactionLocation': 'Chennai' # Assuming the user's transaction location is Chicago
}
# Create a DataFrame with the user features
user_df = pd.DataFrame(user_features, index=[0])
# Preprocess the user data similar to the training data
# Split the 'Timestamp' column into minutes and seconds
user_df[['Minutes', 'Seconds']] = '57:31.1'.split(':')
# Convert minutes and seconds to float
user_df['Minutes'] = user_df['Minutes'].astype(float)
user_df['Seconds'] = user_df['Seconds'].astype(float)
# Convert the time to seconds
```

```
user_df['Timestamp_seconds'] = user_df['Minutes'] * 60 + user_df['Seconds']

# Drop the original 'Timestamp' column and the intermediate columns

user_df = user_df.drop(['Minutes', 'Seconds'], axis=1)

# Encode categorical variables using the same OneHotEncoder

user_encoded = pd. DataFrame(encoder. transform(user_df[['TransactionLocation']]).toarray(), columns=encoder.categories_[0])

# Concatenate the encoded features with the original dataframe

user_df = pd.concat([user_df.drop(['TransactionLocation'], axis=1), user_encoded], axis=1)

# Make predictions for the user

user_prediction = random_forest.predict(user_df)

# Print the prediction

if user_prediction[0] == 1:

    print ("The transaction is predicted to be fraudulent.")

else:

    print ("The transaction is predicted to be legitimate.")
```

The transaction is predicted to be legitimate.

```
import pandas as pd

user_features = {

    'Amount': float (input ("Enter the transaction amount: ")),

    'TransactionLocation': input ("Enter the transaction location: ")  # Assuming the user's transaction location is Chicago

}

# Create a DataFrame with the user features

user_df = pd.DataFrame(user_features, index=[0])
```

```
# Preprocess the user data similar to the training data

# Split the 'Timestamp' column into minutes and seconds

user_df[['Minutes', 'Seconds']] = input("Enter the timestamp (MM:SS.S): ").split(':')

# Convert minutes and seconds to float

user_df['Minutes'] = user_df['Minutes'].astype(float)

user_df['Seconds'] = user_df['Seconds'].astype(float)

# Convert the time to seconds

user_df['Timestamp_seconds'] = user_df['Minutes'] * 60 + user_df['Seconds']

# Drop the original 'Timestamp' column and the intermediate columns

user_df = user_df.drop(['Minutes', 'Seconds'], axis=1)

# Encode categorical variables using the same OneHotEncoder

user_encoded = pd. Data Frame(encoder. transform(user_df[['TransactionLocation']]).toarray(),
columns=encoder.categories_[0])

# Concatenate the encoded features with the original dataframe

user_df = pd.concat([user_df.drop(['TransactionLocation'], axis=1), user_encoded], axis=1)

# Make predictions for the user

user_prediction = random_forest.predict(user_df)

# Print the prediction

if user_prediction[0] == 1:

    print("The transaction is predicted to be fraudulent.")

else:

    print("The transaction is predicted to be legitimate.")
```

The transaction is predicted to be fraudulent.

8036,2763.365182,57:31.1, Chicago,1

CHAPTER 5 – SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE

1. Data Collection:

- Incoming transaction data from various sources such as credit card transactions, online payments, wire transfers, etc., is collected in real-time.

- This data includes transaction amount, timestamp, transaction type, customer information, and any other relevant features.

2. Data Preprocessing:

- Raw transaction data is preprocessed to clean and transform it into a suitable format for machine learning algorithms.

- Preprocessing steps may include data normalization, handling missing values, feature scaling, and encoding categorical variables.

3. Feature Engineering:

- Relevant features are selected or engineered from the transaction data to improve the performance of the fraud detection model.

- Features such as transaction frequency, location, time of day, device information, and transaction history may be useful for identifying fraudulent patterns.

4. Machine Learning Model Selection:

- Various machine learning algorithms are explored and evaluated to select the most suitable ones for fraud detection.

- Ensemble methods like Random Forest or Gradient Boosting, as well as anomaly detection algorithms like Isolation Forest or One-Class SVM, are commonly used for fraud detection tasks.

- Deep learning models such as deep neural networks or recurrent neural networks (RNNs) may also be considered for capturing complex patterns in transaction data.

5. Model Training:

- The selected machine learning models are trained on historical transaction data, including both fraudulent and non-fraudulent transactions.

- Training data should be balanced to ensure the model can effectively learn patterns of fraudulent behavior without being biased towards normal transactions.

6. Real-Time Prediction:

- Once trained, the model is deployed to make real-time predictions on incoming transactions.

- As new transactions arrive, the model evaluates each transaction and assigns a probability score indicating the likelihood of fraud.

- Thresholds can be set to classify transactions as either fraudulent or legitimate based on these scores.

5.2 DATA FLOW DIAGRAM

- The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
- The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
- DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
- DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

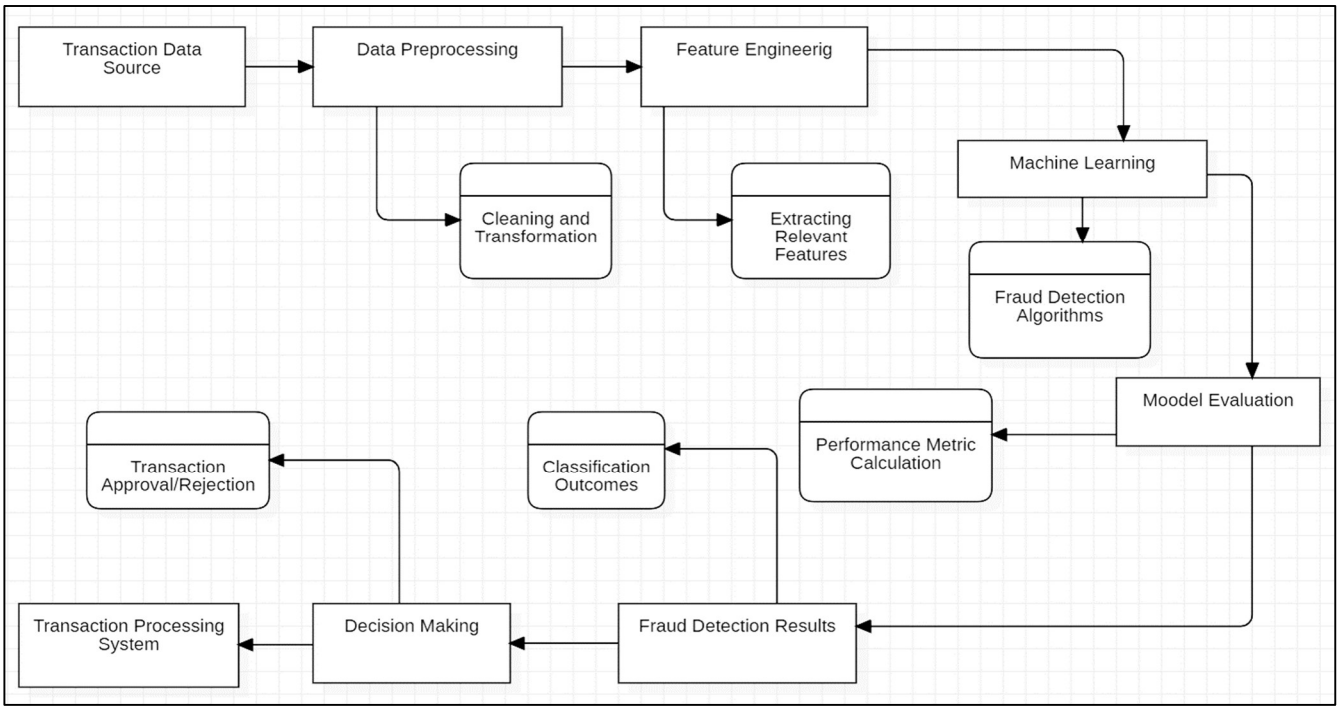


Fig.5.2 Data Flow Diagram

5.3 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.

2. Provide extendibility and specialization mechanisms to extend the core concepts.

3. Be independent of particular programming languages and development process.

4. Provide a formal basis for understanding the modeling language.

5. Encourage the growth of OO tools market.

6. Support higher level development concepts such as collaborations, frameworks, patterns and components.

7. Integrate best practices.

5.4 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

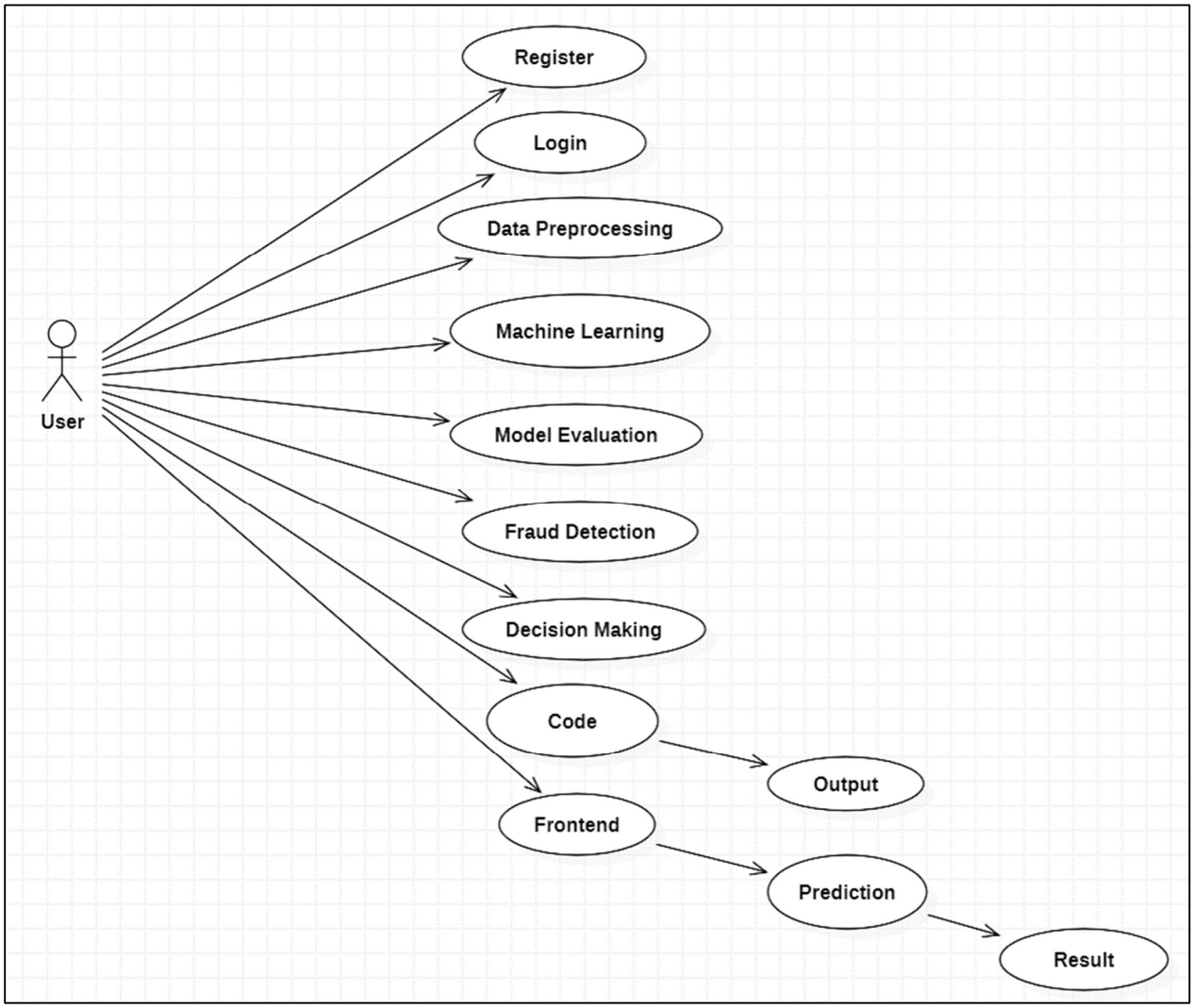


Fig.5.4 Use Case Diagram

5.5 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

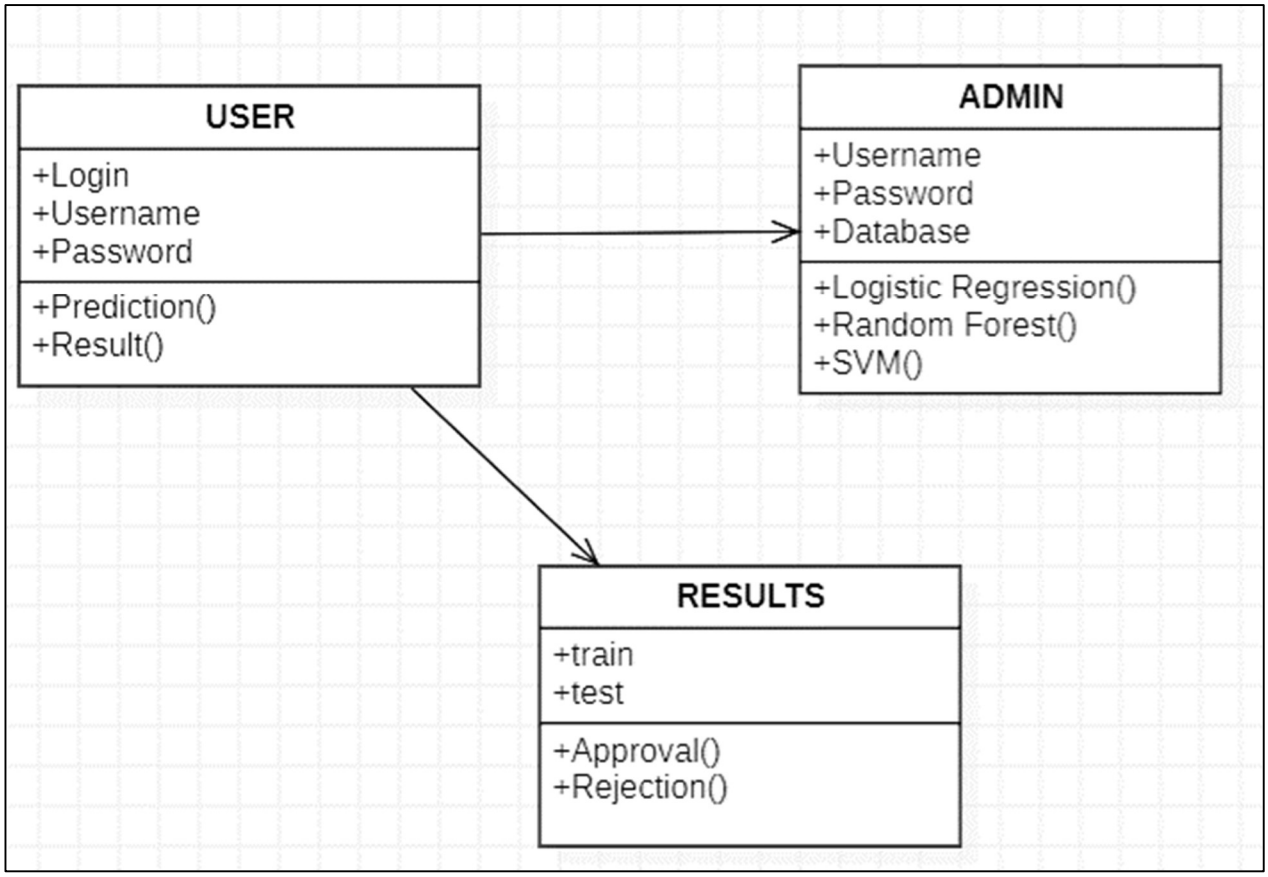


Fig.5.5 Class Diagram

5.6 SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes diagrams, event scenarios, and timing diagrams.

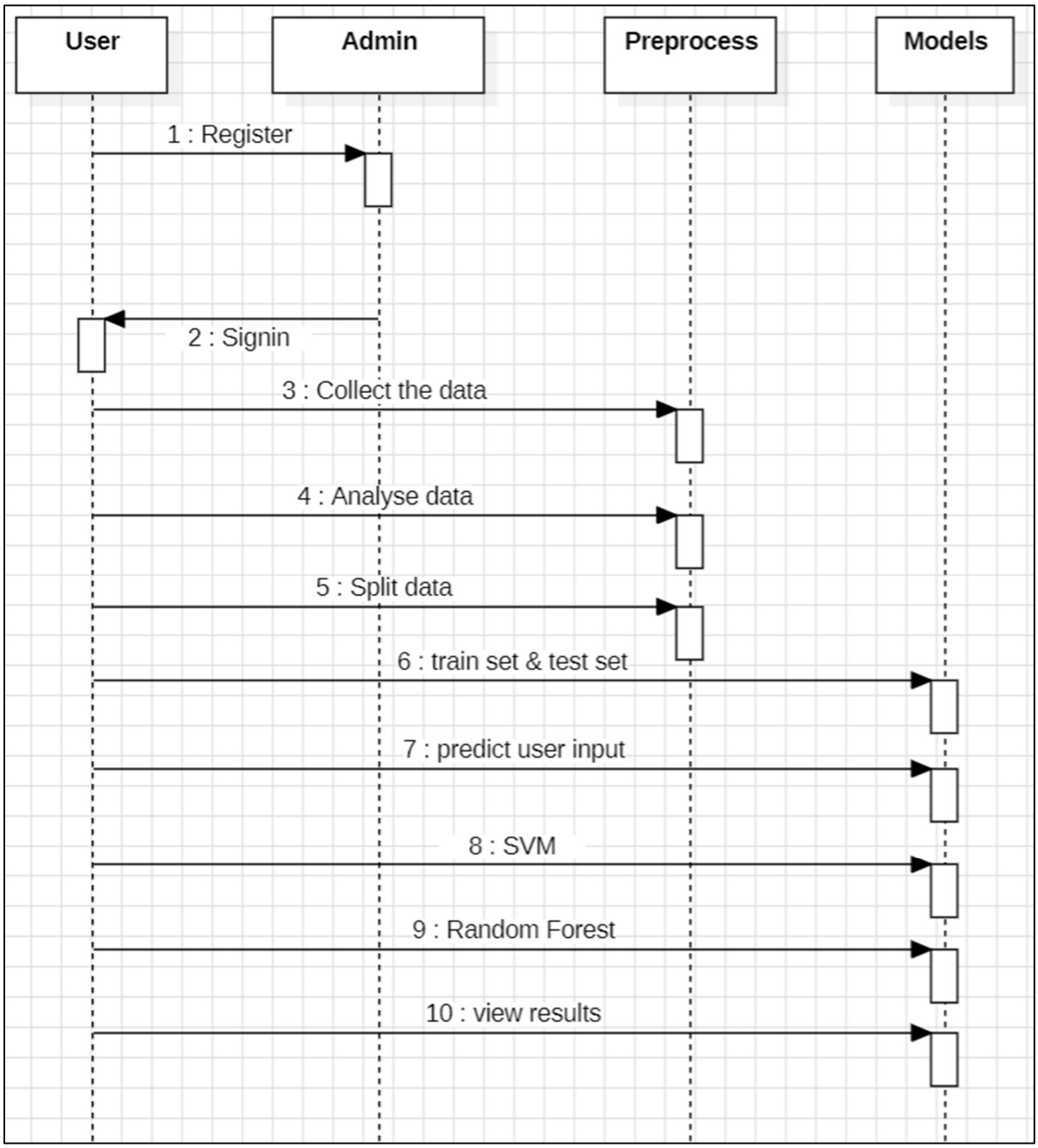


Fig.5.6 Sequence Diagram

5.7 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

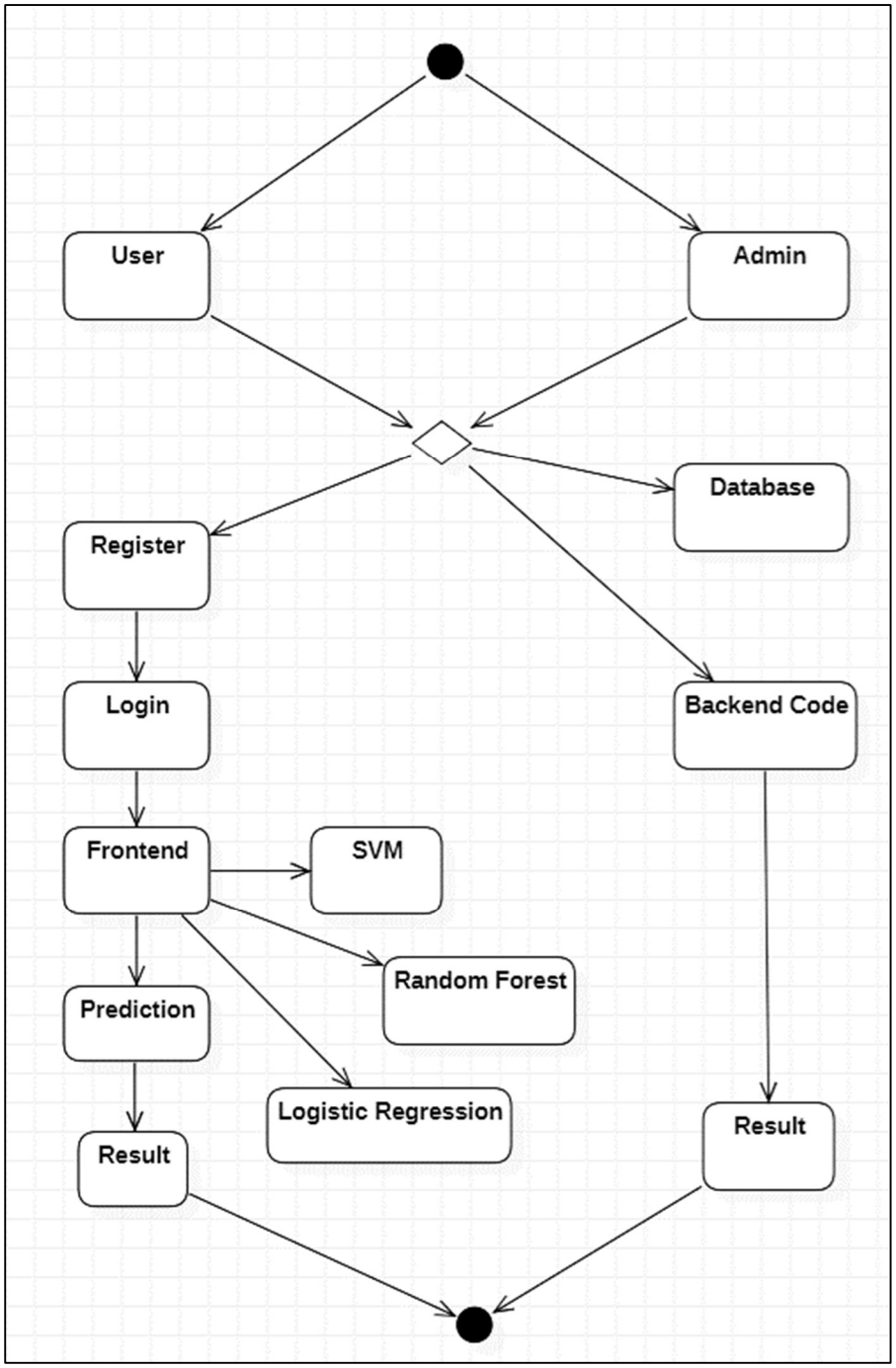


Fig.5.7 Activity Diagram

5.8 INPUT & OUTPUT

Input:

The input is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input should consider the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

Output:

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

CHAPTER 6 – SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.1 TYPES OF TESTING

6.1.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.1.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

6.1.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

6.1.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

6.1.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

6.1.7 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

6.2 TEST STRATEGY AND APPROACH

Certainly! Designing an effective test strategy and approach for **Financial Max's Fraud Detection System** is crucial to ensure its reliability and accuracy. Let's outline the key components:

1. **Test Objectives**:
    - **Functional Verification**: Validate that the system correctly detects fraudulent transactions.
    - **Performance Testing**: Assess the system's responsiveness under varying transaction loads.
    - **Security Testing**: Ensure robustness against unauthorized access and data breaches.
    - **Usability Testing**: Evaluate user-friendliness of alerts and reporting interfaces.
2. **Test Levels**:
    - **Unit Testing**:
        - Verify individual components (e.g., rule-based engine, machine learning models).
        - Test rule execution logic and model predictions.
    - **Integration Testing**:
        - Validate interactions between components (e.g., data ingestion, alert generation).
        - Ensure seamless integration with external systems (e.g., payment gateways).
    - **System Testing (End-to-End)**:
        - Validate the entire system's functionality.
        - Test real-time transaction monitoring, alert generation, and escalation.
        - Assess overall system performance.
3. **Test Data**:
    - Use a mix of synthetic and real-world data.
    - Include valid transactions, known fraudulent patterns, and edge cases.
    - Vary transaction amounts, frequencies, and geographical locations.
4. **Test Scenarios**:
    - **Positive Scenarios**:
        - Valid transactions with no fraud indicators.
        - Rule-based alerts triggered correctly.
        - Machine learning models predict fraud accurately.
    - **Negative Scenarios**:
        - Anomalous transactions (e.g., rapid consecutive withdrawals).
        - Known fraudulent patterns (e.g., card-not-present transactions).
        - Stress testing with high transaction volumes.

5. **Test Automation**:
    - Automate repetitive scenarios (e.g., rule validation, model predictions).
    - Use tools like Selenium, JMeter, or custom scripts.
6. **Performance Testing**:
    - Simulate varying transaction loads (low, moderate, peak).
    - Measure response times, throughput, and resource utilization.
7. **Security Testing**:
    - Verify access controls (user roles, permissions).
    - Test encryption, authentication, and authorization mechanisms.
8. **Usability Testing**:
    - Evaluate user interfaces for clarity and ease of use.
    - Ensure alerts are actionable and informative.
9. **Reporting and Defect Tracking**:
    - Document test results, including false positives and false negatives.
    - Track defects and prioritize fixes.
10. **Regression Testing**:
    - Continuously validate system updates and enhancements.
    - Ensure new features do not impact existing functionality.

6.3 TEST CASES

Certainly! When designing test cases for **Financial Max's Fraud Detection System**, we need to cover various scenarios to ensure robustness. Here are some test case categories:

1. **Positive Test Cases**:
    - **Valid Transactions**:
        - Verify that legitimate transactions are correctly processed.
        - Test different transaction types (e.g., online purchases, ATM withdrawals).
        - Ensure accurate detection of non-fraudulent activities.
    - **Known Fraud Patterns**:
        - Test transactions that match known fraud patterns (e.g., frequent small transactions, unusual time intervals).
        - Validate that the system correctly flags these cases.

2. **Negative Test Cases**:
    - **Anomalous Transactions**:
        - Introduce anomalies (e.g., unusually large amounts, rapid consecutive transactions).
        - Confirm that the system identifies and alerts on such anomalies.
    - **False Positives**:
        - Test scenarios where the system incorrectly flags a valid transaction as fraudulent.
        - Evaluate the impact of false positives on user experience.
3. **Edge Cases**:
    - **Low-Value Transactions**:
        - Test very small transactions (near the lower limit).
        - Ensure they are correctly handled.
    - **High-Value Transactions**:
        - Test large transactions (near the upper limit).
        - Verify that they trigger appropriate alerts.
4. **Concurrency and Load Testing**:
    - **Multiple Simultaneous Transactions**:
        - Simulate concurrent transactions from different users.
        - Assess system performance under load.
    - **Peak Transaction Volume**:
        - Test the system's scalability during peak hours.
        - Measure response times and resource utilization.
5. **Boundary Testing**:
    - **Transaction Amount Limits**:
        - Test transactions just below and above predefined limits.
        - Verify that the system behaves correctly.
    - **Time Intervals**:
        - Test transactions occurring at the boundary of time intervals (e.g., midnight).
        - Confirm accurate detection.
6. **Regression Testing**:
    - **System Updates**:
        - After system enhancements or bug fixes, retest existing scenarios.
        - Ensure new changes do not impact existing functionality.

7. **Negative Scenarios**:
    - **Bypass Attempts**:
        - Test fraudulent transactions designed to bypass detection rules.
        - Verify that the system remains effective.

CHAPTER 7 – SOFTWARE ENVIRONMENT

7.1 PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++or Java. It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Interactive Mode Programming:

Invoking the interpreter without passing a script file as a parameter brings up the following prompt −

$ python

Python 2.4.3 (#1, Nov 11 2010, 13:34:43)

[GCC 4.1.2 20080704 (Red Hat 4.1.2-48)] on linux2

Type "help", "copyright", "credits" or "license" for more information.

>>>

Type the following text at the Python prompt and press the Enter −

>>> print "Hello, Python!"

If you are running new version of Python, then you would need to use print statement with parenthesis as in print ("Hello, Python!"); However in Python version 2.4.3, this produces the following result –

Hello, Python!

Script Mode Programming:

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension .py. Type the following source code in a test.py file −

Live Demo

print "Hello, Python!"

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows −

$ python test.py

This produces the following result −Hello, Python!

Let us try another way to execute a Python script. Here is the modified test.py file

Live Demo #!/usr/bin/python print "Hello, Python!"

We assume that you have Python interpreter available in /usr/bin directory.

Now, try to run this program as follows −

$ chmod +x test.py # This is to make file executable

$./test.py

This produces the following result −Hello, Python!

Python Identifiers:

A Python identifier is a name used to identify a variable, function, class, module or other object. An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).

Python does not allow punctuation characters such as @, $, and % within identifiers. Python is a case sensitive programming language. Thus, Manpower and manpower are two different identifiers in Python.

Here are naming conventions for Python identifiers –

Class names start with an uppercase letter. All other identifiers start with a lowercase letter. Starting an identifier with a single leading underscore indicates that the identifier is private. Starting an identifier with two leading underscores indicates astrongly private identifier.

If the identifier also ends with two trailing underscores, the identifier is a language-defined special name.

1. **Data Preparation and Exploration**:
    - **Pandas**: Use Pandas for data manipulation, cleaning, and feature engineering.
    - **Numpy**: Handle numerical operations efficiently.
2. **Machine Learning and Data Science Libraries**:
    - **Scikit-learn**: Train machine learning models (e.g., decision trees, random forests, logistic regression).
    - **Auto-sklearn** or **H2O**: Automated machine learning tools.
3. **Visualization and Reporting**:
    - **Matplotlib** or **Seaborn**: Create visualizations for monitoring and reporting.
    - **Tableau** or **Power BI**: Build interactive dashboards.
4. **Testing and Quality Assurance**:
    - Design comprehensive test cases for fraud detection scenarios.
    - Perform unit testing, integration testing, and system testing.

7.2 MONGODB

MongoDB, the most popular NoSQL database, is an open-source document-oriented database. The term ‘NoSQL’ means ‘non-relational’. It means that MongoDB isn’t based on the table-like relational database structure but provides an altogether different mechanism for storage and retrieval of data. This format of storage is called BSON (similar to JSON format).

A simple MongoDB document Structure:

```
{
  title: 'FINANCE MAX',
  by: 'BATCH 2',
  type: 'NoSQL'
}
```

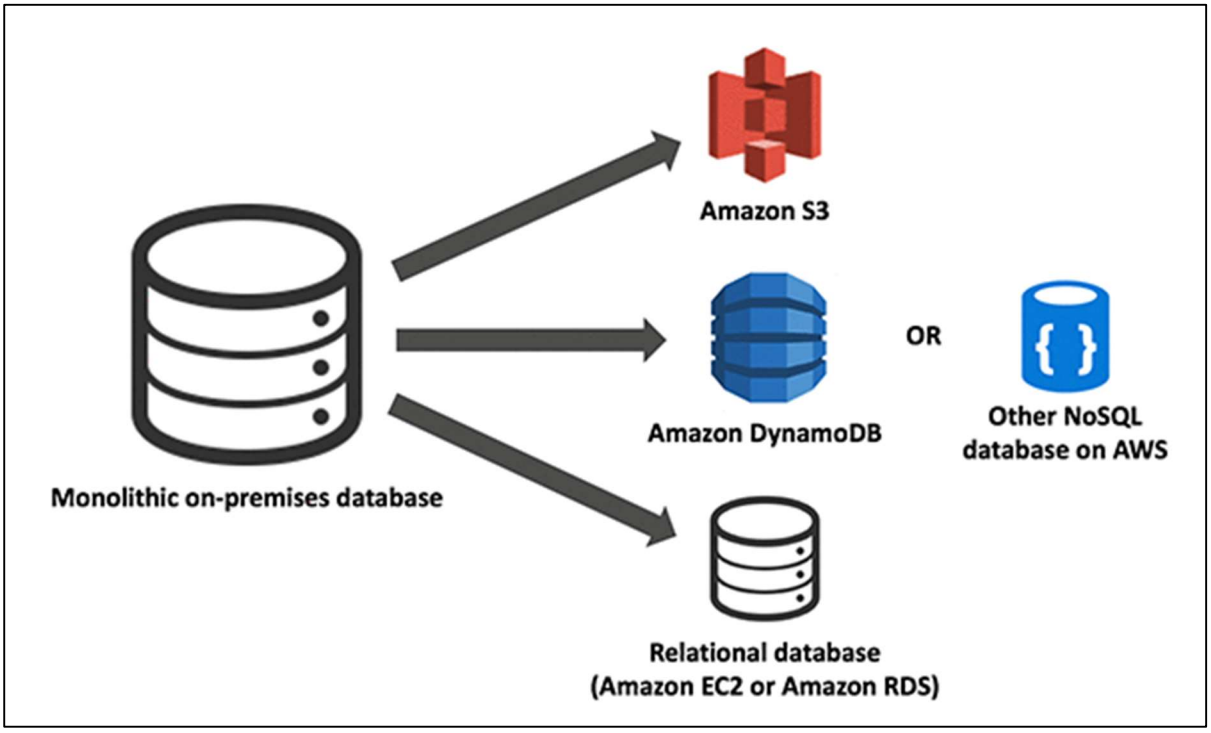


Fig.7.2 Database of MongoDB

SQL databases store data in tabular format. This data is stored in a predefined data model which is not very much flexible for today's real-world highly growing applications. Modern applications are more networked, social, and interactive than ever**.** Applications are storing more and more data and are accessing it at higher rates. Relational Database Management System (RDBMS) i**s** not the correct choice when it comes to handling big data by the virtue of their design since they are not horizontally scalable. If the database runs on a single server, then it will reach a scaling limit. NoSQL databases are more scalable and provide superior performance. MongoDB is such a NoSQL database that scales by adding more and more servers and increases productivity with its flexible document model.

Where do we use MongoDB?

MongoDB is preferred over RDBMS in the following scenarios:

- **Big Data**: If you have huge amount of data to be stored in tables, think of MongoDB before RDBMS databases. MongoDB has built-in solution for partitioning and sharding your database.
- **Unstable Schema**: Adding a new column in RDBMS is hard whereas MongoDB is schema-less. Adding a new field does not affect old documents and will be very easy.
- **Distributed data** Since multiple copies of data are stored across different servers, recovery of data is instant and safe even if there is a hardware failure.

CHAPTER 8 – EXPERIMENTAL RESULTS

8.1 SCREEN SHOTS:

Home Page:

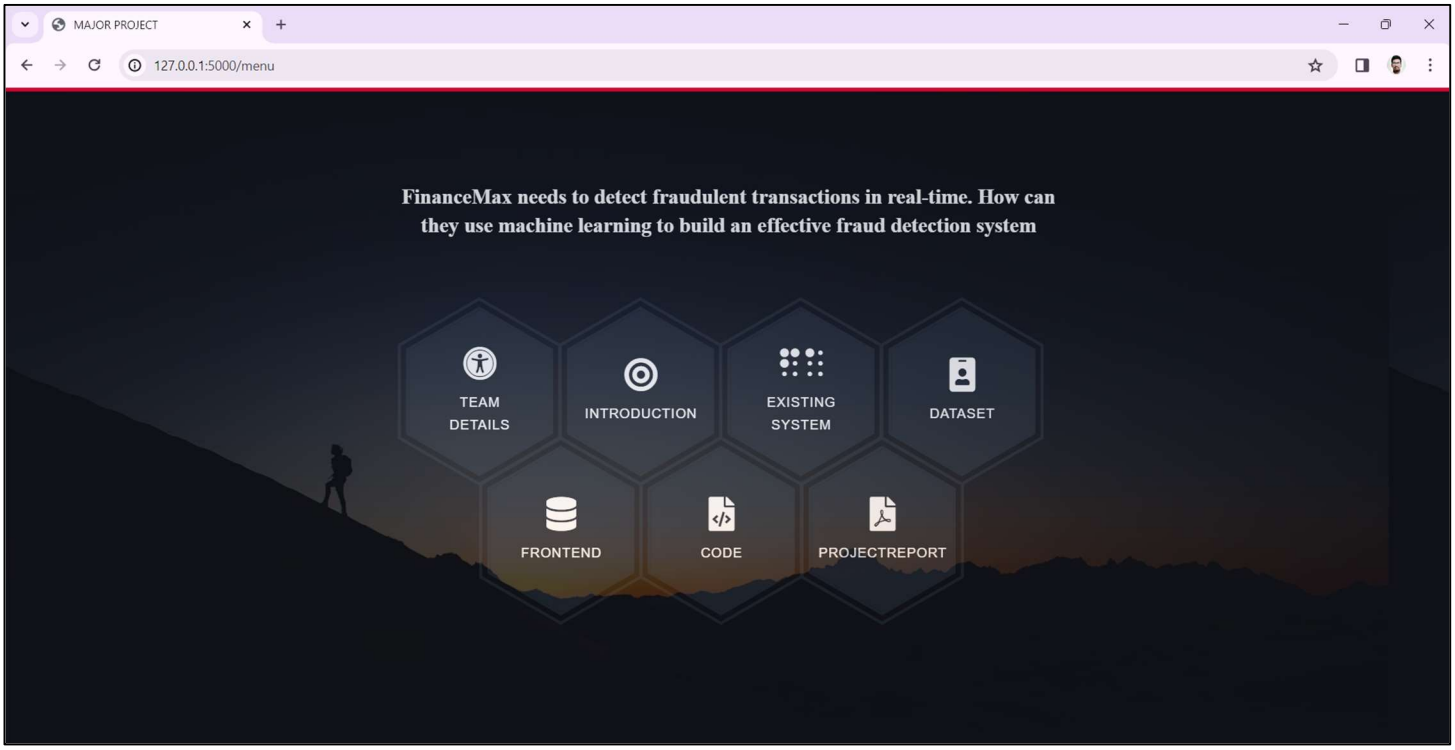


Fig.8.1 Home Page

User Registration:

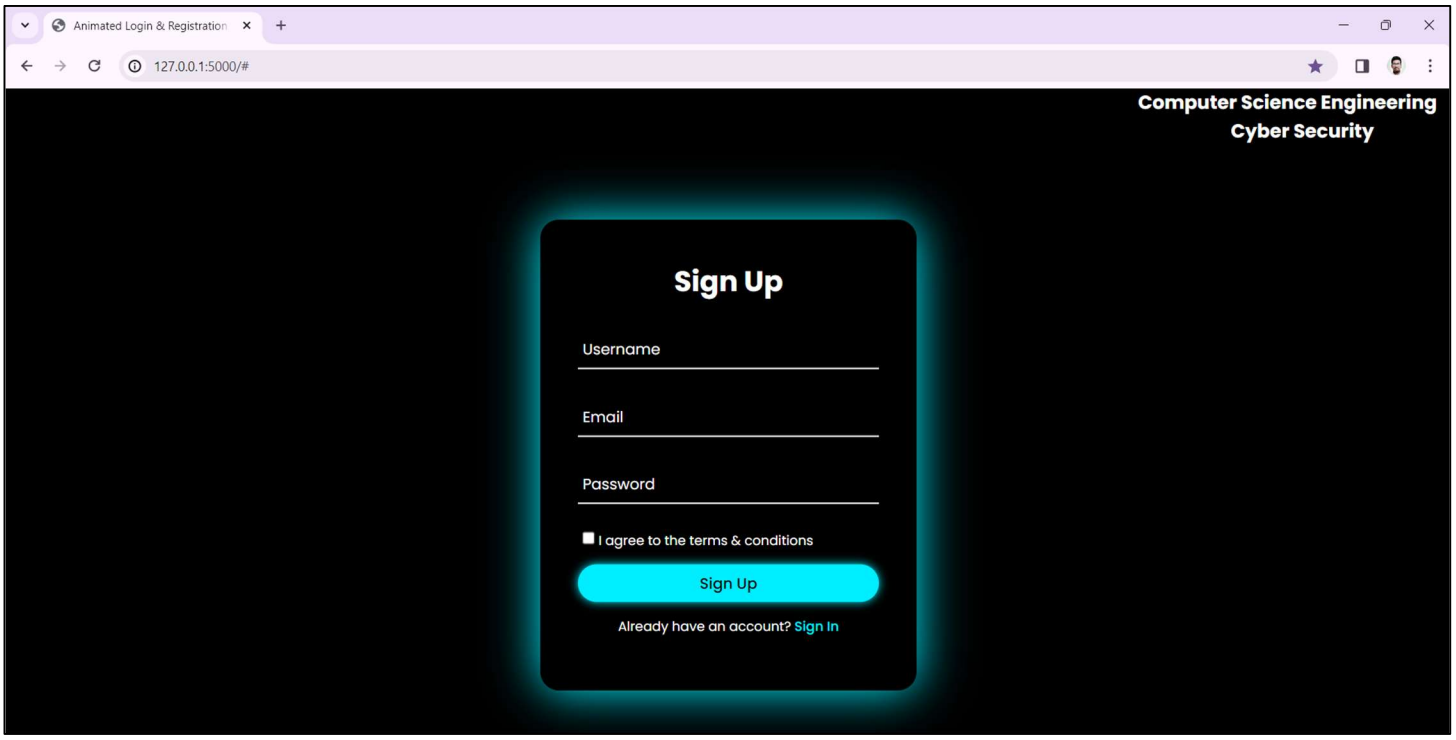


Fig.8.2 Sign Up Page

User Sign In:

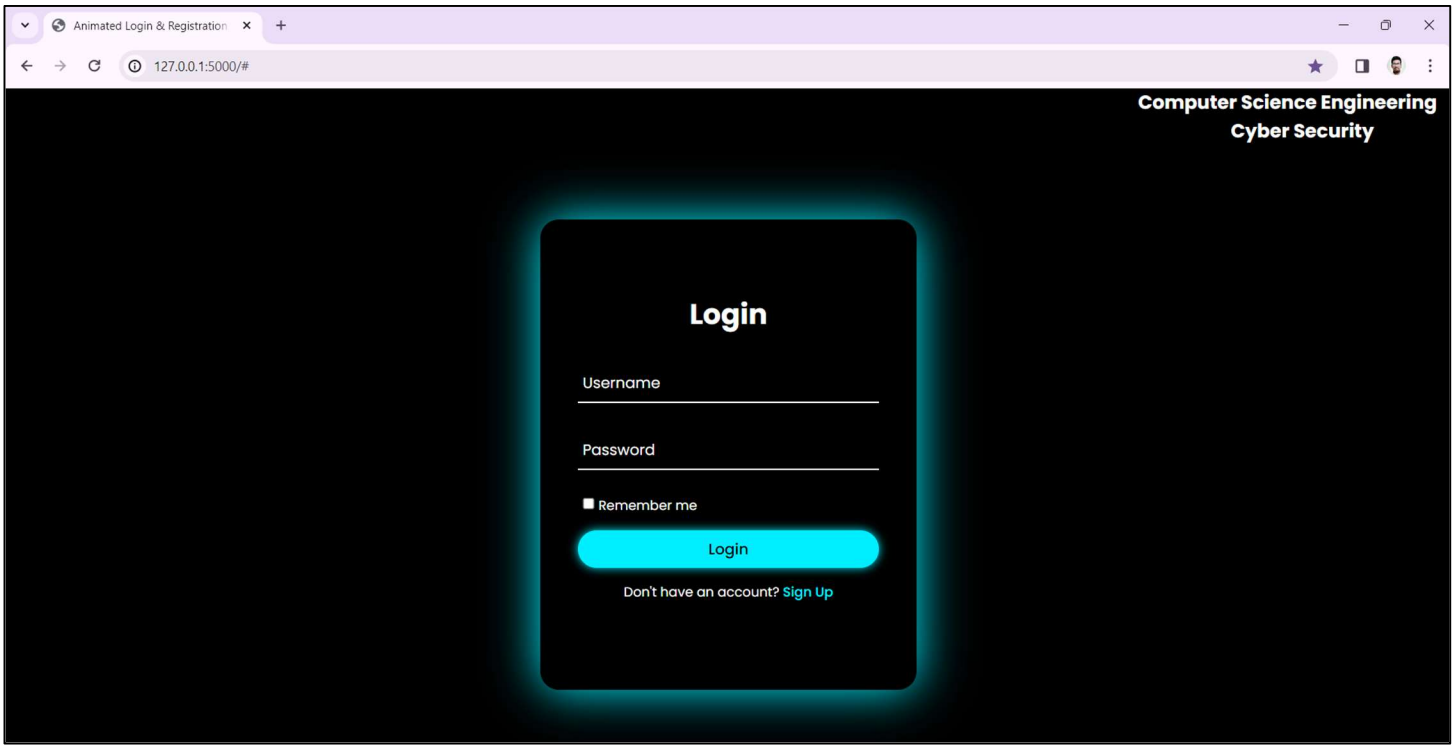


Fig.8.3 Login Page

User Database:

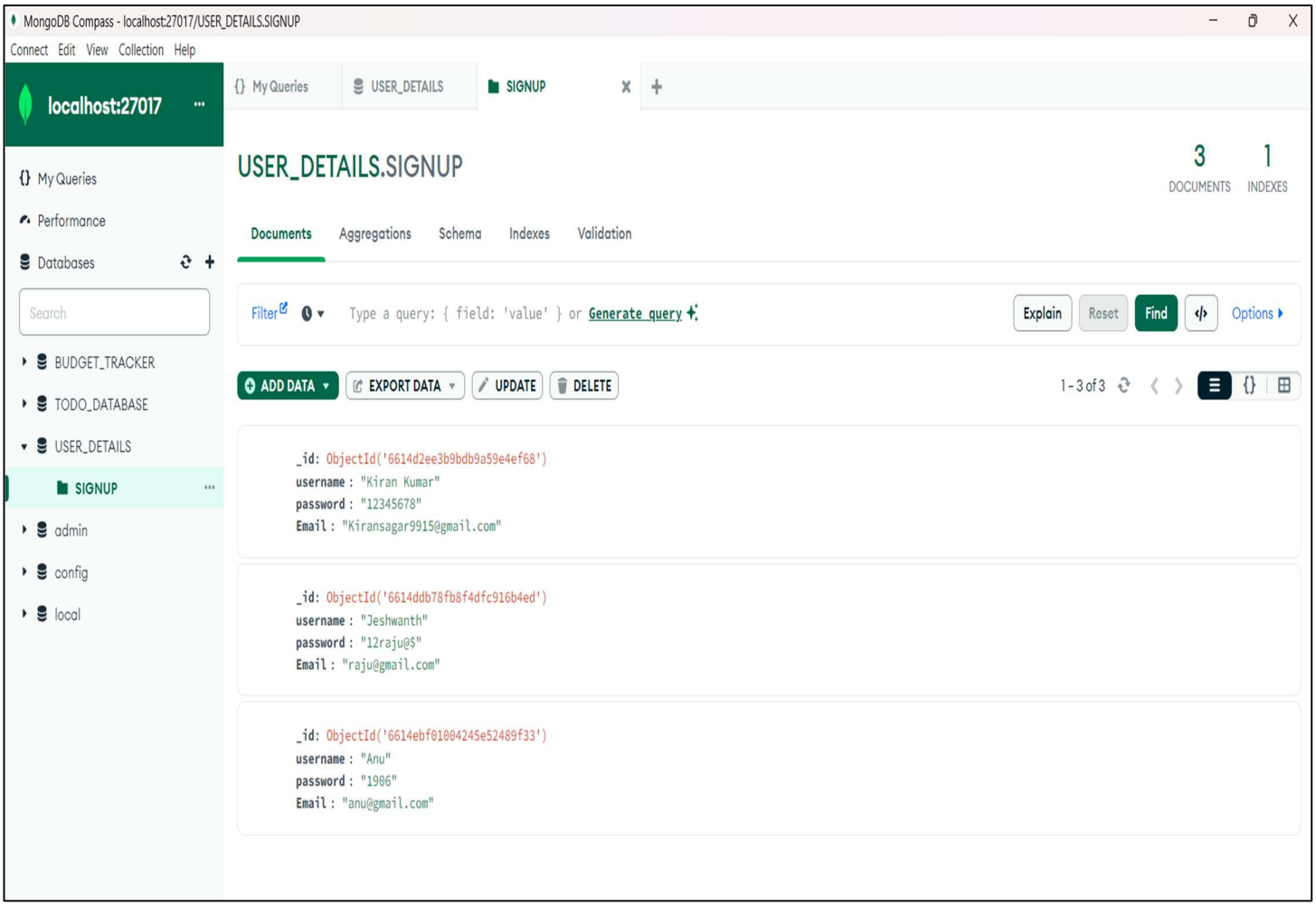
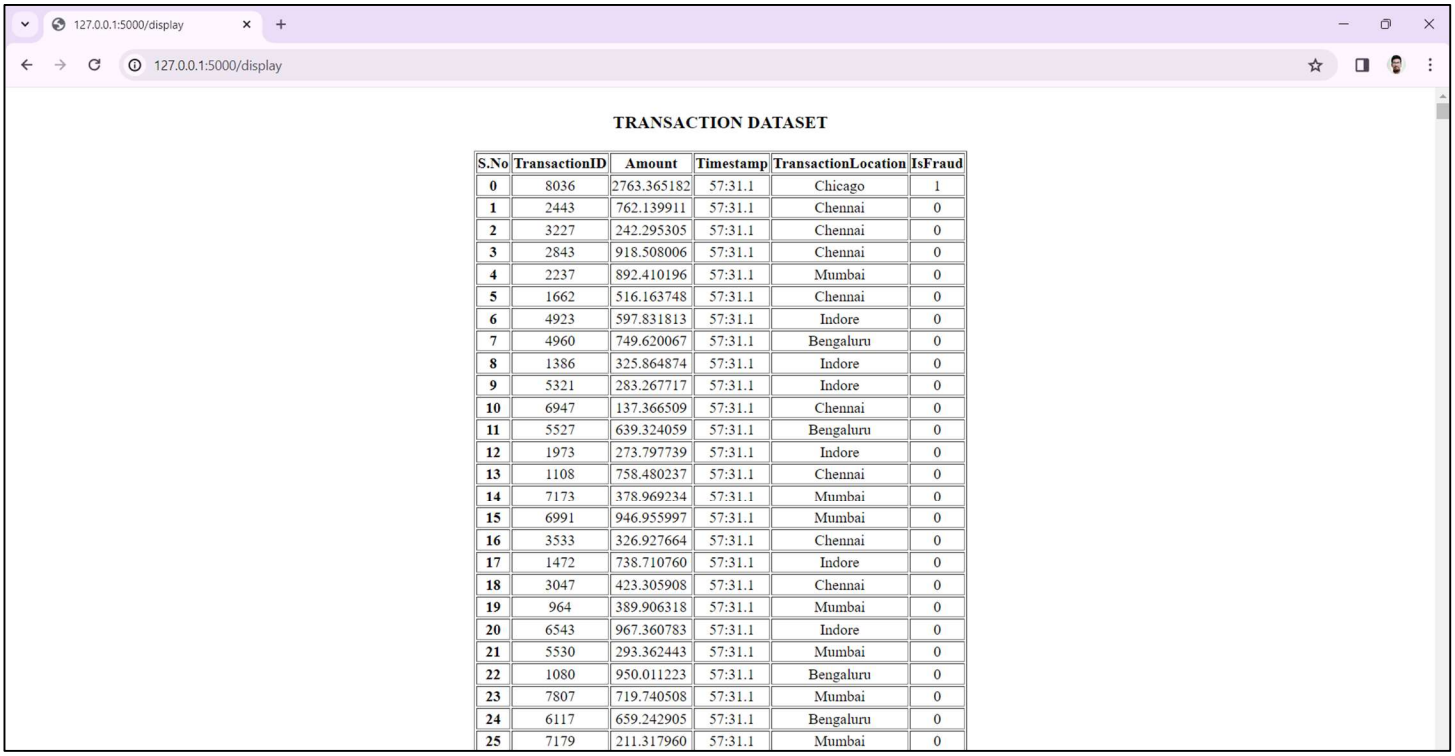


Fig.8.4 Database

Dataset:



TRANSACTION DATASET

S.No	TransactionID	Amount	Timestamp	TransactionLocation	IsFraud
0	8036	2763.365182	57:31.1	Chicago	1
1	2443	762.139911	57:31.1	Chennai	0
2	3227	242.295305	57:31.1	Chennai	0
3	2843	918.508006	57:31.1	Chennai	0
4	2237	892.410196	57:31.1	Mumbai	0
5	1662	516.163748	57:31.1	Chennai	0
6	4923	597.831813	57:31.1	Indore	0
7	4960	749.620067	57:31.1	Bengaluru	0
8	1386	325.864874	57:31.1	Indore	0
9	5321	283.267717	57:31.1	Indore	0
10	6947	137.366509	57:31.1	Chennai	0
11	5527	639.324059	57:31.1	Bengaluru	0
12	1973	273.797739	57:31.1	Indore	0
13	1108	758.480237	57:31.1	Chennai	0
14	7173	378.969234	57:31.1	Mumbai	0
15	6991	946.955997	57:31.1	Mumbai	0
16	3533	326.927664	57:31.1	Chennai	0
17	1472	738.710760	57:31.1	Indore	0
18	3047	423.305908	57:31.1	Chennai	0
19	964	389.906318	57:31.1	Mumbai	0
20	6543	967.360783	57:31.1	Indore	0
21	5530	293.362443	57:31.1	Mumbai	0
22	1080	950.011223	57:31.1	Bengaluru	0
23	7807	719.740508	57:31.1	Mumbai	0
24	6117	659.242905	57:31.1	Bengaluru	0
25	7179	211.317960	57:31.1	Mumbai	0

Fig.8.5 Dataset

Predict Results True:

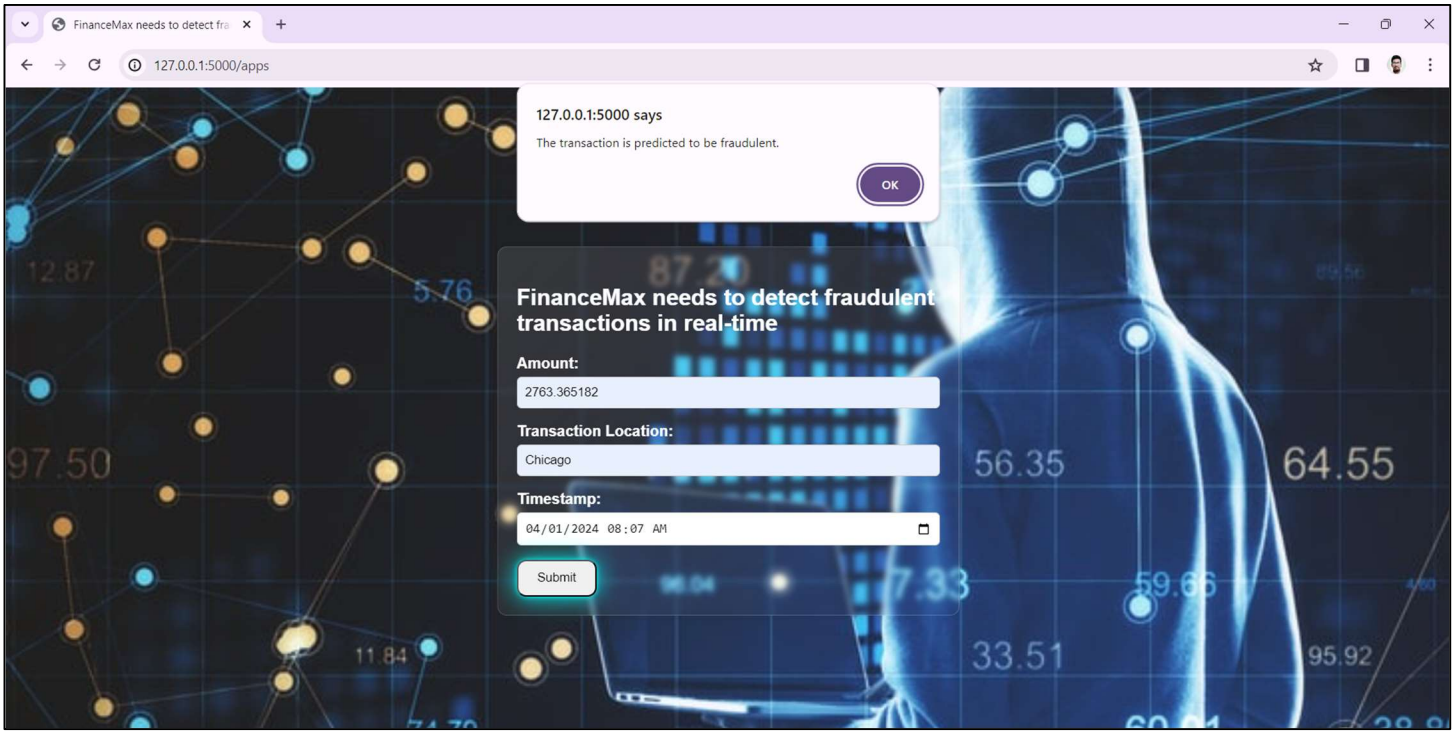


Fig.8.6 Fraudulent Prediction

Predict Results True:

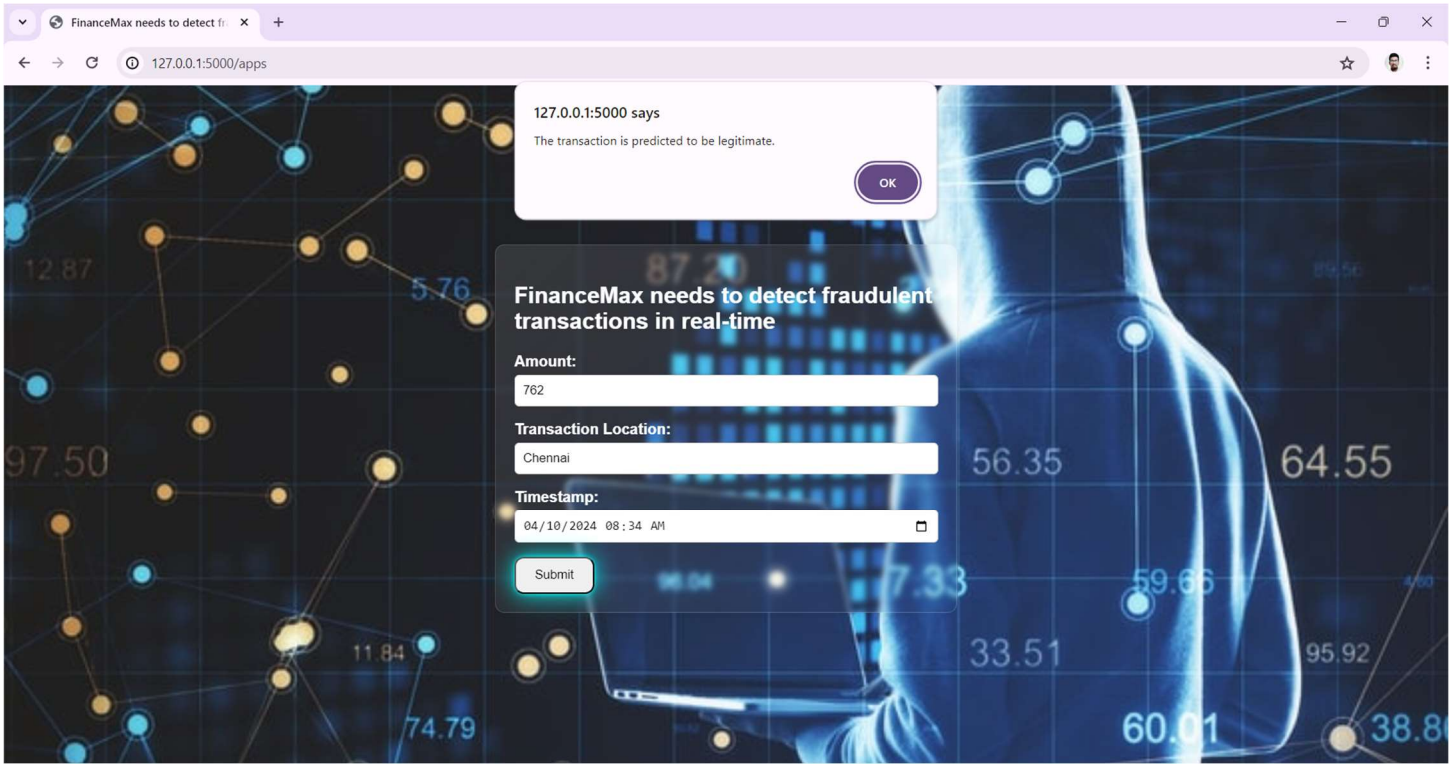


Fig.8.7 Legitimate Prediction

CHAPTER 9 – CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION:

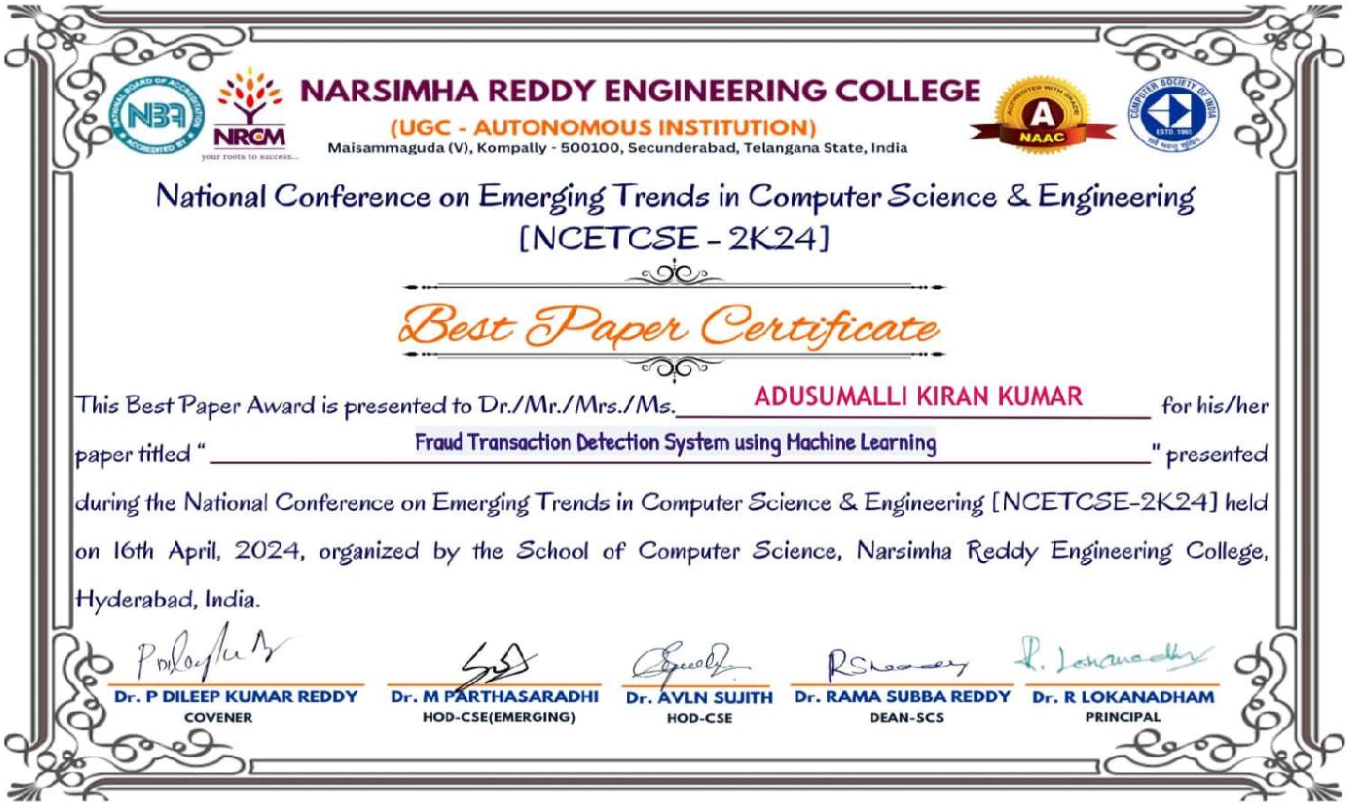
Various machine learning algorithms can be used for fraud detection, including logistic regression, random forest, SVM. The choice of algorithm depends on the specific characteristics of the data and the nature of the fraud. In conclusion, the use of machine learning algorithms, such as logistic regression, random forest, and self-organizing maps, has significantly enhanced the detection of fraudulent transactions in real-time. By leveraging these algorithms, financial institutions can better protect their customers and uphold the integrity of their systems.

FUTURE ENHANCEMENT:

Further research on this topic to improve and design better models to improve accuracy and generate reviews.

REFERENCES

1. Ramakrishna, Arvind, et al. "Fraud Detection in Indian Financial Transactions Using Machine Learning." International Journal of Computer Applications, vol. 186, no. 8, 2018, pp. 1-5.
2. Chandrasekaran, Vignesh, and S. Balamurugan. "Real-Time Fraud Detection System for Indian Financial Transactions Using Deep Learning." International Journal of Recent Technology and Engineering, vol. 8, no. 3, 2019, pp. 1550-1553.
3. Gupta, Priyanka, and Priyanka Singh. "Real-Time Fraud Detection in Indian Banking Transactions: A Case Study." International Journal of Engineering and Management Research, vol. 8, no. 6, 2018, pp. 162-166.
4. Pradhan, Soumya, et al. "Fraud Detection in Indian Financial Transactions: A Comparative Study of Machine Learning Algorithms." International Journal of Computer Sciences and Engineering, vol. 7, no. 8, 2019, pp. 152-156.
5. Financial Fraud Detection Based on Machine Learning: A Systematic Literature Review (2022)**.** This study reviews existing machine learning methods applied to financial transaction fraud detection. It provides insights into ML-based approaches, datasets, and predicting fraudulent activities in financial transactions
6. https://muthu.co/understanding-the-classification-report-in-sklearn/
7. https://developer.twitter.com/en/apps
8. https://text-processing.com/demo/tokenize/
9. https://towards/cybersecurity.com/support-vector-machine-introduction-tomachine-learning-algorithms-934a444fca47
10. https://towards/cybersecurity.com/random-forest-classifier-81d512f50a7c



NARSIMHA REDDY ENGINEERING COLLEGE

(UGC - AUTONOMOUS INSTITUTION)

Maisammaguda (V), Kompally - 500100, Secunderabad, Telangana State, India

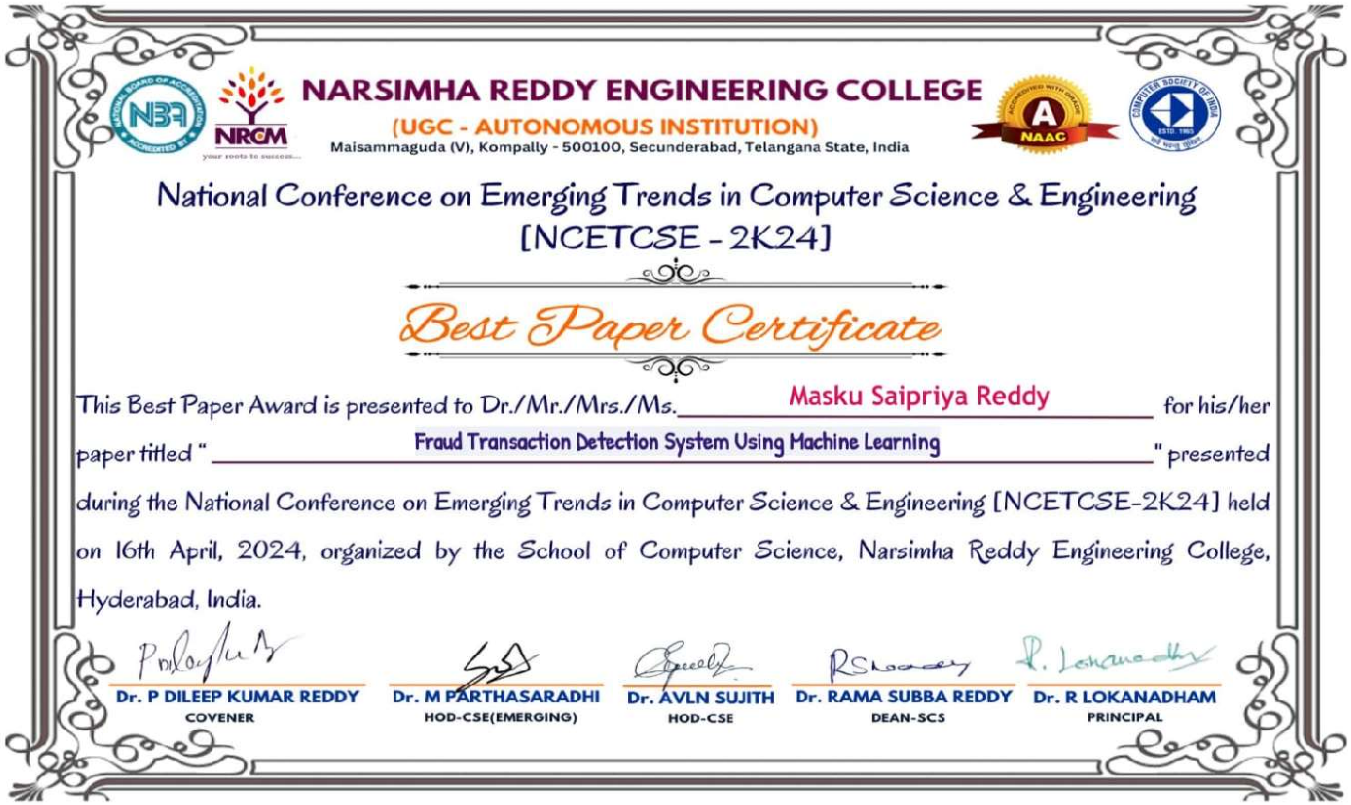
National Conference on Emerging Trends in Computer Science & Engineering

[NCETCSE - 2K24]

Best Paper Certificate

This Best Paper Award is presented to Dr./Mr./Mrs./Ms. ADUSUMALLI KIRAN KUMAR for his/her paper titled " Fraud Transaction Detection System using Machine Learning " presented during the National Conference on Emerging Trends in Computer Science & Engineering [NCETCSE-2K24] held on 16th April, 2024, organized by the School of Computer Science, Narsimha Reddy Engineering College, Hyderabad, India.

Dr. P DILEEP KUMAR REDDY	Dr. M PARTHASARADHI	Dr. AVLN SUJITH	Dr. RAMA SUBBA REDDY	Dr. R LOKANADHAM
COVENER	HOD-CSE(EMERGING)	HOD-CSE	DEAN-SCS	PRINCIPAL



NARSIMHA REDDY ENGINEERING COLLEGE

(UGC - AUTONOMOUS INSTITUTION)

Maisammaguda (V), Kompally - 500100, Secunderabad, Telangana State, India

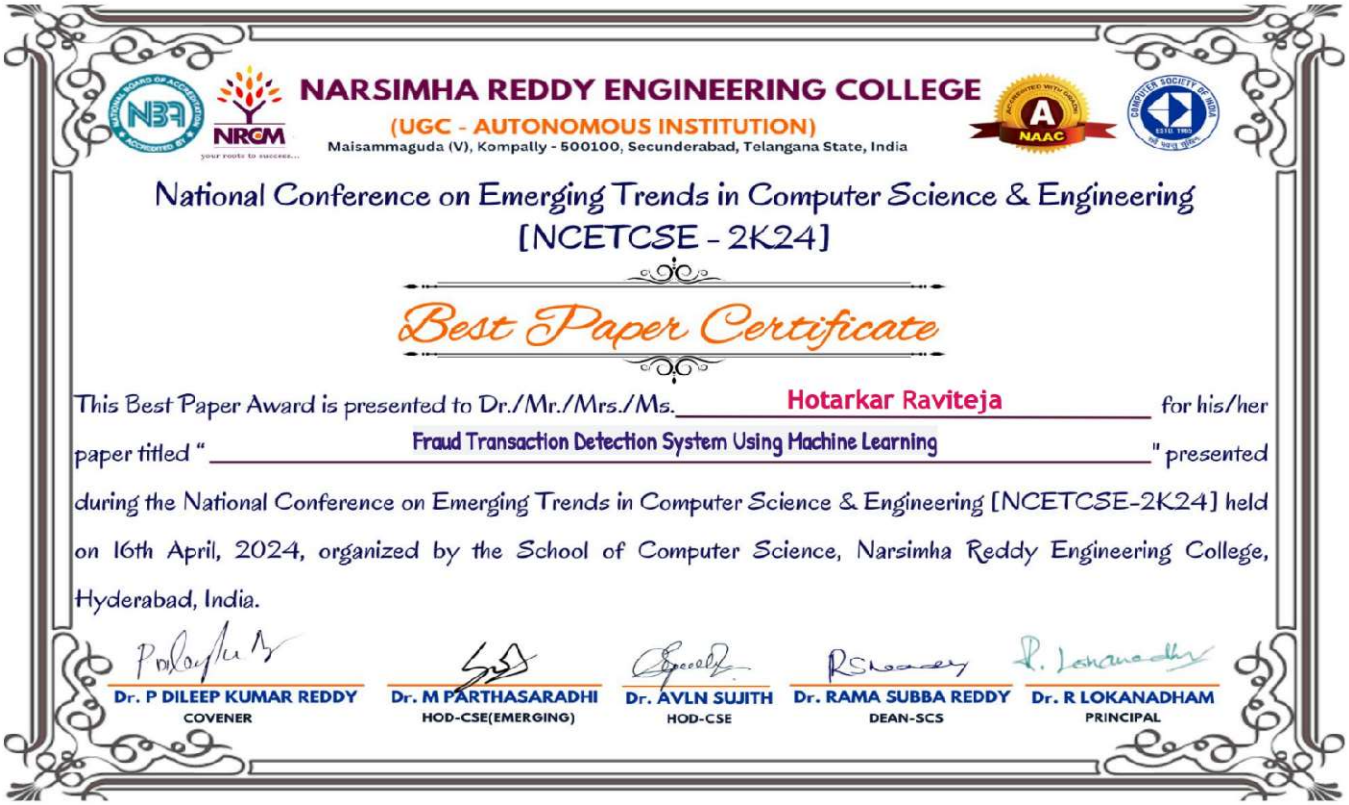
National Conference on Emerging Trends in Computer Science & Engineering

[NCETCSE - 2K24]

Best Paper Certificate

This Best Paper Award is presented to Dr./Mr./Mrs./Ms. Masku Saipriya Reddy for his/her paper titled " Fraud Transaction Detection System Using Machine Learning " presented during the National Conference on Emerging Trends in Computer Science & Engineering [NCETCSE-2K24] held on 16th April, 2024, organized by the School of Computer Science, Narsimha Reddy Engineering College, Hyderabad, India.

Dr. P DILEEP KUMAR REDDY	Dr. M PARTHASARADHI	Dr. AVLN SUJITH	Dr. RAMA SUBBA REDDY	Dr. R LOKANADHAM
COVENER	HOD-CSE(EMERGING)	HOD-CSE	DEAN-SCS	PRINCIPAL



NARSIMHA REDDY ENGINEERING COLLEGE

(UGC - AUTONOMOUS INSTITUTION)

Maisammaguda (V), Kompally - 500100, Secunderabad, Telangana State, India

National Conference on Emerging Trends in Computer Science & Engineering

[NCETCSE - 2K24]

Best Paper Certificate

This Best Paper Award is presented to Dr./Mr./Mrs./Ms. Hotarkar Raviteja for his/her paper titled " Fraud Transaction Detection System Using Machine Learning " presented during the National Conference on Emerging Trends in Computer Science & Engineering [NCETCSE-2K24] held on 16th April, 2024, organized by the School of Computer Science, Narsimha Reddy Engineering College, Hyderabad, India.

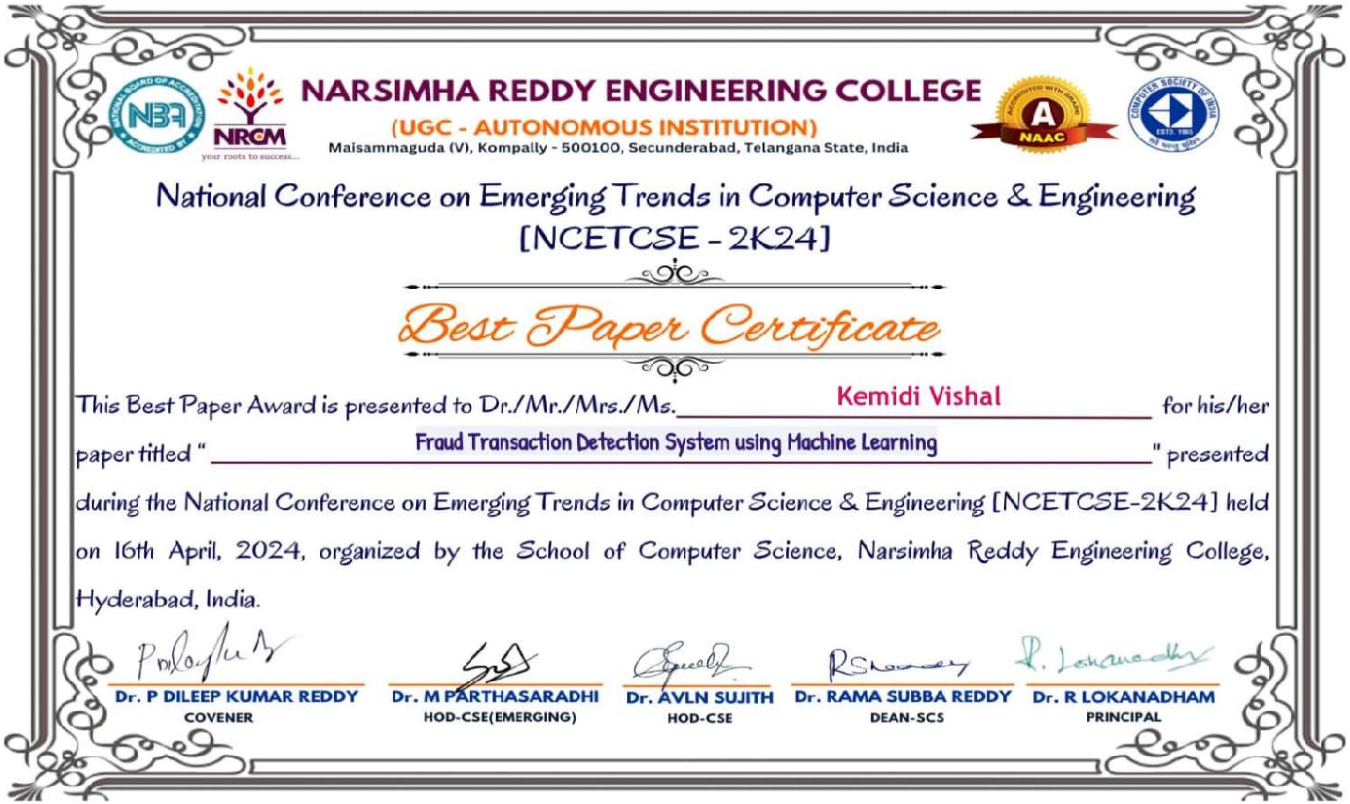
Dr. P DILEEP KUMAR REDDY, COVENER

Dr. M PARTHASARADHI, HOD-CSE(EMERGING)

Dr. AVLN SUJITH, HOD-CSE

Dr. RAMA SUBBA REDDY, DEAN-SCS

Dr. R LOKANADHAM, PRINCIPAL



NARSIMHA REDDY ENGINEERING COLLEGE

(UGC - AUTONOMOUS INSTITUTION)

Maisammaguda (V), Kompally - 500100, Secunderabad, Telangana State, India

National Conference on Emerging Trends in Computer Science & Engineering

[NCETCSE - 2K24]

Best Paper Certificate

This Best Paper Award is presented to Dr./Mr./Mrs./Ms. Kemidi Vishal for his/her paper titled "Fraud Transaction Detection System using Machine Learning" presented during the National Conference on Emerging Trends in Computer Science & Engineering [NCETCSE-2K24] held on 16th April, 2024, organized by the School of Computer Science, Narsimha Reddy Engineering College, Hyderabad, India.

Dr. P DILEEP KUMAR REDDY	Dr. M PARTHASARADHI	Dr. AVLN SUJITH	Dr. RAMA SUBBA REDDY	Dr. R LOKANADHAM
COVENER	HOD-CSE(EMERGING)	HOD-CSE	DEAN-SCS	PRINCIPAL