PHISHING URL DETECTION USING RANDOM FOREST CLASSIFIER AND SUPPORT VECTOR MACHINE

Major project report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

B BHAVANA	(20UECS0110)	**(16825)**
M SAI SUSHMA SARANYA	(20UECS0550)	**(17058)**
L LAVANYA	(20UECS1049)	**(16698)**

Under the guidance of
Dr.D.UMANANDHINI,M.E., Ph.D.
PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

PHISHING URL DETECTION USING RANDOM FOREST CLASSIFIER AND SUPPORT VECTOR MACHINE

Major project report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

B BHAVANA	(20UECS0110)	**(16825)**
M SAI SUSHMA SARANYA	(20UECS0550)	**(17058)**
L LAVANYA	(20UECS1049)	**(16698)**

Under the guidance of
Dr.D.UMANANDHINI,M.E., Ph.D.
PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled ”PHISHING URL DETECTION USING SUPPORT VECTOR MACHINE AND RANDOM FOREST CLASSIFIER” by ”B BHAVANA (20UECS0110), M SAI SUSHMA SARANYA (20UECS0550), L LAVANYA (20UECS1049)” has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

Signature of Professor In-charge
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

DECLARATION

We declare that this written submission represents our ideas in our own words and where others ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

B BHAVANA

Date: / /

M SAI SUSHMA SARANYA

Date: / /

L LAVANYA

Date: / /

APPROVAL SHEET

This project report entitled ”PHISHING URL DETECTION USING RANDOM FOREST CLASSIFIER AND SUPPORT VECTOR MACHINE” by B BHAVANA (20UECS0110), M SAI SUSHMA SARANYA (20UECS0550), L LAVANYA (20UECS1049) is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners **Supervisor**

Dr.D.UMANANDHINI,M.E., Ph.D. PROFESSOR.,

Date: / /

Place: AVADI

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering,Dr.M.S. MURALI DHAR, M.E., Ph.D.,** for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our Internal Supervisor **Dr.D.UMANANDHINI,M.E.,Ph.D.,** for her cordial support, valuable information and guidance, he/she helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. C. SHYAMALA KUMARI, M.E.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

B BHAVANA **(20UECS0110)**
M SAI SUSHMA SARANYA **(20UECS0550)**
L LAVANYA **(20UECS1049)**

ABSTRACT

Phishing is the process of stealing confidential personal and corporate information through deceptive Emails, URLs and text messages. Phishing via URLs (Uniform Resource Locators) is one of the most common types, and its primary goal is to steal the data from the user when the user accesses the malicious website. Detecting a malicious URL is a significant challenge. This project aims to provide a solution for detecting such websites with the help of machine learning algorithms focused on the behaviours and qualities of the suggested URLs.Dataset contains nearly 400 URL datasets in those 80 percent are testing and 20 percent are training. Comparison of various machine learning algorithms like random forest and support vector machine and find the deceptive URLs based on the accuracy of the machine learning algorithms will be done.The methodology involves feature extraction from URLs, including lexical and domain-based features, followed by the application of various supervised learning algorithms such as Random Forest, Support Vector Machine(SVM), and Neural Networks.My SQL dataset used for training and testing consists of a diverse collection of legitimate and phishing URLs.The results demonstrate the effectiveness of the proposed approach in accurately identifying phishing URLs, thereby providing a valuable tool for enhancing cybersecurity measures and safeguarding against online threats.

Keywords:Cyber Security,My SQL,Phishing,Random Forest,Support Vector Machine,Uniform Resource Locator.

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

CNN	Convolutional Neural Networks
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
LR	Logistic Regression
ML	Machine Learning
PILU	Phishing Index Login URL
RFC	Random Forest Classifier
SQL	Structured Query Language
SVM	Support Vector Machine
URL	Uniform Resource Locator
UI	User Interface

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

Phishing is a social engineering cyber attack where the criminals steal the sensitive information of the users and obtain the credentials through a login form that submits the data to a malicious servers.Phishing URL detection refers to the process of identifying URLs that are associated with phishing attacks.Phishing URLs are web addresses created by malicious actors to deceive individuals to steal sensitive information such as login credentials, financial details, or personal data.Phishing URL detection aims to distinguish between legitimate URLs and fraudulent ones, thereby helping users and the organisations to protect themselves from falling victim to phishing scams.There are various algorithms for supervised learning processes, such as naïve bayes, neural networks, linear regression, logistic regression, decision tree, support vector machine, K-nearest neighbour, and random forest.The existing system uses any one of the suitable machine learning algorithms for the detection of phishing URLs and predicts its accuracy. phishing URL dataset using legitimate login websites to obtain the URLs from such pages.Then evaluation of machine and deep learning techniques for recommending the method with higher accuracy will be done.Next will show how models trained with legitimate homepages struggle to classify legitimate login URLs, demonstrating hypothesis about phishing detection and legitimate login URLs.Additionally,how the accuracy decrease with the time on models trained with datasets from 2016 and evaluated on data collected in 2020. Finally an overview of current phishing encounters, explaining attacker tricks and approaches will be provided.Demonstrated empirically how an URL phishing detection model struggles in classifying login URLs when it was trained on the URLs of the homepage of phishing and legitimate URLs.

1.2 Aim of the project

The aim of the project for ”Phishing URL Detection Using Random Forest Classifier and Support Vector Machine” is to develop a system capable of identifying and categorizing URLs as either legitimate or potentially malicious/phishing.

1.3 Project Domain

The domain of a project on phishing URL detection using random forest and support vector machine typically involves cybersecurity and machine learning. In this context, the domain encompasses identifying and classifying URLs as either legitimate or phishing based on various features extracted from the URLs, such as domain length, presence of suspicious keywords, and use of certain characters. The goal is to develop models, like random forest and support vector machine, that can effectively distinguish between legitimate and phishing URLs to enhance online security. In a project context, collection of a dataset of labeled URLs (legitimate or phishing), extract features, preprocess the data, split it into training and testing sets, train the models using algorithms like random forest and SVM, and then evaluate the models' performance using metrics like accuracy, precision, recall, and F1-score. The ultimate aim is to develop a robust and accurate phishing URL detection system to enhance cybersecurity measures.

1.4 Scope of the Project

The scope is to develop an phishing URL detection system using machine learning algorithms such as Support Vector Machines (SVM) and Random Forest Classifier (RFC). Analyzing common phishing tactics and their evolution. Developing algorithms to identify patterns specific to phishing login URLs. Integrating the detection

mechanism into existing cybersecurity systems. Evaluating the effectiveness of the detection system through real-world scenarios.

Chapter 2

LITERATURE REVIEW

[1] SK Hasane Ahammad ,Sunil D. Kale et al., (2022) Phishing is the primary step to detect the URL in cyber crime .When a user accesses the website attacker can steal the information using malicious software,here author used different machine learning algorithms to detect and remove the fake URL. Created a blacklisting service to identify the threat and malicious websites.

[2] R. Arora et al.,(2023) proposed that by analysing patterns in the URLs, machine learning models can identify malicious URLs with a high degree of accuracy.Mischievous URLs, certain machine learning algorithms are used for different techniques and algorithms to investigate the crime .

[3] Manuel Sánchez-Paniagua et al.,(2023) proposed that criminals obtain the user credentials through login submission forms.Author used the datasets for training how models decrease the accuracy while testing.Here Author used Logistic Regression models to train and test the data sets.To prove the statements they use 60 login urls through login websites.

[4] M. Al Mousa and M. Anwar et al.,(2023) Novel character-aware language models for detecting URL-based social semantic attacks.They experimented with three different techniques for testing and identifying the cyber attacks like LSTM,CNN,RNN etc.Users are tricked into clicking on obfuscated URLs, downloading a harmful attachment, or sharing sensitive and private information.

[5] A. S. Rafsanjani et al., (2023) Adapted of Quick Response codes with malicious URLs is a growing concern and is an open security issue. The result shows

that QsecR outperforms others and achieves a detection accuracy.

[6] M. Alsaedi et al.,(2023) proposed Multimodal representation approach that fuses textual and image-based features to enhance the performance of the malicious website detection.A decision-making artificial neural network classifier was trained using the combined output layers of the two models.

[7] S. Asiri et al.,((2023) It is hard to detect because attackers can design a phishing message that looks legitimate to a user. This URL leads the victim to a fake website that steals information, such as login information, payment information, etc.Content-based approaches extract information from webpage content such as images, JavaScript, text, HyperText Markup Language (HTML) code.

[8]Y. Liang et al.,(2022) To fuse heterogeneous features, a self-paced wide deep learning strategy is proposed to train a robust model effectively. This solution is evaluated on a large-scale URL dataset. The deep structural features, position embedding is introduced for token vectorization to reduce the ambiguity of URL tokens.

[9]B. Sabir et al., (2022) A cost-effective Adversarial URL generator URL bug that created an Adversarial URL dataset and tested the considered MLPU systems on Adversarial data and analysed their robustness and reliability using box plots and heat maps.the considered MLPU models Matthew Correlation Coefficient dropped from median 0.92 to 0.02 when tested against Adversarial data , indicating that the baseline MLPU models are unreliable in their current form.

[10]A. Karim et al.,(2023) Based on the phishing URL-based dataset extracted from the famous dataset repository, which consists of phishing and legitimate URL attributes collected from 11000+ website datasets in vector form and they used decision tree,random forest,svc they tried different approaches for detection .

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

The existing system involves a comprehensive process aimed at effectively identifying malicious URLs. Initially, a substantial dataset of URLs is collected, categorized as either phishing or legitimate. Subsequently, relevant features such as domain age, URL length, and the presence of special characters are extracted from the URLs. After data preprocessing, which includes cleaning and standardizing the data, machine learning models are trained on the prepared dataset. Popular algorithms like Decision Trees, Random Forest, and Support Vector Machines are commonly employed for this task.This models are then evaluated using various machine learning algorithms.The best-performing model is selected and integrated into the system for real-time phishing URL detection. Continuous monitoring and periodic updates are essential to ensure the system remains effective against evolving phishing techniques. Popular datasets like PhishTank and OpenPhish are often utilized for training, along with feature engineering techniques like TF-IDF(Term Frequency Inverse Document Frequency) to enhance model accuracy.

3.1.1 Disadvantages on Existing System

- As a list of blacklisted URLs is being used to detect the accuracy of predicting thecorrect results may be very low. There isn't any standard list of URLs published byany standard organization.
- As a part of daily life, We may encounter many new URL's which seems to

besame as original ones and not present in the list. It is hard to differentiate suchURL's.

3.2 Proposed System

Phishing URL dataset uses legitimate login websites to obtain the URLs from such pages. Then evaluating machine and deep learning techniques for recommending the method with higher accuracy. Next, the models are trained with legitimate homepages struggle to classify legitimate login URLs, demonstrating the hypothesis about phishing detection and legitimate login URLs. Additionally, will demonstrate how the accuracy decrease with the time on models trained with datasets from 2016 and evaluated on data collected in 2020. Finally,an overview of current phishing encounters, explaining attacker tricks and approaches. Finally the extension previous dataset PILU-60K (Phishing Index Login URL), from 60K to 90K URLs equally distributed among three classes: phishing, the legitimate homepage, and legitimate login.Making this extended dataset, PILU-90K, publicly available for research purposes.

Using PILU-90K, implemented and evaluated three pipelines for URL phishing detection: used the 38 handcrafted feature descriptors proposed by Sahingoz et al for training eight supervised machine learning classifiers and also (ii) automatic feature extraction using Term Frequency Inverse Document Frequency (TF-IDF) at character N-gram level combined with Logistic Regression (LR) algorithm, and a Convolutional Neural Network (CNN) at character level too.demonstration of how an URL phishing detection model struggles in classifying login URLs when it was trained on the URLs of the homepage of phishing and legitimate URLs. Evaluation of the robustness of the proposed phishing detection over time and trained the model on a dataset collected between March 2016 and April 2016, and evaluated the model on

other datasets collected between 2017 and 2020. Phishing websites were analyzed using domain frequency.

3.2.1 Advantages of Proposed System

- We can identify fake websites and helps users safe from exploitation of theirinformation.
- We can stop unsafe transactions identifying these fake websites.
- We can add this approach to any browser to make it secure browser.

3.3 Feasibility Study

Feasibility study tells about how the system is feasible, what are its feasible conditions, how to achieve different types of feasibility. In the conduction of the feasibility study, the analyst will usually consider seven distinct, but inter-related types of feasibility. The feasibility of project provides the various constraints to the quality of being weak or strong to plan the purpose of business needs. To estimate the costs, performance, the designed implementation and the resource being defined for the environment. The various accepts that perform through the description of project, the operation of technical knowledge, managing the resources and mainly capable for the success. It is carried out during the proposed system; the future requirements may also include the level of system resources.

3.3.1 Economic Feasibility

The economic feasibility hinges on several factors. Initially, there may be significant upfront costs associated with acquiring and preprocessing a sufficiently large and diverse dataset for training the machine learning models. Additionally, investment in

computational resources for training and deploying the models is necessary. However, once developed, the operational costs can be relatively low, primarily consisting of maintenance, monitoring, and periodic retraining of the models. The potential benefits of such a system include mitigating the financial losses incurred from falling victim to phishing attacks, protecting sensitive data, and preserving the reputation of individuals and organizations. Assessing the economic feasibility requires weighing the initial investment against the anticipated benefits, including potential cost savings and risk reduction associated with improved phishing URL detection capabilities. Furthermore, considering scalability and adaptability to evolving threats is crucial for ensuring long-term economic viability.

3.3.2 Technical Feasibility

The technical feasibility depends on several factors. Firstly, the availability of labeled datasets containing both phishing and legitimate URLs is essential for training robust machine learning models. Additionally, the preprocessing and feature extraction steps must be carefully designed to ensure the models receive relevant and informative input data. The implementation of random forest and SVM classifiers requires suitable libraries and frameworks, along with sufficient computational resources for training and inference. Furthermore, integrating the trained models into a production environment for real-time URL classification necessitates robust software engineering practices to ensure scalability, reliability, and performance. While these technical challenges can be overcome with appropriate expertise and resources, ongoing monitoring and adaptation to emerging phishing techniques are crucial for maintaining the effectiveness of the detection system over time.

3.3.3 Social Feasibility

The social feasibility depends on various factors related to acceptance, adoption, and impact on stakeholders. Public awareness campaigns about the importance of phishing detection and the role of technology in safeguarding against cyber threats can foster acceptance and support for such projects. Collaboration with cybersecurity experts, industry stakeholders, and regulatory bodies can help ensure alignment with best practices and compliance requirements. Moreover, transparent communication about the capabilities and limitations of the detection system is essential for building trust among end users. Addressing privacy concerns and implementing measures to protect user data can further enhance social acceptability. Additionally, considering the societal impact of reducing phishing-related financial losses, protecting personal information, and preserving trust in online communication channels underscores the importance of social feasibility in advancing such projects.

3.4 System Specification

- URL Analysis component responsible for analyzing URLs to determine their legitimacy.
- A database containing known phishing URLs and associated metadata.
- whitelist helps reduce false positives by exempting trusted URLs from further scrutiny.
- Real-time monitoring enables rapid detection and response to phishing attempts as they occur.
- Pattern recognition helps identify suspicious URLs even if they are not already known to be malicious.

3.4.1 Hardware Specification

- Processor - Pentium –IV
- RAM - 4 GB (min)
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

3.4.2 Software Specification

- Operating system : Windows 7 Ultimate
- Coding Language : Python
- Front-End : Python
- Back-End : Django-ORM
- Designing : Html, css, javascript
- Data Base : MySQL , WAMP S

3.4.3 Standards and Policies

- URL Blacklists: Maintaining and regularly updating blacklists of known phishing URLs is a fundamental approach. These blacklists can be curated manually or automatically through various sources such as threat intelligence feeds, security vendors, and community contributions.
- Real-time Analysis: Conducting real-time analysis of URLs as users access them. This can involve comparing URLs against known phishing databases,

checking for URL reputation scores, and inspecting the content of the linked webpage for phishing indicators.

- User Reporting: Encouraging users to report suspicious URLs they encounter through designated channels. This can include implementing reporting buttons in email clients, web browsers, and security awareness training programs.
- Email Security Policy: This policy defines procedures for email usage, including authentication mechanisms (such as SPF, DKIM, and DMARC), encryption requirements, and guidelines for identifying and handling suspicious emails, including phishing attempts.

Anaconda Prompt

Anaconda prompt is a type of command line interface which explicitly deals with the Machine Learning modules.And navigator is available in all the Windows,Linux and MacOS.The anaconda prompt has many number of (Integrated Development Environment)IDE's which make the coding easier. The User Interface can also be implemented in python.

Standard Used: ISO/IEC 27001

Jupyter

It's like an open source web application that allows us to share and create the documents which contains the live code, equations, visualizations and narrative text. It can be used for data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning.

Standard Used: ISO/IEC 27001

Chapter 4

METHODOLOGY

4.1 Url Detection Architecture

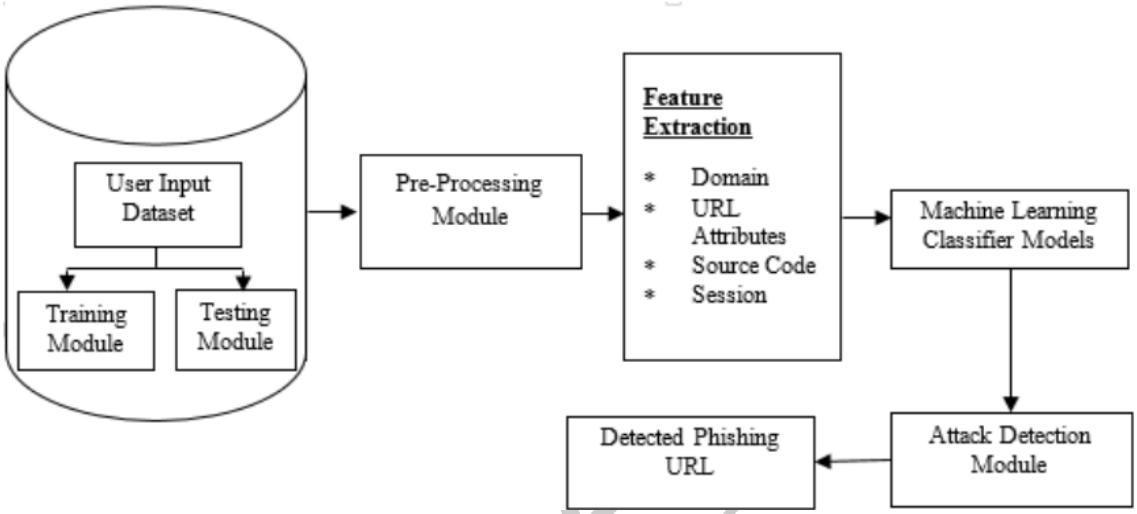


Figure 4.1: **Phishing URL Detection**

Ref fig.no.4.1 :The architecture of a phishing URL detection project employing machine learning algorithms typically encompasses data collection, feature extraction, preprocessing, model training, evaluation, model selection, deployment, and ongoing monitoring and maintenance. Initial data collection involves compiling a dataset of labeled URLs, followed by feature extraction to capture relevant characteristics. Preprocessing involves cleaning and normalizing the data. Machine learning models are then trained on the processed data and evaluated using various metrics. The best-performing model is selected for deployment in a production environment where it can classify incoming URLs in real-time. Continuous monitoring and periodic retraining ensure the model remains effective over time, completing the architecture loop for a robust phishing URL detection system.

4.2 Design Phase

4.2.1 Data Flow Diagram

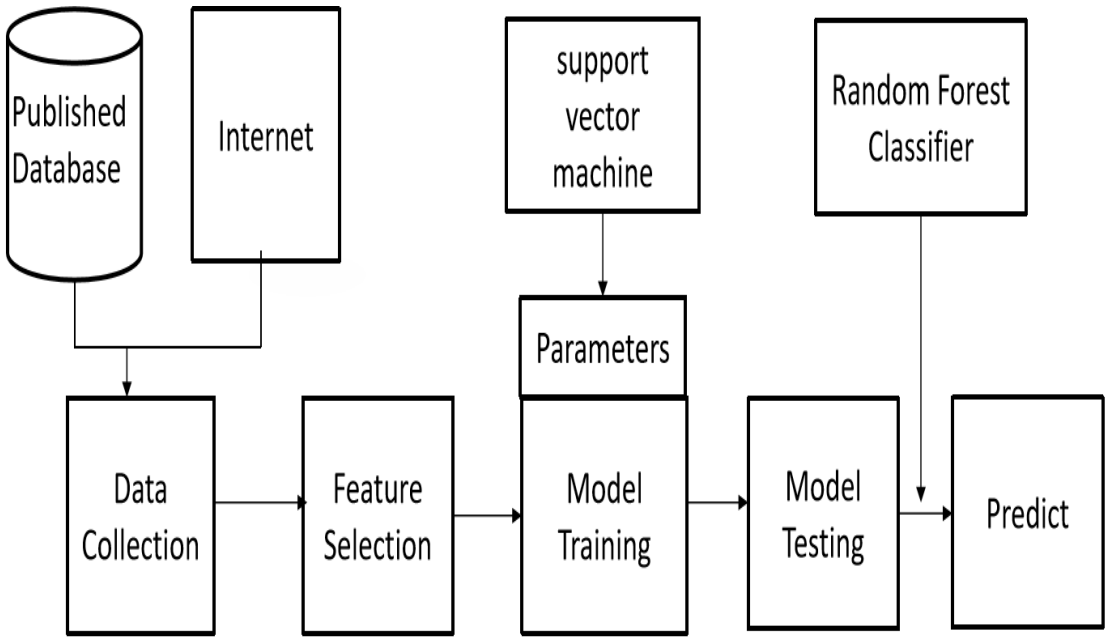


Figure 4.2: **Data Flow Diagram**

Ref fig.no.4.2 :The data flow diagram for a phishing URL detection using random forest and support vector machine depicts the sequential flow of information within the system. It begins with the collection of urls and extracting the features.These features serve as inputs to the machine learning models, which are trained on a portion of the data. Once trained, the models are deployed to classify incoming URLs as either phishing or legitimate. The classified URLs are then subjected to further analysis and potentially stored in a database for future reference or analysis. Additionally, feedback from the deployment phase can be looped back to the training phase to continuously improve the model's performance. Overall, the data flow diagram illustrates the systematic progression of data from collection to deployment and beyond, facilitating the effective detection of phishing URLs.

4.2.2 Use Case Diagram

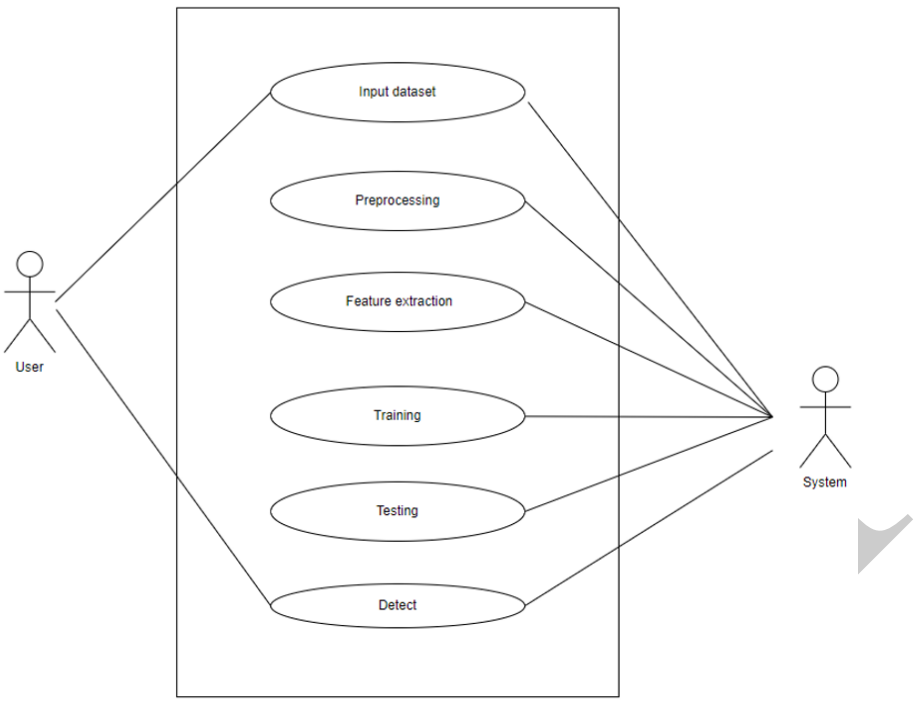


Figure 4.3: **Use case Diagram**

Ref fig.no.4.3 :The use case diagram for a phishing URL detection project employing random forest and SVM involves various actors and their interactions with the system. The primary actors include the system administrator, who oversees the entire process, and the end users, who interact with the system to submit URLs for classification. The system facilitates multiple use cases, such as training the random forest and SVM models with labeled data, deploying the trained models for real-time URL classification, and providing feedback mechanisms for users to report misclassifications or new phishing URLs. Additionally, the system supports administrative tasks like monitoring model performance, updating model parameters, and managing user access. Overall, the use case diagram illustrates the diverse functionalities and interactions within the phishing URL detection system, catering to the needs of both administrators and end users while leveraging the capabilities of random forest and SVM algorithms.

4.2.3 Class Diagram

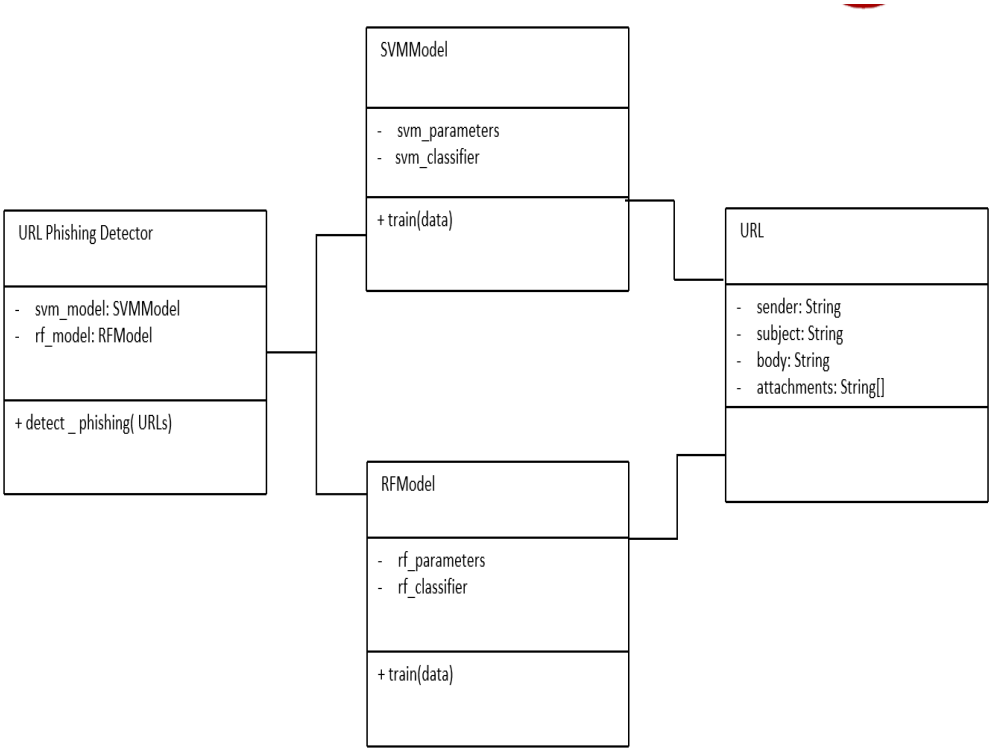


Figure 4.4: **Class Diagram**

Ref fig.no.4.4 :In the class diagram for a phishing URL detection project utilizing random forest and Support Vector Machine, key classes include those representing the models themselves, such as Random Forest Classifier and Support Vector Machine Classifier, which inherit from a generic Classifier class. Additionally, there are classes for data preprocessing, feature extraction, and model evaluation. The Data Processor class handles tasks like cleaning and normalization, while the Feature Extractor class extracts relevant features from URLs. Evaluation Metrics class calculates various performance metrics like accuracy, precision, and recall. Other classes include Dataset for managing training and testing data, and User Interface for interaction with end users. Each class encapsulates specific functionalities essential for the successful implementation and operation of the phishing URL detection system, leveraging the capabilities of random forest and Support Vector Machine algorithms.

4.2.4 Sequence Diagram

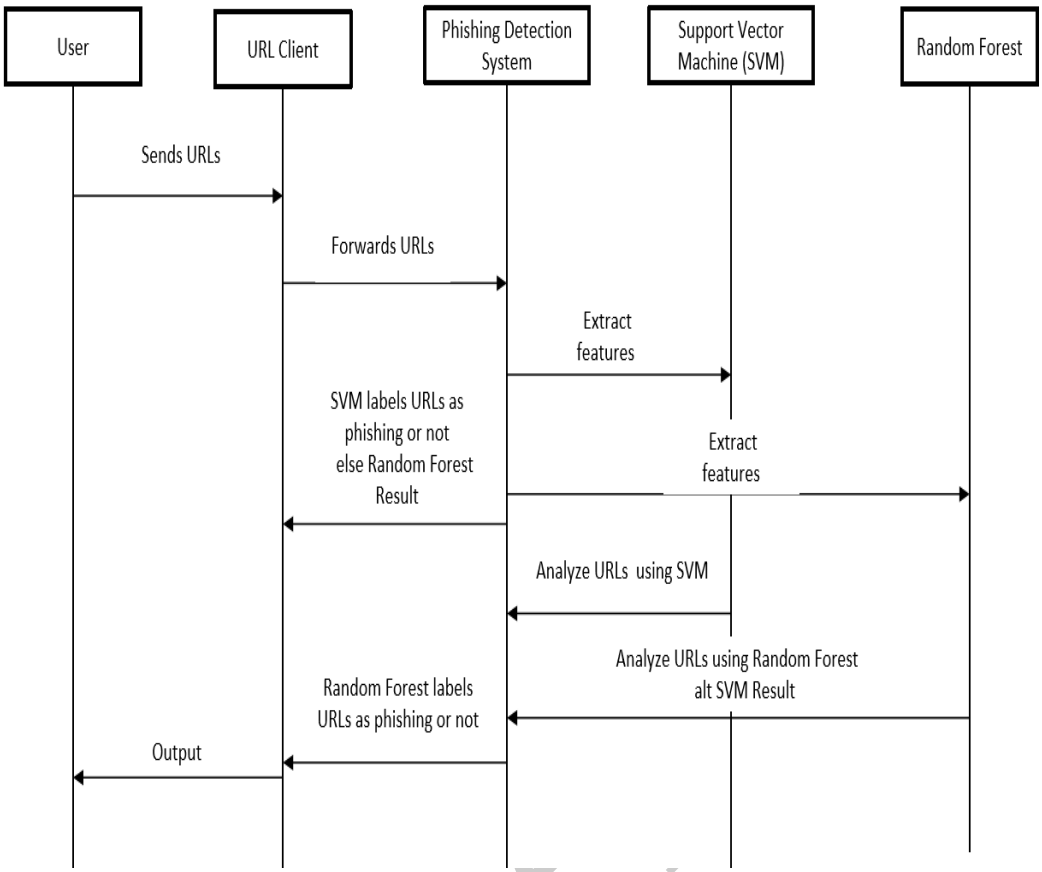


Figure 4.5: **Sequence Diagram**

Ref fig.no.4.5 :The sequence diagram for a phishing URL detection project utilizing random forest and support vector machine illustrates the flow of interactions between different components in the system. It begins with the submission of a URL by the user, triggering the preprocessing phase where the URL undergoes cleaning and normalization. Next, the feature extraction process extracts relevant characteristics from the URL. These features are then passed to both the random forest and SVM classifiers, where they undergo classification to determine whether the URL is phishing or legitimate. The classifiers return their respective predictions, which are combined or compared to improve overall accuracy. Finally, the result of the classification, along with any feedback provided by the user, is communicated back to the user interface

for display and further action. Throughout this sequence, each component interacts seamlessly to facilitate the efficient detection of phishing URLs using random forest and SVM algorithms.

4.2.5 Activity Diagram

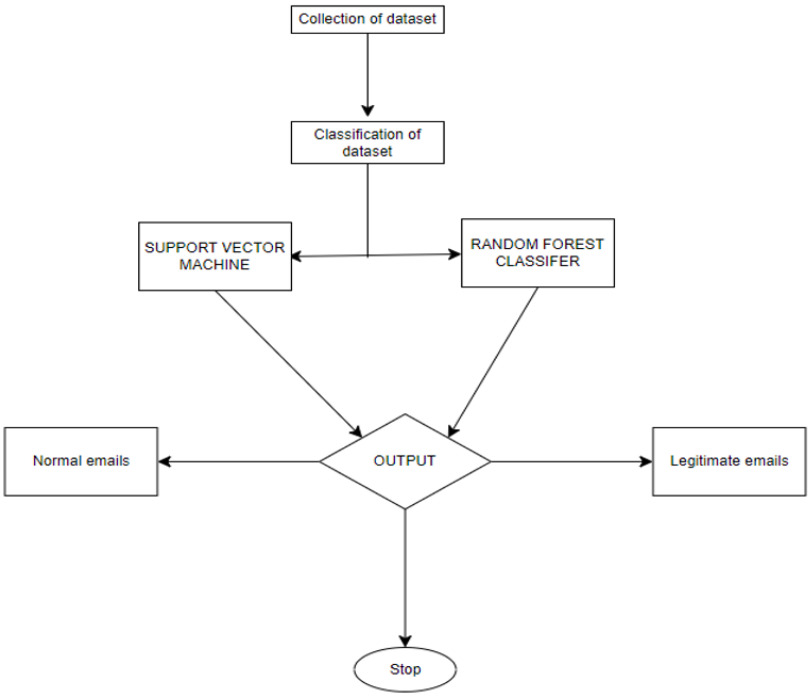


Figure 4.6: **Activity Diagram**

Ref fig.no.4.6 :The activity diagram for a phishing URL detection using random forest and support vector machine illustrates the sequence of activities involved in the detection process. It begins with the user submitting a URL for classification, triggering the preprocessing step to clean and normalize the URL data. Subsequently, the feature extraction activity extracts relevant characteristics from the URL, followed by the classification activity where the URL features are input into both the random forest and SVM classifiers. Each classifier independently processes the features and generates a classification result. These results are then combined or compared to improve overall accuracy. Finally, the classification result, along with any user

feedback, is communicated back to the user interface for display and potential action. The activity diagram provides a visual representation of the sequential flow of actions involved in detecting phishing URLs using random forest and SVM algorithms.

4.3 Algorithm & Pseudo Code

4.3.1 Algorithm

Step 1: Get the link to website

Step 2: Register/Login

Step 3: Enter URL to be checked

Step 4: Prediction Phase

Step 5: Display result

4.3.2 Pseudo Code

```
# Assuming you have a dataset with features (X) and
labels (y)
1. Split the dataset into training and testing sets
(X_train, X_test, y_train, y_test)
2. Choose a machine learning algorithm
(e.g., decision trees)
# Train the model
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
# Evaluate the model
accuracy = model.score(X_test, y_test)
```

```
print("Model accuracy:", accuracy)
# Deploy the model for real-time phishing URL detection
```

4.4 Module Description

4.4.1 Module1

Data Collection: Gather a diverse dataset of login URLs from various sources, including phishing repositories, security forums, and user-reported incidents. Ensure the dataset encompasses a wide range of phishing techniques and targets.

Data Preprocessing :Extract relevant features from the URL data. These might include, URLs sender's address Keywords indicative of phishing (e.g., "urgent," "verify your account," etc.) Clean and preprocess the collected data to remove noise, standardize formats, and handle missing values. This step may involve URL canonicalization, domain resolution, and text normalization techniques.

Feature Extraction: Utilize various techniques to extract relevant features from phishing URLs. These features may include URL structure, domain reputation, presence of suspicious keywords, and similarity to known phishing URLs.

Feature Selection: Employ methods such as information gain, chi-square test, or correlation analysis to select the most discriminative features. This step helps reduce dimensionality and focus computational resources on the most informative attributes.

4.4.2 Module2

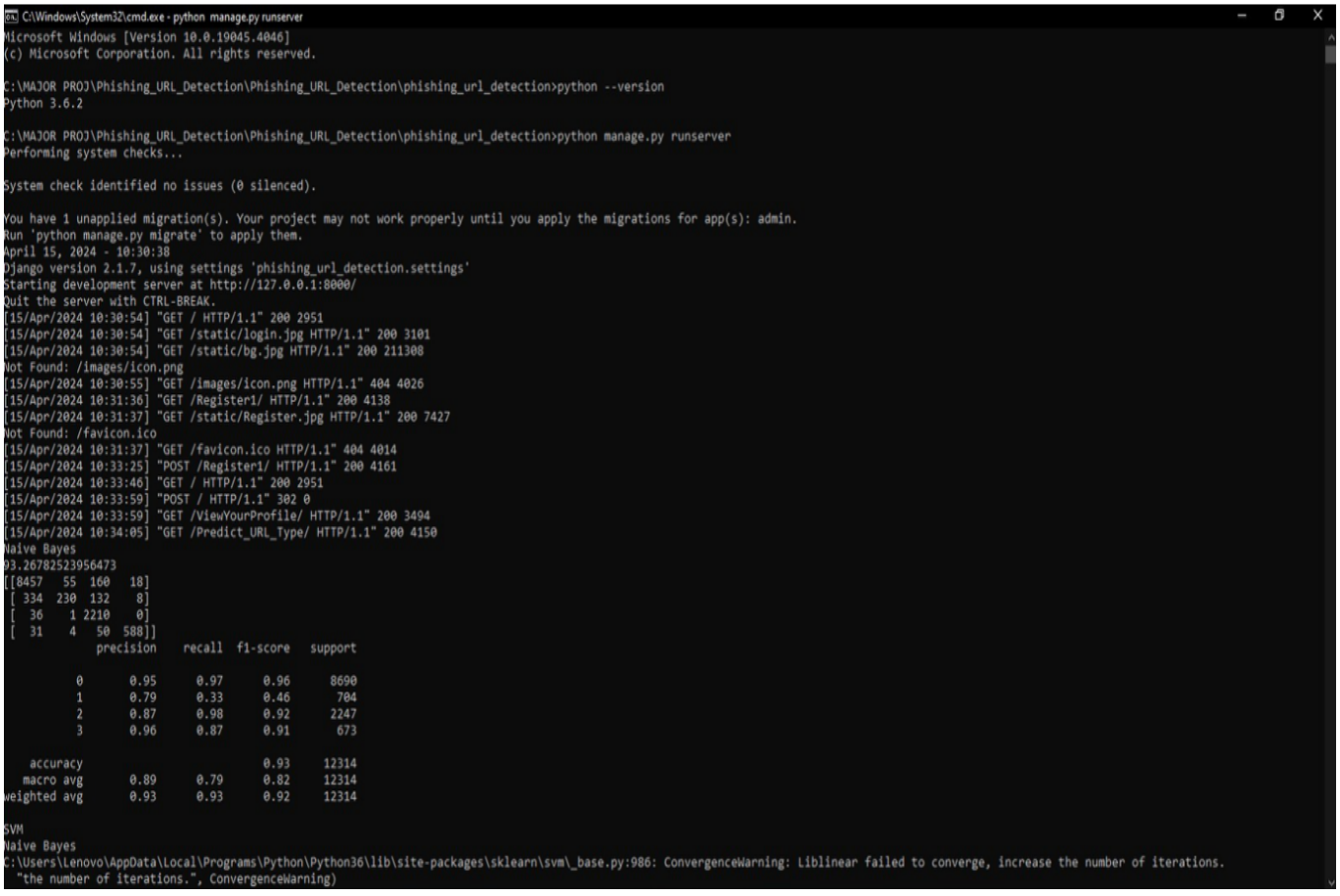
Machine Learning Models: Implement supervised learning algorithms, such as decision trees, random forests, or support vector machines, to train classifiers using the selected features. These models learn to distinguish between legitimate and phishing URLs based on the extracted features.

Integration with Security Infrastructure: Integrate the phishing URL detection system with existing security infrastructure, such as web browsers, email clients, or network proxies. This integration enables real-time detection and prevention of phishing attacks targeting login credentials.

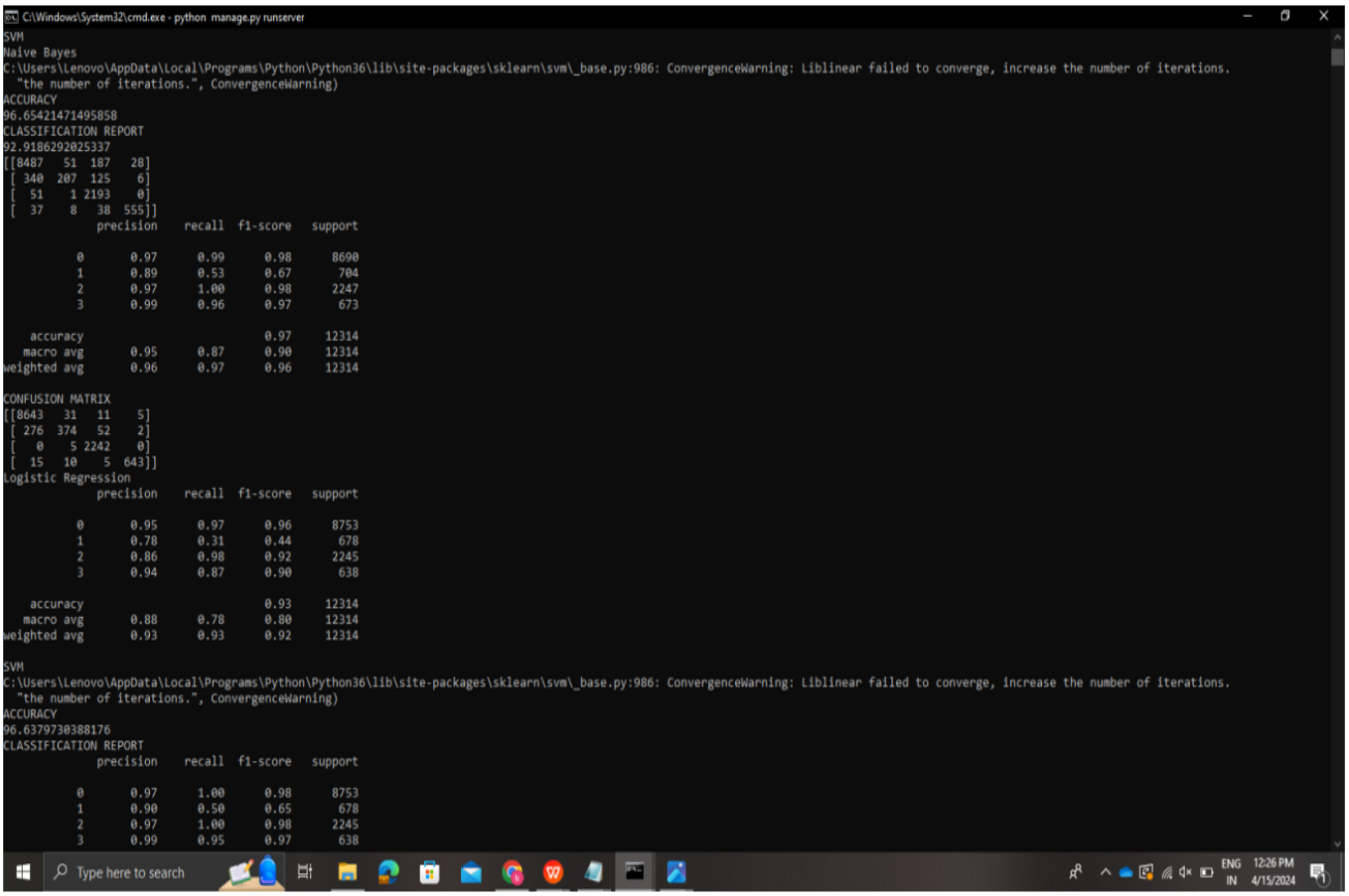
Alert Mechanisms: Implement alert mechanisms to notify users or administrators when a potential phishing URL is detected. These alerts may include warning messages, browser extensions, or automated blocking of suspicious URLs.

4.4.3 Module3

Model creation



Model evaluation



4.5 Steps to execute/run/implement the project

4.5.1 Step1:Service Provider:

- Login the URL page
- Train Test URL Data Sets
- View URL Data Sets Trained and Tested Accuracy in Bar Chart
- View URL Data Sets Trained and Tested Accuracy Results
- View Prediction Of URL Type
- View URL Type Ratio
- Download Predicted Data Sets
- View URL Type Ratio Results

• View All Remote Users

4.5.2 Step2: Remote User:

• Register and Login phishig url page

• Predicting the URL type and algortihm with accuracy value

• Viewing the profile page of Phishing URL detection using RF Classifier and SVM

4.5.3 Step3: Members

• Enter Or Paste the URL to predict the data

• Prediction completed after entering the url and and click submit option .It will detect the Phishing mode

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

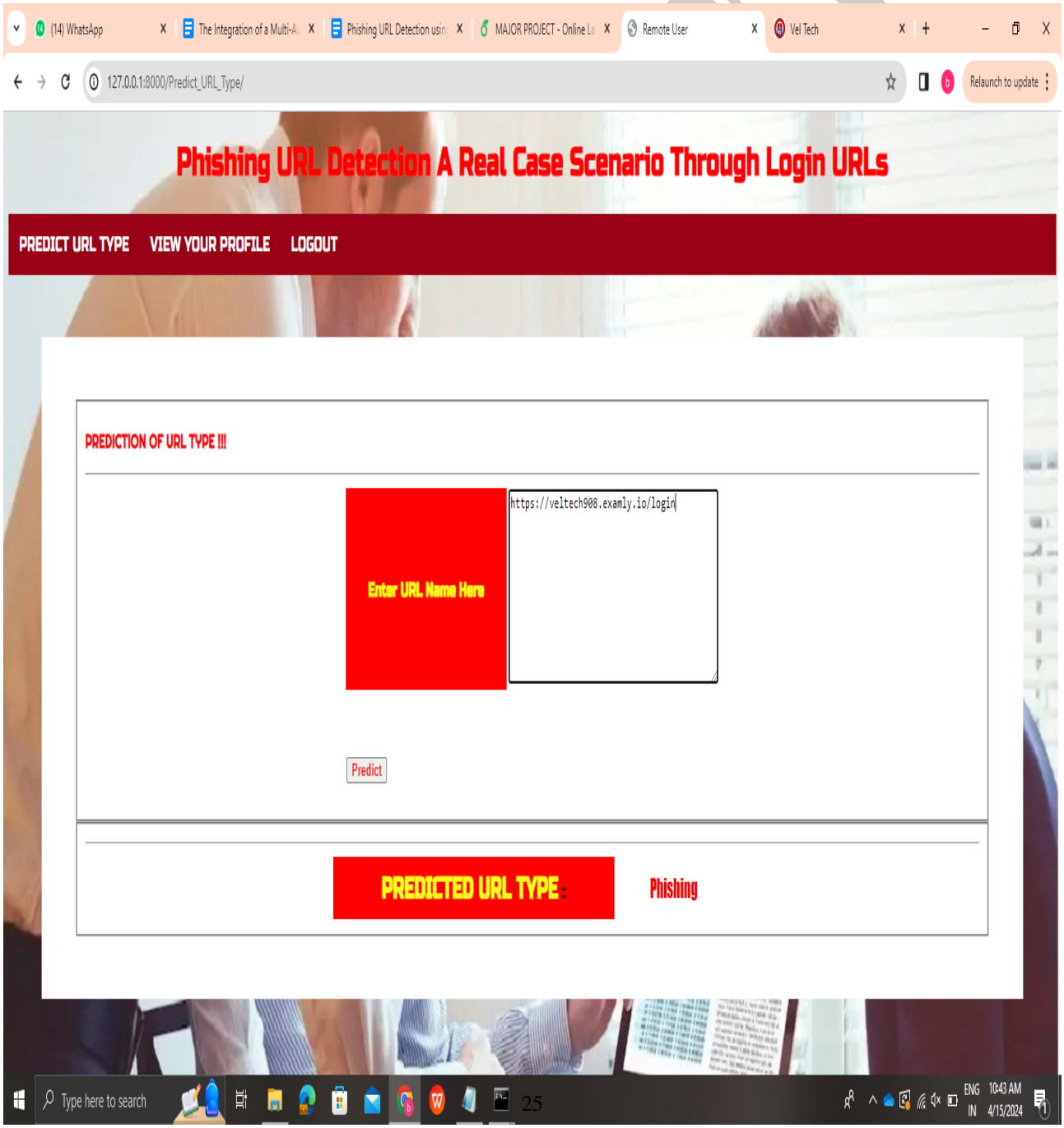


Figure 5.1: **Register Form**

5.1.2 Output Design

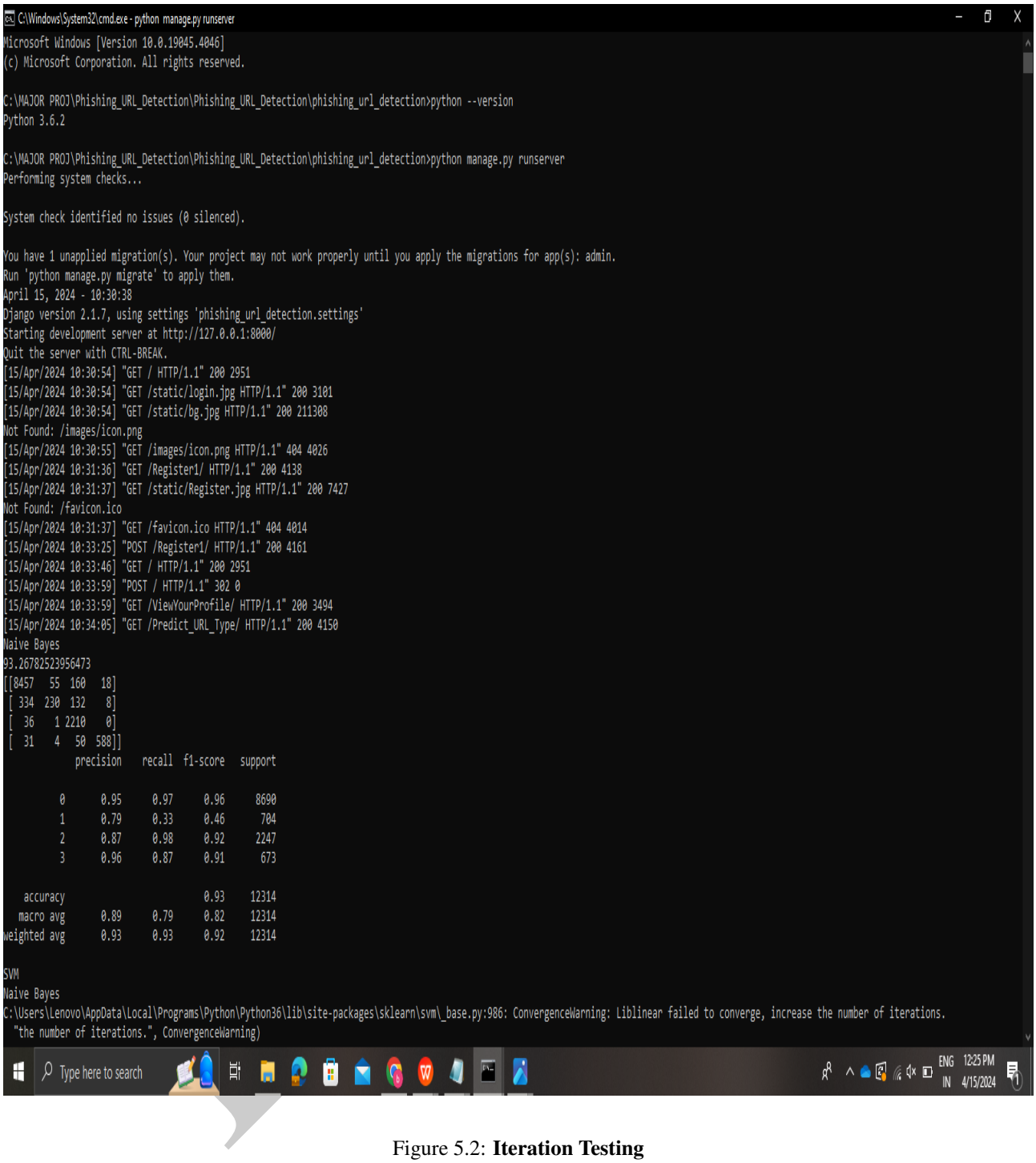


Figure 5.2: **Iteration Testing**

5.2 Testing

The main aim of the testing is to analyze the performance and to evaluate the errors that occur when the program is executed with different input sources and running in different operating environments.

5.3 Types of Testing

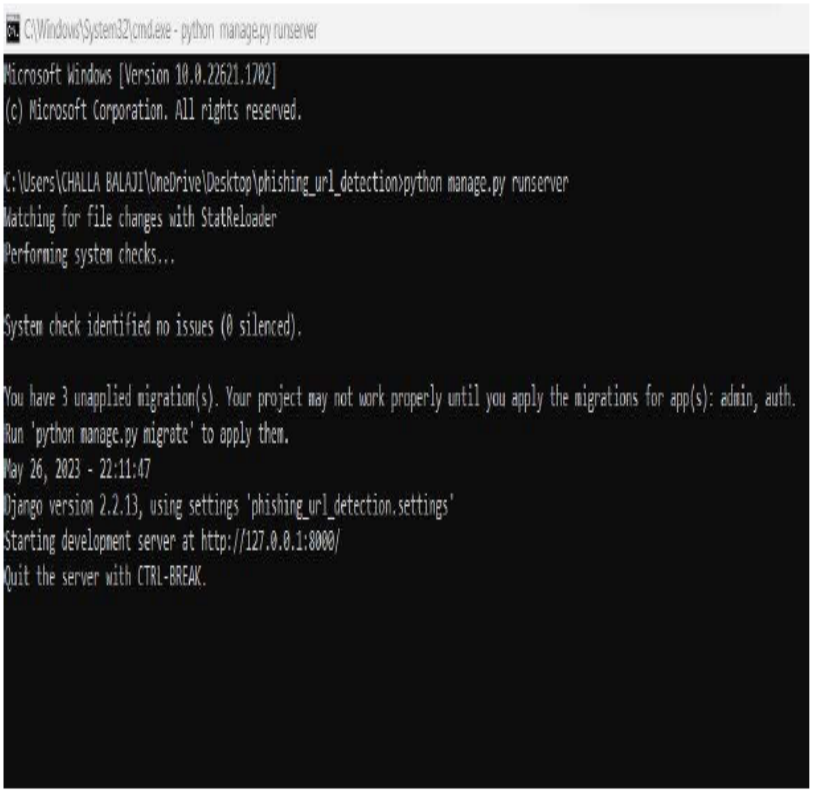
5.3.1 Unit testing

Unit testing, also known as component testing refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the classlevel, and the minimal unit tests include the constructors and destructors. Unit testing is a software development process that involves synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time and costs. The following Unit testing table shows the functions that were tested at the time of programming. The first column gives all the modules which were tested, and the second column gives the test results. Test results indicate if the functions, for given inputs are delivering valid outputs.

Test Result:Verified the sysytem results and procedure and passed all the test cases.

5.3.2 Integration testing

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together. Normally the former is considered a better practice since it allows interface issues to be located more quickly and fixed. Integration testing works to expose defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system.

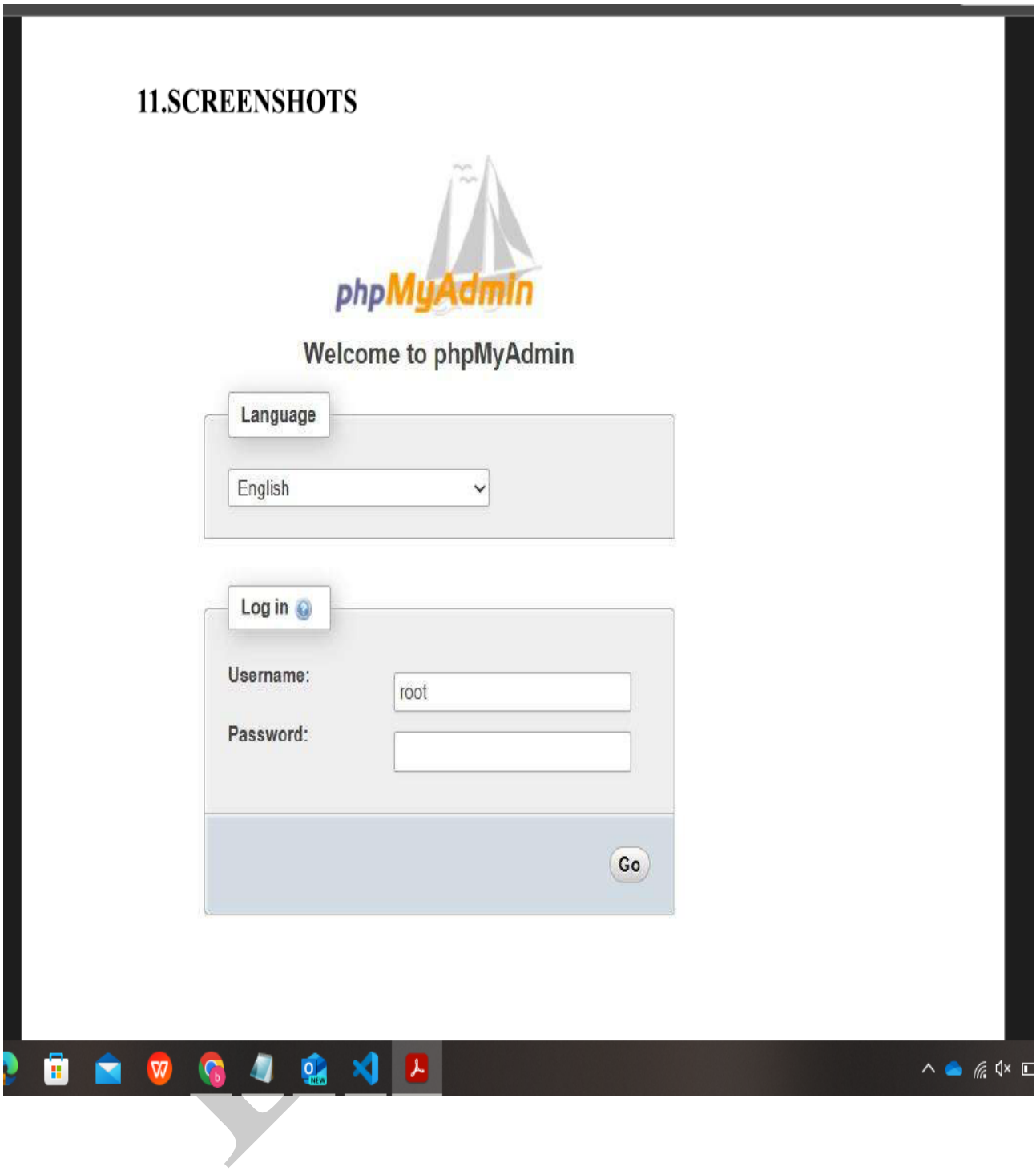


5.3.3 System testing

System testing for a project like ”phishing URL detection using random forest and support vector machine” would involve evaluating the entire system to ensure it meets its functional and non-functional requirements. This includes testing the integration of the random forest and support vector machine models, verifying that the system correctly identifies phishing URLs, and assessing its performance, scalability,

and reliability under various conditions

11.SCREENSHOTS



5.3.4 Test Result

Figure 5.3: **Test Image**

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

The proposed system is based on the Random forest Algorithm that creates many decision trees. Accuracy of proposed system is done by using random forest gives the ouput approximately 76 to 78 percent. Random forest implements many decision trees and also gives the most accurate output when compared to the decision tree. Random Forest algorithm is used in the two phases. Firstly, the RF algorithm extracts subsamples from the original samples by using the bootstrap resampling method and creates the decision trees for each testing sample and then the algorithm classifies the decision trees and implements a vote with the help of the largest vote of the classification as a final result of the classification. The random Forest algorithm always includes some of the steps as follows: Selecting the training dataset:Using the bootstrap random sampling method we can derive the K training sets from the original dataset properties using the size of all training set the same as that of original training dataset. Building the random forest algorithm: Creating a classification regression tree each of the bootstrap training set will generate the K decision trees to form a random forest model, uses the trees that are not pruned. Looking at the growth of the tree, 31 this approach is not chosen the best feature as the internal nodes for the branches but rather the branching process is a random selection of all the trees gives the best features.

6.2 Comparison of Existing and Proposed System

Existing system:(Decision tree)

In the Existing system, we implemented a decision tree algorithm that predicts whether to grant the loan or not. When using a decision tree model, it gives the training dataset the accuracy keeps improving with splits. We can easily overfit the dataset and doesn't know when it crossed the line unless we are using the cross validation. The advantages of the decision tree are model is very easy to interpret we can know that the variables and the value of the variable is used to split the data. But the accuracy of decision tree in existing system gives less accurate output that is less when compared to proposed system.

Proposed system:(Random forest algorithm)

Random forest algorithm generates more trees when compared to the decision tree and other algorithms. We can specify the number of trees we want in the forest and also we also can specify maximum of features to be used in the each of the tree. But, we cannot control the randomness of the forest in which the feature is a part of the algorithm. Accuracy keeps increasing as we increase the number of trees but it becomes static at one certain point. Unlike the decision tree it won't create more biased and decreases variance. Proposed system is implemented using the Random forest algorithm so that the accuracy is more when compared to the existing system.

6.3 Sample Code

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";
CREATE DATABASE IF NOT EXISTS `phishing_url_detection`
```

```
DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
USE `phishing_url_detection`;
CREATE TABLE IF NOT EXISTS `auth_group` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(80) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `name` (`name`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `auth_group_permissions` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `group_id` int(11) NOT NULL,
  `permission_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `auth_group_permissions_group_id_permission_
  id_0cd325b0_uniq` (`group_id`,`permission_id`),
  KEY `auth_group_permissio_permission_id_84c5c92e_fk_auth
  _perm` (`permission_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `auth_permission` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL,
  `content_type_id` int(11) NOT NULL,
  `codename` varchar(100) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `auth_permission_content_type_id_codename_
  01ab375a_uniq` (`content_type_id`,`codename`)
```

```
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=28 ;
INSERT INTO `auth_permission` (`id`, `name`,
`content_type_id`, `codename`) VALUES
(1, 'Can add log entry', 1, 'add_logentry'),
(2, 'Can change log entry', 1, 'change_logentry'),
(3, 'Can delete log entry', 1, 'delete_logentry'),
(4, 'Can add permission', 2, 'add_permission'),
(5, 'Can change permission', 2, 'change_permission'),
(6, 'Can delete permission', 2, 'delete_permission'),
(7, 'Can add group', 3, 'add_group'),
(8, 'Can change group', 3, 'change_group'),
(9, 'Can delete group', 3, 'delete_group'),
(10, 'Can add user', 4, 'add_user'),
(11, 'Can change user', 4, 'change_user'),
(12, 'Can delete user', 4, 'delete_user'),
(13, 'Can add content type', 5, 'add_contenttype'),
(14, 'Can change content type', 5, 'change_contenttype'),
(15, 'Can delete content type', 5, 'delete_contenttype'),
(16, 'Can add session', 6, 'add_session'),
(17, 'Can change session', 6, 'change_session'),
(18, 'Can delete session', 6, 'delete_session'),
(19, 'Can add client register_ model', 7, 'add_clientregister
_model'),
(20, 'Can change client register_ model', 7,
'change_clientregister_model'),
(21, 'Can delete client register_ model', 7,
```

```
'delete_clientregister_model'),
(22, 'Can add client posts_ model', 8,
'add_clientposts_model'),
(23, 'Can change client posts_ model', 8
'change_clientposts_model'),
(24, 'Can delete client posts_ model', 8,
'delete_clientposts_model'),
(25, 'Can add feedbacks_ model', 9,
'add_feedbacks_model'),
(26, 'Can change feedbacks_ model', 9,
'change_feedbacks_model'),
(27, 'Can delete feedbacks_ model', 9,
'delete_feedbacks_model');
CREATE TABLE IF NOT EXISTS `auth_user` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `password` varchar(128) NOT NULL,
  `last_login` datetime(6) DEFAULT NULL,
  `is_superuser` tinyint(1) NOT NULL,
  `username` varchar(150) NOT NULL,
  `first_name` varchar(30) NOT NULL,
  `last_name` varchar(150) NOT NULL,
  `email` varchar(254) NOT NULL,
  `is_staff` tinyint(1) NOT NULL,
  `is_active` tinyint(1) NOT NULL,
  `date_joined` datetime(6) NOT NULL,
  PRIMARY KEY (`id`),
```

```
  UNIQUE KEY `username` (`username`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `auth_user_groups` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `group_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `auth_user_groups_user_id_group_id_94350c0c
  _uniq` (`user_id`,`group_id`),
  KEY `auth_user_groups_group_id_97559544_fk_auth_group_id`
  (`group_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `auth_user_user_permissions` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `permission_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `auth_user_user_permissions_user_id_permission_
  id_14a6b632_uniq` (`user_id`,`permission_id`),
  KEY `auth_user_user_permi_permission_id_1fbb5f2c
  _fk_auth_perm` (`permission_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `django_admin_log` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `action_time` datetime(6) NOT NULL,
  `object_id` longtext,
```

```
  `object_repr` varchar(200) NOT NULL,
  `action_flag` smallint(5) unsigned NOT NULL,
  `change_message` longtext NOT NULL,
  `content_type_id` int(11) DEFAULT NULL,
  `user_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `django_admin_log_content_type_id_c4bce8eb_
  fk_django_co` (`content_type_id`),
  KEY `django_admin_log_user_id_c564eba6_
  fk_auth_user_id` (`user_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `django_content_type` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `app_label` varchar(100) NOT NULL,
  `model` varchar(100) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `django_content_type_app_label_model
  _76bd3d3b_uniq` (`app_label`,`model`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=10 ;

INSERT INTO `django_content_type` (`id`, `app_label`,
`model`) VALUES
(1, 'admin', 'logentry'),
(3, 'auth', 'group'),
(2, 'auth', 'permission'),
(4, 'auth', 'user'),
```

```
(8, 'Client_Site', 'clientposts_model'),
(7, 'Client_Site', 'clientregister_model'),
(9, 'Client_Site', 'feedbacks_model'),
(5, 'contenttypes', 'contenttype'),
(6, 'sessions', 'session');
CREATE TABLE IF NOT EXISTS `django_migrations` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `app` varchar(255) NOT NULL,
  `name` varchar(255) NOT NULL,
  `applied` datetime(6) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=22 ;
CREATE TABLE IF NOT EXISTS `remote_user_url_detection_type` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `url_name` varchar(30000) NOT NULL,
  `Prediction` varchar(300) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=9 ;
INSERT INTO `remote_user_url_detection_type` (`id`,
`url_name`, `Prediction`) VALUES
(1, 'allmusic.com/album/crazy-from-the-heat-r16990',
'Non-Phishing'),
(2, 'http://portal.dddgaming.com/docs/rules/15027/cn/game_cn.
html?amluMjAxNQ%3D%3D', 'Malware'),
(3, 'http://mmc-leipzig.com/index.php', 'Defacement'),
(4, '192.com/atoz/people/sturgeon/craig/\r\n', 'Non-Phishing'),
```

```
(5, 'tophyipsites.com', 'Non-Phishing'),
(6, 'http://update-information001.albayan.ly/', 'Phishing'),
(7, 'http://42.227.166.214:52835/Mozi.m', 'Malware'),
(8, 'http://florsiris.com/index.php?option=com_jevents&
task=day.listevents&year=2013&month=01&day=25&Itemid=59', 'Defa
```

Output

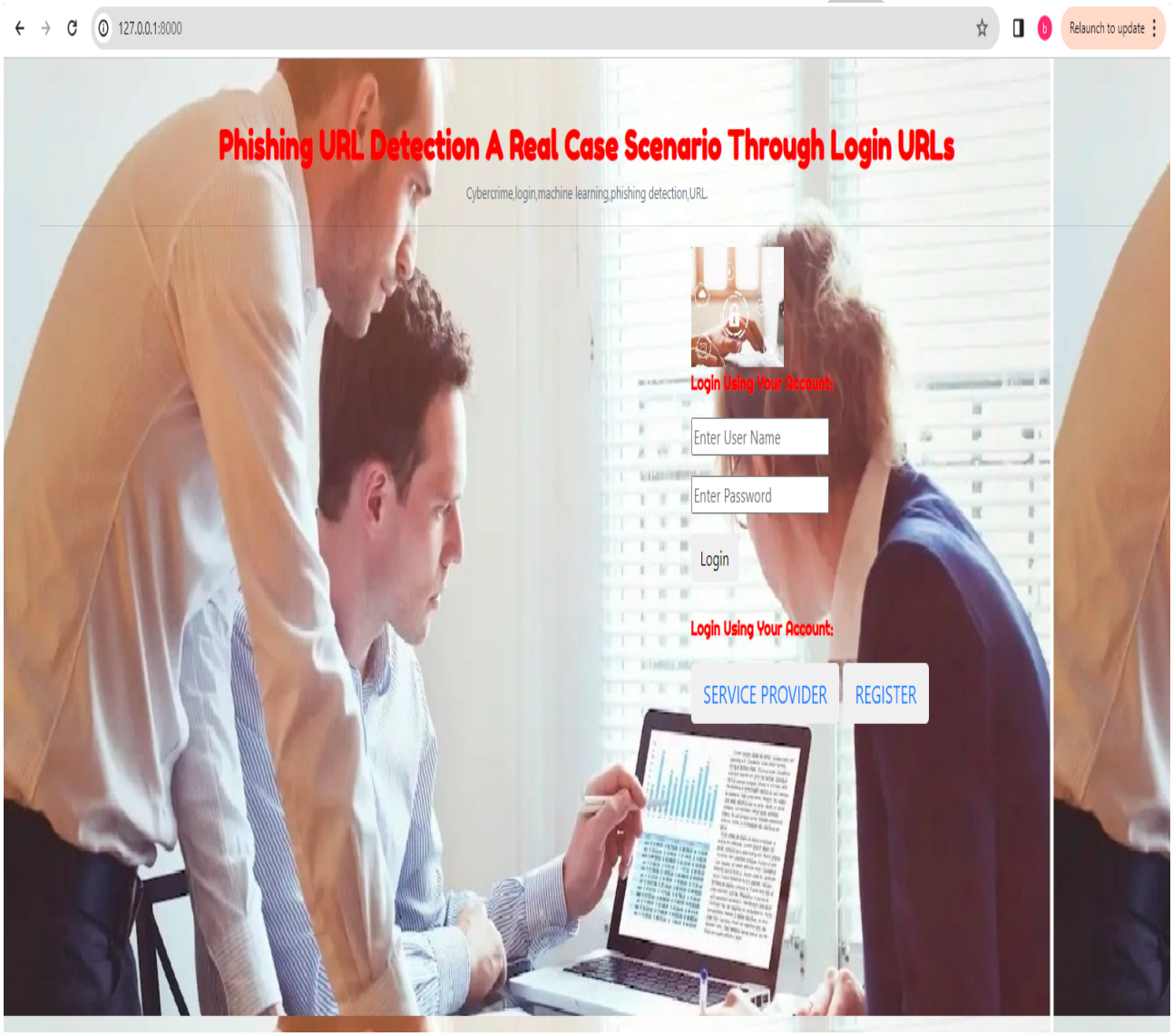


Figure 6.1: **Output 1**

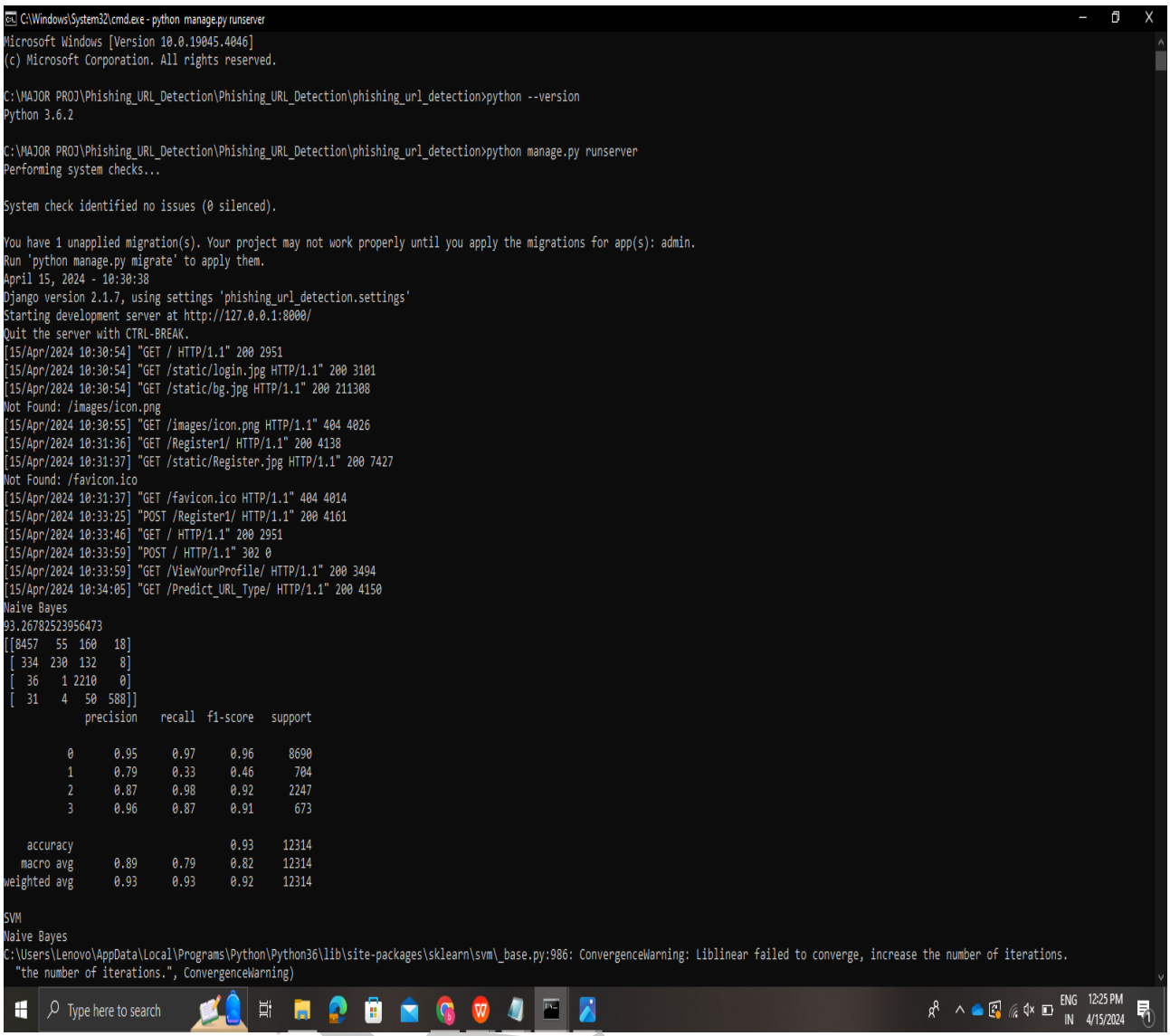


Figure 6.2: **Output 2**

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

Phishing detection mechanism aims to improve current blacklist methods, protecting users from malicious login forms. Our work provides an updated dataset PILU-90K for researchers to train and test their approaches. This dataset includes legitimate login URLs which are the most representative scenario for real-world phishing detection. We explored several URL-based detection models using deep learning and machine learning solutions trained with phishing and legitimate home URLs. The main advantage of our approach is the low false-positive rate when classifying this type of URL. Among the different evaluated models, TF-IDF combined with N-gram and LR algorithm obtained the best results .

No dependence on external services. A limitation of the description methods that use features such as domain age, page ranking on Google or Alexa or online blacklists, is their dependence on those services. Network slowdowns and service shortages can negatively impact analysis time, making real-time execution infeasible. Since phishing websites have a short lifespan low detection times are required to warn users before accessing phishing websites. Login website detection. Unlike other methods, which are trained with homepage URLs as representatives of the legitimate class, our model was trained with legitimate login websites. This ensures the correct classification of those websites. Therefore, our approach can be applied

to the real case scenario where users have to predict whether a login form page is legitimate or phishing. Updated and real-world dataset. PLU-60K is focused on using updated legitimate login URLs. As demonstrated, models trained with old datasets were not able to endure their performance over time. We provide an updated phishing URL dataset for models to learn from nowadays phishing URLs and trends, which are crucial for real-world performance. We demonstrated that phishing URL detection systems trained with legitimate land page URLs fail to classify legitimate login URLs correctly. Different categories for current phishing attacks were identified by using a domain frequency analysis. While standalone and compromised domains were the most common approaches, free hosting services, cloud web servers and malware blog posts represent many current phishing attacks due to their cost and effectiveness for phishing campaigns.

7.2 Future Enhancements

Future enhancements for the project on phishing URL detection leveraging random forest and support vector machine entail exploring ensemble learning methods to amalgamate predictions from multiple classifiers, thereby enhancing accuracy. Additionally, investigating deep learning architectures like convolutional or recurrent neural networks could extract intricate features from URLs, potentially improving detection efficacy. Real-time URL scanning mechanisms could be implemented for instantaneous protection against emerging threats. Feature engineering could be refined by considering additional URL attributes or incorporating domain reputation scores from diverse sources. Advanced anomaly detection algorithms could be explored to identify subtle deviations indicative of phishing attempts. A user-friendly interface or integration with existing cybersecurity tools could facilitate seamless deployment and usage. Comprehensive performance evaluations across diverse datasets

would ensure robustness and generalizability. Collaboration with cybersecurity experts would aid in updating the model with the latest phishing tactics. Techniques for handling imbalanced datasets could mitigate the impact of skewed class distributions, while research into interpretable machine learning methods would provide insights into the detection model's decision-making process, enhancing transparency and trustworthiness.

Chapter 8

PLAGIARISM REPORT

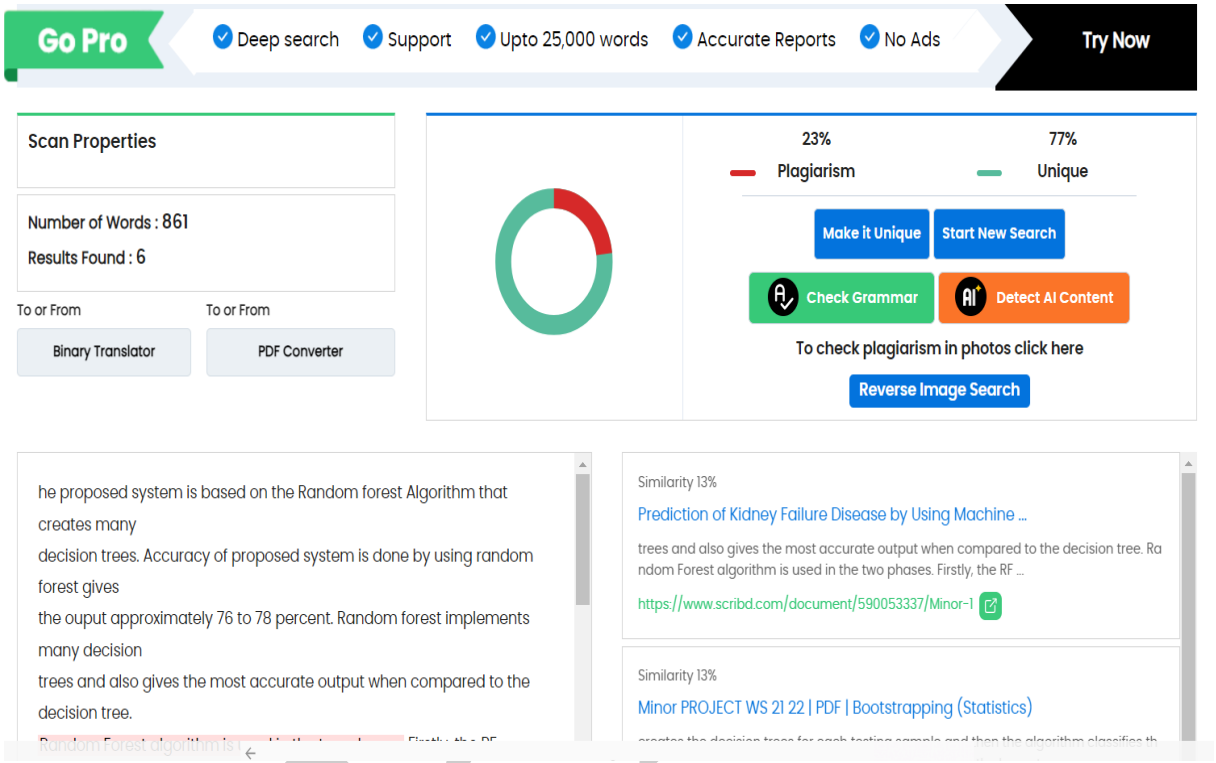


Figure 8.1: **Plagarism Report**

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```
<?xml version="1.0" encoding="UTF-8"?>
<module type="PYTHON_MODULE" version="4">
  <component name="NewModuleRootManager">
    <content url="file://$MODULE_DIR$">
      <excludeFolder url="file://$MODULE_DIR$/venv" />
    </content>
    <orderEntry type="inheritedJdk" />
    <orderEntry type="sourceFolder" forTests="false" />
  </component>
  <component name="TestRunnerService">
    <option name="PROJECT_TEST_RUNNER" value="Unittests" />
  </component>
</module>
```

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";
CREATE DATABASE IF NOT EXISTS `phishing_url_detection` DEFAULT
CHARACTER SET latin1 COLLATE latin1_swedish_ci;
```

```
USE `phishing_url_detection`;
CREATE TABLE IF NOT EXISTS `auth_group` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(80) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `name` (`name`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `auth_group_permissions` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `group_id` int(11) NOT NULL,
  `permission_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `auth_group_permissions_group_id_permission_id_0cd
  325b0_uniq` (`group_id`,`permission_id`) KEYauth_group_permi
  permission_id_84c5c92e_fk_auth_perm` (`permission_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `auth_permission` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL,
  `content_type_id` int(11) NOT NULL,
  `codename` varchar(100) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `auth_permission_content_type_id_codename
  _01ab375a_uniq` (`content_type_id`,`codename`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=28 ;
INSERT INTO `auth_permission` (`id`, `name`, `content_type_id`,
```

```
`codename`) VALUES
(1, 'Can add log entry', 1, 'add_logentry'),
(2, 'Can change log entry', 1, 'change_logentry'),
(3, 'Can delete log entry', 1, 'delete_logentry'),
(4, 'Can add permission', 2, 'add_permission'),
(5, 'Can change permission', 2, 'change_permission'),
(6, 'Can delete permission', 2, 'delete_permission'),
(7, 'Can add group', 3, 'add_group'),
(8, 'Can change group', 3, 'change_group'),
(9, 'Can delete group', 3, 'delete_group'),
(10, 'Can add user', 4, 'add_user'),
(11, 'Can change user', 4, 'change_user'),
(12, 'Can delete user', 4, 'delete_user'),
(13, 'Can add content type', 5, 'add_contenttype'),
(14, 'Can change content type', 5, 'change_contenttype'),
(15, 'Can delete content type', 5, 'delete_contenttype'),
(16, 'Can add session', 6, 'add_session'),
(17, 'Can change session', 6, 'change_session'),
(18, 'Can delete session', 6, 'delete_session'),
(19, 'Can add client register_ model', 7,
'add_clientregister_model'),
(20, 'Can change client register_ model', 7,
'change_clientregister_model'),
(21, 'Can delete client register_ model', 7,
'delete_clientregister_model'),
(22, 'Can add client posts_ model', 8,
```

```
'add_clientposts_model'),
(23, 'Can change client posts_ model', 8,
'change_clientposts_model'),
(24, 'Can delete client posts_ model', 8,
'delete_clientposts_model'),
(25, 'Can add feedbacks_ model', 9,
'add_feedbacks_model'),
(26, 'Can change feedbacks_ model', 9,
'change_feedbacks_model'),
(27, 'Can delete feedbacks_ model', 9,
'delete_feedbacks_model');
CREATE TABLE IF NOT EXISTS `auth_user` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `password` varchar(128) NOT NULL,
  `last_login` datetime(6) DEFAULT NULL,
  `is_superuser` tinyint(1) NOT NULL,
  `username` varchar(150) NOT NULL,
  `first_name` varchar(30) NOT NULL,
  `last_name` varchar(150) NOT NULL,
  `email` varchar(254) NOT NULL,
  `is_staff` tinyint(1) NOT NULL,
  `is_active` tinyint(1) NOT NULL,
  `date_joined` datetime(6) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `username` (`username`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
```

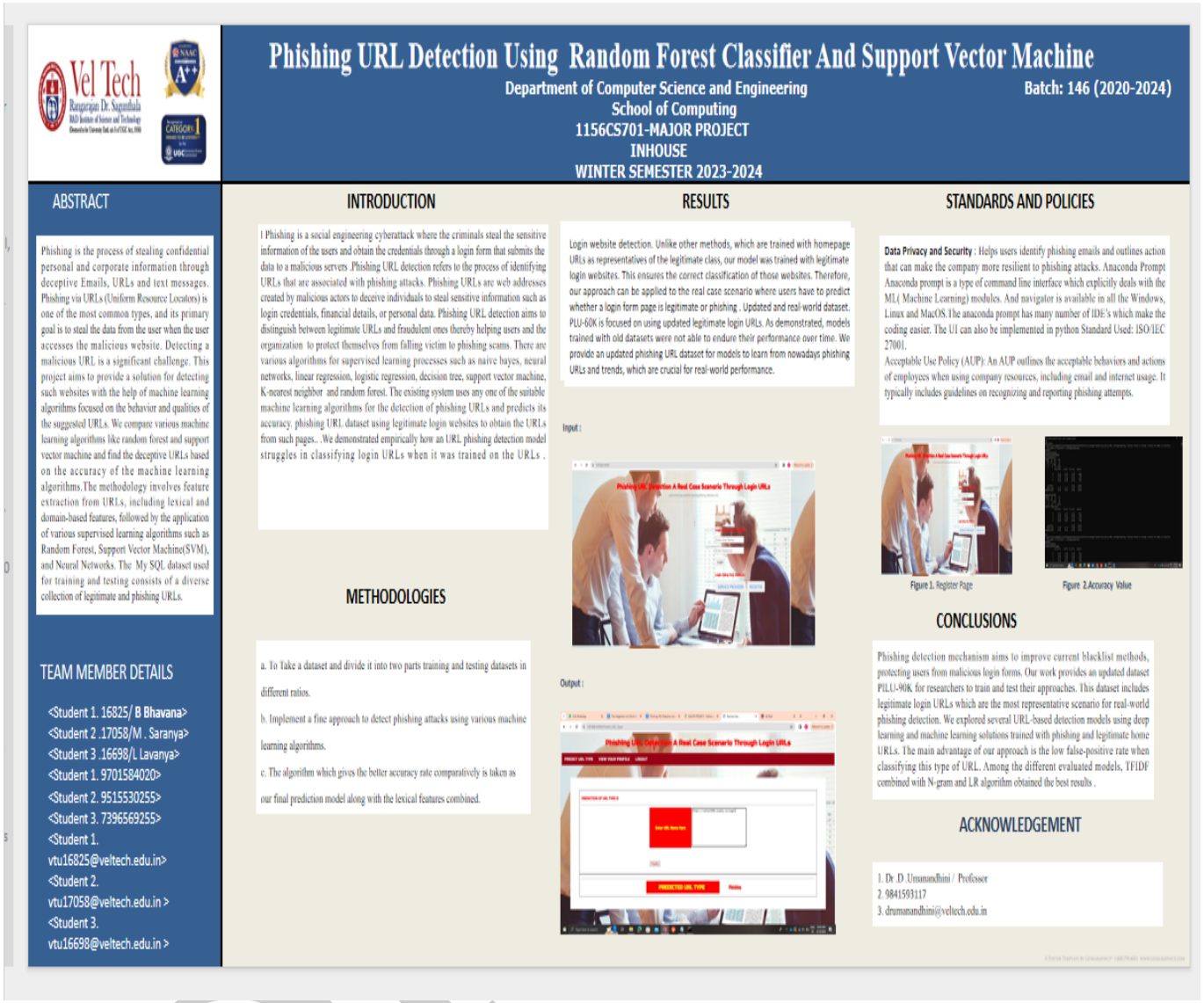
```
CREATE TABLE IF NOT EXISTS `auth_user_groups` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `group_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `auth_user_groups_user_id_group_id_94350c0c
  _uniq` (`user_id`,`group_id`),
  KEY `auth_user_groups_group_id_97559544_fk_auth_
  group_id` (`group_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `auth_user_user_permissions` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `permission_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `auth_user_user_permissions_user_id_permission_
  id_14a6b632_uniq` (`user_id`,`permission_id`),
  KEY `auth_user_user_permi_permission_id_1fbb5f2c_
  fk_auth_perm` (`permission_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `django_admin_log` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `action_time` datetime(6) NOT NULL,
  `object_id` longtext,
  `object_repr` varchar(200) NOT NULL,
  `action_flag` smallint(5) unsigned NOT NULL,
```

```
  `change_message` longtext NOT NULL,
  `content_type_id` int(11) DEFAULT NULL,
  `user_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `django_admin_log_content_type_id_c4bce8eb_fk_django_co`
  (`content_type_id`),
  KEY `django_admin_log_user_id_c564eba6_fk_auth_
  user_id` (`user_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE IF NOT EXISTS `django_content_type` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `app_label` varchar(100) NOT NULL,
  `model` varchar(100) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `django_content_type_app_label_model_76bd3d3b_
  uniq` (`app_label`,`model`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=10 ;
INSERT INTO `django_content_type` (`id`, `app_label`, `model`)
VALUES
(1, 'admin', 'logentry'),
(3, 'auth', 'group'),
(2, 'auth', 'permission'),
(4, 'auth', 'user'),
(8, 'Client_Site', 'clientposts_model'),
(7, 'Client_Site', 'clientregister_model'),
(9, 'Client_Site', 'feedbacks_model'),
```

```
(5, 'contenttypes', 'contenttype'),
(6, 'sessions', 'session');
CREATE TABLE IF NOT EXISTS `django_migrations` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `app` varchar(255) NOT NULL,
  `name` varchar(255) NOT NULL,
  `applied` datetime(6) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=22 ;
CREATE TABLE IF NOT EXISTS `remote_user_url_detection_type` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `url_name` varchar(30000) NOT NULL,
  `Prediction` varchar(300) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=9 ;
INSERT INTO `remote_user_url_detection_type`
(`id`, `url_name`, `Prediction`) VALUES
(1, 'allmusic.com/album/crazy-from-the-heat-r16990',
'Non-Phishing'),
(2, 'http://portal.dddgaming.com/docs/rules/15027/cn/game_
cn.html?amluMjAxNQ%3D%3D', 'Malware'),
(3, 'http://mmc-leipzig.com/index.php', 'Defacement'),
(4, '192.com/atoz/people/sturgeon/craig/\r\n',
'Non-Phishing'),
(5, 'tophyipsites.com', 'Non-Phishing'),
(6, 'http://update-information001.albayan.ly/', 'Phishing'),
```

```
(7, 'http://42.227.166.214:52835/Mozi.m', 'Malware'),
(8, 'http://florsiris.com/index.php?option=com_jevents&
task=day.listevents&year=2013&
month=01&day=25&Itemid=59', 'Defacement');
```

9.2 Poster Presentation



References

[1] Ahammad, SK Hasane, Sunil D. Kale, Gopal D. Upadhye, Sandeep Dwarkanath Pande, E. Venkatesh Babu, Amol V. Dhumane, and Mr Dilip Kumar Jang Bahadur. ”Phishing URL detection using machine learning methods.” Advances in Engineering Software 173 (2022): 103288.

[2] R. Arora, R. Gupta and P. Yadav, ”Mischievous URL Prediction Through Supervised Machine Learning Algorithms,” 2023 IEEE International Conference on Contemporary Computing and Communications (InC4), Bangalore, India, 2023, pp. 1-5, doi: 10.1109/InC457730.2023.10262881.

[3] Manuel Sánchez-Paniagua, Eduardo Fidalgo, Enrique Alegre, Rocío Alaiz-Rodríguez, Phishing websites detection using a novel multipurpose dataset and web technologies features,Expert Systems with Applications,Volume 207,2022,118010,ISSN 0957-4174.

[4] M. Almousa and M. Anwar, ”A URL-Based Social Semantic Attacks Detection With Character-Aware Language Model,” in IEEE Access, vol. 11, pp. 10654-10663, 2023, doi: 10.1109/ACCESS.2023.3241121.

[5] A. S. Rafsanjani, N. B. Kamaruddin, H. M. Rusli and M. Dabbagh, ”QsecR: Secure QR Code Scanner According to a Novel Malicious URL Detection Framework,” in IEEE Access, vol. 11, pp. 92523-92539, 2023, doi: 10.1109/ACCESS.2023.3291811.

[6] M. Alsaedi, F. A. Ghaleb, F. Saeed, J. Ahmad and M. Alasli, ”Multi-Modal Features Representation-Based Convolutional Neural Network Model for Malicious Website Detection,” in IEEE Access, vol. 12, pp. 7271-7284, 2024, doi: 10.1109/ACCESS.2023.3348071.

[7] S. Asiri, Y. Xiao, S. Alzahrani, S. Li and T. Li, ”A Survey of Intelligent Detection Designs of HTML URL Phishing Attacks,” in IEEE Access, vol. 11, pp. 6421-6443, 2023, doi: 10.1109/ACCESS.2023.3237798.

[8] Y. Liang, Q. Wang, K. Xiong, X. Zheng, Z. Yu and D. Zeng, ”Robust Detection of Malicious URLs With Self-Paced Wide Deep Learning,” in IEEE Transactions on Dependable and Secure Computing, vol. 19, no. 2, pp. 717-730, 1 March-April 2022, doi: 10.1109/TDSC.2021.3121388.

[9] B. Sabir, M. A. Babar, R. Gaire and A. Abuadbba, ”Reliability and Robustness analysis of Machine Learning based Phishing URL Detectors,” in IEEE Transactions on Dependable and Secure Computing, doi: 10.1109/TDSC.2022.3218043.

[10] A. Karim, M. Shahroz, K. Mustofa, S. B. Belhaouari and S. R. K. Joga, ”Phishing Detection System Through Hybrid Machine Learning Based on URL,” in IEEE Access, vol. 11, pp. 36805-36822, 2023, doi: 10.1109/ACCESS.2023.3252366.

General Instructions

- Cover Page should be printed as per the color template and the next page also should be printed in color as per the template
- **Wherever Figures applicable in Report , that page should be printed in color**
- Dont include general content , write more technical content
- Each chapter should minimum contain 3 pages
- Draw the notation of diagrams properly
- Every paragraph should be started with one tab space
- Literature review should be properly cited and described with content related to project
- All the diagrams should be properly described and dont include general information of any diagram
- Example Use case diagram - describe according to your project flow
- All diagrams,figures should be numbered according to the chapter number and it should be cited properly
- **Testing and codequality should done in Sonarqube Tool**
- Test cases should be written with test input and test output
- All the references should be cited in the report
- **AI Generated text will not be considered**
- **Submission of Project Execution Files with Code in GitHub Repository**
- **Thickness of Cover and Rear Page of Project report should be 180 GSM**

- **Internship Offer letter and neccessary documents should be attached**
- **Strictly dont change font style or font size of the template, and dont customize the latex code of report**
- **Report should be prepared according to the template only**
- **Any deviations from the report template,will be summarily rejected**
- **Number of Project Soft Binded copy for each and every batch is (n+1) copies as given in the table below**
- For **Standards and Policies** refer the below link
  https://law.resource.org/pub/in/manifest.in.html
- Plagiarism should be less than 15%
- **Journal/Conference Publication proofs should be attached in the last page of Project report after the references section**

width=!,height=!,page=-