RV Educational Institutions ®
RV College of Engineering ®

Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi

Approved by AICTE, New Delhi

Go, change the world

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PHISHING WEBSITE DETECTION

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

VI SEMESTER

PROJECT REPORT

Submitted by

Name 1: Prajwal T S	**USN1: 1RV20CS114**
Name2: Prasanna Suresh Naik	**USN2: 1RV20CS118**

Under the guidance of

Dr. Vinay Hegde
Associate Professor
Dept of CSE
RV College of Engineering

Bachelor of Engineering
in
Computer Science and Engineering

2022-2023

RV COLLEGE OF ENGINEERING®, BENGALURU-59

(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the **Artificial Intelligence and Machine Learning Project Work** titled "**Phishing Website Detection**" is carried out by **Prajwal T S (1RV20CS114) and Prasanna Suresh Naik (1RV20CS118)** who are bonafide students of RV College of Engineering, Bengaluru, in partial fulfilment for the **Internal Assessment of Course: ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING LAB (18CS62)** during the year 2022-2023. It is certified that all corrections/suggestions indicated for the Internal Assessment have been incorporated in the report.

Dr Vinay Hegde
Associate Professor,
Department of CSE,
R.V.C.E., Bengaluru –59

Head of Department,
Department of CSE,
R.V.C.E., Bengaluru–59

External Viva

Name of Examiners	Signature with Date
1	
2	

RV COLLEGE OF ENGINEERING®, BENGALURU-59
(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We, **Prajwal T S (1RV20CS114) and Prasanna Suresh Naik (1RV20CS118)** the students of Sixth Semester B.E., Department of Computer Science and Engineering, RV College of Engineering, Bengaluru hereby declare that project titled **"Phishing website detection"** has been carried out by us and submitted in partial fulfillment for the **Internal Assessment of the Course: ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING LAB (18CS62)** during the academic year 2022-2023. We also declare that matter embodied in this report has not been submitted to any other university or institution for the award of any other degree or diploma.

Place: Bengaluru

Date:

Name	**Signature**
1. Prajwal T S (1RV20CS114)	
2. Prasanna Suresh Naik (1RV20CS118)	

ACKNOWLEDGEMENT

We are indebted to our Faculty (Lab)**, Faculty name,** designation**, Dept of CSE, RV College of Engineering** for his/her wholehearted support, suggestions and invaluable advice throughout our project work and helped in the preparation of this report.

We also express our gratitude to our theory Faculty(Theory) **Names** with designations**,** Department of Computer Science and Engineering for their valuable comments and suggestions.

Our sincere thanks to **Dr. Ramakanth Kumar P.**, Professor and Head, Department of Computer Science and Engineering, RVCE for his support and encouragement.

We express sincere gratitude to our beloved Principal, **Dr. K. N. Subramanya** for his appreciation towards this project work.

We thank all the **teaching staff and technical staff** of Computer Science and Engineering department, RVCE for their help.

Lastly, we take this opportunity to thank our **family** members and **friends** who provided all the backup support throughout the project work.

ABSTRACT

Phishing attacks are a rapidly expanding threat in the cyber world, costing internet users billions of dollars each year. It is a criminal crime that involves the use of a variety of social engineering tactics to obtain sensitive information from users. Phishing techniques can be detected using a variety of types of communication, including email, instant chats, pop-up messages, and web pages. This study develops and creates a model that can predict whether a URL link is legitimate or phishing.

The data set used for the classification was sourced from an open source service called ‘Phish Tank’ which contain phishing URLs in multiple formats such as CSV, JSON, etc. and also from the University of New Brunswick dataset bank which has a collection of benign, spam, phishing, malware & defacement URLs. Over six (6) machine learning models and deep neural network algorithms all together are used to detect phishing URLs.

This study aims to develop a web application software that detects phishing URLs from the collection of over 5,000 URLs which are randomly picked respectively and are fragmented into 80,000 training samples & 20,000 testing samples, which are equally divided between phishing and legitimate URLs. The URL dataset is trained and tested base on some feature selection such as address bar-based features, domain-based features, and HTML & JavaScript-based features to identify legitimate and phishing URLs.

In conclusion, the study provided a model for URL classification into phishing and legitimate URLs. This would be very valuable in assisting individuals and companies in identifying phishing attacks by authenticating any link supplied to them to prove its validity.

TABLE OF CONTENTS

List of Figures

CHAPTER-1

INTRODUCTION

CHAPTER 1
INTRODUCTION

1.1 Background Information

The Internet has become an important part of our lives for gathering and disseminating information, particularly through social media. According to Pamela (2021), the Internet is a network of computers containing valuable data, so there are many security mechanisms in place to protect that data, but there is a weak link: the human. When a user freely gives away their data or access to their computer, security mechanisms have a much more difficult time protecting their data and devices. Therefore, Imperva (2021) defines social engineering (a type of attack used to steal user data, including login credentials and credit card numbers) as a type of attack that is one of the most common social engineering attacks. The attack happens when an attacker fools a victim into opening an email, instant message, or text message as if it were from a trusted source. Upon clicking the link, the recipient is fooled into believing that they've received a gift and unsuspectingly clicks a malicious link, resulting in the installation of malware, the freezing of the system as part of a ransomware attack, or the disclosure of sensitive information.

Computer security threats have increased substantially in recent years, owing to the rapid adoption of technology improvements, while simultaneously increasing the vulnerability of human exploitation. Users should know how the phishers do it, and they should also be aware of techniques to help protect themselves from becoming phished.

2 The strategies employed by cybercriminals are becoming more complex as technology advances. Other than phishing, there are a variety of methods for obtaining personal information from users. KnowBe4 (2021) stated the following techniques:

a) Vishing (Voice Phishing): This kind of phishing includes the phisher calling the victim to get personal information about the bank account. The most common method of phone phishing is to use a phony caller ID.

b) Smishing (SMS Phishing): Phishing via Short Message Service (SMS) is known as Smishing. It is a method of luring a target through the SMS text message service by sending a link to a phishing website.

c) Ransomware: A ransomware attack is a type of attack that prevents users from accessing a device or data unless they pay up.

d) Malvertising: Malvertising is malicious advertising that uses active scripts to download malware or push undesirable information onto your computer. The most prevalent techniques used in malvertisements are exploits in Adobe PDF and Flash.
Hence, this is a rapidly evolving threat to individuals as well as big and small corporations. Criminals now have access to industrial-strength services on the dark web, resulting in an increase in the amount of these phishing links and emails, and, more frighteningly, they are increasing in ‘quality,' making them tougher to detect

1.2. Motivation

The motivation behind the project "Phishing Website Detection using Machine Learning" is to address the growing threat of phishing attacks on the internet. Phishing is a form of cybercrime where attackers impersonate legitimate websites to trick users into revealing sensitive information such as login credentials, credit card details, or personal data. These attacks can lead to identity theft, financial loss, and other serious consequences for individuals and organizations. Machine learning can play a crucial role in detecting phishing websites due to its ability to analyze large amounts of data, identify patterns, and make predictions based on learned patterns.

1.3 Problem Statement

Phishing attacks have gotten increasingly complex, it is very difficult for an average person to determine if an email message link or website is legitimate. Cyber-attacks by criminals that employ phishing schemes are so prevalent and successful nowadays. Hence, this project seeks to address fake URLs and domain names by identifying phishing website links. Therefore, having a web application that provides the user an interface to check if a URL is Phishing or legitimate will help decrease security risks to individuals and organizations.

1.4. Objectives

To accomplish the project's purpose, the following particular objectives have been established:

i. dataset collection and pre-processing;
ii. ii. machine-learning model selection and development;

iii. iii. development of a web-based application for detection;
iv. iv. Integration of the developed model to a web application.

1.5. Scope

i. This study explores data science and machine learning models that use datasets gotten from open-source platforms to analyze website links and distinguish between phishing and legitimate URL links.
ii. The model will be integrated into a web application, allowing a user to predict if a URL link is legitimate or phishing.
iii. This online application is compatible with a variety of browsers.

1.6. Methodology

An extensive review was done on related topics and existing documented materials such as journals, e-books, and websites containing related information gathered which was examined and reviewed to retrieve essential data to better understand and know how to help improve the system.

The methodology used to achieve the earlier stated objectives is explained below. The dataset collection consists of phishing and legitimate URLs which were obtained from open-source platforms. The dataset was then pre-processed that is cleaned up from any abnormality such as missing data to avoid data imbalance. Afterward, expository data analysis was done on the dataset to explore and summarize the dataset. Once the dataset was free from all anomalies, website content-based features were extracted from the dataset to get accurate features to train and test the model. An extensive review was done on existing works of literature and machine learning models on detecting phishing websites to best decide the classification models to solve the problem of detecting phishing websites.

Hence, Series of these machine learning classification models such as Decision Tree and Random Forest was deployed on the dataset to distinguish between phishing and legitimate URLs. The best model with high training accuracy out of all the deployed models was selected then integrated into a developed web application. Thus, a user can enter a URL link on the web application to predict if it is phishing or legitimate.

CHAPTER-2

Overview of AI and ML Component in the Problem Domain

CHAPTER 2

Overview of AI and ML Component in the Problem Domain

2.1. Introduction

Phishing Website Detection is a fascinating application of Artificial Intelligence (AI) and Machine Learning (ML) that aims to interpret and identify human emotions from facial expressions. With the advancements in computer vision and deep learning techniques, AI and ML play a pivotal role in building accurate and efficient systems for recognizing emotions based on facial cues. This technology has numerous practical applications, from human-computer interaction to healthcare and security.

AI and ML are the core components driving the Phishing Website Detection System. Machine learning algorithms are employed to extract meaningful features from Phishing Website, while ML models are trained on extensive datasets to recognize and classify various domainstates. Through pattern recognition and learning from labeled examples, ML models can accurately identify which is legitimate website or phishing website.

1.Feature Extraction and Representation

Feature extraction is a crucial step in building a machine learning model for phishing website detection. It involves transforming raw website data into a format that can be understood and used by the learning algorithm. The selected features should capture relevant information about the websites that can help discriminate between legitimate and phishing websites effectively. Here are some common feature extraction techniques for phishing website detection:

URL-based Features:

Domain-based features: Extracting information from the website's domain, such as the length of the domain name, the number of dots in the domain, the presence of hyphens or numbers, etc.

Subdomain and path features: Analyzing the structure of the URL to check for unusual or suspicious patterns.

URL length: Phishing URLs are often longer and more complex than legitimate URLs.

Use of redirection: Counting the number of redirects, as excessive redirection is more common in phishing attacks.

Content-based Features:

Text analysis: Extracting keywords, n-grams, and phrases from the website content to identify common phishing terms or malicious phrases.

HTML analysis: Examining the structure of the HTML code to detect suspicious elements like hidden forms or iframes that may be used for phishing.

SSL/TLS Certificate Features:

Extracting information from the SSL/TLS certificate, such as the certificate's expiration date, issuer, and certificate authority (CA) details.

Checking for the use of self-signed certificates or certificates from untrusted CAs.

Website Traffic Features:

Age of the domain: Phishing websites are often created recently, so the age of the domain can be a relevant feature.

Website popularity: Comparing the website's traffic with known legitimate websites can help identify anomalies.

2.Training and Optimization:

Machine Learning plays a crucial role in training the phishing website detection model. Supervised learning is commonly used, where the model is trained on a labeled dataset containing facial images and corresponding emotion labels. The training process involves optimizing model parameters through techniques like backpropagation and gradient descent to minimize the classification error. Hyperparameter tuning ensures the model generalizes well on unseen data.

2.2. Relevant Technical and Mathematical Details

Data Preprocessing: Before feeding the data into the machine learning model, it needs to be preprocessed. This step includes tasks like data cleaning (handling missing values, outliers), data normalization (scaling numerical features to a common range), and data encoding (converting categorical variables into numerical representations, e.g., one-hot encoding).

Feature Selection: Selecting the most relevant features from the dataset is essential to avoid noise and reduce computational overhead. Techniques like information gain, chi-square, correlation analysis, or recursive feature elimination can help identify the most informative features.

Feature Engineering: As mentioned earlier, feature engineering involves creating new features or transforming existing ones to enhance the model's performance. This step requires domain knowledge and creativity. For example, generating textual features from the website content or calculating age from domain registration date.

Dataset Splitting: The dataset is typically split into training, validation, and testing sets. The training set is used to train the model, the validation set is used for hyperparameter tuning, and the testing set is used to evaluate the final model's performance on unseen data.

Machine Learning Algorithms: Various machine learning algorithms can be used for phishing website detection. Common choices include decision trees, random forests, logistic regression, support vector machines (SVM), k-nearest neighbors (KNN), and deep learning models like neural networks.

Model Evaluation Metrics: To assess the performance of the model, relevant evaluation metrics should be chosen. Common metrics for binary classification tasks like phishing website detection include accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC).

Cross-Validation: Cross-validation is used to evaluate the model's generalization performance. Techniques like k-fold cross-validation divide the dataset into k subsets and iteratively train and validate the model on different combinations of the subsets.

Hyperparameter Tuning: Each machine learning algorithm has specific hyperparameters that influence its performance. Tuning these hyperparameters can significantly improve the model's performance. Techniques like grid search or random search can be used to find the optimal hyperparameter values.

Imbalanced Data Handling: In real-world scenarios, the dataset may suffer from class imbalance, where one class (e.g., legitimate websites) significantly outnumbers the other (e.g., phishing websites). Proper techniques like oversampling, undersampling, or using class weights can help address this issue.

Ensemble Methods: Ensemble methods, such as bagging (e.g., random forests) and boosting (e.g., AdaBoost, XGBoost), can be employed to combine multiple weak classifiers into a stronger, more robust model.

Model Interpretability: In some cases, model interpretability is essential for understanding the reasoning behind the model's predictions. Techniques like feature importance analysis and SHAP

(SHapley Additive exPlanations) values can provide insights into the model's decision-making process.

Model Deployment: Once the model is trained and evaluated, it can be deployed in production environments. For web-based applications, API endpoints can be created to receive website URLs as input and return predictions (legitimate or phishing) as output.

Mathematically, machine learning algorithms involve optimization techniques for finding the best model parameters. For instance, gradient descent is often used in training neural networks. Additionally, various loss functions are used to measure the error between predicted and actual labels during training. Examples include the binary cross-entropy loss for binary classification tasks and the mean squared error (MSE) for regression tasks.

Overall, phishing website detection using machine learning is a multi-faceted task that requires a combination of data preprocessing, feature engineering, algorithm selection, and model evaluation to build an effective and accurate solution.

CHAPTER-3

Software Requirements Specification

CHAPTER 3

Software Requirements Specification

3.1 Software Requirements

Following are the software requirement necessary of the project:

a) Python Programming Language: Python is widely used in AI and ML development due to its rich libraries and frameworks.

b)MachineLearning Frameworks: Choose a machine learning framework like TensorFlow, Scikit-learn to build and train the models for phishing website detection.

c)Development Environment: IDEs like Visual Studio Code, PyCharm, or Jupyter Notebook can be used for coding and experimentation.

d)Website Dataset: A labeled dataset containing facial images annotated with corresponding website domains labels is essential for training the model.

3.2 Hardware Requirements

Following are the hardware requirement that is most important for the project:

1.CPU: A modern multi-core CPU is necessary for running the software components, data preprocessing, and training the facial emotion recognition model. The more cores, the faster the processing speed, which can be beneficial for large datasets.

2.GPU (Graphics Processing Unit): While not mandatory, having a dedicated GPU significantly speeds up the training process for deep learning models. GPUs are highly parallel processors and excel at performing matrix operations, which are fundamental to neural network training. A high-end GPU can drastically reduce the training time and enable real-time inference.

3.RAM: Sufficient RAM is crucial, especially when working with large datasets. At least 16 GB of RAM is recommended to handle data preprocessing and deep learning tasks effectively.

4.Storage: Adequate storage space is required to store the datasets, model checkpoints, and trained models. Consider using a fast SSD (Solid State Drive) for improved data access speeds.

CHAPTER-4

High Level Design of Phishing Website Detection System

CHAPTER 4

High Level Design of Phishing Website Detection System

4.1 System Architecture

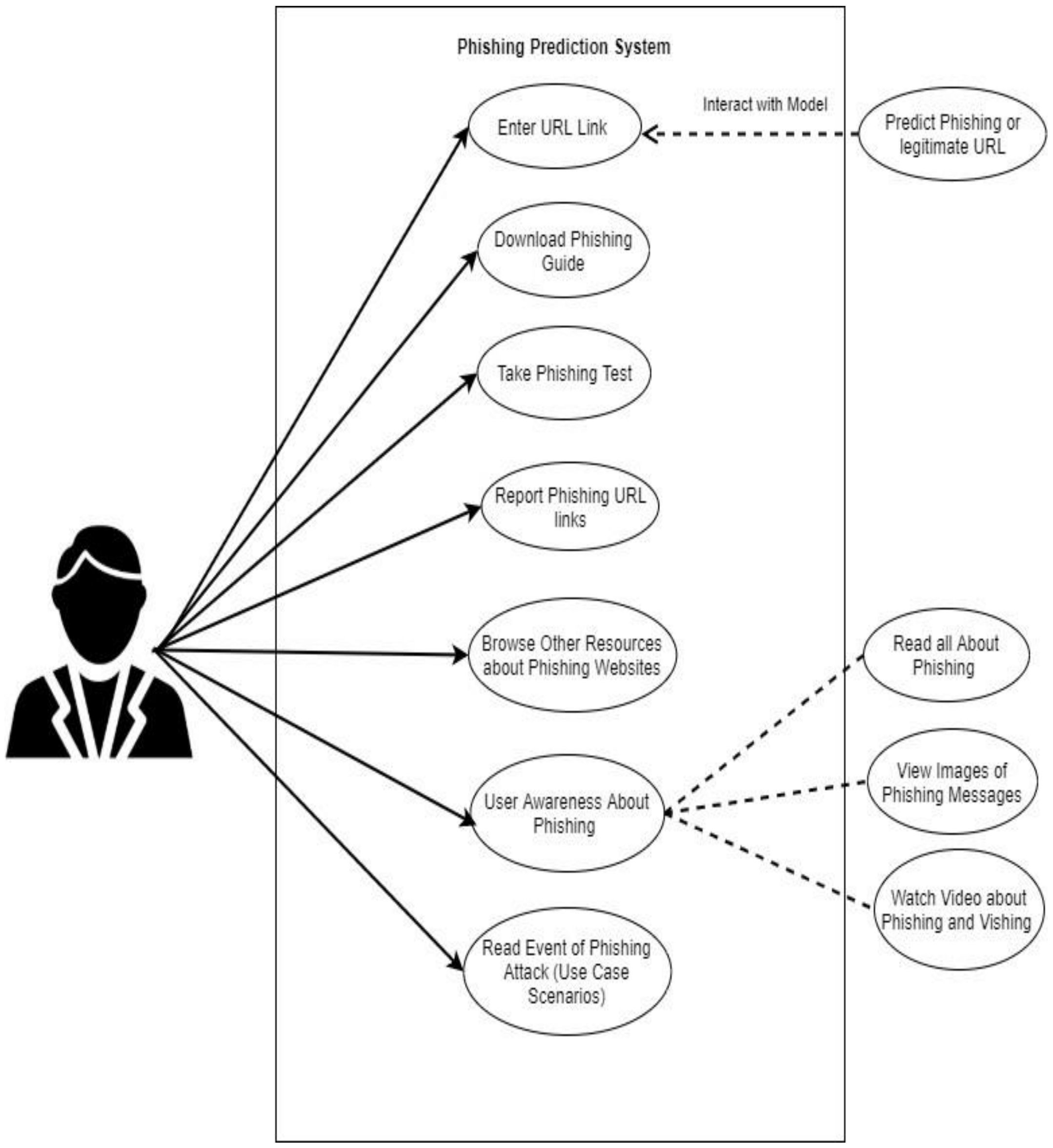

Phishing Prediction System
Enter URL Link
Interact with Model
Predict Phishing or legitimate URL
Download Phishing Guide
Take Phishing Test
Report Phishing URL links
Browse Other Resources about Phishing Websites
Read all About Phishing
View Images of Phishing Messages
User Awareness About Phishing
Watch Video about Phishing and Vishing
Read Event of Phishing Attack (Use Case Scenarios)

Figure 1. Use Case Diagram

4.2 State-machine Diagram

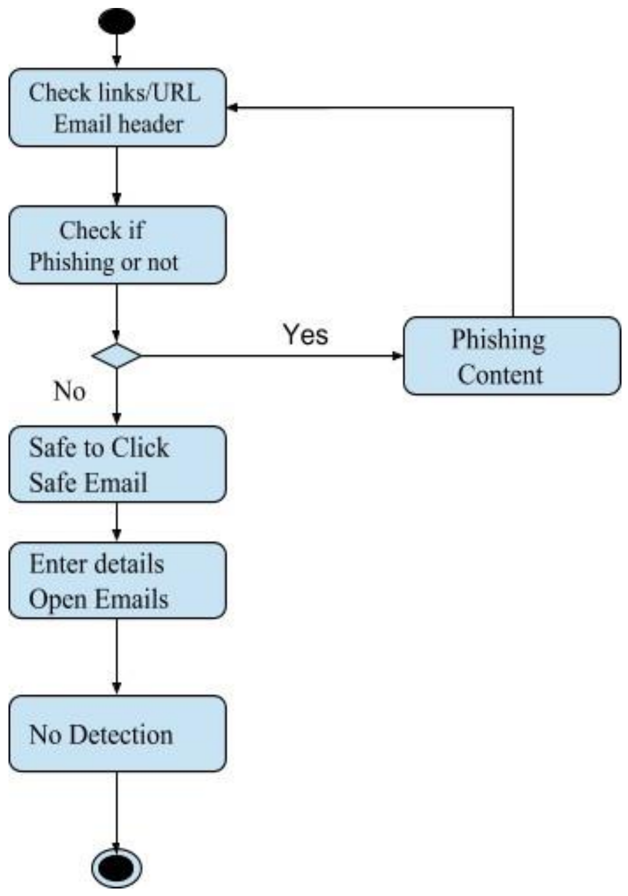


Figure 2. State-Machine Diagram

As we can see the block machine diagram is for phishing process. Although this diagram is old technique, but it can be used to explain the system better here, so we will explain based on project requirements. The first black circle means the start of the process and the end black circle with round ring meaning end of the process.

So first any user will check the URL or link of any website he/she would like to visit, then there is a condition check symbol which decides yes or no. If yes means the website is spoofed so not safe and go back again to start. In the second step is the content is legitimate then you can open an email or enter the sensitive information in web page which is safe, and process ends.

4.3 Activity Diagram

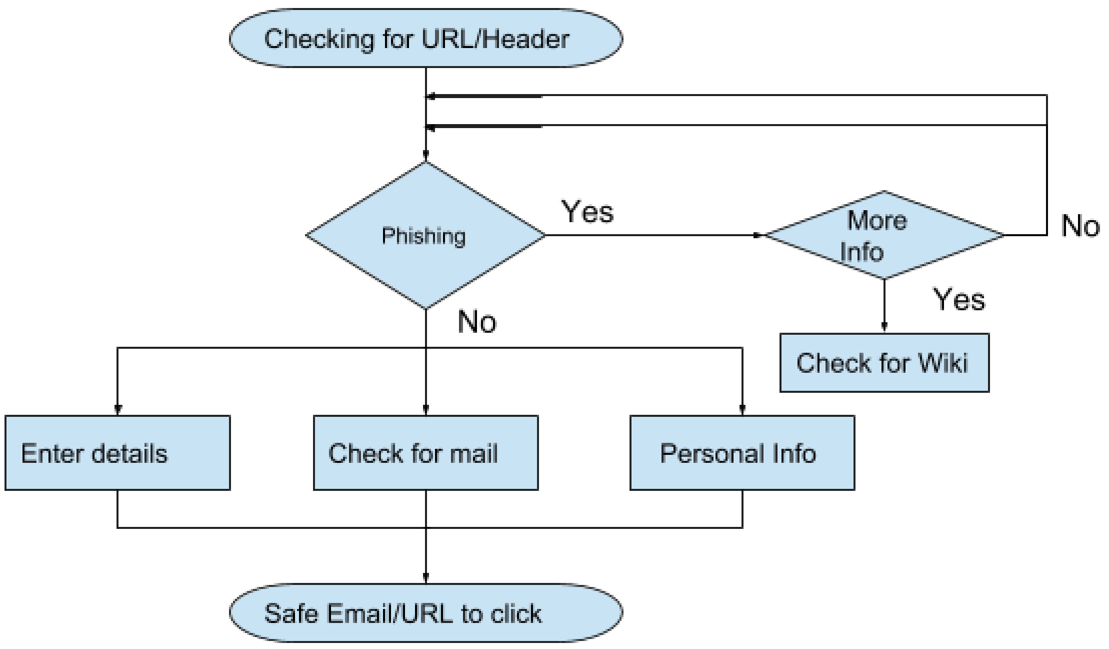


Figure 3. Activity Diagram

An activity diagram is the way to explain the flow of the process with the step by step when order of the process matters. IF we start with the first phase of user side action then user perform the header check for mail or URL check for web page he/she wants to visit before giving any sensitive info. The condition phishing check for the possible ways; if YES means not safe then if you want more info then visit the wiki page for details or else exit. If the condition NO implies the safe surfing on the internet. Enter detail, personal info and check mails are the actions can be taken by user and last stage gives the safe operation output.

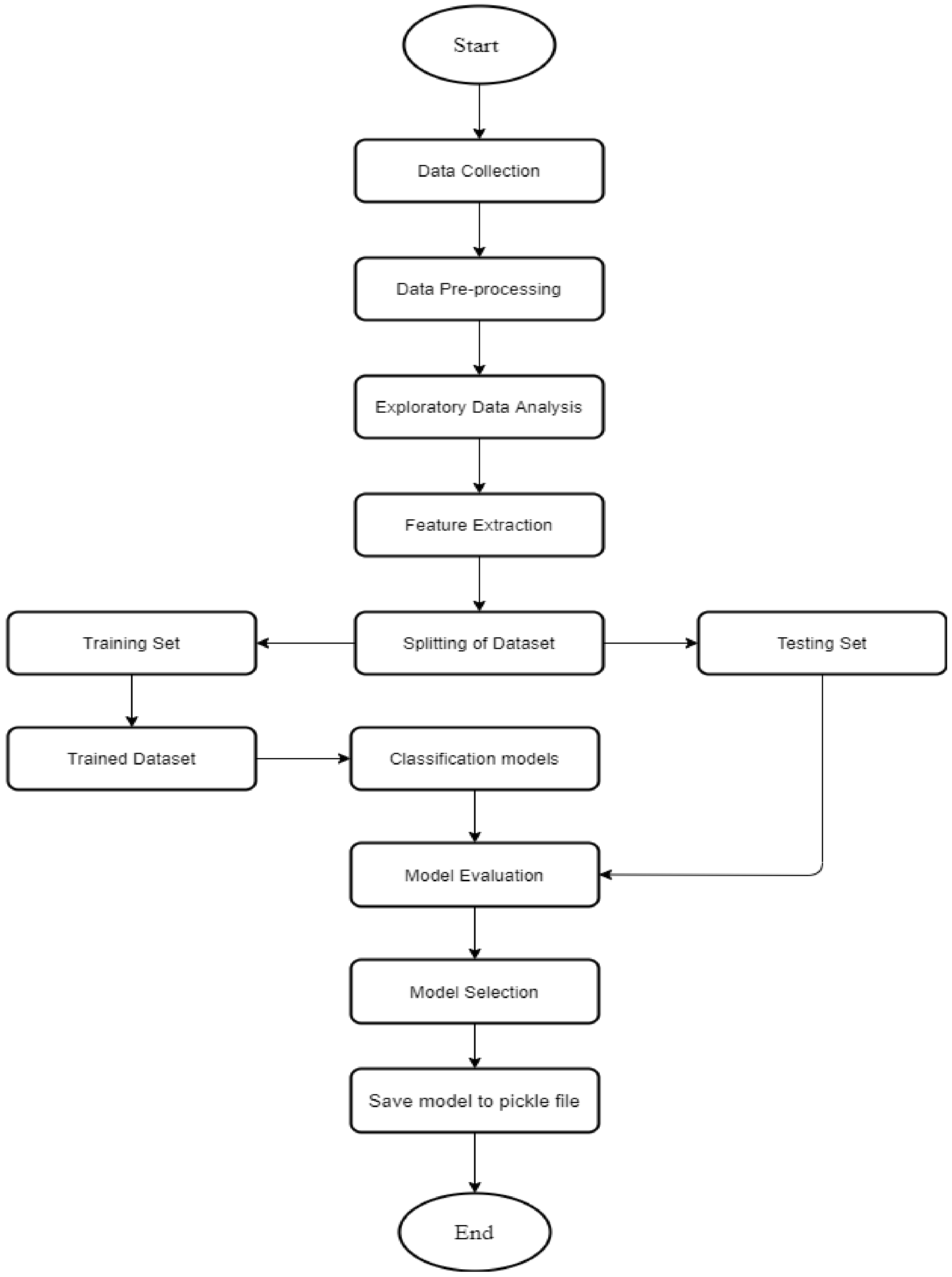
CHAPTER-5

Detailed Design of Phishing Website Detection System

CHAPTER 5

Detailed Design of Phishing Website Detection System

5.1. Structure Chart



5.2. Functional Description of the Modules

Data Collection: Obtain a comprehensive dataset containing labelled examples of phishing and legitimate websites. This dataset will be used to train and evaluate the machine learning model.

The dataset should include various features of websites, such as URL attributes, content, SSL certificate information, and more.

Data Preprocessing: Clean and preprocess the dataset to handle missing values, outliers, and standardize the features. This step is crucial for ensuring data quality and model performance.

Feature Selection: Identify relevant features that are informative for distinguishing between phishing and legitimate websites. Use techniques like correlation analysis and feature importance ranking to select the most significant features.

Model Selection: Choose an appropriate machine learning algorithm for the task. Commonly used algorithms for phishing website detection include Logistic Regression, Random Forest, Gradient Boosting, Support Vector Machines, and Deep Learning models.

Model Training: Split the pre-processed dataset into training and testing sets. Use the training set to train the selected machine learning model on the labelled examples.

Tune the hyperparameters of the model using techniques like cross-validation to optimize its performance.

Model Evaluation: Evaluate the trained model using the testing set to assess its accuracy, precision, recall, F1-score, and ROC-AUC. These metrics will help measure the model's effectiveness in detecting phishing websites.

Addressing Imbalance: If the dataset suffers from class imbalance (i.e., more legitimate websites than phishing websites), use techniques like oversampling, under sampling, or class weighting to balance the dataset and avoid biased results.

Model Optimization: Based on the evaluation results, fine-tune the model to improve its performance and reduce false positives and false negatives.

Real-time Implementation: Deploy the trained and optimized model to work in real-time. Integrate the model into web browsers, network appliances, or security software to provide real-time protection against phishing threats.

Continuous Monitoring and Updates: Continuously monitor the model's performance in real-world scenarios. Regularly update the model with new data to adapt to emerging phishing tactics and maintain its effectiveness.

.

CHAPTER-6

Implementation of Phishing Website Detection System

CHAPTER 6

Implementation of Phishing Website Detection System

6.1. Programming Language Selection

The choice of programming language for phishing website detection depends on various factors, including the available libraries, ease of implementation, performance requirements, and the expertise of the development team.

Python: Python is one of the most popular languages for machine learning and computer vision tasks. It offers a rich ecosystem of libraries such as OpenCV, Dlib, TensorFlow, and Keras, which provide excellent support for facial detection, landmark detection, and building deep learning models for emotion recognition. Python's simplicity and readability make it a preferred choice for rapid prototyping and development.

Selecting Python as the programming language for Facial Emotion Recognition is an excellent choice due to several reasons:

1.Rich Ecosystem of Libraries: Python has a vast ecosystem of libraries and frameworks that are well-suited for machine learning, computer vision, and facial emotion recognition tasks. Libraries like OpenCV, Dlib, TensorFlow, Keras, and PyTorch provide robust functionalities for face detection, facial landmark detection, feature extraction, and deep learning model development.

2.Ease of Implementation: Python is known for its simplicity and readability, making it easy to understand and write code. It allows developers to focus more on the logic and algorithms rather than worrying about complex syntax.

3.Rapid Prototyping: Python's versatility and quick development cycle enable rapid prototyping of facial emotion recognition systems. This is especially beneficial when experimenting with different models, features, and configurations.

4.Integration with Other Technologies: Python can seamlessly integrate with other technologies and platforms, making it suitable for building web-based applications, APIs, and cross-platform solutions.

6.2 Model Development

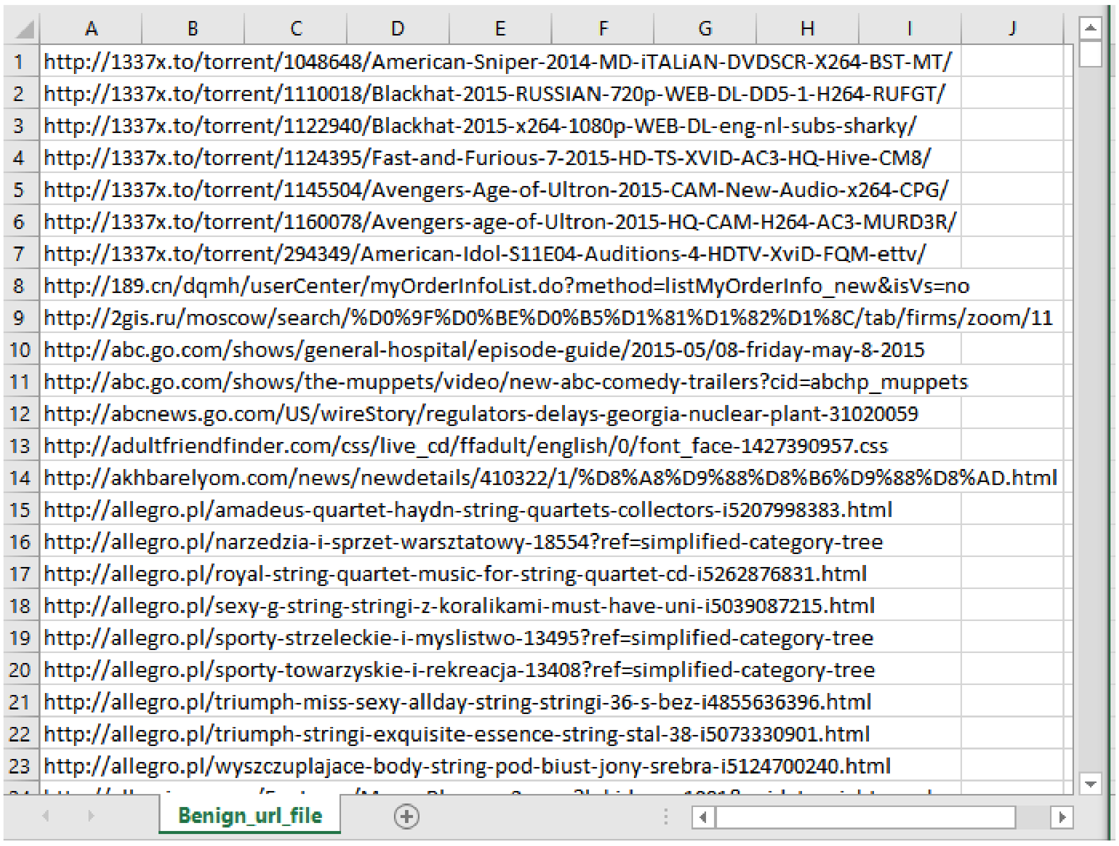
The model for detecting phishing URL websites was built using a python programminglanguage

with over six (6) machine learning models and deep neural network algorithms altogether and

the most accurate test score on the tested 5,000 datasets wereused.

6.3 Data collection

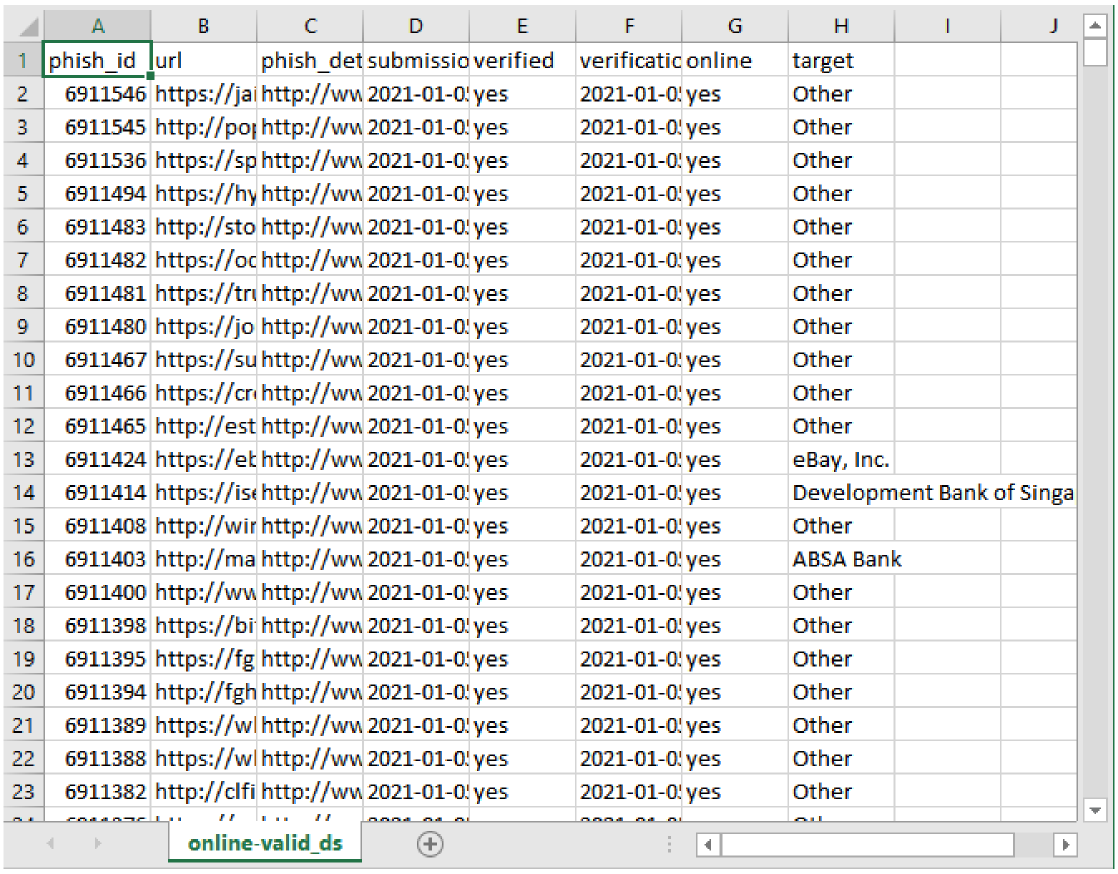
The dataset used for the classification was sourced from was gotten from multiple sources listed in the earlier stated methodology.The dataset used for classifying the dataset into phishing and legitimate URLs was sourced from open source websites, samples of which are shown below in figure 6.1 and 6.2 respectively.



	A
1	http://1337x.to/torrent/1048648/American-Sniper-2014-MD-iTALiAN-DVDSCR-X264-BST-MT/
2	http://1337x.to/torrent/1110018/Blackhat-2015-RUSSIAN-720p-WEB-DL-DD5-1-H264-RUFGT/
3	http://1337x.to/torrent/1122940/Blackhat-2015-x264-1080p-WEB-DL-eng-nl-subs-sharky/
4	http://1337x.to/torrent/1124395/Fast-and-Furious-7-2015-HD-TS-XVID-AC3-HQ-Hive-CM8/
5	http://1337x.to/torrent/1145504/Avengers-Age-of-Ultron-2015-CAM-New-Audio-x264-CPG/
6	http://1337x.to/torrent/1160078/Avengers-age-of-Ultron-2015-HQ-CAM-H264-AC3-MURD3R/
7	http://1337x.to/torrent/294349/American-Idol-S11E04-Auditions-4-HDTV-XviD-FQM-ettv/
8	http://189.cn/dqmh/userCenter/myOrderInfoList.do?method=listMyOrderInfo_new&isVs=no
9	http://2gis.ru/moscow/search/%D0%9F%D0%BE%D0%B5%D1%81%D1%82%D1%8C/tab/firms/zoom/11
10	http://abc.go.com/shows/general-hospital/episode-guide/2015-05/08-friday-may-8-2015
11	http://abc.go.com/shows/the-muppets/video/new-abc-comedy-trailers?cid=abchp_muppets
12	http://abcnews.go.com/US/wireStory/regulators-delays-georgia-nuclear-plant-31020059
13	http://adultfriendfinder.com/css/live_cd/ffadult/english/0/font_face-1427390957.css
14	http://akhbarelyom.com/news/newdetails/410322/1/%D8%A8%D9%88%D8%B6%D9%88%D8%AD.html
15	http://allegro.pl/amadeus-quartet-haydn-string-quartets-collectors-i5207998383.html
16	http://allegro.pl/narzedzia-i-sprzet-warsztatowy-18554?ref=simplified-category-tree
17	http://allegro.pl/royal-string-quartet-music-for-string-quartet-cd-i5262876831.html
18	http://allegro.pl/sexy-g-string-stringi-z-koralikami-must-have-uni-i5039087215.html
19	http://allegro.pl/sporty-strzeleckie-i-myslistwo-13495?ref=simplified-category-tree
20	http://allegro.pl/sporty-towarzyskie-i-rekreacja-13408?ref=simplified-category-tree
21	http://allegro.pl/triumph-miss-sexy-allday-string-stringi-36-s-bez-i4855636396.html
22	http://allegro.pl/triumph-stringi-exquisite-essence-string-stal-38-i5073330901.html
23	http://allegro.pl/wyszczuplajace-body-string-pod-biust-jony-srebra-i5124700240.html

Figure 6.1 Dataset of Phishing URLs

Source: The Dataset is collected from an open-source service called Phish-Tank. This datasetconsists of 5,000 random phishing URLs which are collected to train the ML models.



	A	B	C	D	E	F	G	H	I	J
1	phish_id	url	phish_det	submissio	verified	verificatic	online	target		
2	6911546	https://jai	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
3	6911545	http://po	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
4	6911536	https://sp	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
5	6911494	https://hy	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
6	6911483	http://sto	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
7	6911482	https://oc	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
8	6911481	https://tr	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
9	6911480	https://jo	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
10	6911467	https://su	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
11	6911466	https://cr	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
12	6911465	http://est	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
13	6911424	https://eb	http://ww	2021-01-0	yes	2021-01-0	yes	eBay, Inc.		
14	6911414	https://ise	http://ww	2021-01-0	yes	2021-01-0	yes	Development Bank of Singa		
15	6911408	http://wir	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
16	6911403	http://ma	http://ww	2021-01-0	yes	2021-01-0	yes	ABSA Bank		
17	6911400	http://ww	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
18	6911398	https://bi	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
19	6911395	https://fg	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
20	6911394	http://fgh	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
21	6911389	https://wl	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
22	6911388	https://wl	http://ww	2021-01-0	yes	2021-01-0	yes	Other		
23	6911382	http://clfi	http://ww	2021-01-0	yes	2021-01-0	yes	Other		

Figure 6.2 Dataset of Legitimate URLs

Source: The Dataset were obtained from the open datasets of the University of New Brunswick, The dataset consists of collections of benign, spam, phishing, malware & defacement URLs. Out of all these types, the benign URL dataset is considered for thisproject. This dataset consists of 5,000 random legitimate URLs which are collected to train the ML models.

6.4 Feature extraction on the datasets

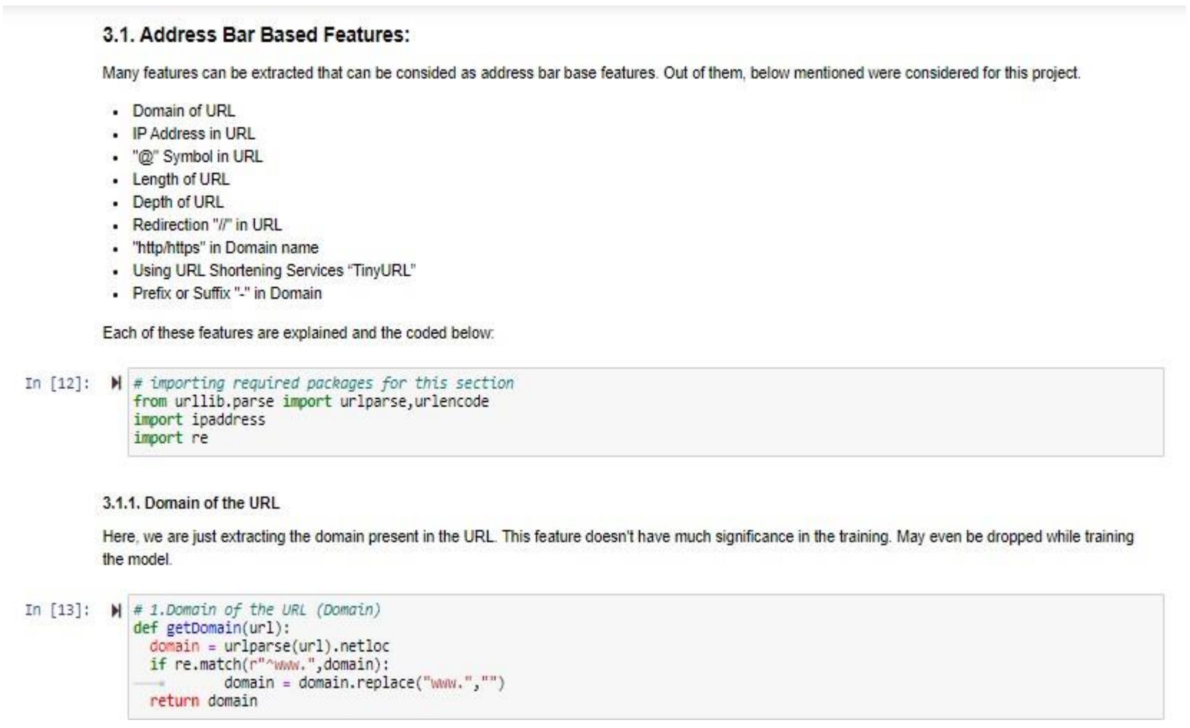
The features extraction used on the dataset are categorized into

i. Address bar based features

ii. Domain-based features

iii. Html & java-script based features

In figure 4.3, figure 4.4, and figure 4.5 the images show the list of code feature extraction done on the dataset while figure 4.6 shows the code computation for all the feature extraction used on the dataset.



3.1. Address Bar Based Features:

Many features can be extracted that can be consided as address bar base features. Out of them, below mentioned were considered for this project.

- Domain of URL
- IP Address in URL
- "@" Symbol in URL
- Length of URL
- Depth of URL
- Redirection "//" in URL
- "http/https" in Domain name
- Using URL Shortening Services "TinyURL"
- Prefix or Suffix "-" in Domain

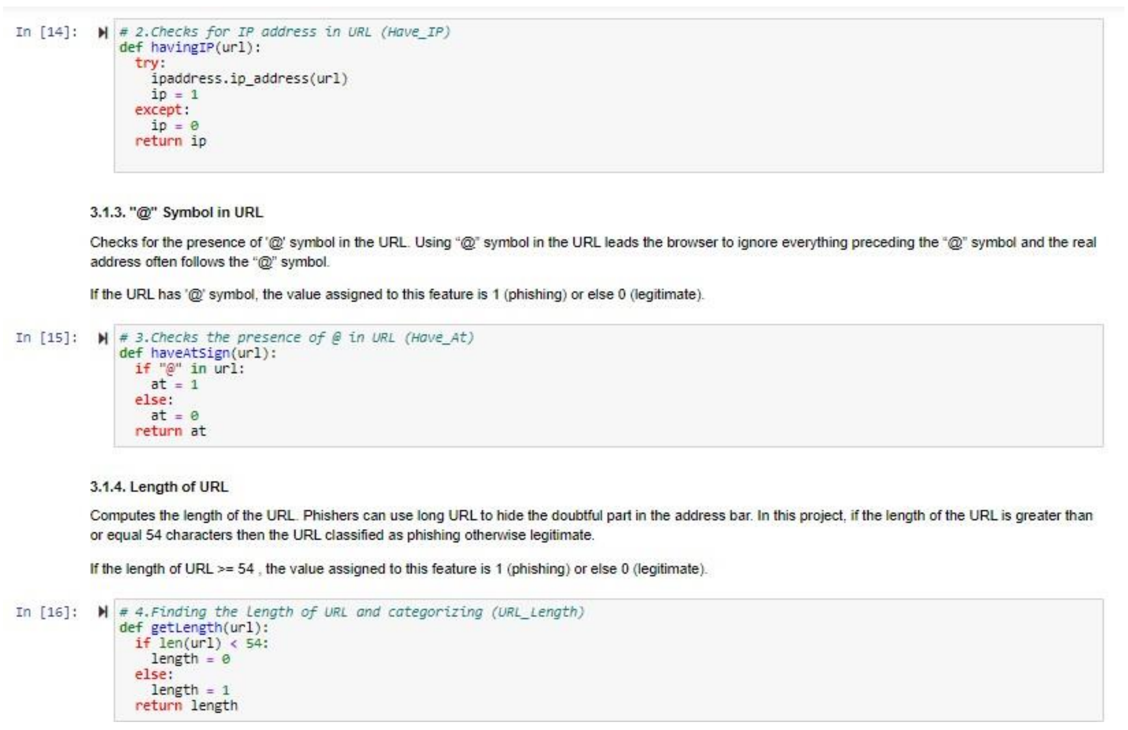
Each of these features are explained and the coded below:

In [12]:
```
# importing required packages for this section
from urllib.parse import urlparse,urlencode
import ipaddress
import re
```

3.1.1. Domain of the URL

Here, we are just extracting the domain present in the URL. This feature doesn't have much significance in the training. May even be dropped while training the model.

In [13]:
```
# 1.Domain of the URL (Domain)
def getDomain(url):
  domain = urlparse(url).netloc
  if re.match(r"^www.",domain):
          domain = domain.replace("www.","")
  return domain
```



In [14]:
```
# 2.Checks for IP address in URL (Have_IP)
def havingIP(url):
  try:
    ipaddress.ip_address(url)
    ip = 1
  except:
    ip = 0
  return ip
```

3.1.3. "@" Symbol in URL

Checks for the presence of '@' symbol in the URL. Using "@" symbol in the URL leads the browser to ignore everything preceding the "@" symbol and the real address often follows the "@" symbol.

If the URL has '@' symbol, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

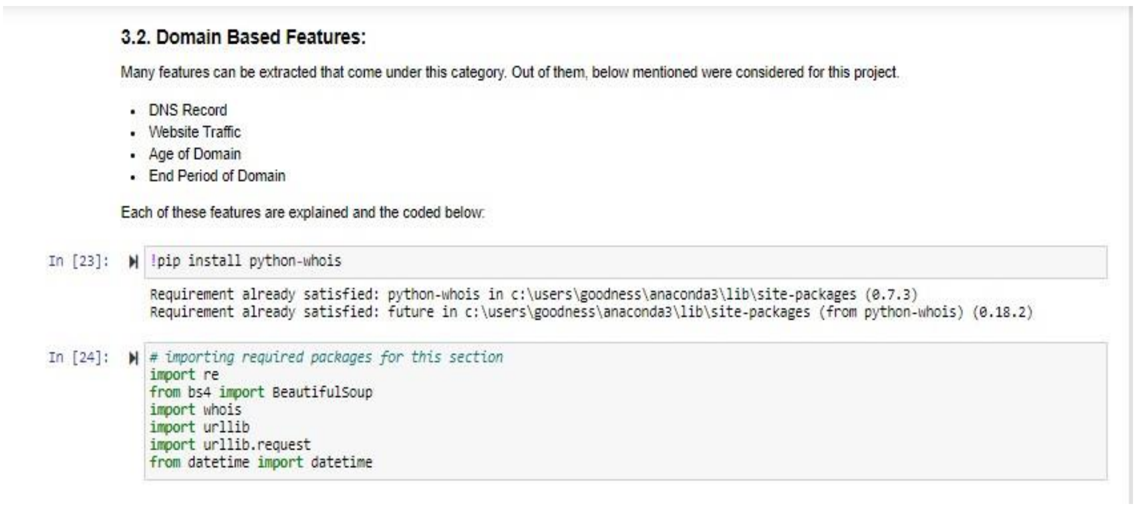
In [15]:
```
# 3.Checks the presence of @ in URL (Have_At)
def haveAtSign(url):
  if "@" in url:
    at = 1
  else:
    at = 0
  return at
```

3.1.4. Length of URL

Computes the length of the URL. Phishers can use long URL to hide the doubtful part in the address bar. In this project, if the length of the URL is greater than or equal 54 characters then the URL classified as phishing otherwise legitimate.

If the length of URL >= 54 , the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

In [16]:
```
# 4.Finding the length of URL and categorizing (URL_Length)
def getLength(url):
  if len(url) < 54:
    length = 0
  else:
    length = 1
  return length
```



3.2. Domain Based Features:

Many features can be extracted that come under this category. Out of them, below mentioned were considered for this project.

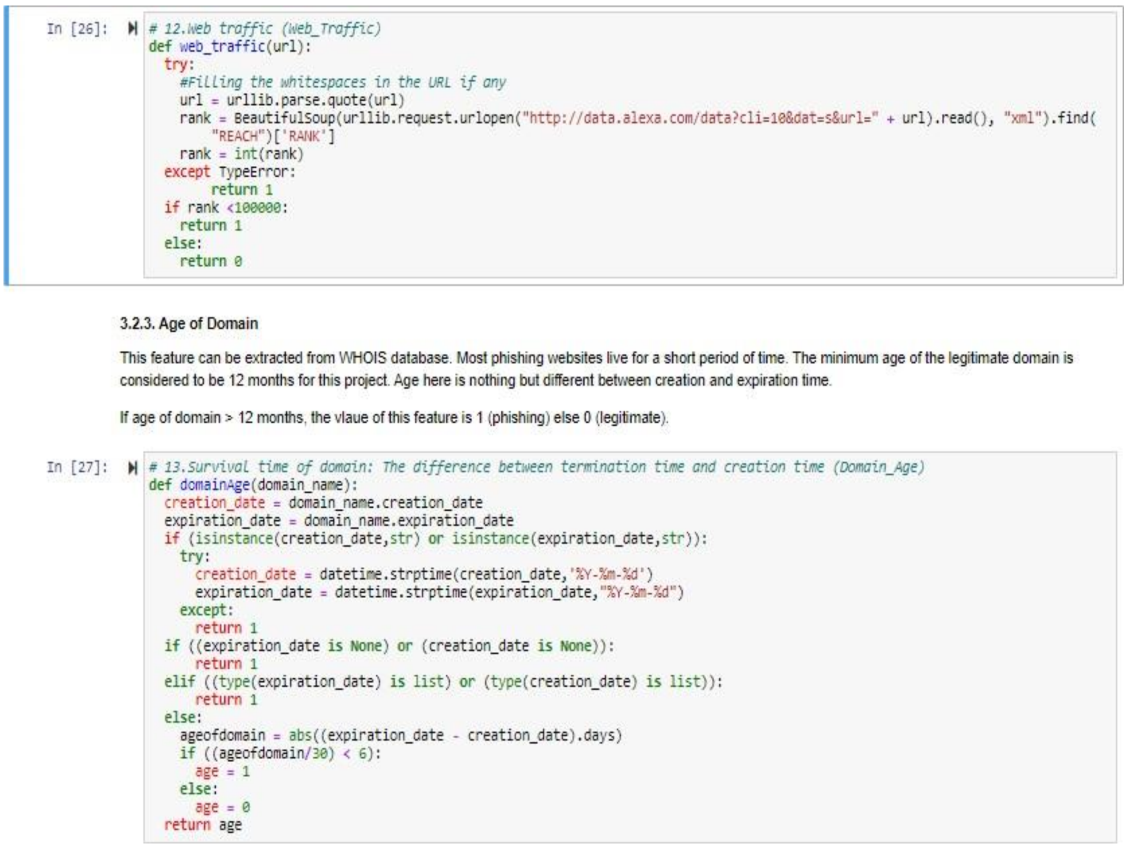
- DNS Record
- Website Traffic
- Age of Domain
- End Period of Domain

Each of these features are explained and the coded below.

```
In [23]:  !pip install python-whois

          Requirement already satisfied: python-whois in c:\users\goodness\anaconda3\lib\site-packages (0.7.3)
          Requirement already satisfied: future in c:\users\goodness\anaconda3\lib\site-packages (from python-whois) (0.18.2)
```

```
In [24]:  # importing required packages for this section
          import re
          from bs4 import BeautifulSoup
          import whois
          import urllib
          import urllib.request
          from datetime import datetime
```



```
In [26]:  # 12.Web traffic (Web_Traffic)
          def web_traffic(url):
            try:
              #Filling the whitespaces in the URL if any
              url = urllib.parse.quote(url)
              rank = BeautifulSoup(urllib.request.urlopen("http://data.alexa.com/data?cli=10&dat=s&url=" + url).read(), "xml").find(
                  "REACH")['RANK']
              rank = int(rank)
            except TypeError:
                  return 1
            if rank <100000:
              return 1
            else:
              return 0
```

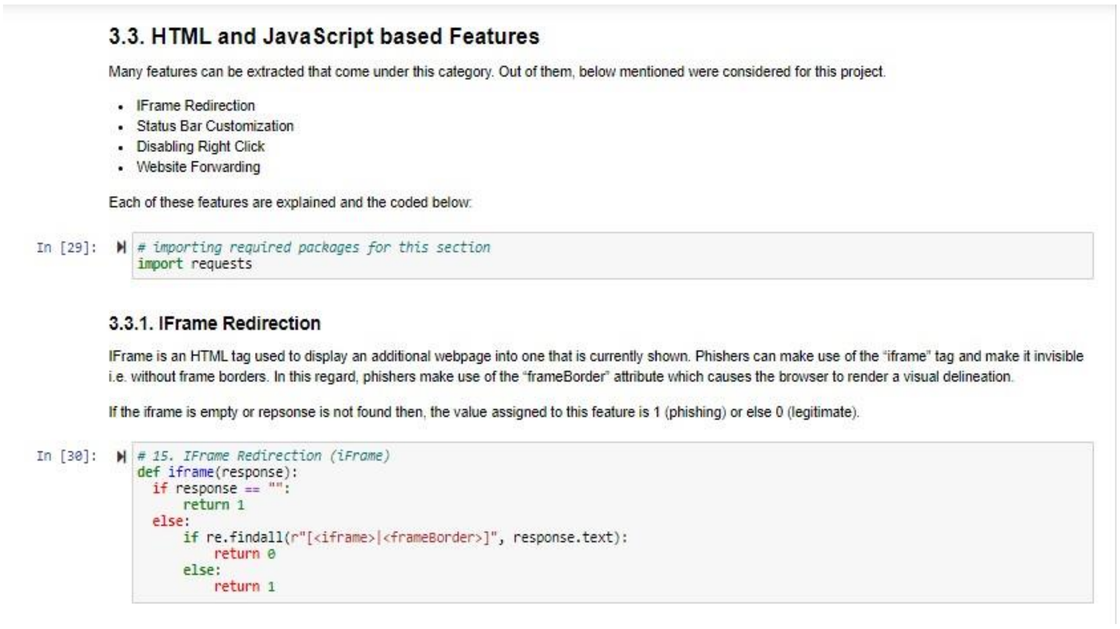
3.2.3. Age of Domain

This feature can be extracted from WHOIS database. Most phishing websites live for a short period of time. The minimum age of the legitimate domain is considered to be 12 months for this project. Age here is nothing but different between creation and expiration time.

If age of domain > 12 months, the vlaue of this feature is 1 (phishing) else 0 (legitimate).

```
In [27]:  # 13.Survival time of domain: The difference between termination time and creation time (Domain_Age)
          def domainAge(domain_name):
            creation_date = domain_name.creation_date
            expiration_date = domain_name.expiration_date
            if (isinstance(creation_date,str) or isinstance(expiration_date,str)):
              try:
                creation_date = datetime.strptime(creation_date,'%Y-%m-%d')
                expiration_date = datetime.strptime(expiration_date,"%Y-%m-%d")
              except:
                return 1
            if ((expiration_date is None) or (creation_date is None)):
                return 1
            elif ((type(expiration_date) is list) or (type(creation_date) is list)):
                return 1
            else:
              ageofdomain = abs((expiration_date - creation_date).days)
              if ((ageofdomain/30) < 6):
                age = 1
              else:
                age = 0
            return age
```

Figure 6.4: Code for domain-based features extraction



3.3. HTML and JavaScript based Features

Many features can be extracted that come under this category. Out of them, below mentioned were considered for this project.

- IFrame Redirection
- Status Bar Customization
- Disabling Right Click
- Website Forwarding

Each of these features are explained and the coded below:

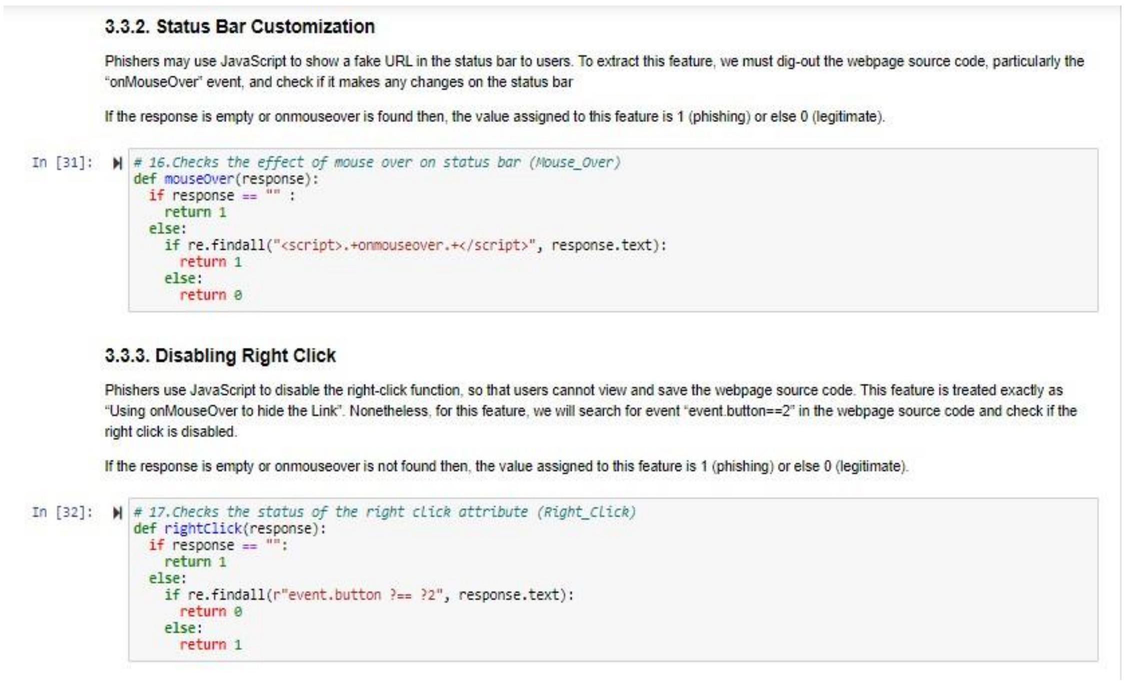
```
In [29]:  # importing required packages for this section
          import requests
```

3.3.1. IFrame Redirection

IFrame is an HTML tag used to display an additional webpage into one that is currently shown. Phishers can make use of the "iframe" tag and make it invisible i.e. without frame borders. In this regard, phishers make use of the "frameBorder" attribute which causes the browser to render a visual delineation.

If the iframe is empty or repsonse is not found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

```
In [30]:  # 15. IFrame Redirection (iFrame)
          def iframe(response):
            if response == "":
                return 1
            else:
                if re.findall(r"[<iframe>|<frameBorder>]", response.text):
                    return 0
                else:
                    return 1
```



3.3.2. Status Bar Customization

Phishers may use JavaScript to show a fake URL in the status bar to users. To extract this feature, we must dig-out the webpage source code, particularly the "onMouseOver" event, and check if it makes any changes on the status bar

If the response is empty or onmouseover is found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

```
In [31]:  # 16.Checks the effect of mouse over on status bar (Mouse_Over)
          def mouseOver(response):
            if response == "" :
              return 1
            else:
              if re.findall("<script>.+onmouseover.+</script>", response.text):
                return 1
              else:
                return 0
```

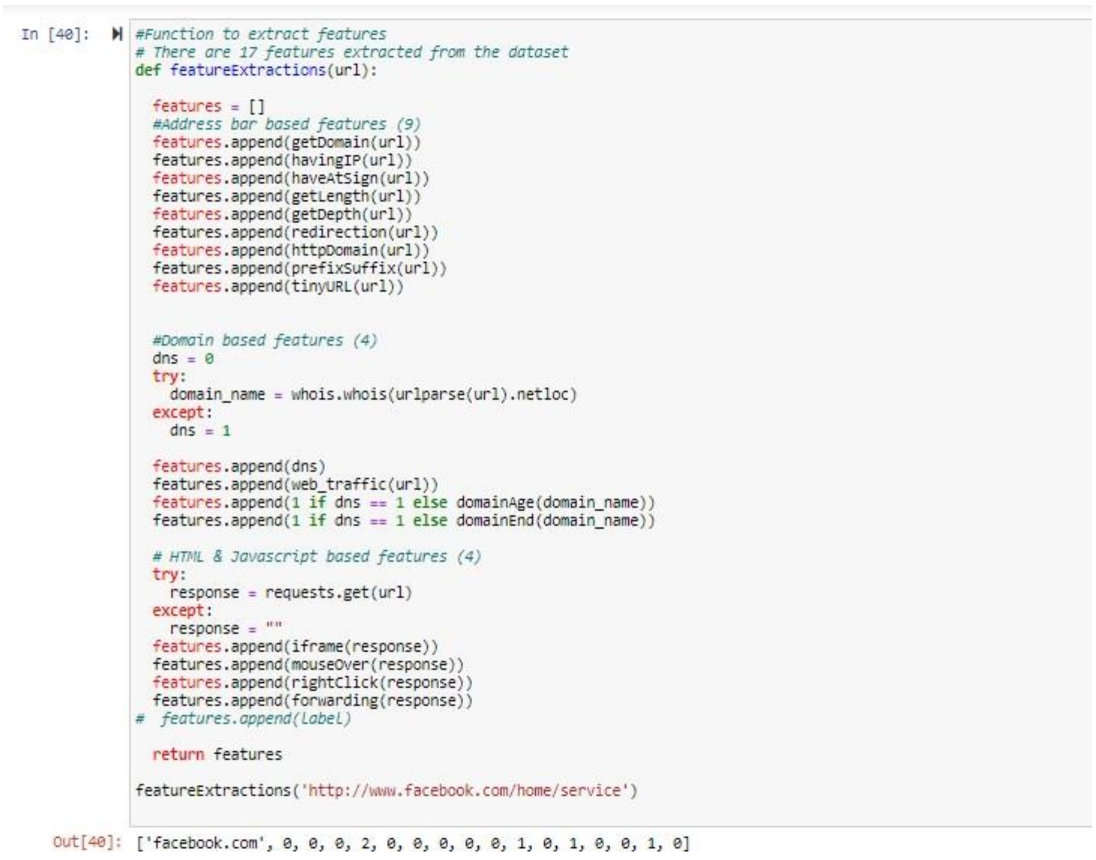
3.3.3. Disabling Right Click

Phishers use JavaScript to disable the right-click function, so that users cannot view and save the webpage source code. This feature is treated exactly as "Using onMouseOver to hide the Link". Nonetheless, for this feature, we will search for event "event.button==2" in the webpage source code and check if the right click is disabled.

If the response is empty or onmouseover is not found then, the value assigned to this feature is 1 (phishing) or else 0 (legitimate).

```
In [32]:  # 17.Checks the status of the right click attribute (Right_Click)
          def rightClick(response):
            if response == "":
              return 1
            else:
              if re.findall(r"event.button ?== ?2", response.text):
                return 0
              else:
                return 1
```

Figure 6.5: Code for Html & java-script based features extraction



```
In [40]:  #Function to extract features
          # There are 17 features extracted from the dataset
          def featureExtractions(url):

            features = []
            #Address bar based features (9)
            features.append(getDomain(url))
            features.append(havingIP(url))
            features.append(haveAtSign(url))
            features.append(getLength(url))
            features.append(getDepth(url))
            features.append(redirection(url))
            features.append(httpDomain(url))
            features.append(prefixSuffix(url))
            features.append(tinyURL(url))


            #Domain based features (4)
            dns = 0
            try:
              domain_name = whois.whois(urlparse(url).netloc)
            except:
              dns = 1

            features.append(dns)
            features.append(web_traffic(url))
            features.append(1 if dns == 1 else domainAge(domain_name))
            features.append(1 if dns == 1 else domainEnd(domain_name))

            # HTML & Javascript based features (4)
            try:
              response = requests.get(url)
            except:
              response = ""
            features.append(iframe(response))
            features.append(mouseOver(response))
            features.append(rightClick(response))
            features.append(forwarding(response))
          #  features.append(label)

            return features

          featureExtractions('http://www.facebook.com/home/service')

Out[40]:  ['facebook.com', 0, 0, 0, 2, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0]
```

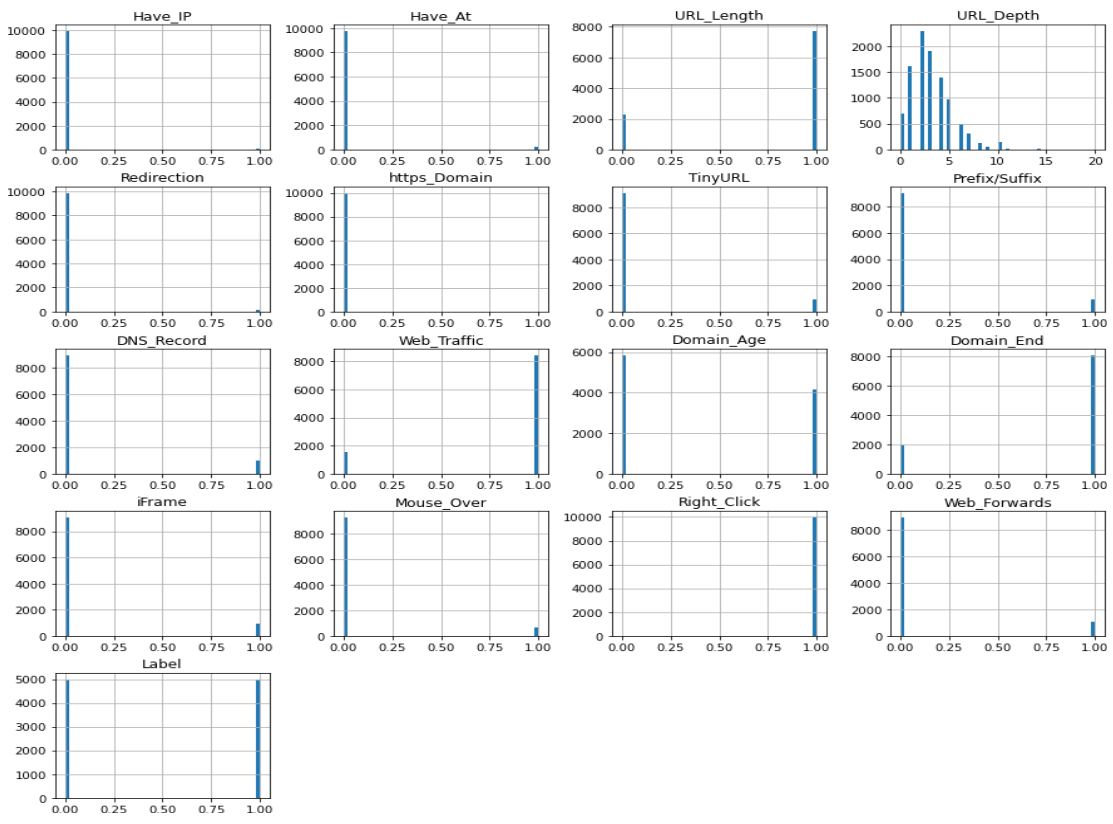
Figure 6.6: Code computation for all the feature extraction used dataset.

6.5 Data Analysis & Visualization

The image as shown in figure 4.7 shows the distribution plot of how legitimate and phishing datasets are distributed base on the features selected and how they are relatedto each other.

In figure 4.8 shows the plot of a correlation heat-map of the dataset. The plot shows correlation between different variables in the dataset.

In figure 4.9 and figure 4.10, it shows the feature importance in the model for Decisiontree classifier and Random forest classifier respectively.



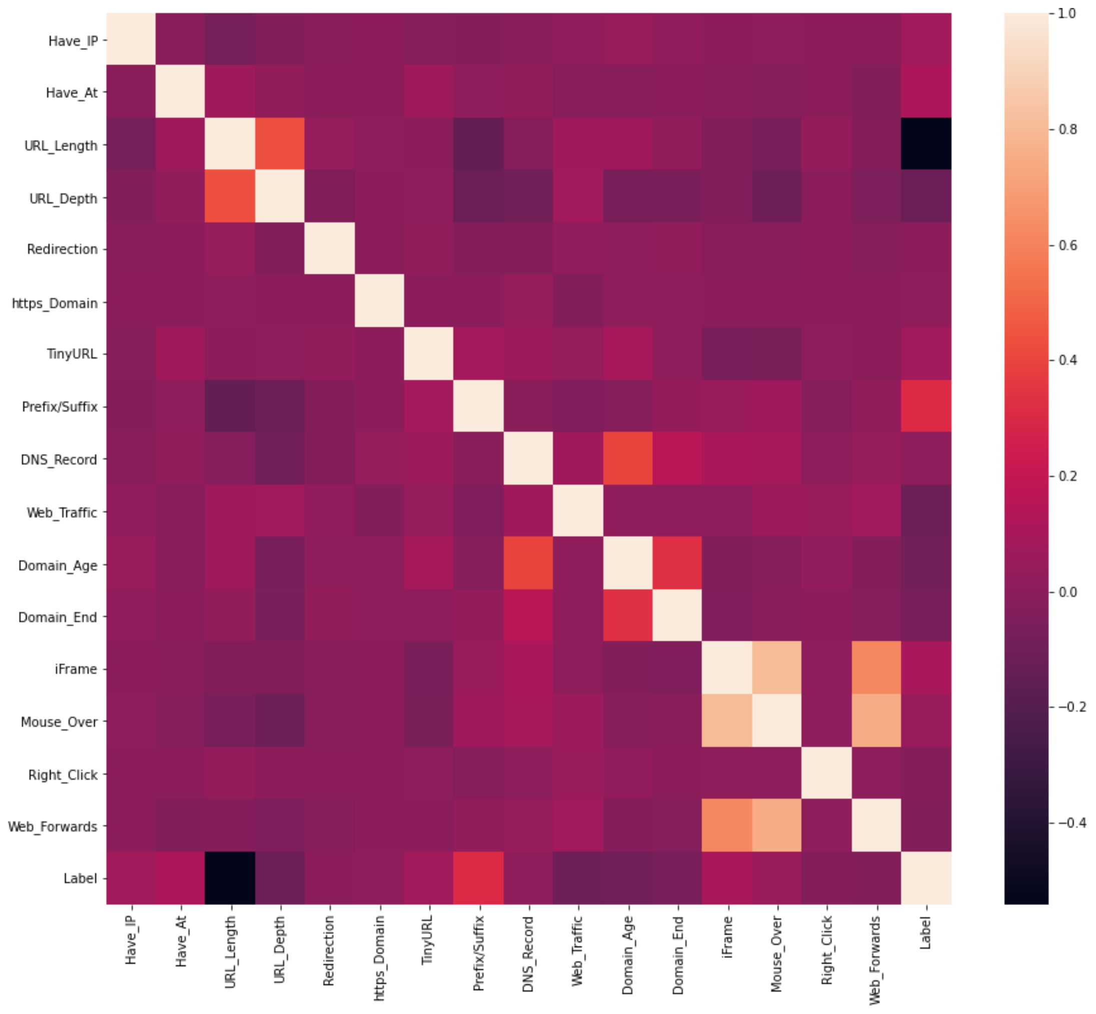


Figure 6.8: Correlation heat map of the dataset

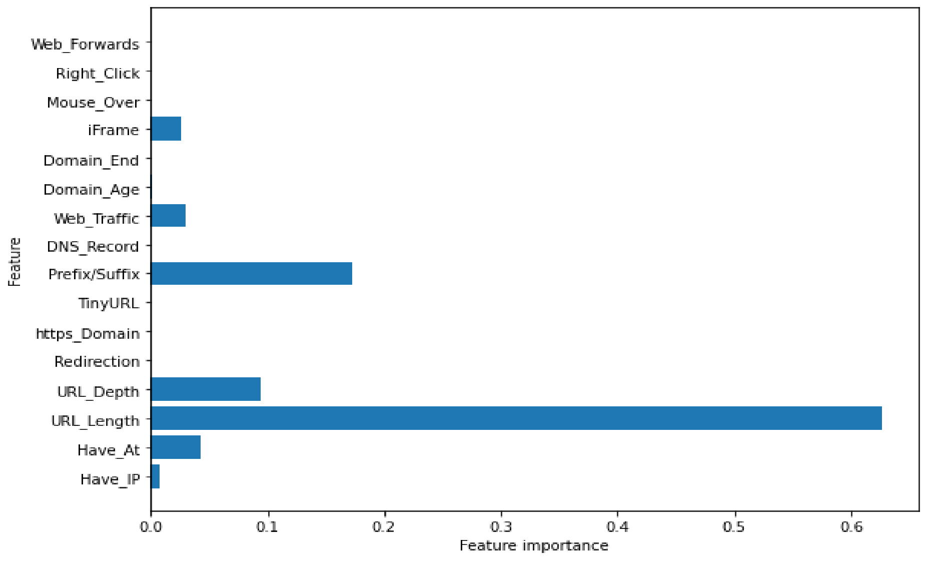


Figure 6.9: Feature importance for Decision Tree classifier

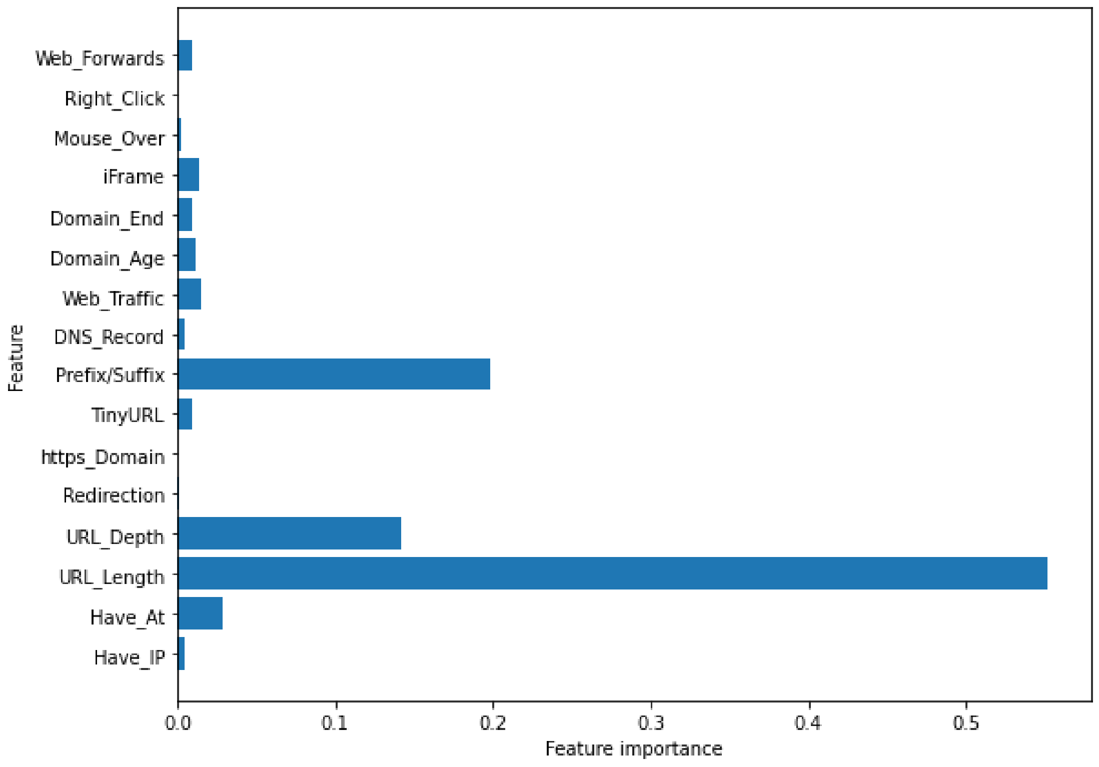


Figure 6.10: Feature importance for Random forest classifier

CHAPTER-7

Software Testing of Phishing Website Detection System

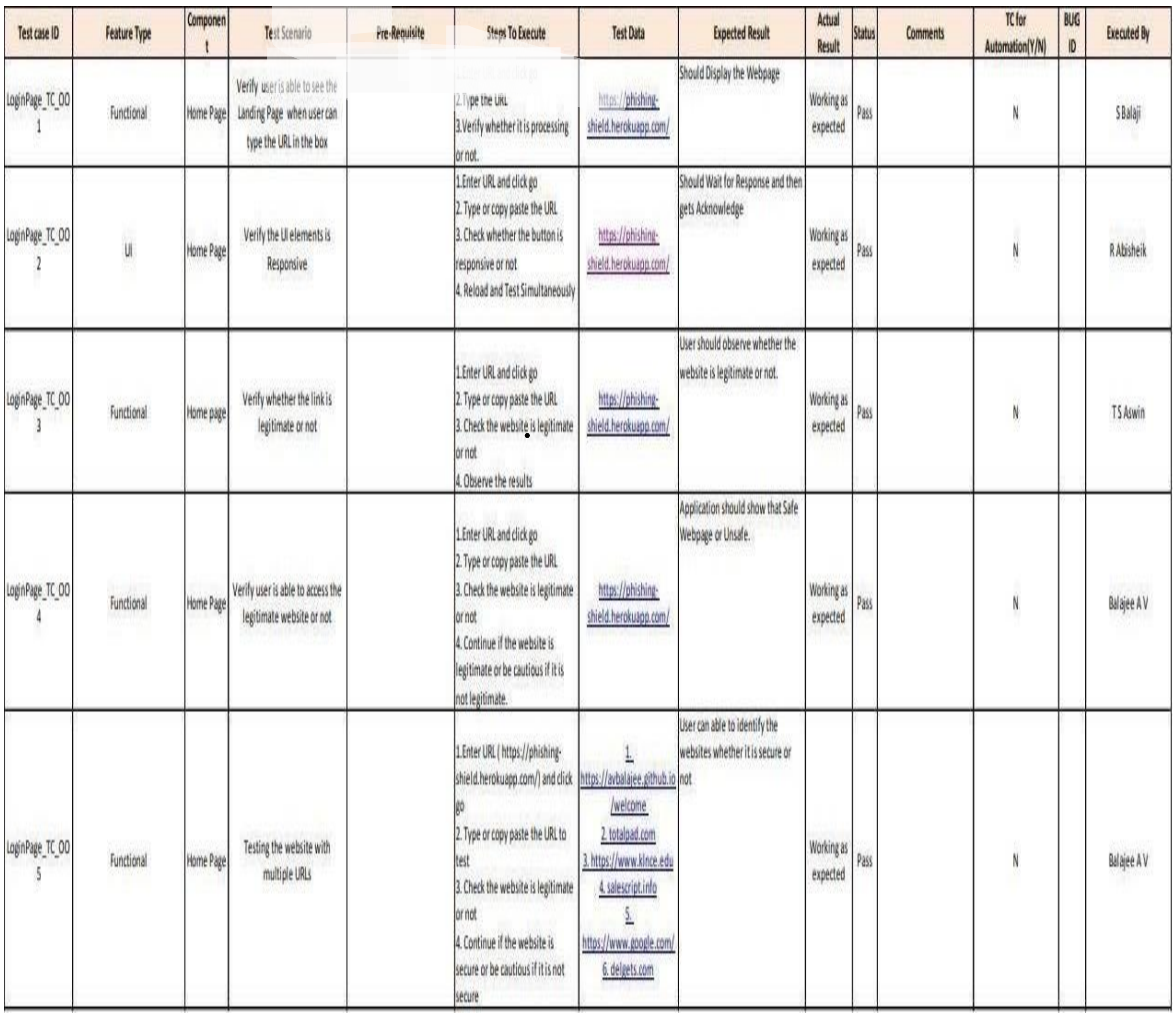
CHAPTER 7

Software Testing of Phishing Website Detection System

7.1. Unit Testing

Unit testing is a software testing technique that involves testing individual units or components of a software application in isolation to ensure that they function correctly and meet their specified requirements. A unit refers to the smallest testable part of an application, such as a function, method, or class. The main purpose of unit testing is to validate that each unit of code behaves as intended and to identify any bugs or defects early in the development process.

7.1.1. Unit Testing of Module1



Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
LoginPage_TC_OO 1	Functional	Home Page	Verify user is able to see the Landing Page when user can type the URL in the box		1.Enter URL and click go 2.Type the URL 3.Verify whether it is processing or not.	https://phishing-shield.herokuapp.com/	Should Display the Webpage	Working as expected	Pass		N		S Balaji
LoginPage_TC_OO 2	UI	Home Page	Verify the UI elements is Responsive		1.Enter URL and click go 2. Type or copy paste the URL 3. Check whether the button is responsive or not 4. Reload and Test Simultaneously	https://phishing-shield.herokuapp.com/	Should Wait for Response and then gets Acknowledge	Working as expected	Pass		N		R Abisheik
LoginPage_TC_OO 3	Functional	Home page	Verify whether the link is legitimate or not		1.Enter URL and click go 2. Type or copy paste the URL 3. Check the website is legitimate or not 4. Observe the results	https://phishing-shield.herokuapp.com/	User should observe whether the website is legitimate or not.	Working as expected	Pass		N		T S Aswin
LoginPage_TC_OO 4	Functional	Home Page	Verify user is able to access the legitimate website or not		1.Enter URL and click go 2. Type or copy paste the URL 3. Check the website is legitimate or not 4. Continue if the website is legitimate or be cautious if it is not legitimate.	https://phishing-shield.herokuapp.com/	Application should show that Safe Webpage or Unsafe.	Working as expected	Pass		N		Balajee A V
LoginPage_TC_OO 5	Functional	Home Page	Testing the website with multiple URLs		1.Enter URL (https://phishing-shield.herokuapp.com/) and click go 2. Type or copy paste the URL to test 3. Check the website is legitimate or not 4. Continue if the website is secure or be cautious if it is not secure	1. https://avbalajee.github.io/welcome 2. totalpad.com 3. https://www.kince.edu 4. salescript.info 5. https://www.google.com/ 6. delgets.com	User can able to identify the websites whether it is secure or not	Working as expected	Pass		N		Balajee A V

7.2. Acceptance testing

Step 5: Verify Results:After the tests are executed, verify that the Feature Extraction module correctly extracts features from the test facial images based on the expected outcomes of each test case.

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	10	2	4	20	36
Not Reproduced	0	0	1	0	1
Skipped	0	0	0	0	0
Won't Fix	0	0	2	1	3
Totals	23	9	12	25	60

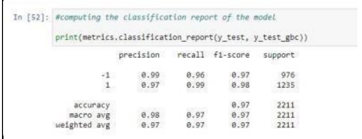
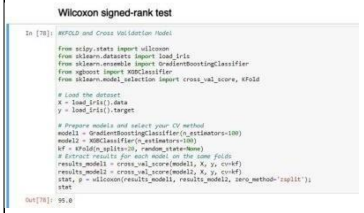
CHAPTER-8

Experimental Results and Analysis of Phishing Website Detection System

CHAPTER 8

Experimental Results and Analysis of Phishing Website Detection System

8.1 Model Performance :

S.no.	Parameter	Values	Screenshot
1.	Metrics	**Classification Model:** **Decision tree** Accuray Score- 97.4%	
2.	Tune the Model	Hyperparameter Tuning - 97% Validation Method – KFOLD & Cross Validation Method	

```
In [52]: #computing the classification report of the model

         print(metrics.classification_report(y_test, y_test_gbc))
                      precision    recall  f1-score   support

                  -1       0.99      0.96      0.97       976
                   1       0.97      0.99      0.98      1235

            accuracy                           0.97      2211
           macro avg       0.98      0.97      0.97      2211
        weighted avg       0.97      0.97      0.97      2211
```

Wilcoxon signed-rank test

```
In [78]: #KFOLD and Cross Validation Model

         from scipy.stats import wilcoxon
         from sklearn.datasets import load_iris
         from sklearn.ensemble import GradientBoostingClassifier
         from xgboost import XGBClassifier
         from sklearn.model_selection import cross_val_score, KFold

         # Load the dataset
         X = load_iris().data
         y = load_iris().target

         # Prepare models and select your CV method
         model1 = GradientBoostingClassifier(n_estimators=100)
         model2 = XGBClassifier(n_estimators=100)
         kf = KFold(n_splits=20, random_state=None)
         # Extract results for each model on the same folds
         results_model1 = cross_val_score(model1, X, y, cv=kf)
         results_model2 = cross_val_score(model2, X, y, cv=kf)
         stat, p = wilcoxon(results_model1, results_model2, zero_method='zsplit');
         stat

Out[78]: 95.0
```

8.2. Evaluation Metrics

Evaluation metrics are essential for assessing the performance and effectiveness of a Facial Emotion Recognition system. These metrics help quantify how well the system is identifying and predicting emotions from facial expressions. Depending on the specific requirements and objectives of the system, various evaluation metrics can be used. Here are some commonly used evaluation metrics for Facial Emotion Recognition:

Accuracy: Accuracy is a fundamental metric that measures the proportion of correctly predicted emotions out of the total number of samples. It provides an overall assessment of the system's correctness.

Confusion Matrix: A confusion matrix displays the number of true positive, true negative, false positive, and false negative predictions for each emotion class. It allows a more detailed understanding of the system's performance for individual emotions.

Precision: Precision measures the proportion of correctly predicted positive emotions (true positives) out of all predicted positive emotions (true positives + false positives). It indicates how well the system avoids false positive predictions.

Recall (Sensitivity): Recall measures the proportion of correctly predicted positive emotions (true positives) out of all actual positive emotions (true positives + false negatives). It indicates how well the system captures positive emotions.

F1 Score: The F1 score is the harmonic mean of precision and recall. It provides a balanced measure between precision and recall and is useful when there is an imbalance in the emotion classes.

Mean Absolute Error (MAE): MAE measures the average absolute difference between the predicted emotion values and the true emotion labels. It is useful for regression-based emotion recognition systems.

Mean Squared Error (MSE): MSE measures the average squared difference between the predicted emotion values and the true emotion labels. It is also commonly used for regression-based systems.

Cohen's Kappa: Cohen's Kappa measures the agreement between the predicted emotions and the true emotions, considering the agreement that could occur by chance.

Receiver Operating Characteristic (ROC) Curve: The ROC curve is used for binary emotion recognition problems. It plots the true positive rate against the false positive rate at various classification thresholds.

Area Under the Curve (AUC): The AUC is the area under the ROC curve and provides a single-value metric to assess the overall performance of a binary classifier.

The choice of evaluation metrics depends on the problem formulation (binary or multi-class emotion recognition), the dataset, and the specific goals of the Facial Emotion Recognition system. It is essential to consider multiple metrics to gain a comprehensive understanding of the system's performance and identify areas for improvement.

8.3. Performance Analysis

Performance analysis in Facial Emotion Recognition involves evaluating and analyzing the system's overall performance, accuracy, efficiency, and robustness. It helps identify strengths, weaknesses, and areas for improvement in the system. Here are key aspects to consider during performance analysis:

Accuracy and Emotion Recognition Performance:

Calculate the overall accuracy of the system in correctly recognizing emotions. Analyze the confusion matrix to understand the performance for each emotion class.

Identify emotions that the system performs well on and those where it struggles. Investigate the reasons behind misclassifications or low accuracy for specific emotions.

1.Speed and Efficiency:

Measure the average processing time required for emotion recognition on individual frames or images. Evaluate whether the system meets real-time processing requirements.

Identify any bottlenecks or computational inefficiencies that affect the system's speed and responsiveness.

2.Robustness and Generalization:

Test the system on diverse datasets with varying lighting conditions, poses, and facial expressions to assess its robustness and ability to generalize to new data.

Evaluate the system's performance on data that is different from the training set to ensure it can handle real-world variations.

3.Impact of Preprocessing Techniques:

Evaluate the impact of preprocessing techniques (e.g., image resizing, normalization) on the system's performance. Determine the optimal preprocessing pipeline for best results.

4.Cross-Validation:

Implement cross-validation to assess the system's performance on different subsets of the data. This helps validate the system's generalization capabilities and reduces the risk of overfitting.

5.Performance Metrics Comparison:

Compare the system's performance using different evaluation metrics, such as accuracy, precision, recall, F1 score, etc. This provides a comprehensive understanding of the system's strengths and weaknesses.

6.Error Analysis:

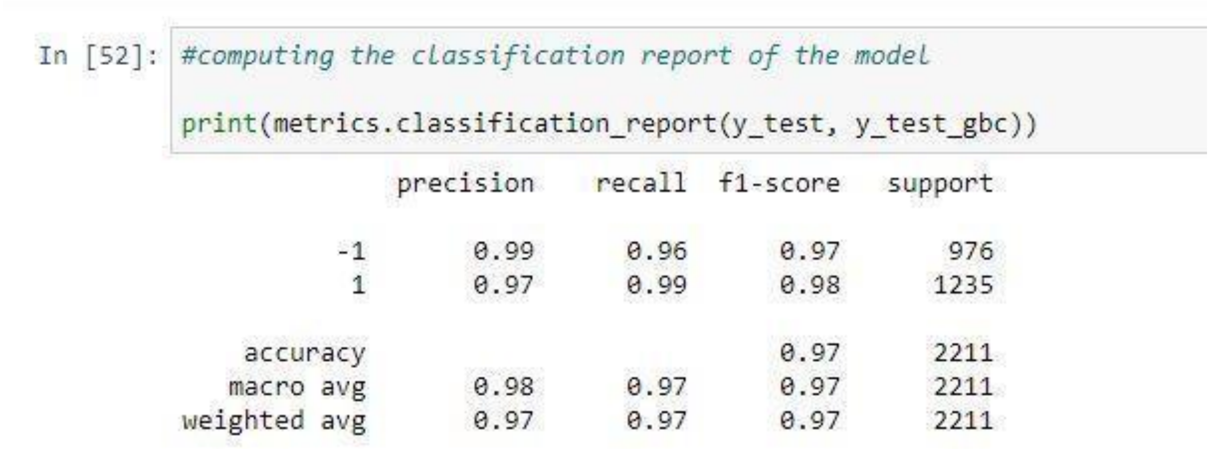
Conduct an in-depth error analysis to understand common patterns of misclassifications. Identify scenarios or facial expressions that are particularly challenging for the system.

7.Resource Utilization:

Measure the system's resource utilization, such as CPU and memory usage, to ensure it operates efficiently and within acceptable resource constraints.

8.User Feedback:

Gather feedback from users or human evaluators on the system's performance. This provides insights into the system's real-world effectiveness and user satisfaction.



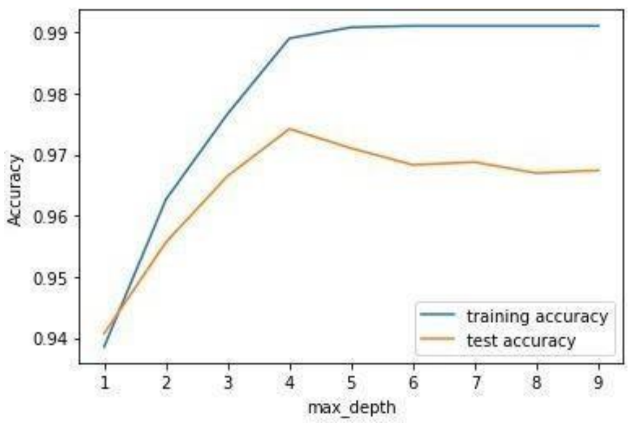
```
In [52]: #computing the classification report of the model

         print(metrics.classification_report(y_test, y_test_gbc))
```

```
              precision    recall  f1-score   support

          -1       0.99      0.96      0.97       976
           1       0.97      0.99      0.98      1235

    accuracy                           0.97      2211
   macro avg       0.98      0.97      0.97      2211
weighted avg       0.97      0.97      0.97      2211
```



CHAPTER-9

Conclusion and Future Enhancement

CHAPTER 9

Conclusion and Future Enhancement

9.1. Limitations of the Project

Phishing website detection, while crucial for enhancing cybersecurity, comes with several limitations and challenges. Some of the common limitations of phishing website detection projects include:

Evolving Phishing Techniques: Phishing attacks are continually evolving and becoming more sophisticated. Attackers may use obfuscation, encryption, and other techniques to make it challenging to detect phishing websites using conventional methods.

Zero-Day Attacks: Zero-day attacks refer to new and previously unknown vulnerabilities that hackers exploit before they are discovered and patched. Detecting phishing websites in such cases can be challenging, as there may not be enough data to train the models on these new threats.

Small Data Samples: Obtaining labeled data for training machine learning models can be difficult, especially for newer or less common phishing websites. Limited data can lead to overfitting or reduced model performance.

Imbalanced Data: Phishing websites are relatively less common compared to legitimate websites. This class imbalance can affect the performance of machine learning models, as they may become biased towards the majority class.

Feature Engineering: Extracting relevant and discriminative features from websites for training the models is complex. Choosing the right set of features that can effectively differentiate between phishing and legitimate websites requires domain expertise.

Domain-Specific Phishing: Phishing attacks can be tailored to specific industries or organizations, making it harder to detect using generic models. Customizing detection for each domain can be resource-intensive and time-consuming.

Time Sensitivity: Detecting phishing websites quickly is crucial to prevent users from falling victim to the attacks. However, some machine learning models might be computationally expensive and slow for real-time detection.

Adversarial Attacks: Phishers might attempt to evade detection by intentionally modifying their websites to bypass machine learning algorithms. Adversarial attacks can be challenging to defend against.

Privacy Concerns: Phishing website detection often involves analyzing website content and URLs, raising privacy concerns, especially when dealing with user-related data.

False Positives and False Negatives: Machine learning models can produce false positives (legitimate websites incorrectly flagged as phishing) or false negatives (phishing websites not detected), impacting the effectiveness of the system.

To mitigate these limitations, it's essential to continuously update and improve the detection system, leverage ensemble methods, explore anomaly detection techniques, and combine machine learning with other security measures for a comprehensive defense against phishing attacks. Regular monitoring and evaluation of the system's performance are also crucial to identify and address any emerging issues.

9.2. Future Enhancements

Facial Emotion Recognition is a dynamic field with ongoing research and advancements. Several future enhancements can be explored to improve the accuracy, usability, and practicality of Facial Emotion Recognition systems. Here are some potential areas for future development:

1.Large and Diverse Datasets: Collecting and using large, diverse, and balanced datasets representing various demographics, cultural backgrounds, and expressions can improve the generalization of models and reduce bias.

2.Transfer Learning: Leveraging pre-trained deep learning models for related tasks (e.g., face recognition) and fine-tuning them for emotion recognition can lead to better performance, especially with limited training data.

3.Temporal Modeling: Incorporating temporal information from video sequences to capture the dynamics of facial expressions and emotions over time can improve recognition accuracy.

4.Data Augmentation: Applying data augmentation techniques (e.g., rotation, translation, and brightness adjustment) can help increase the diversity and size of the training dataset.

9.3. Summary

Phishing attacks are a rapidly expanding threat in the cyber world, costing internet users billions of dollars each year. It involves the use of a variety of social engineering tactics to obtain sensitive information from users. Hence, Phishing techniques can be detected using a variety of types of communication, including email, instant chats, pop-up messages, and web pages.

This project was able to categorize and recognize how phishers carry out phishing attacks and the different ways in which researchers have helped to solve phishing detection. Hence, the proposed system of this project worked with different feature selection and machine learning and deep neural networks such as Decision Tree, and Random Forest to identify patterns in which URL links can be detected easily.

The Model with the highest accuracy based on the feature extraction algorithm used to identify phishing URL from legitimate URL links was integrated to a web application where users can input website URL links to detect if it is legitimate or phishing.

References

1.Abdelhamid, N., Thabtah F., & Abdel-Jaber, H. Phishing detection: A recent intelligent machine learning comparison based on models' content and features," 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), Beijing, 2017, pp. 72-77, DOI: 10.1109/ISI.2017.8004877.

2.Anjum N. S., Antesar M. S., & Hossain M.A. (2016). A Literature Review on Phishing Crime, Prevention Review and Investigation of Gaps. Proceedings of the 10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA), Chengdu, China, 2016, pp. 9-15, DOI: 10.1109/SKIMA.2016.7916190.

3.Almomani, A., Gupta, B. B., Atawneh, S., Meulenberg, A., & Almomani, E. (2013). A survey of phishing email filtering techniques, Proceedings of IEEE Communications Surveys and Tutorials, vol. 15, no. 4, pp. 2070–2090.

4.Ashritha, J. R., Chaithra, K., Mangala, K., & Deekshitha, S. (2019). A Review Paper on Detection of Phishing Websites using Machine Learning.Proceedings of International Journal of Engineering Research & Technology (IJERT), 7, 2. Retrieved from www.ijert.org.

5.Anti-Phishing Working Group (APWG) Phishing activity trends report the first quarter. (2014) Retrieved from http://docs.apwg.org/reports/apwg trends report q1 2014.pdf

6. APWG report. (2014). Retrieved from http://apwg.org/download/document/245/APWG Global Phishing Report 2H 2014.

pdf 111 Ayush, P. (2019). Workflow of a Machine Learning project. Retrieved from https://towardsdatascience.com/workflow-of-a-machine-learning-projectec1dba419b94

7.Camp W. (2001). Formulating and evaluating theoretical frameworks for career and technical education research. Journal of Vocational Education Research, 26(1), 4- 25.

8.DeepAI (n.d.). About clinical psychology. Retrieved from https://deepai.org/machine-learning-glossary-and-terms/feature-extraction

9. Engine K., & Christopher K. (2005). Protecting Users Against Phishing Attacks. Proceedings of the Oxford University Press on behalf of The British Computer Society, Oxford University,0,2005,Retrievedfrom: https://sites.cs.ucsb.edu/~chris/research/doc/cj06_phish.pdf

10.Gandhi, V. (2017). A Theoretical Study on Different ways to identify the Phishing URL and Its Prevention Approaches: presented at International Conference on Cyber Criminology, Digital Forensics and Information Security at DRBCCC Hindu College, Chennai. Retrieved from https://www.researchgate.net/publication/319006943_A_Theoretical_Study_on_Different_ways_to_Identify_the_Phishing_URL_and_Its_Prevention_Approache s

11.Gupta, B. B., Tewari, A., Jain, A. K., & Agrawal, D. P. (2016). Fighting against phishing attacks: state of the art and future challenges, Neural Computing and Applications. Internet world stats usage and population statistics. (2014). Retrieved from http://www.internetworldstats.com/stats.htm.

12.Imperva.(2021). Phishing attacks. Retrieved from https://www.imperva.com/learn/application-security/phishing-attack-scam/

13.Kiruthiga, R., Akila, D. (2019, September). Phishing Websites Detection Using Machine Learning. Retrieved from https://www.researchgate.net/publication/337049054 Phishing Websites Detection Using_Machine_Learning.

14.KnowBe4 (2021). Phishing Techniques. Retrieved from https://www.phishing.org/phishing-techniques

15.Kondeti, P. S., Konka, R. C., & Kavishree, S. (2021). Phishing Websites Detection using Machine Learning Techniques. International Research Journal of Engineering and Technology, 08(4), Page 1471-1473. Retrieved from https://www.irjet.net/archives/V8/i4/IRJET-V8I4274.pdf

16.Noel, B. (2016). Support Vector Machines: A Simple Explanation. Retrieved from https://www.kdnuggets.com/2016/07/support-vector-machines-simpleexplanation.html

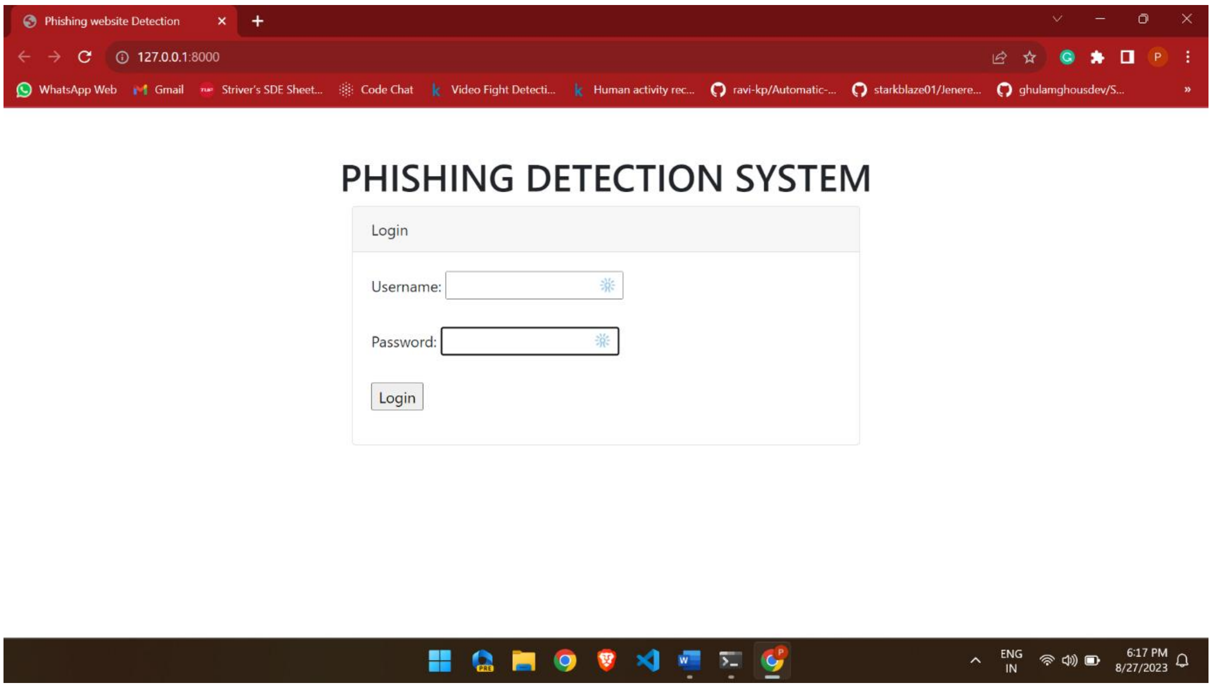
17.Osanloo, A., & Grant, C. (2016). Understanding, selecting, and integrating a theoretical framework in dissertation research: creating the blueprint for your "house". Administrative issues journal: connecting education, practice and research 4(2), 7.

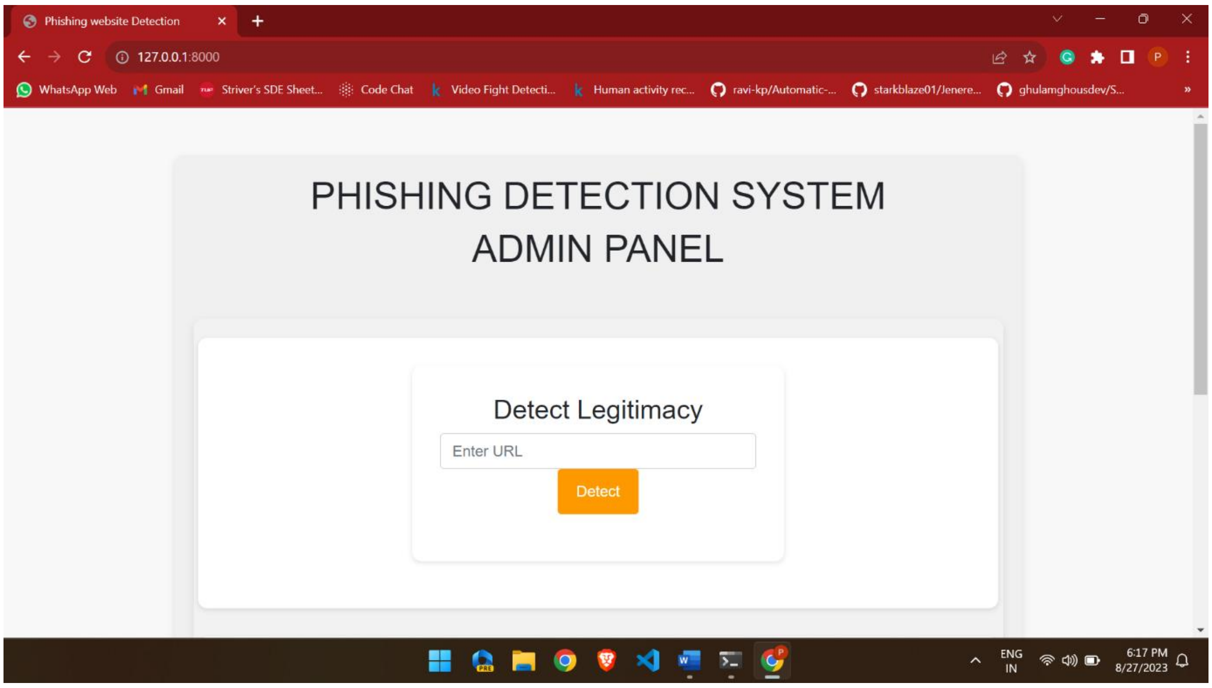
18.Peng, T., Harris, I., & Sawa, I. (2018). Detecting Phishing Attacks Using Natural Language Processing and Machine Learning. Proc. - 12th IEEE Int. Conf. Semant. Comput. ICSC 2018, vol. 2018–Janua, pp. 300–301.

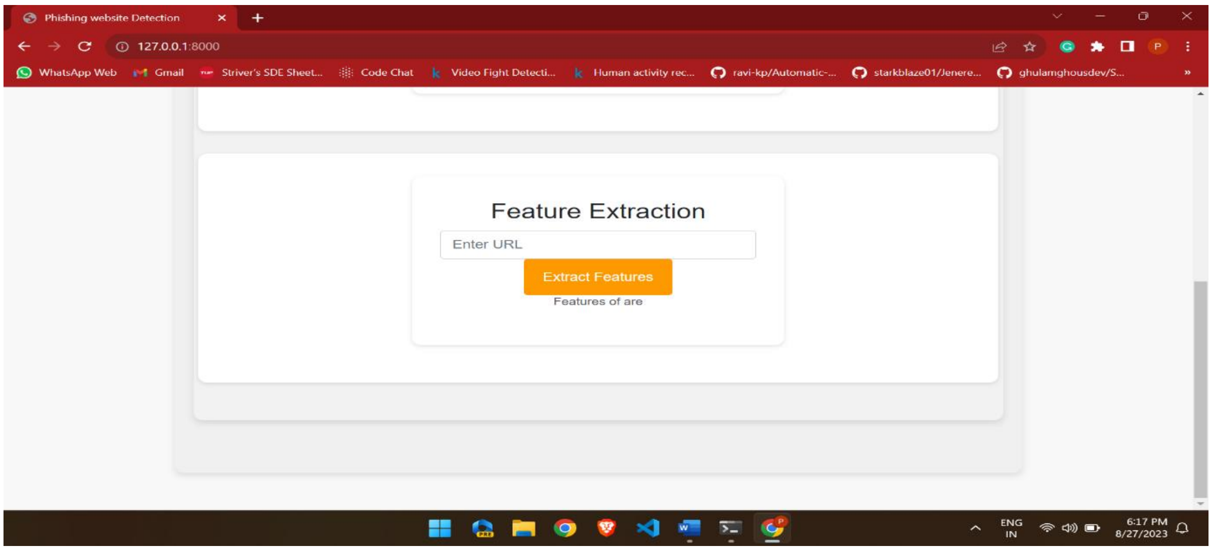
19.J. J. Pao, "Phishing website identification" p.6, 2018.

20.N. Zeng, H. Zhang, B. Song, W. Liu, Y. Li, and A. M. Dobaie, "Phishing websites" Neurocomputing, vol. 273, pp. 643–649, 2018.

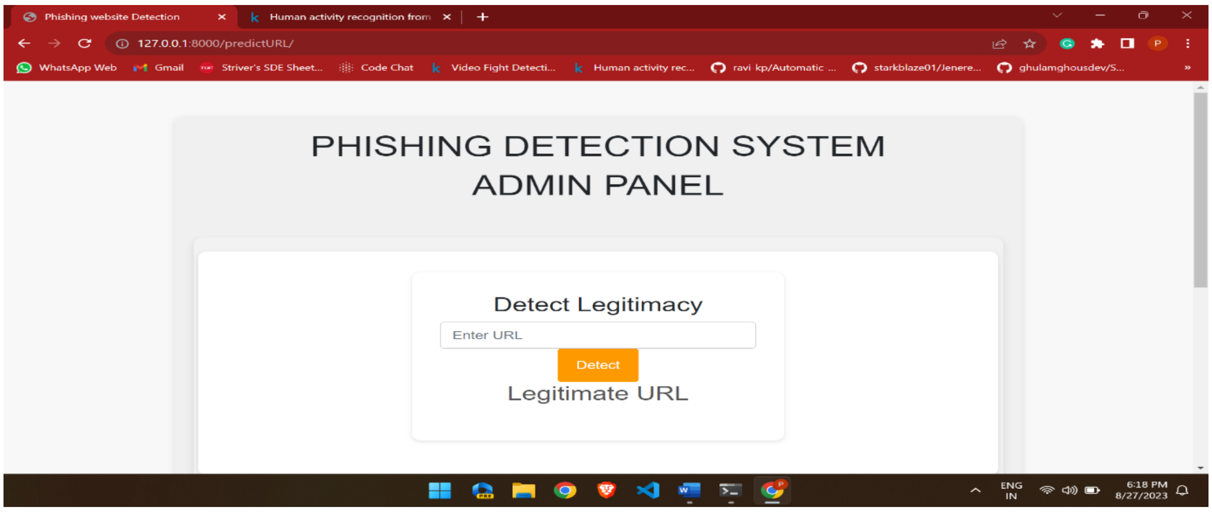
Appendix -1


Phishing website Detection
127.0.0.1:8000
PHISHING DETECTION SYSTEM
Login
Username:
Password:
Login

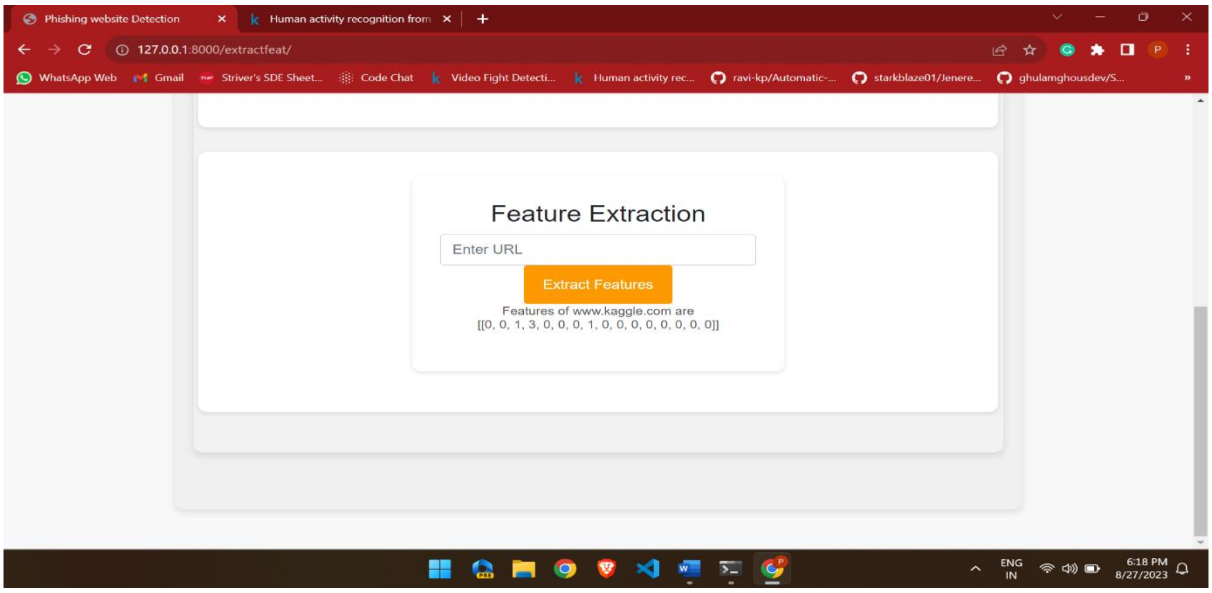

Phishing website Detection
127.0.0.1:8000
PHISHING DETECTION SYSTEM
ADMIN PANEL
Detect Legitimacy
Enter URL
Detect



Features of are


Phishing website Detection
Human activity recognition from
127.0.0.1:8000/predictURL/
PHISHING DETECTION SYSTEM
ADMIN PANEL
Detect Legitimacy
Enter URL
Detect
6:18 PM
8/27/2023

Legitimate URL


Phishing website Detection
Human activity recognition from
127.0.0.1:8000/extractfeat/
Feature Extraction
Enter URL
Extract Features
6:18 PM
8/27/2023

Features of www.kaggle.com are
[[0, 0, 1, 3, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]]