A

Mini Project

On

# PHISHING WEBSITE DETECTION USING MACHINE LEARNING

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

N. SOWMYA (207R1A05A3)

V. CHIDRUP (207R1A05B6)

CH. KUSHAL (207R1A0588)

Under the Guidance of

**SABA SULTANA**

(Assistant Professor)


**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CMR TECHNICAL CAMPUS**

**UGC AUTONOMOUS**

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

**2020-2024**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**


# CERTIFICATE

This is to certify that the project entitled **"PHISHING WEBSITE DETECTION USING MACHINE LEARNING"** being submitted by **N. SOWMYA (207R1A05A3), V. CHIDRUP (207R1A05B6) & CH. KUSHAL (207R1A0588)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

**Saba Sultana**
(Assistant Professor)
INTERNAL GUIDE

**Dr. A. Raji Reddy**
DIRECTOR

**Dr. K. Srujan Raju**
HOD

**EXTERNAL EXAMINER**

**Submitted for viva voce Examination held on _____**

# ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **Saba Sultana,** Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **Dr. J. Narasimharao, G. Vinesh Shanker, Ms. Shilpa, Dr. K. Maheswari & Saba Sultana** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

**N. SOWMYA (207R1A05A3)**

**V. CHIDRUP (207R1A05B6)**

**CH. KUSHAL (207R1A0588)**

# ABSTRACT

Phishing attack is a simplest way to obtain sensitive information from innocent users. Aim of the phishers is to acquire critical information like username, password and bank account details. Cyber security persons are now looking for trustworthy and steady detection techniques for phishing websites detection. This paper deals with machine learning technology for detection of phishing URLs by extracting and analyzing various features of legitimate and phishing URLs. Decision Tree, random forest and Support vector machine algorithms are used to detect phishing websites. Aim of the paper is to detect phishing URLs as well as using light gbm and svm algorithm.

# LIST OF FIGURES/TABLES

# LIST OF SCREENSHOTS

# TABLE OF CONTENTS

# TABLE OF CONTENTS

# 1. INTRODUCTION

# 1. INTRODUCTION

## 1.1 PROJECT SCOPE

This project is titled “Phishing Website Detection using Machine Learning”. It aims to develop a machine learning model for detecting phishing URLs. It is essential because phishing attacks keep growing. They trick people by pretending to be real websites. ML helps find these fake sites fast and can work for lots of people. It learns to spot new tricks, reduces mistakes, and is cost-effective. This project works alongside other security efforts, protecting against data theft and fraud, and keeping people safe online.

## 1.2 PROJECT PURPOSE

The purpose of a project focused on phishing website detection using machine learning is to develop a system that can automatically identify and flag websites or web pages that are designed to deceive users and steal their sensitive information. Phishing is a cybercrime tactic where attackers create fake websites that mimic legitimate ones to trick users into divulging personal information such as login credentials, credit card details, or other sensitive data. Detecting and preventing phishing attacks is crucial for protecting individuals, businesses, and organizations from financial losses and data breaches.

## 1.3 PROJECT FEATURES

The main features of this project are that this model classifies phishing website and legitimate website by using ml algorithm. A phishing URL and its accompanying website have various characteristics that distinguish them from harmful URLs. Using Content-Based Features to construct a quick detection mechanism capable of analyzing a huge number of domains may not be feasible. Page Based Features are not very effective when analyzing registered domains. As a result, the features that the detection mechanism will use are determined by the detection mechanism's purpose.

# 2. SYSTEM ANALYSIS

# 2. SYSTEM ANALYSIS

## 2. SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

## 2.1 PROBLEM DEFINITION

The problem of phishing website detection using machine learning revolves around the critical need to protect individuals and organizations from deceptive online threats. The goal is to develop an automated system that can accurately distinguish between genuine websites and phishing attempts by analyzing various features, patterns, and behaviors associated with these deceptive sites.

## 2.2 EXISTING SYSTEM

The existing system of phishing website detection using machine learning represents a significant advancement in cybersecurity. Traditional methods of identifying phishing websites relied heavily on static rules and blacklists, which struggled to keep up with the rapidly evolving tactics employed by cybercriminals. These systems analyze a wide range of features, including URL characteristics, domain attributes, webpage content, and user interactions, to build predictive models that can accurately classify websites as either legitimate or fraudulent.

### 2.2.1 DISADVANTAGES OF EXISTING SYSTEM

Following are the disadvantages of existing system:

- Limited training data
- False positives
- Black box models
- Overfitting

## 2.3 PROPOSED SYSTEM

This project presents a system that recognizes people in video sequences using image information. More specifically we are interested in locating shots where some particular person appears in the image while talking, so that both face and voice are out of use. Examples of these shots include taped footage of news anchors, and head and shoulders sequences of people being interviewed. Moreover recording conditions for this type of shots are usually more controlled, making the recognition task more accurate.

### 2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

There are several advantages to the proposed system of phishing website detection using machine learning:

- Increased accuracy
- Scalability
- Flexibility

## 2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

### 2.4.1 ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on a project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also, all the resources are already available, it gives an indication that the system is economically possible for development.

### 2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

### 2.4.3 BEHAVIORAL FEASIBILITY

This includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible

## 2.5 HARDWARE & SOFTWARE REQUIREMENTS

### 2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

- Processor: Intel Dual Core I5 and above
- Hard disk: 8GB and above
- RAM: 8GB and above
- Input devices: Keyboard, mouse.

### 2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements,

- Operating system: Windows 8 and above
- Languages: Python, Html, CSS
- Tools: Python IDLE 3.7 version, Anaconda - Jupyter, Spyder

# 3. ARCHITECTURE

# 3. ARCHITECTURE

## 3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

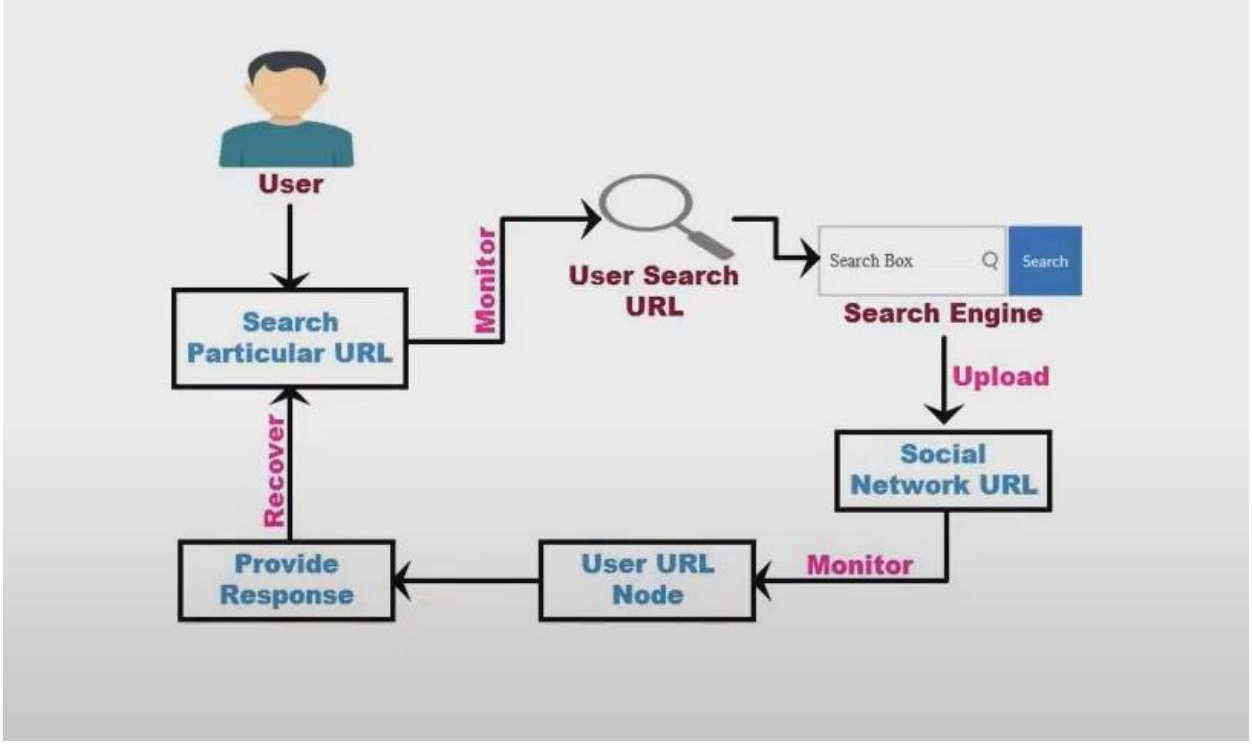



Figure 3.1: Project Architecture of Phishing Website Detection

## 3.2 DESCRIPTION

In this project a model is built using SVM and LightGBM algorithms which classifies the URLs as either phishing or genuine websites. User needs to enter the URL in the form of text and click on submit. The model will predict the output as phishing or genuine and displays it on the screen. By clicking on run SVM and run LightGBM button the algorithms are applied and accuracies of both are displayed.

## 3.3 USE CASE DIAGRAM

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

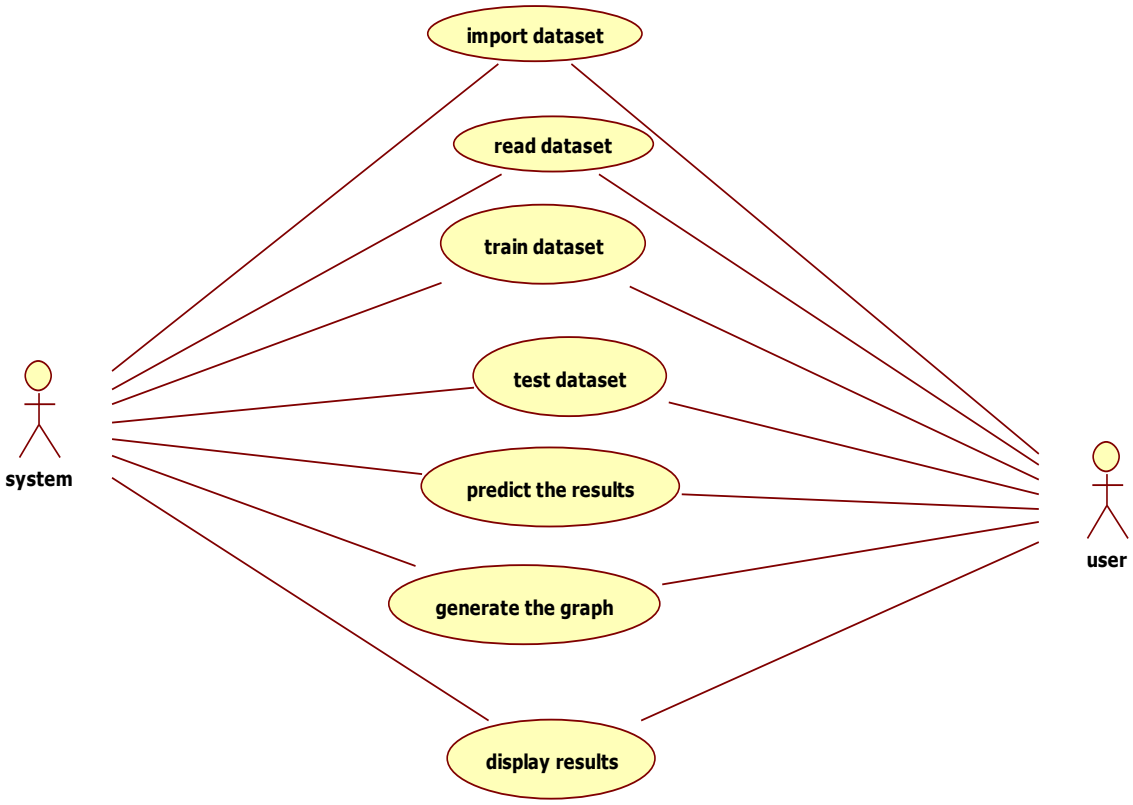



Figure 3.2: Use Case Diagram for Phishing Website Detection

## 3.4 CLASS DIAGRAM

Class diagram 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 objects.

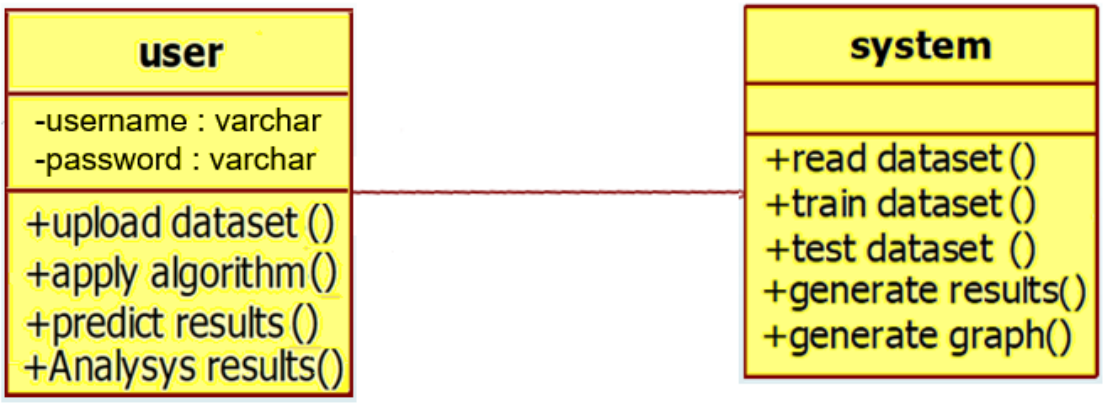



Figure 3.3: Class Diagram for Phishing Website Detection

## 3.5 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

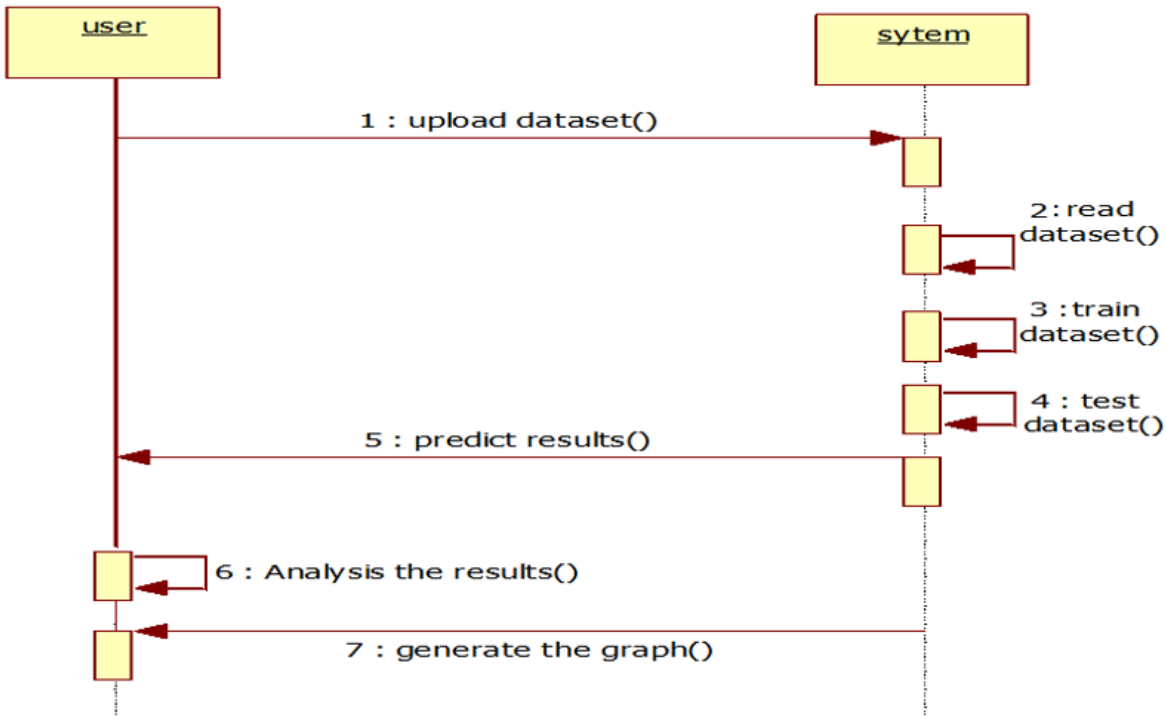



Figure 3.4: Sequence Diagram for Phishing Website Detection

## 3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

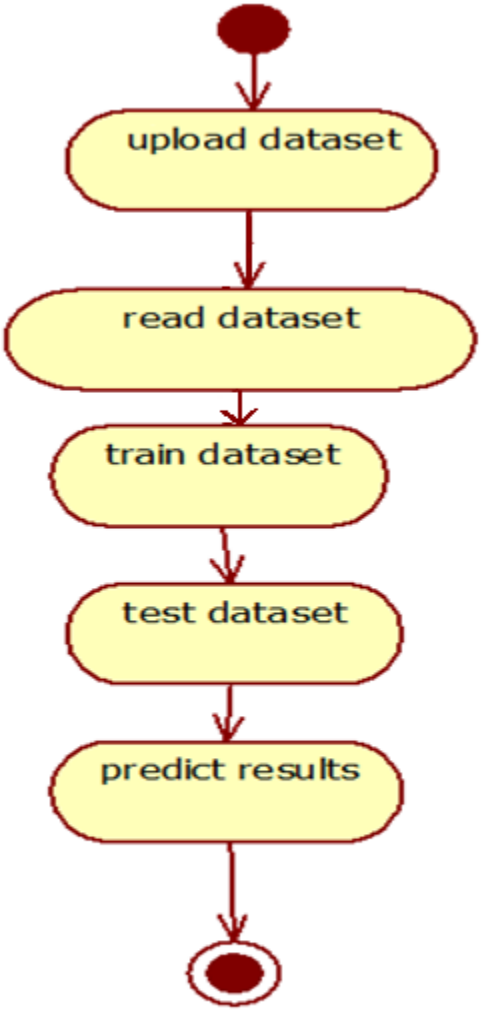



Figure 3.5: Activity Diagram for Phishing Website Detection

# 4. IMPLEMENTATION

## 4.1 SAMPLE CODE

```
from django.shortcuts import render
from django.template import RequestContext
from django.contrib import messages
from django.http import HttpResponse
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import pickle
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import accuracy_score
from sklearn import svm
from lightgbm import LGBMClassifier
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import confusion_matrix
import seaborn as sns

global precision, recall, fscore, accuracy
```

```
X = np.load("model/X.txt.npy")
Y = np.load("model/Y.txt.npy")
indices = np.arange(X.shape[0])
np.random.shuffle(indices)
X = X[indices]
Y = Y[indices]

with open('model/tfidf.txt', 'rb') as file:
    tfidf = pickle.load(file)
file.close()
X = tfidf.fit_transform(X).toarray()
print(X.shape)
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)

if os.path.exists('model/svm.txt'):
    with open('model/svm.txt', 'rb') as file:
        svm_cls = pickle.load(file)
    file.close()
else:
    svm_cls = svm.SVC()
    svm_cls.fit(X_train, y_train)
    with open('model/svm.txt', 'wb') as file:
        pickle.dump(svm_cls, file)
    file.close()

if os.path.exists('model/lgbm.txt'):
    with open('model/lgbm.txt', 'rb') as file:
        lgbm_cls = pickle.load(file)
    file.close()
```

```
else:
    lgbm_cls = LGBMClassifier()
    lgbm_cls.fit(X_train, y_train)
    with open('model/lgbm.txt', 'wb') as file:
        pickle.dump(lgbm_cls, file)
    file.close()

with open('model/rf.txt', 'rb') as file:
    rf_cls = pickle.load(file)
file.close()

def RunSVM(request):
    if request.method == 'GET':
        global precision, recall, fscore, accuracy
        global X_train, X_test, y_train, y_test
        precision = []
        accuracy = []
        fscore = []
        recall = []
        predict = svm_cls.predict(X_test)
        acc = accuracy_score(y_test,predict)*100
        p = precision_score(y_test,predict,average='macro') * 100
        r = recall_score(y_test,predict,average='macro') * 100
        f = f1_score(y_test,predict,average='macro') * 100
        precision.append(p)
        recall.append(r)
        fscore.append(f)
        accuracy.append(acc)
```

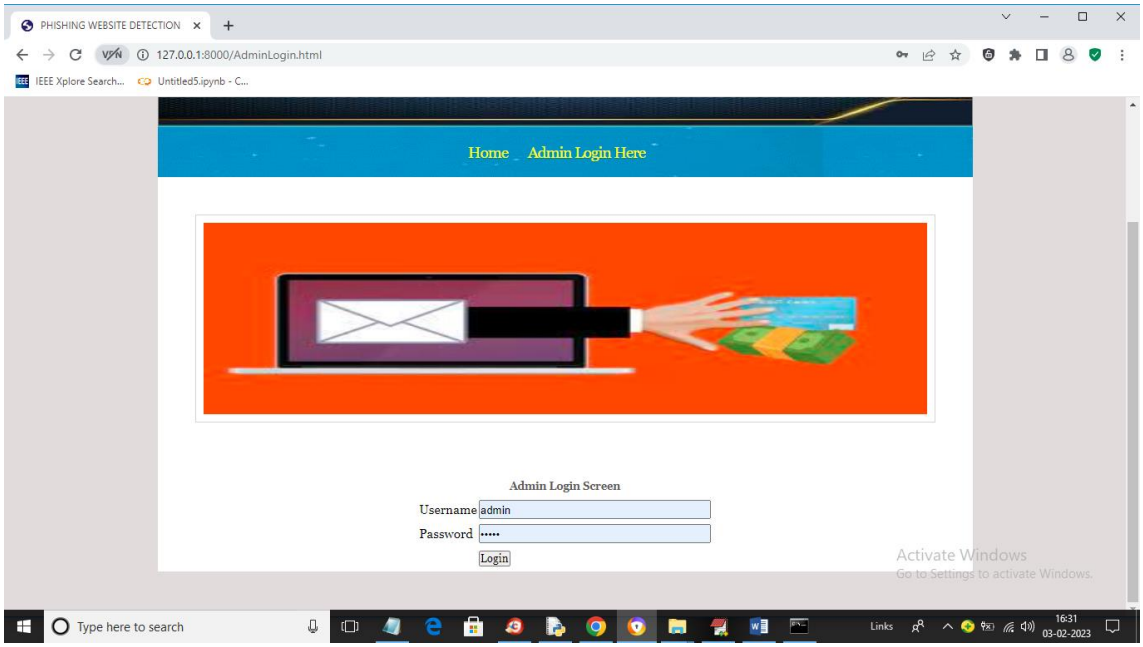
```
        output = ""
        output+='<tr><td><font size="" color="black">SVM</td>'
        output+='<td><font size="" color="black">'+str(accuracy[0])+'</td>'
        output+='<td><font size="" color="black">'+str(precision[0])+'</td>'
        output+='<td><font size="" color="black">'+str(recall[0])+'</td>'
        output+='<td><font size="" color="black">'+str(fscore[0])+'</td>'

        LABELS = ['Normal URL','Phishing URL']
        conf_matrix = confusion_matrix(y_test, predict)
        plt.figure(figsize =(6, 6))
        ax = sns.heatmap(conf_matrix, xticklabels = LABELS, yticklabels = LABELS, annot =
True, cmap="viridis" ,fmt ="g");
        ax.set_ylim([0,2])
        plt.title("SVM Confusion matrix")
        plt.ylabel('True class')
        plt.xlabel('Predicted class')


 plt.show()
        context= {'data':output}
        return render(request, 'ViewOutput.html', context)

def RunLGBM(request):
    if request.method == 'GET':
        global precision, recall, fscore, accuracy
        global X_train, X_test, y_train, y_test
        predict = lgbm_cls.predict(X_test)
        acc = accuracy_score(y_test,predict)*100
        p = precision_score(y_test,predict,average='macro') * 100
        r = recall_score(y_test,predict,average='macro') * 100
        f = f1_score(y_test,predict,average='macro') * 100
```

```
        precision.append(p)
        recall.append(r)
        fscore.append(f)
        accuracy.append(acc)
        output = ""
        output+='<tr><td><font size="" color="black">SVM</td>'
        output+='<td><font size="" color="black">'+str(accuracy[0])+'</td>'
        output+='<td><font size="" color="black">'+str(precision[0])+'</td>'
        output+='<td><font size="" color="black">'+str(recall[0])+'</td>'
        output+='<td><font size="" color="black">'+str(fscore[0])+'</td>'

        output+='<tr><td><font size="" color="black">Light GBM</td>'
        output+='<td><font size="" color="black">'+str(accuracy[1])+'</td>'
        output+='<td><font size="" color="black">'+str(precision[1])+'</td>'
        output+='<td><font size="" color="black">'+str(recall[1])+'</td>'
        output+='<td><font size="" color="black">'+str(fscore[1])+'</td>'

        LABELS = ['Normal URL','Phishing URL']
        conf_matrix = confusion_matrix(y_test, predict)


 plt.figure(figsize =(6, 6))
        ax = sns.heatmap(conf_matrix, xticklabels = LABELS, yticklabels = LABELS, annot =
True, cmap="viridis" ,fmt ="g");
        ax.set_ylim([0,2])
        plt.title("LightGBM Confusion matrix")
        plt.ylabel('True class')
        plt.xlabel('Predicted class')
        plt.show()
        context= {'data':output}
        return render(request, 'ViewOutput.html', context)
```

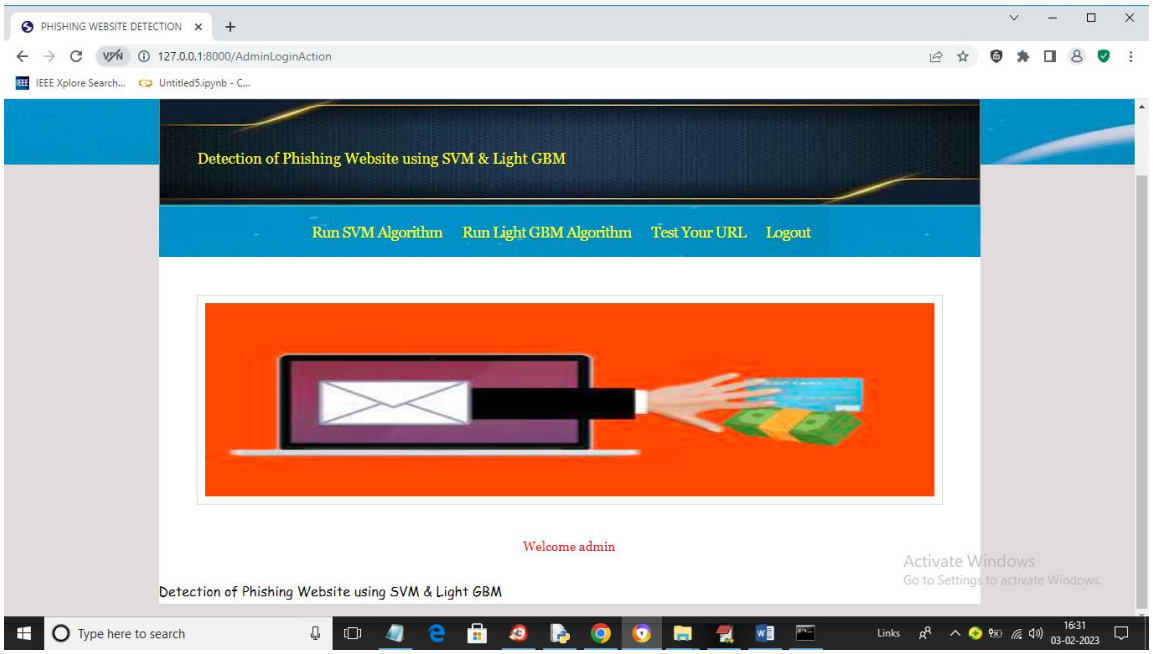
```
def getData(arr):
    data = ""
    for i in range(len(arr)):
        arr[i] = arr[i].strip()
        if len(arr[i]) > 0:
            data += arr[i]+" "
    return data.strip()

def PredictAction(request):
    if request.method == 'POST':
        global rf_cls, tfidf
        url_input = request.POST.get('t1', False)
        test = []
        arr = url_input.split("/")
        if len(arr) > 0:
            data = getData(arr)
            print(data)
            test.append(data)
            test = tfidf.transform(test).toarray()
            print(test)
            print(test.shape)
            predict = rf_cls.predict(test)

             print(predict)
            predict = predict[0]
            output = ""
            if predict == 0:
                output = url_input+" Given URL Predicted as Genuine"
            if predict == 1:
                output = url_input+" PHISHING Detected in Given URL"
            context= {'data':output}
            return render(request, 'Predict.html', context)
```

```
 else:
            context= {'data':"Entered URL is not valid"}
            return render(request, 'Predict.html', context)



def index(request):
    if request.method == 'GET':
       return render(request, 'index.html', {})

def Predict(request):
    if request.method == 'GET':
      return render(request, 'Predict.html', {})

def AdminLogin(request):
    if request.method == 'GET':
       return render(request, 'AdminLogin.html', {})

def AdminLoginAction(request):
    if request.method == 'POST':
        global userid
        user = request.POST.get('t1', False)
        password = request.POST.get('t2', False)
        if user == "admin" and password == "admin":
            context= {'data':'Welcome '+user}
            return render(request, 'AdminScreen.html', context)

 else:
            context= {'data':'Invalid Login'}
            return render(request, 'AdminLogin.html', context)
```

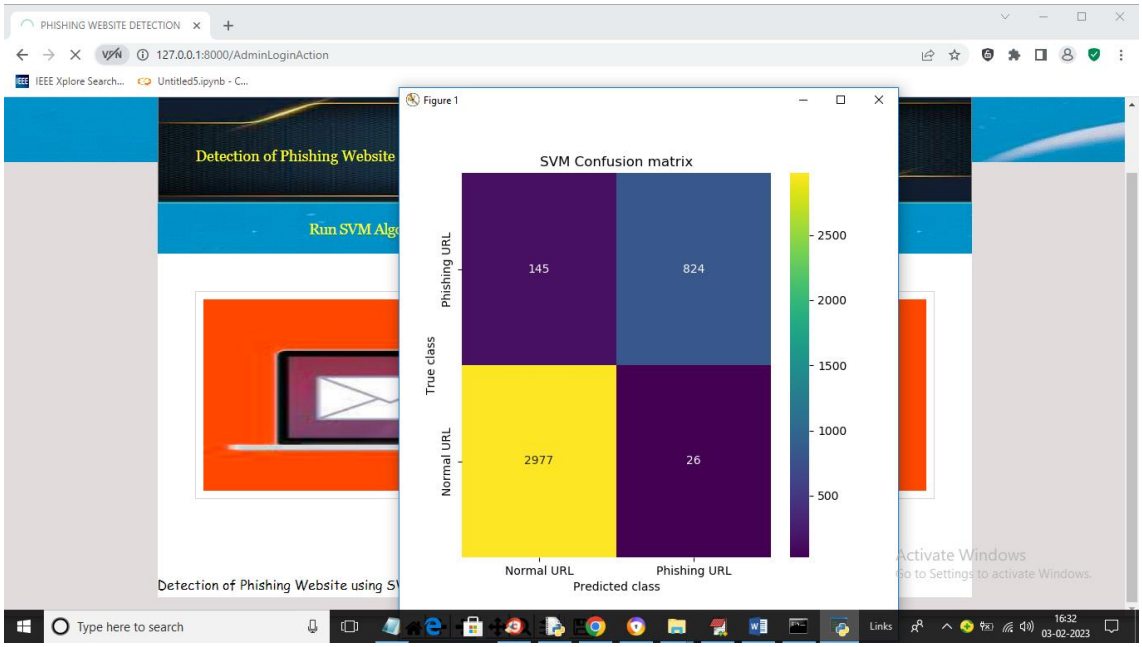
# 5. SCREENSHOTS



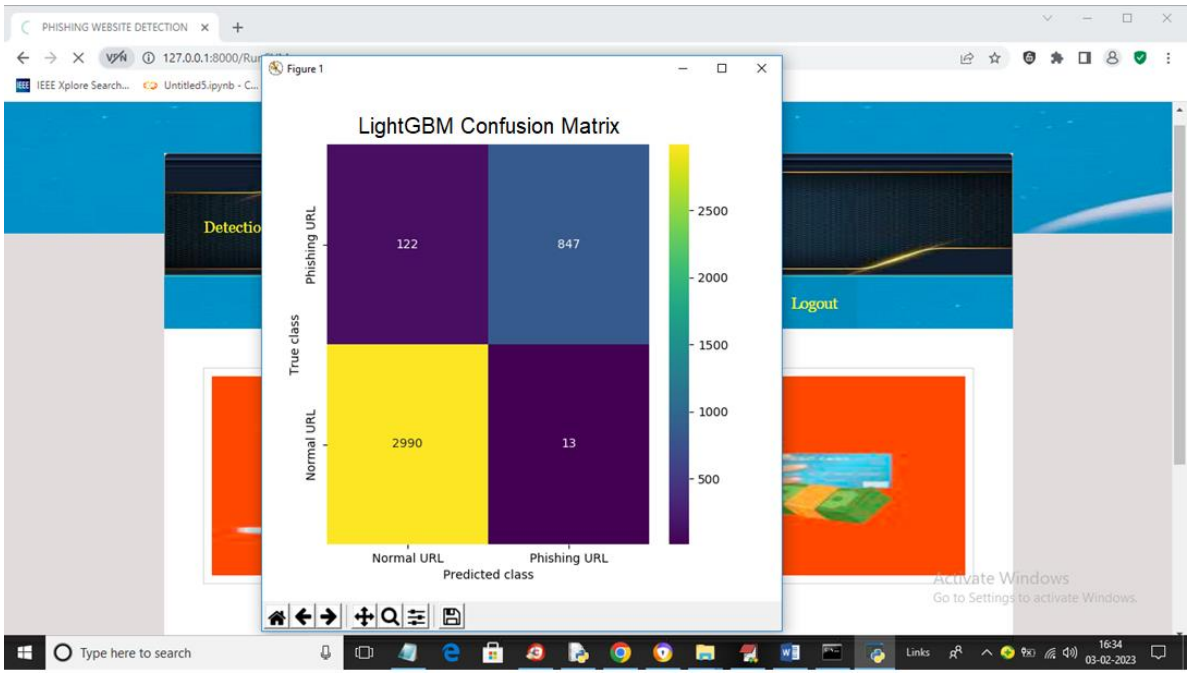

Screenshot 5.1: Login Window



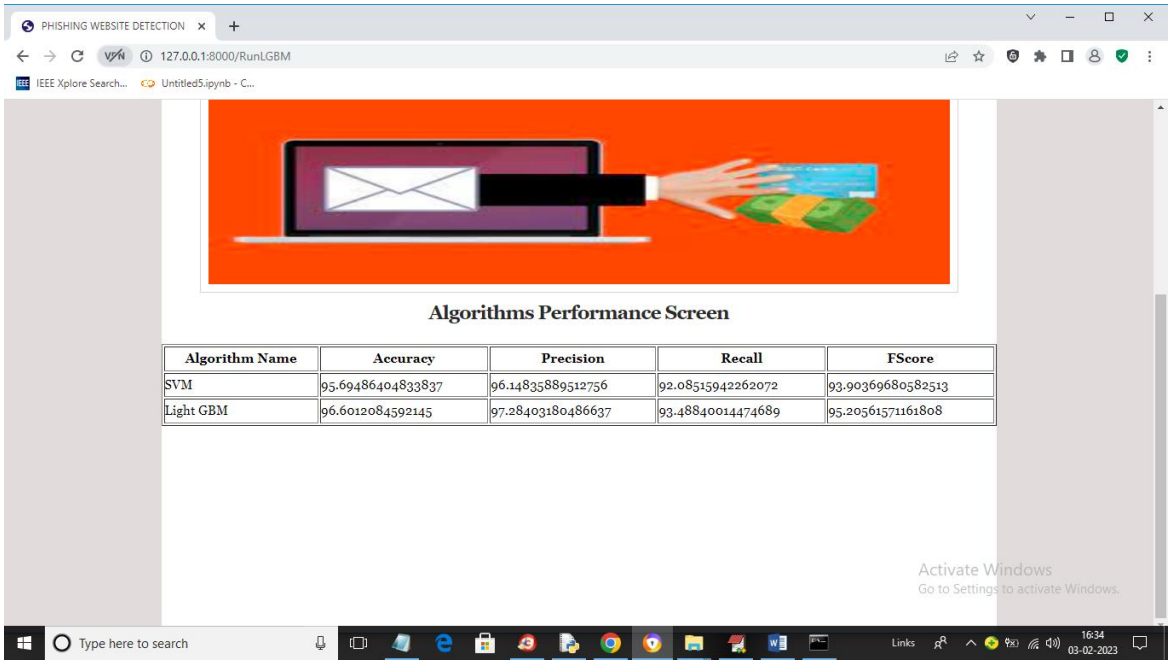

Screenshot 5.2: Home Page



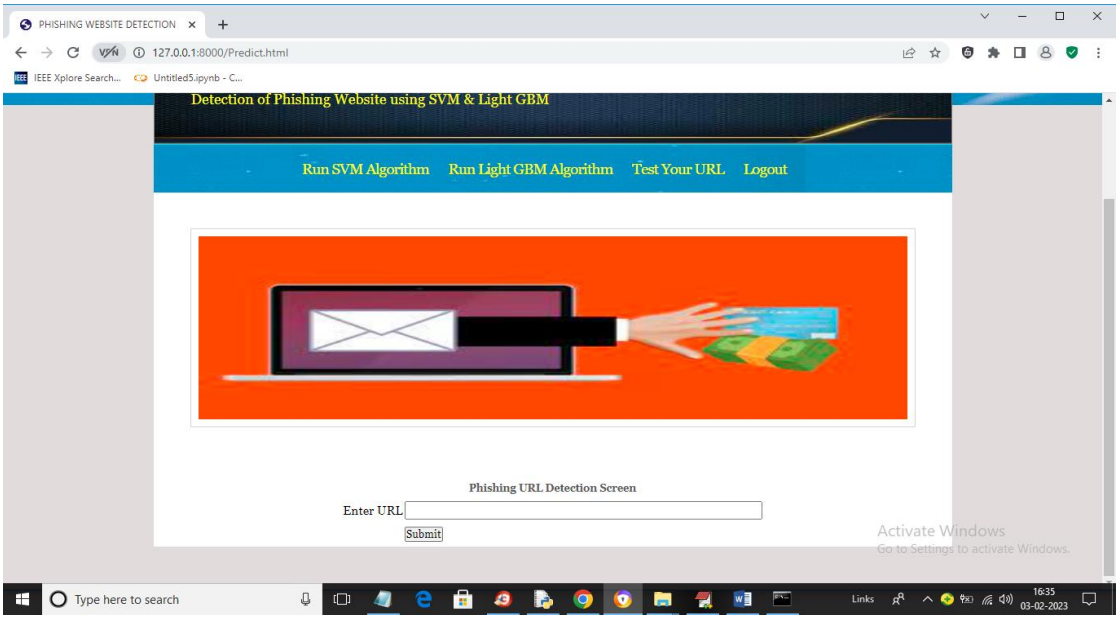

Screenshot 5.3: SVM Confusion Matrix



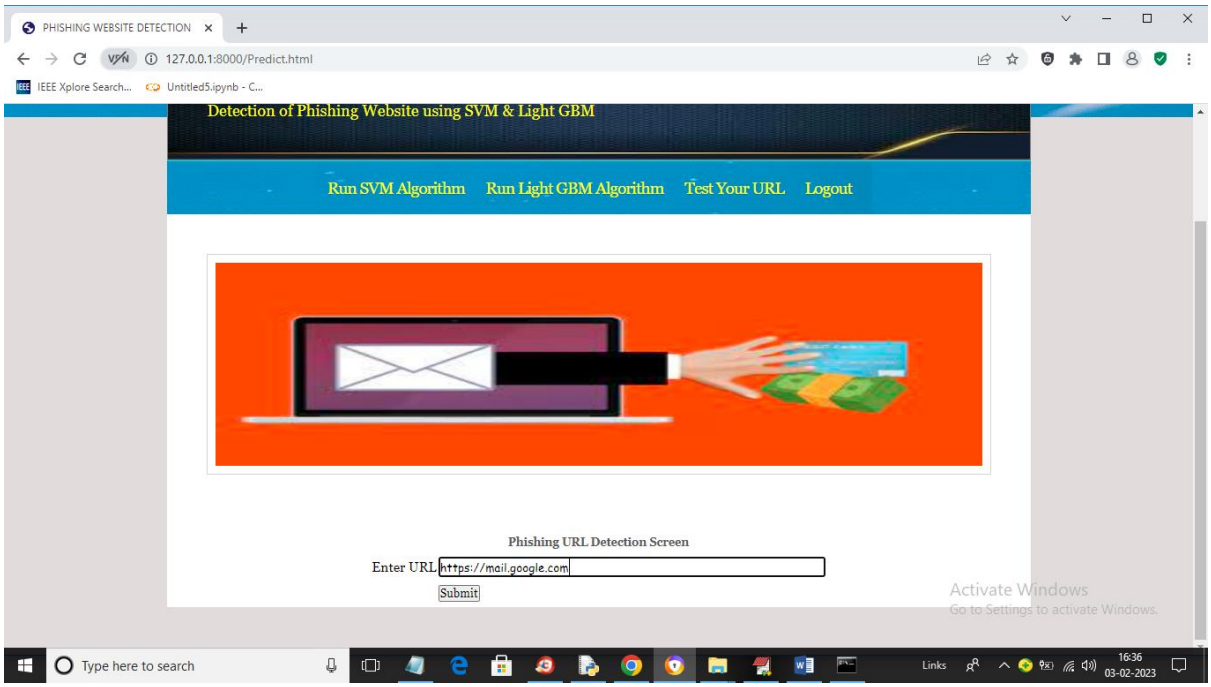

Screenshot 5.4: LightGBM Confusion Matrix



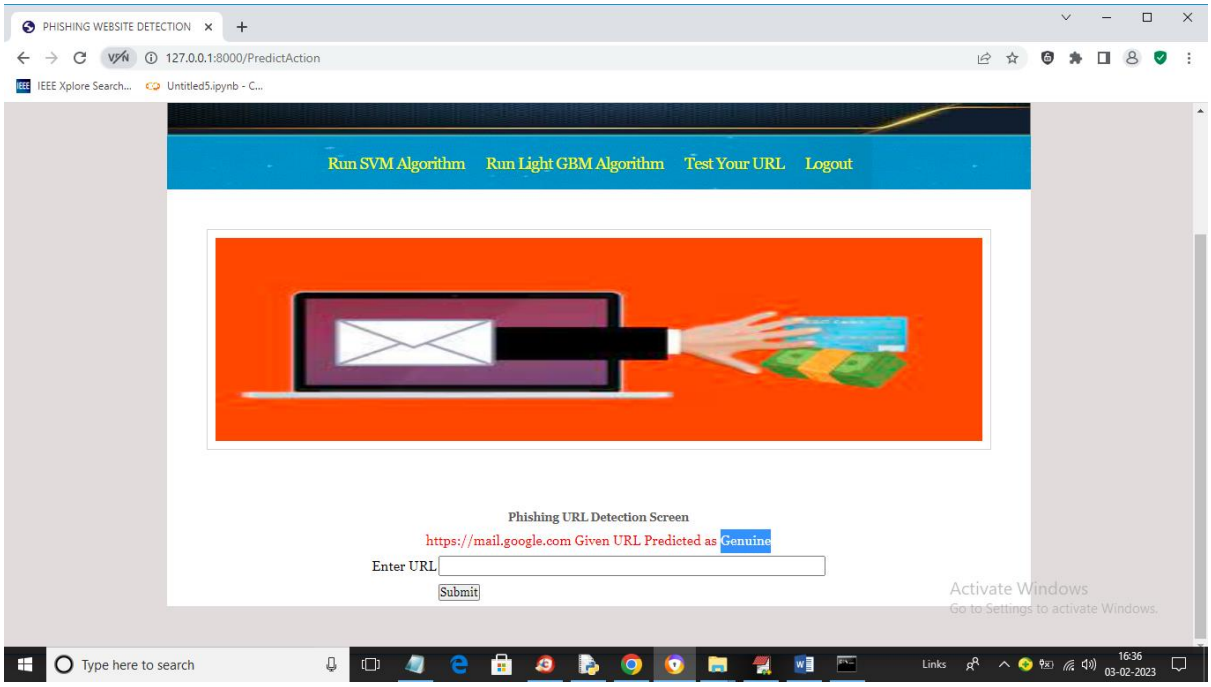

Screenshot 5.5: Algorithms Performance Screen




Screenshot 5.6: Test URL Window




Screenshot 5.7: URL entered for detection




Screenshot 5.8: URL predicted as Genuine

# 6. TESTING

# 6.TESTING

## 6.1 INTRODUCTION TO 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, subassemblies, 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 tests. Each test type addresses a specific testing requirement.

## 6.2 TYPES OF TESTING

### 6.2.1 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.

### 6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Integration tests demonstrate that although the components were individually satisfactory, 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.

### 6.2.3 FUNCTIONAL TESTING

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.

## 6.3 TEST CASES

### 6.3.1 CLASSIFICATION

| Test case ID | Test case name | Purpose | Input | Output | Result |
|---|---|---|---|---|---|
| 1 | User Login | To validate the user | Username as text | Incorrect Username | Pass |
| 2 | User Login | To validate the user | Password as text | Incorrect Password | Pass |
| 3 | User Login | To validate the user | Username and Password as text | Login Successful | Pass |
| 4 | Enter URL | To classify the URL | The user gives the input in the form of a text | An output is classification of URL as genuine. | Pass |
| 5 | Enter URL | To classify the URL | The user gives the input in the form of a text | An output is classification of URL as phishing. | Pass |

# 7. CONCLUSION

# 7. CONCLUSION & FUTURE SCOPE

## 7.1 PROJECT CONCLUSION

In conclusion, a phishing website detection project using machine learning (ML) can provide a reliable and effective solution to the growing problem of phishing attacks. This project can help prevent financial losses, protect sensitive information, and safeguard online identities from phishing attacks. However, it is important to note that phishing attacks are constantly evolving and becoming more sophisticated, and ML-based detection systems may not always be foolproof. Overall, a phishing website detection project using ML is a valuable contribution to the field of cybersecurity, providing a scalable and automated solution to a pervasive problem that affects millions of users worldwide.

## 7.2 FUTURE SCOPE

The future scope of phishing website detection using machine learning is promising as it combines the strength of two powerful machine learning algorithms. With the increasing sophistication of phishing attacks, leveraging advanced techniques like ensemble methods (LightGBM) alongside SVM can enhance detection accuracy. Furthermore, the ongoing growth of online transactions and sensitive data sharing highlights the crucial need for robust phishing detection systems. Future developments may include the incorporation of deep learning models, real-time data feeds, and user behavior analysis to create more resilient and adaptive systems for safeguarding users from evolving phishing threats in an increasingly digital world.

# 8. BIBLIOGRAPHY

# 8. BIBLIOGRAPHY

## 8.1 REFERENCES

[1] Ms. Sophiya Shigalkar, Mrs. Swati Narwane (2019), Detecting of URL based Phishing Attack using Machine Learning. vol., 8 Issue 11, November – 2019.

[2] Rashmi Karnik, Dr. Gayathri M Bhandari, Support Vector Machine Based Malware and Phishing Website Detection.

[3] Arun Kulkarni, Leonard L. Brown, III2, Phishing Websites Detection using Machine Learning., vol. 10, No. 7,2019.

[4] R. Kiruthiga, D. Akila, Phishing Websites Detection using Machine Learning.

[5] Ademola Philip Abidoye, Boniface Kabaso, Hybrid Machine Learning: A Tool to detect Phishing Attacks in Communication Networks., vol. 11 No. 6,2020.

[6] Andrei Butnaru, Alexios Mylonas and Nikolaos Pitropakis, Article Towards Lightweight URL-Based Phishing Detection.13 June 2021.

[7] Ashit Kumar Dutta (2021), Detecting phishing websites using machine learning technique. Oct 11 2021.

[8] Nguyet Quang Do, Ali Selamat, Ondrej Krejcar, Takeru Yokoi and Hamido Fujita (2021) Phishing Webpage Classification via Deep Learning-Based Algorithms: An Empirical study.

[9] Ammara Zamir, Hikmat Ullah Khan and Tassawar Iqbal, Phishing website detection using diverse machine learning algorithms.

[10] Valid Shahrivari, Mohammad Mahdi Darabi and Mohammad Izadi (2020), Phishing Detection Using Machine Learning Techniques.

[11] A. A. Orunsolu, A. S. Sodiya and A.T. Akinwale (2019), A predictive model for phishing detection.

[12] Wong, R. K. K. (2019). An Empirical Study on Performance Server Analysis and URL Phishing Prevention to Improve System Management Through Machine Learning. In Economics of Grids, Clouds, Systems, and Services: 15th International Conference, GECON 2018, Pisa, Italy, September 18-20, 2018, Proceedings (Vol. 11113, p. 199). Springer.

[13] Desai, A., Jatakia, J., Naik, R., & Raul, N. (2017, May). Malicious web content detection using machine leaning. In 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT) (pp. 1432-1436). IEEE.

## 8.2 GITHUB LINK

https://github.com/chidrup29/Phishing-Website-Detection