WEB PHISHING DETECTION

A PROJECT REPORT

Submitted by

KESHAV KHANTH T (410719104042)

HARISH R (410719104031)

SARATH KUMAR I (410719104089)

LOKESH SUNIL D (410719104049)

TEAM ID : PNT2022TMID28668

of

COMPUTER SCIENCE AND ENGINEERING

DHANALAKSHMI COLLEGE OF ENGINEERING

CHENNAI – 601301

ABSTRACT

Phishing URL is a widely used and common technique for cyber security attacks. Phishing is a cybercrime that tries to trick the targeted users into exposing their private and sensitive information to the attacker. The motive of the attacker is to gain access to personal information such as usernames, login credentials, passwords, financial account details, social networking data, and personal addresses. These private credentials are then often used for malicious activities such as identity theft, notoriety, financial gain, reputation damage, and many more illegal activities. This paper aims to provide a comprehensive and comparative study of various existing free service systems and research based systems used for phishing website detection. The systems in this survey range from different detection techniques and tools used by many researchers. The approach included in these researched papers ranges from Blacklist and Heuristic features to visual and content-based features. The studies presented here use advanced machine learning and deep learning algorithms to achieve better precision and higher accuracy while categorizing websites as phishing or benign. This article would provide a better understanding of the current trends and existing systems in the phishing detection domain.

CHAPTER 1

INTRODUCTION

1.1 Project Overview: Web phisha website which isused to detect phishing sites to improve the customer's sense ofsafety wheneverhe/she attempts to provide any sensitive information to a site. Also, by which people won't access them which will reduce the revenue of malicious site owners. This applicationcan be accessed online without paying instead, can be accessed via any browser of the customer'schoice to detect any site with high accuracy. This system uses machine learning algorithm which implements classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy. The design and implementation of a comprehensive web phishing detection system instils a cyber security culture which prevents the need for the deployment of targeted anti-phishing solutions in acorporate to meet industry's compliance obligations.

1.2 Purpose:

Web phishing is a threat in variousaspects of securityon the internet, which might involve scams andprivate information disclosure. Some of the common threats of web phishing are:

i. Attempt to fraudulently solicit personalinformation from an individual or organization.

ii. Attempt to deliver malicious softwareby posing as a trustworthy organization or entity.

iii. Installing those malwaresinfects the data that cause data breach or even natu re'sforcesthat takes down your company's data headquarters, disrupting access.

For this purpose, the objective of our project involves building an efficient and intelligent system to detect such websites by applying a machine-learning algorithm which implements classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy and asa result of which whenever a user makes a transaction online and makes payment through an e- banking website our system will use a data mining algorithm to detect whether the e-banking website is a phishing websiteor not.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing problem:

There are phishing detection sites out in the web. But they charge users after a limit of usage. Most of them are built on a clean set of features. We have carefully analysed and identified several factorsthat could be used to detect a phishing site. These factors fall under the categories of address bar- based features, domain-based features, HTML & JavaScript based features. Using these features, we build an intelligent system which can identify a phishing site with high accuracy and efficiency. It is also an open-source website which will be easily accessible to all users.

Problem statement definition:

Phishing detection techniques do suffer low detection accuracy and high false alarm especially when novel phishing approaches are introduced. Besides, the most common technique used, blacklist-based method is inefficient in responding to emanating phishing attacks since registering new domain has become easier, no comprehensive blacklist can ensure a perfect up-to-date database. Furthermore, page content inspection has been used by some strategies to overcome the false negative problems and complement the vulnerabilities of the stale lists. Moreover, page content inspection algorithms each have different approach to phishing website detection with varying degrees of accuracy. Therefore, ensemble can be seen to be a better solution as it can combine the similarity in accuracy and different error-detection rate properties in selected algorithms. Therefore, this study will address a couple of research:

Internet has dominated the world by dragging half of the world’s population exponentially into the cyber world. With the booming of internet transactions, cybercrimes rapidly increased and with anonymity presented by the internet, Hackers attempt to trap the end-users through various forms such as phishing, SQL injection, malware, man-in-the-middle, domain name system tunnelling, ransom ware, web Trojan, and so on. Among all these attacks, phishing reports to be the most deceiving attack. Our main aim of this paper is classification of a phishing website with the aid of various machine learning techniques to achieve maximum accuracy and concise model.

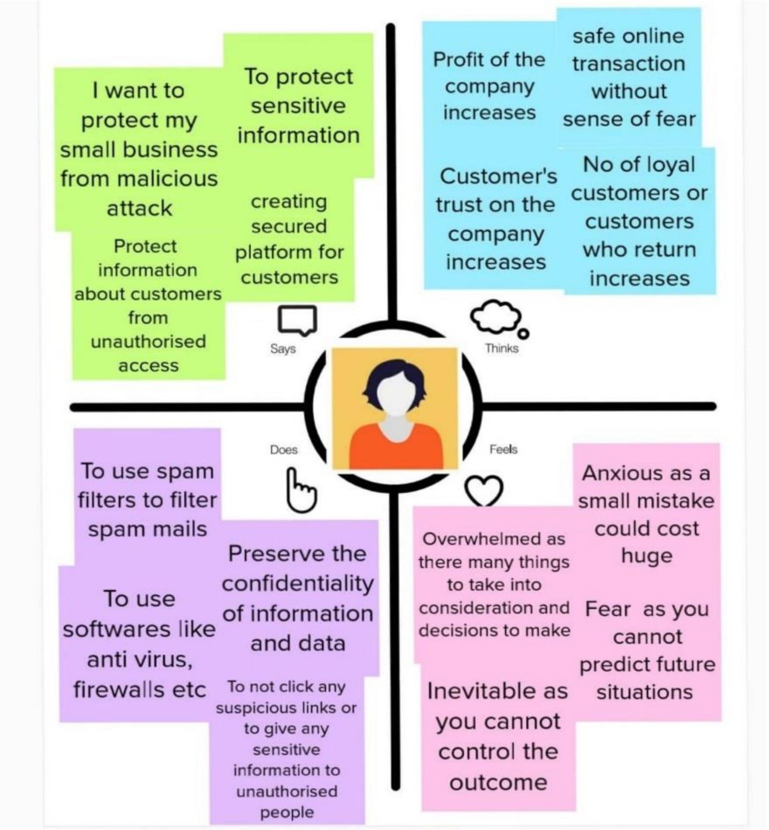
Nowadays, many people are losing considerable wealth due to online scams. Phishing is one of the means that a scammer can use to deceitfully obtain the victim's personal identification, bank account information, or any other sensitive data. There are a number of anti-phishing techniques and tools in place, but unfortunately phishing still works. One of the reasons is that phishers usually use human behaviour to design and then utilise a new phishing technique. Therefore, identifying the psychological and sociological factors used by scammers could help us to tackle the very root causes of fraudulent phishing attacks

CHAPTER 3

IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas:

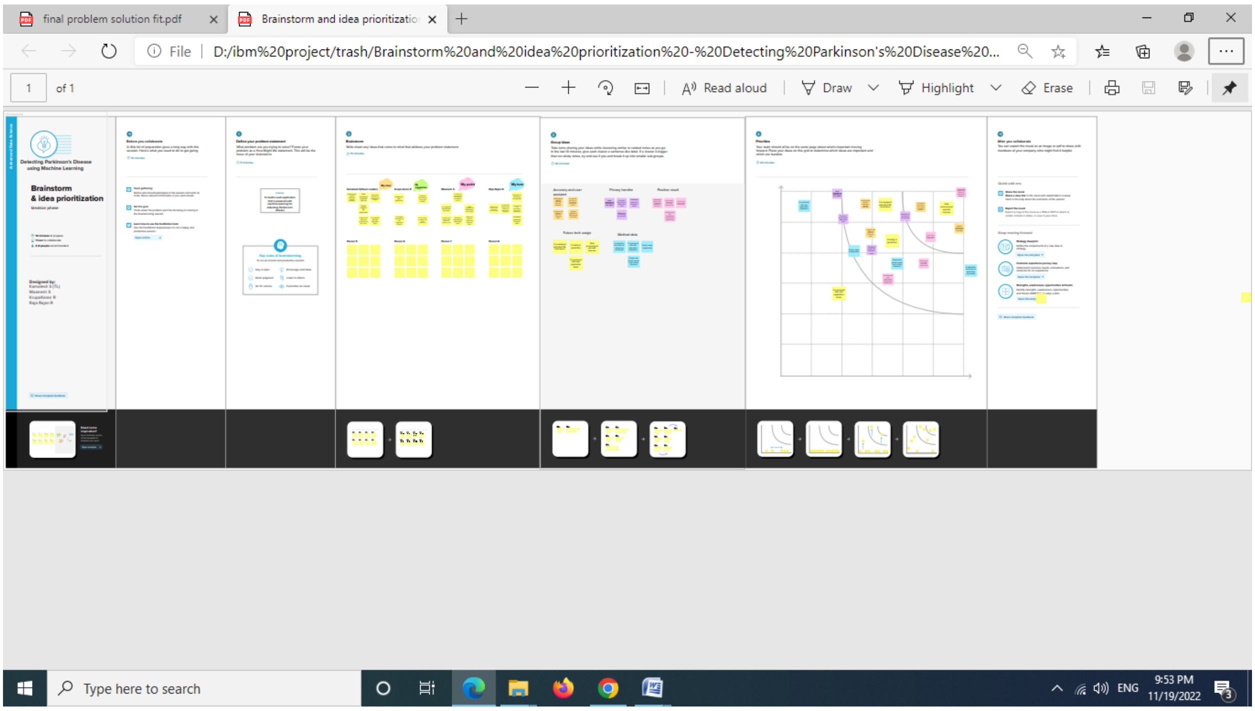
Phishing attack is a simplest way to obtain sensitive information from innocent users. Aim of the phishers is to acquire critical information like username, password and bank account details. Cyber security persons are now looking for trustworthy and steady detection techniques for phishing websites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


I want to protect my small business from malicious attack
To protect sensitive information
Protect information about customers from unauthorised access
creating secured platform for customers
Says
Profit of the company increases
safe online transaction without sense of fear
Customer's trust on the company increases
No of loyal customers or customers who return increases
Thinks
Does
To use spam filters to filter spam mails
Preserve the confidentiality of information and data
To use softwares like anti virus, firewalls etc
To not click any suspicious links or to give any sensitive information to unauthorised people
Feels
Overwhelmed as there many things to take into consideration and decisions to make
Anxious as a small mistake could cost huge
Fear as you cannot predict future situations
Inevitable as you cannot control the outcome

3.2 Ideation & Brainstorming:

Ideation essentially refers to the whole creative process of coming up with and communicating new ideas. Ideation is innovative thinking, typically aimed at solving a problem orproviding a more efficient means of doing or accomplishing something.

Ideation is often closely related to the practice of brainstorming, a specific technique that isutilized to generate new ideas. A principal difference between ideation and brainstorming is that ideation is commonly more thought of as being an individual pursuit, while brainstorming is almost always a group activity.

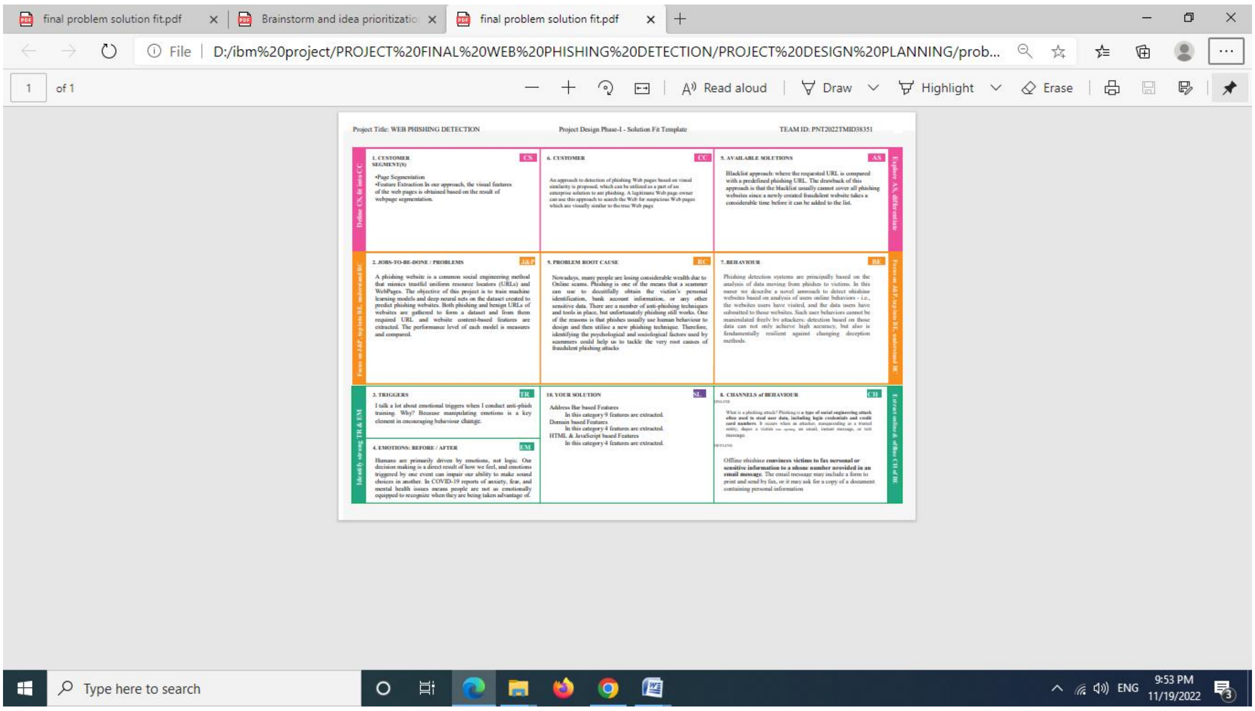

final problem solution fit.pdf
Brainstorm and idea prioritizatio
File
D:/ibm%20project/trash/Brainstorm%20and%20idea%20prioritization%20-%20Detecting%20Parkinson's%20Disease%20...
1
of 1
Read aloud
Draw
Highlight
Erase
Detecting Parkinson's Disease using Machine Learning
Brainstorm & idea prioritization
Type here to search
ENG
9:53 PM
11/19/2022

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Internet has dominated the world by dragging half of the world's population exponentially into the cyber world. With the booming of internet transactions, cybercrimes rapidly increased and with anonymity presented by the internet, Hackers attempt to trap the end-users through various forms such as phishing, SQL injection, malware, man-in-the-middle, domain name system tunnelling, ransom ware, web Trojan, and so on. Among all these attacks, phishing reports to be the most deceiving attack. Our main aim of this paper is classification of a phishing website with the aid of various machine learning techniques to achieve maximum accuracy and concise model.
2.	Idea / Solution description	Detection and prevention of phishing websites endure measure continuously a major space for analysis. There are different types of phishing techniques that offer torrential and essential ways that offer attackers to penetrate the data of people and organizations. Uniform resource locator URLs sometimes are also referred to as "Web links" play a vital role in a phishing attack. Uniform resource locator has a vulnerability of redirecting the pages i.e., through the hyperlink; which could redirect to the legitimate website or the phishing site. Different techniques in making phishing sites are emerging day by day. This actually motivated several researchers to put up their concentrate on finding the phishing sites.
3.	Novelty / Uniqueness	Microsoft. Microsoft Security Index Report.
4.	Social Impact / Customer Satisfaction	An exhaustive systematic search was performed on all the indexing databases. The state-of-the-art research related to the web phishing detections was collected. The papers were classified based on the methodologies. Taxonomy was derived by performing a deep scan on the classified papers. The contributions listed in this survey are exhaustive and lists all the state-of-the-art development in this area.
5.	Business Model (Revenue Model)	An exhaustive systematic search was performed on all the indexing databases. The state-of-the-art research related to the web phishing detections was collected. The papers were classified based on the methodologies. Taxonomy was derived by performing a deep scan on the classified papers. The contributions listed in this survey are exhaustive and lists all the state-of-the-art development in the area. A phishing scan starts with spreading bogus e-mail. After receiving an e-mail, ant phishing techniques start working, either by redirecting the phishing mail in the spam folder or by showing a warning when an online user clicks on the link of phishing URL. The lifecycle of phishing attackt in this area.
6.	Scalability of the Solution	The key notable points of our initial work embed: Phishing sites and their domains reveal the features that are different from other sites and domains. (For example, Google; www.google.com and some random phishing website be like; www.google.com).Phishing Uniform Resource Locators and 'domain names' typically have a different length when compared to other websites and domain names the training accuracy and testing accuracy of all the models. The difference between the values of train and test accuracy shows that the models are not over fitting over large dataset

3.4 Problem Solution fit

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it solves the customer’s problem. It helps entrepreneurs, marketers


final problem solution fit.pdf
Brainstorm and idea prioritizatio
final problem solution fit.pdf
File
D:/ibm%20project/PROJECT%20FINAL%20WEB%20PHISHING%20DETECTION/PROJECT%20DESIGN%20PLANNING/prob...
1
of 1
Read aloud
Draw
Highlight
Erase
Type here to search
ENG
9:53 PM
11/19/2022

Project Title: WEB PHISHING DETECTION   Project Design Phase-I - Solution Fit Template   TEAM ID: PNT2022TMID38351

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

- •Page Segmentation
- •Feature Extraction In our approach, the visual features of the web pages is obtained based on the result of webpage segmentation.

6. CUSTOMER CC

An approach to detection of phishing Web pages based on visual similarity is proposed, which can be utilized as a part of an enterprise solution to ant phishing. A legitimate Web page owner can use this approach to search the Web for suspicious Web pages which are visually similar to the true Web page

5. AVAILABLE SOLUTIONS AS

Blacklist approach: where the requested URL is compared with a predefined phishing URL. The drawback of this approach is that the blacklist usually cannot cover all phishing websites since a newly created fraudulent website takes a considerable time before it can be added to the list.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

A phishing website is a common social engineering method that mimics trustful uniform resource locators (URLs) and WebPages. The objective of this project is to train machine learning models and deep neural nets on the dataset created to predict phishing websites. Both phishing and benign URLs of websites are gathered to form a dataset and from them required URL and website content-based features are extracted. The performance level of each model is measures and compared.

9. PROBLEM ROOT CAUSE RC

Nowadays, many people are losing considerable wealth due to Online scams. Phishing is one of the means that a scammer can use to deceitfully obtain the victim’s personal identification, bank account information, or any other sensitive data. There are a number of anti-phishing techniques and tools in place, but unfortunately phishing still works. One of the reasons is that phishes usually use human behaviour to design and then utilise a new phishing technique. Therefore, identifying the psychological and sociological factors used by scammers could help us to tackle the very root causes of fraudulent phishing attacks

7. BEHAVIOUR BE

Phishing detection systems are principally based on the analysis of data moving from phishes to victims. In this paper we describe a novel approach to detect phishing websites based on analysis of users online behaviors - i.e., the websites users have visited, and the data users have submitted to those websites. Such user behaviors cannot be manipulated freely by attackers; detection based on those data can not only achieve high accuracy, but also is fundamentally resilient against changing deception methods.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

I talk a lot about emotional triggers when I conduct anti-phish training. Why? Because manipulating emotions is a key element in encouraging behaviour change.

4. EMOTIONS: BEFORE / AFTER EM

Humans are primarily driven by emotions, not logic. Our decision making is a direct result of how we feel, and emotions triggered by one event can impair our ability to make sound choices in another. In COVID-19 reports of anxiety, fear, and mental health issues means people are not as emotionally equipped to recognize when they are being taken advantage of.

10. YOUR SOLUTION SL

Address Bar based Features
In this category 9 features are extracted.
Domain based Features
In this category 4 features are extracted.
HTML & JavaScript based Features
In this category 4 features are extracted.

8. CHANNELS of BEHAVIOUR CH

ONLINE

What is a phishing attack? Phishing is **a type of social engineering attack often used to steal user data, including login credentials and credit card numbers.** It occurs when an attacker, masquerading as a trusted entity, dupes a victim into opening an email, instant message, or text message.

OFFLINE

Offline phishing **convinces victims to fax personal or sensitive information to a phone number provided in an email message.** The email message may include a form to print and send by fax, or it may ask for a copy of a document containing personal information

Extract online & offline CH of BE

CHAPTER 4
REQUIREMENT ANALYSIS

4.1 Functional requirements:

FR No.	Functional Requirement(Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Input	User inputs an URLin required field to checkits validations
FR-2	Website Comparison	Model compares the websites usingBlacklistand White list approach
FR-3	Feature extraction	After comparing, if none found on comparison then it extracts feature usingheuristic and visual similarity approach.
FR-4	Prediction	Model predicts the URL using Machine Learning algorithms suchasLogistic Regression , KNN
FR-5	Classifier	Model sends all output to classifier andproduces final result.
FR-6	Announcement	Model then displays whether website is alegalsite or a phishing site.

FR-7	Events	This modelneeds the capability of retrievinganddisplaying accurate resultfor a website

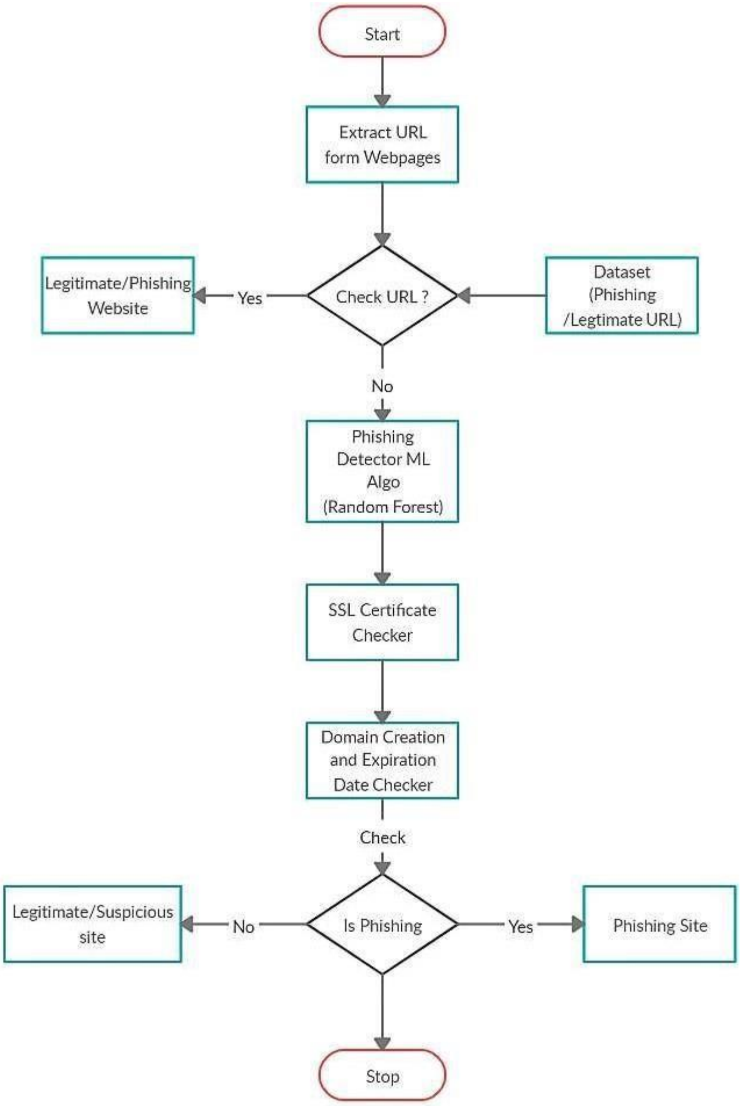
4.2 Non-functional requirements

FR No.	**Non-FunctionalRequirement**	**Description**
NFR-1	Usability	It is an easy to use and access interface which results ingreater efficiency.
NFR-2	Security	It is a secure website which protects the sensitive information of the userand prevents malicious attacks.
NFR-3	Reliability	The system can detect phishing websites with greateraccuracy usingML algorithms.
NFR-4	Performance	The system produces responses within seconds andexecution is faster.
NFR-5	Availability	Users can accessthe website via any browserfromanywhere at any time.
NFR-6	Scalability	This application can be accessed online without paying. It can detectany web site withhigh accuracy.

CHAPTER 5

PROJECT DESIGN

5.1Data Flow diagram: A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. Itshows how data enters and leaves the system, what changes the information, and where data is stored.


Start
Extract URL form Webpages
Check URL ?
Dataset (Phishing /Legtimate URL)
Yes
Legitimate/Phishing Website
No
Phishing Detector ML Algo (Random Forest)
SSL Certificate Checker
Domain Creation and Expiration Date Checker
Check
Is Phishing
No
Legitimate/Suspicious site
Yes
Phishing Site
Stop

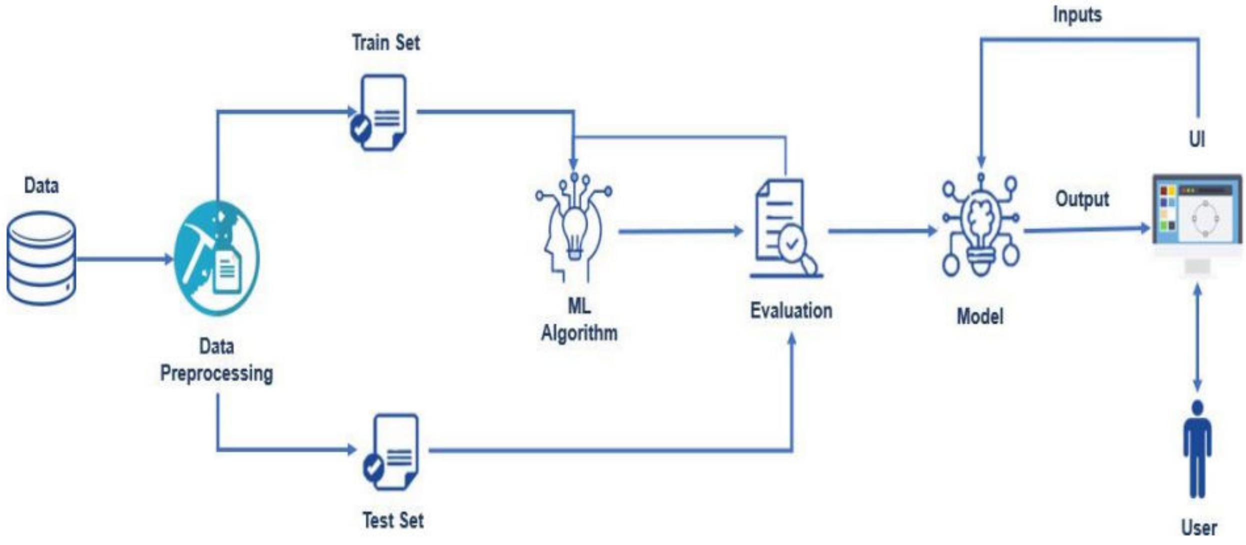
5.2 Solution &Technical Architecture:

SOLUTION:

Our solution is to build an efficient and intelligent system to detect phishing sites by applying a machine learning algorithmwhich implements classification algorithms and techniquesto extract the phishing datasets criteria to classify their legitimacy by carefully analysing and identifying variousfactors that could be used to detect a phishing site. These factors fall under the categories of addressbar-based features, domain-based features, HTML & JavaScript based features. Using these features,we can identifya phishing site with highaccuracy.

TECHNICAL ARCHITECTURE:

Technical architecture which is also often referred to as application architecture includes the major components of the system, their relationships, and the contracts that define the interactions between the components. The goal of technical architects is to achieve all the business needs with an application that isoptimized for both performance and security.



1. The application developer builds a Python-based app and deploysit.
2. The user enters the URL of a websitein the application to checkfor its genuineness.
3. The user submits the URL throughthe web-based application and gets back the result.
4. The user makes a decision whetherto proceed surfingin that website or move to anotherone.

User Stories:

User type	**Functional requirement (Epic)**	**User Story number**	**User story/task**	**Acceptance criteria**
Customer (web user)	Login	USN-1	As a user, I can navigate into the website.	I can access the page.
	Dashboard	USN-2	As a user, I will paste the URL that needs to be checked if it's a phishing website or not.	I can paste the URL in the textbox.
		USN-3	As a user, I can see the output.	I can see if it's a safe site.
Administrator		USN-4	If the new URL is found, I can add the new state into the database.	I can add the new URL.

CHAPTER 6

POJECT PLANNING& SCHEDULIN

Sprint	**Functional Requirement (Epic)**	**User StoryNumber**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-1	User input	USN-1	User inputs an URLin the required fieldto check itsvalidation.	15	High	Keshav Khanth Harish

Sprint-1	WebsiteComparison	USN-2	Model compares the websites using Blacklistand Whitelist approach.	15	High	Keshav Khanth Harish Sarath
Sprint-1	Storage	USN-3	Storing the Blacklisted websites in Databaseusing IBM Cloud.	10	Medium	Harish Sarath Lokesh Sunil
Sprint-2	Feature Extraction	USN-4	After comparison, if none found on comparison then it extract featureusing heuristic andvisual similarity.	5	low	Lokesh Sunil Keshav Khanth
Sprint-2	Prediction	USN-5	Model predicts theURL using Machine learningalgorithms such as logistic Regression, MLP.	15	High	Keshav Khanth Harish Sarath Kumar
Sprint-2	Accuracy Test	USN-6	Selecting the bestaccurate model and top rocess further steps.	15	High	Keshav Khanth Lokesh Sunil Sarth Kumar
Sprint-3	Classifier	USN-7	Model sends all the output to the classifier andproduces the finalresult.	10	Medium	Lokesh Sunil Harish Sarth Kumar
Sprint-3	Hosting	USN-8	Setting Up the Application and hosting in IBMcloud	5	low	Lokesh Sunil Sarth Kumar
Sprint-4	Announcement	USN-9	Model then displays whether the website is legal site or a phishing	15	High	Keshav Khanth Sarath Kumar

			site.			
Sprint-4	Events	USN-10	This model needs the capability of retrievingand displaying accurate result for a website.	15	HIgh	Keshav Khanth Harish

Sprint DeliverySchedule:

Sprint	**Total StoryPoints**	**Duration**	**Sprint Start Date**	**Sprint End Date (Planned)**	**Story Points Completed (as on Planned End Date)**	**Sprint Release Date(Actual)**
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	12 Nov 2022

CHAPTER 7
CODING & SOLUTIONING

Feature 1 – Classificationof URL:

The primary feature of this project is to classify the given URL as phishing or benign. Various classificationalgorithmsare used to achieve this.

Methodology:

XGBoost

Data collection:

URL features of legitimate websites and phishing websites were collected. The data set consists of total 11,055 URLs which include 6,157 legitimate URLs and 4,898 phishing URLs. Legitimate URLs arelabelled as “1” and phishing URLs are labelled as “-1”. The features that are present in the data set include:

IP Address in URL

- Length of URL
- Using URL Shortening Services
- "@"Symbol in URL
- Redirection "//" in URL
- Prefix or Suffix"-" in Domain
- Having Sub Domain
- Length of DomainRegistration
- Favicon
- Port Number
- HTTPS Token
- Request URL
- URL of Anchor

- Links in Tags
- SFH
- Email Submission
- Abnormal URL
- Status Bar Customization (on mouse over)
- Disabling Right Click
- Presence of Popup Window
- IFrame Redirection
- Age of Domain
- DNS Record
- Web Traffic
- Page Rank
- Google Index
- Links pointingto the page
- Statistical Report
- Result

Using IBM Cloud Storage this data is accessed throughout the project. The code written below isused to import the dataset.

Model building:

From the dataset above, it is clear that this is a supervised machine learning task. There are twomajortypes of supervised machine learning problems,called classification and regression.

This data set comes under classification problem, as the input URL is classified as phishing (-1) orlegitimate (1). The supervised machine learning models (classification) considered to train the dataset in this notebook are:

a. XGBoost

b. Decision Tree

c. Random Forest

d. Support Vector Machines

XGBoost:

XGBoost is one of the most popular machine learning algorithms these days. XGBoost stands for eXtreme Gradient Boosting. Regardless of the type of prediction task at hand; regression or classification. XG Boost is an implementation of gradient boosted decision trees designed for speedand performance.

```
import os, types importpandas as pd
  from botocore.client import Configimportibm_boto3
  def___iter__(self): return 0

  # The followingcode accesses a file in your IBM Cloud ObjectStorage. It includes your credentials.

  # You might want to remove those credentials before you share thenotebook.

  cos_client =
          ibm_boto3.client(service_name='s3', ibm_api_key_id='', ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/t
          oken", config=Config(signature_version='oauth'),
          endpoint_url='https://s3.private.us.cloud-object- storage.appdomain.cloud')


bucket = 'webphishingdetection-donotdelete-pr-icmjtvktnzli2s'object_key = 'dataset_website.csv'


  body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']

  # add missing_iter_method, so pandas accepts body as file-like object

  if not hasattr(body, "iter__"): body.iter___= types.MethodType(
  _iter, body )



  data0 = pd.read_csv(body)data0.head()
```

Data pre-processing and Exploratory Data Analysis:

Few plots and graphs were drawn to find how the data is distributed and the how features arerelated to each other

Univariate analysis provides an understanding in the characteristics of each feature in the data set. Different characteristics are computed for numerical and categorical data. For the numerical features characteristics are standard deviation, skewness, kurtosis, percentile, interquartile range (IQR) and range. For the categorical features characteristics are count, cardinality, list of unique values, top and freq.

Bivariate analysis:

```
plt.figure(figsize=(15,13))
sns.heatmap(data0.corr()) plt.show()
```

From this correlation matrix, it is evident that there is no correlation with many features. So, it iscrucialto eliminate these features.

Multivariate analysis:

```
data0.hist(bins = 50,figsize = (15,15))plt.show()
```

From data distribution graph and correlation matrix, we can conclude that the following features donot have much impact on the result:

- having_Sub_Domain
- Domain_registeration_length
- Favicon
- Request_URL
- URL_of_Anchor
- Links_in_tags
- Submitting_to_email
- Redirect
- web_traffic
- Page_Rank
- Google_Index
- Links_pointing_to_page

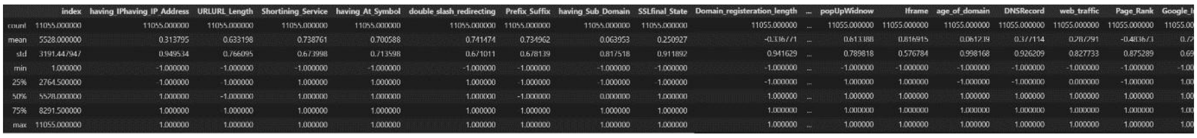
All the above features will not be included in further processing.

#Removing the features which do not have much impact on Result

data=data0.iloc[:,[1,2,3,4,5,6,12,20,21,22,23,24,25,30,31]]

data.head()

```
data0.describe()
```



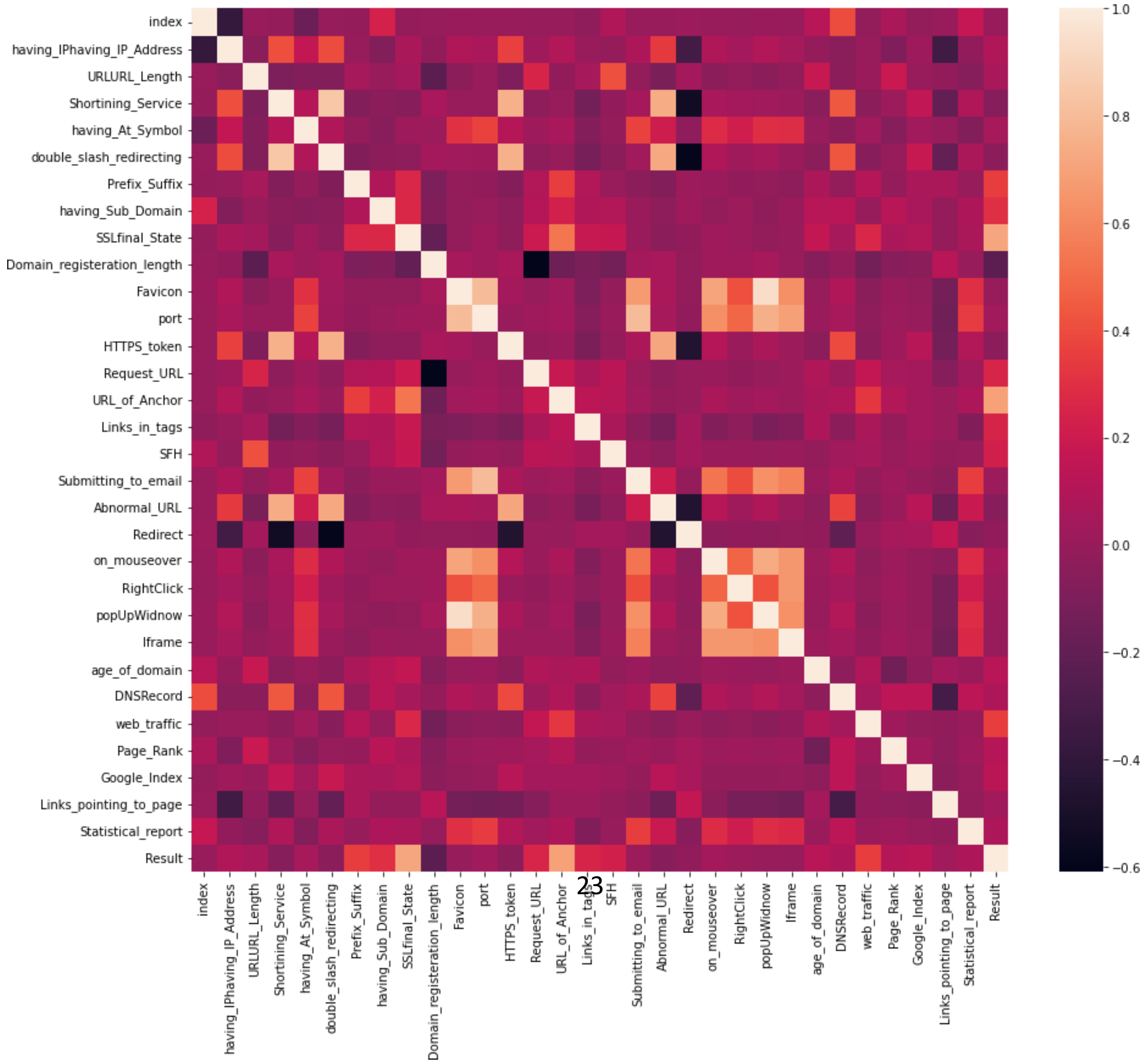
	index	having_IPhaving_IP_Address	URLURL_Length	Shortining_Service	having_At_Symbol	double_slash_redirecting	Prefix_Suffix	having_Sub_Domain	SSLfinal_State	Domain_registeration_length	...	popUpWidnow	Iframe	age_of_domain	DNSRecord	web_traffic	Page_Rank	Google_I
count	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	...	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.00
mean	5528.000000	0.313795	0.633198	0.738761	0.700588	0.741474	0.734962	0.063953	0.250927	-0.336771	...	0.613388	0.816915	0.061239	0.377114	0.287291	-0.483673	0.72
std	3191.447947	0.949534	0.766095	0.673998	0.713598	0.671011	0.678139	0.817518	0.911892	0.941629	...	0.789818	0.576784	0.998168	0.926209	0.827733	0.875289	0.69
min	1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	...	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.00
25%	2764.500000	-1.000000	-1.000000	1.000000	1.000000	1.000000	-1.000000	-1.000000	-1.000000	-1.000000	...	1.000000	1.000000	-1.000000	-1.000000	0.000000	-1.000000	1.00
50%	5528.000000	1.000000	-1.000000	1.000000	1.000000	1.000000	-1.000000	0.000000	1.000000	1.000000	...	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.00
75%	8291.500000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	...	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.00
max	11055.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	...	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.00

Bivariate analysis:

```
plt.figure(figsize=(15,13))

sns.heatmap(data0.corr())

plt.show()
```

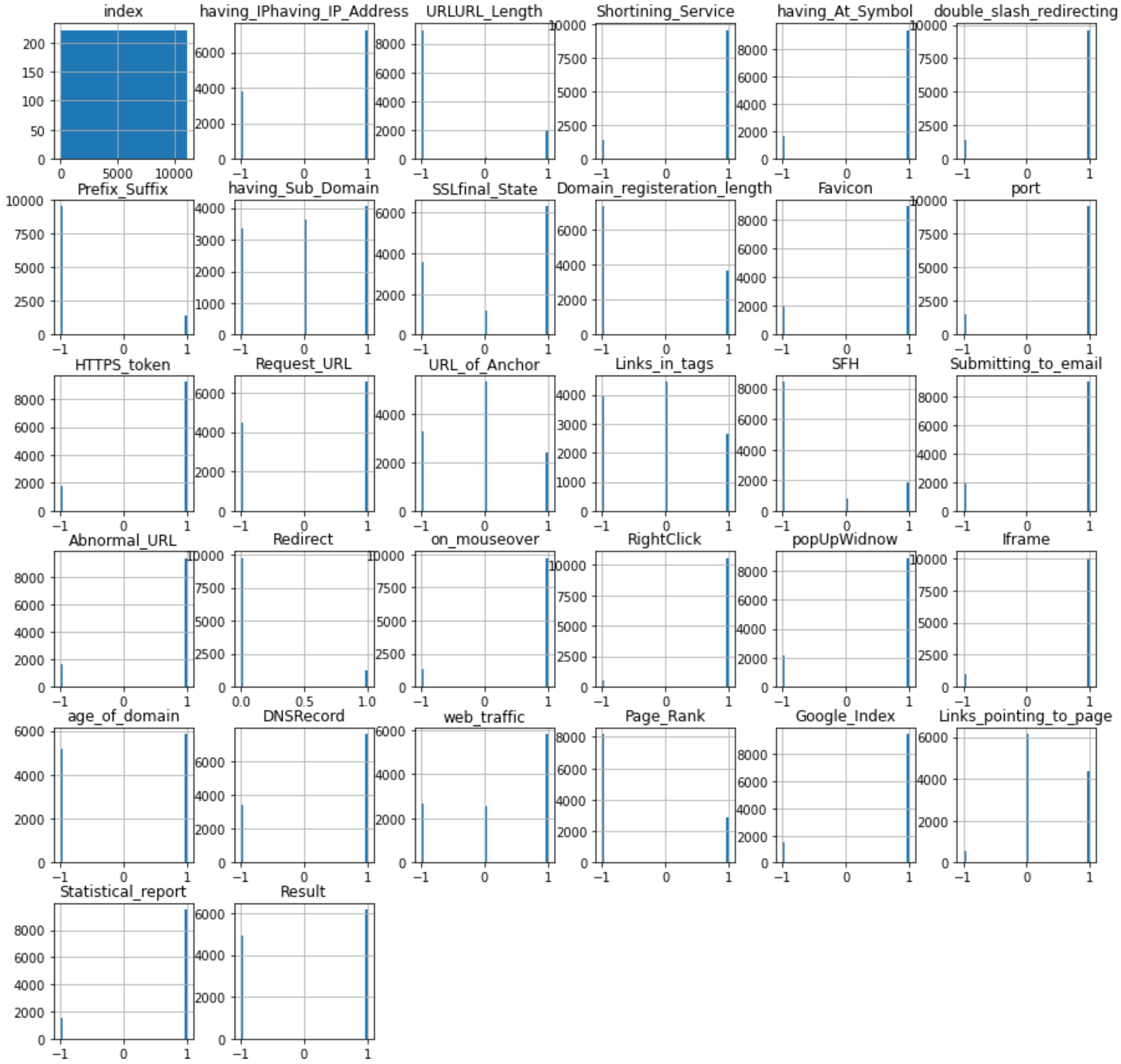


From this correlation matrix, it is evident that there is no correlation with many features. So, it iscrucial to eliminate these features.

Multivariate analysis:

```
data0.hist(bins = 50,figsize = (15,15))

plt.show()
```



From data distribution graph and correlation matrix, we can conclude that the following features donot have much impact on the result:

Checking for null values:

Model building:

From the dataset above, it is clear that this is a supervised machine learning task. There are two majortypes of supervised machine learning problems,called classification and regression.

This data set comes under classification problem, as the input URL is classified as phishing (-1) orlegitimate (1). The supervised machine learning models (classification) considered to train the dataset in this notebook are:

a. XGBoost

b. Decision Tree

c. Random Forest

d. Support Vector Machines

XGBoost:

XGBoost is one of the most popular machine learning algorithms these days. XGBoost stands for eXtreme Gradient Boosting. Regardless of the type of prediction task at hand; regression or classification. XGBoost is an implementation of gradient boosted decision trees designed for speedand performance.

```
#XGBoost Classification model

   from xgboost import XGBClassifier



   import warnings

   warnings.filterwarnings("ignore", category=UserWarning)



   # instantiate the model

   xgb = XGBClassifier(learning_rate=0.4,max_depth=7,verbosity = 0)

   #fit the model

   xgb.fit(X_train, y_train)

   #predicting the target value from the model for the samples
```

```
y_test_xgb = xgb.predict(X_test) y_train_xgb = xgb.predict(X_train)
#computing the accuracy of the model performance acc_train_xgb =
accuracy_score(y_train,y_train_xgb)acc_test_xgb = accuracy_score(y_test,y_test_xgb)


print("XGBoost: Accuracyon training Data:
{:.3f}".format(acc_train_xgb))

print("XGBoost : Accuracy on test Data: {:.3f}".format(acc_test_xgb))
```

Random Forest Classifier:

Random forests for regression and classification are currently among the most widely used machinelearning methods. A random forest is essentially a collection of decision trees, where each tree is slightly different from the others. The idea behind random forests is that each tree might do a relatively good job of predicting, but will likely over fit on part of the data.If we build many trees, all of which work well and overfit in different ways, we can reduce the amount of overfitting by averaging their results. To build a random forest model, you need to decideon the number of trees to build (the n_estimators parameter of RandomForestRegressor or RandomForestClassifier). They are very powerful, often work well without heavy tuning of the parameters, and don't require scaling of the data.

Support Vector Machines:

In machine learning, support-vector machines (SVMs, also support-vector networks) are supervisedlearning models with associated learning algorithms that analyse data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the otherof two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other,making it a non-probabilistic binary linear classifier.

User interface:

The user opens the site and inputs a URL to check its legitimacy. Necessary features are extractedfrom this URL and predictions are made.

Feature extraction:

We will extract the 13 features that we used to train our model.

IP Address in URL:

Checks for the presence of IP address in the URL. URLs may have IP address instead of domain name.If an IP address is used as an alternative of the domain name in the URL, we can be sure that someoneis trying to steal personal information with this URL.,

If the domain part of URL has IP address, the value assigned to this feature is -1 (phishing) or else 1(legitimate).

```
def having_IPhaving_IP_Address(self):try:
    ipaddress.ip_address(self.url) return -1
except:
    return 1
```

Length of URL:

Computes the length of the URL. Phishers can use long URL to hide the doubtful part in the addressbar. 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 featureis -1 (phishing) or else 1 (legitimate).

```
def URLURL_Length(self):if len(self.url) < 54:
    return 1else:
    return -1
```

Using URL Shortening Services:

URL shortening is a method on the "World Wide Web" in which a URL may be made considerably smaller in length and still lead to the required webpage. This is accomplished by means of an "HTTPRedirect" on a domain name that is short, which links to the webpage that has a long URL.

If the URL is using Shortening Services, the value assigned to this feature is -1 (phishing) or else 1(legitimate).

```
def
      Shortining_Service(self):shortening_servi
      ces =
r"bit\.ly|goo\.gl|shorte\.st|go2l\.ink|x\.co|ow\.ly|t\.co|tinyurl|tr\.im|is\.gd|cli\.gs|"
\ r"yfrog\.com|migre\.me|ff\.im|tiny\.cc|url4\.eu|twit\.ac|su\.pr|twurl\.nl|snipurl\.com|"
\r"short\.to|BudURL\.com|ping\.fm|post\.ly|Just\.as|bkite\.com|snipr\.com|fic\.kr|loopt\. us|"
\ r"doiop\.com|short\.ie|kl\.am|wp\.me|rubyurl\.com|om\.ly|to\.ly|bit\.do|t\.co|lnkd\.in|db\.tt|"
\ r"qr\.ae|adf\.ly|goo\.gl|bitly\.com|cur\.lv|tinyurl\.com|ow\.ly|bit\.ly|ity\.im|q\.gs|is\.gd| "
\ r"po\.st|bc\.vc|twitthis\.com|u\.to|j\.mp|buzurl\.com|cutt\.us|u\.bb|yourls\.org|x\.co|"
\ r"prettylinkpro\.com|scrnch\.me|filoops\.info|vzturl\.com|qr\.net|1url\.com|tweez\.me|v
\.gd|" \
```

```
            r"tr\.im|link\.zip\.net" match=re.search(shortening_servi
      ces,self.url)if match:
        return -1else:
        return 1
```

"@" Symbol in URL:

Checks for the presence of '@' symbol in the URL. Using "@" symbol in the URL leads the browser toignore 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 1 (legitimate)

```
def having_At_Symbol(self):if "@" in self.url:
      return -1else:
```

```
 return 1
```

Redirection "//" in URL:

Checks the presence of "//" in the URL. The existenceof “//” within the URL path means that the user will be redirected to another website.The location of the “//” in URL is computed. We find that

if the URL starts with “HTTP”, that means the “//” should appear in the sixth position. However, ifthe URL employs “HTTPS” then the “//” should appear in seventh position.

If the "//" is anywhere in the URL apart from after the protocol, thee value assigned to this feature is-

1 (ph**Redirection "//" in URL:**

Checks the presence of "//" in the URL. The existenceof “//” within the URL path means that the user will be redirected to another website.The location of the “//” in URL is computed. We find thatif the URL starts with “HTTP”, that means the “//” should appear in the sixth position. However, ifthe URL employs “HTTPS” then the “//” should appear in seventh position.
If the "//" is anywhere in the URL apart from after the protocol, thee value assigned to this feature is-
1 (phishing) or else 1 (legitimate).

HTTPS Token:

Checks for the pesenceof "http/https" in the domain part of the URL. The phishers may addthe“HTTPS”token to the domain part of a URL in order to trick users. If the URL has "http/https" in the domain part, the value assigned to this feature is -1 (phishing) orelse1 (legitimate).

```
 def HTTPS_token(self):
    domain = urlparse(self.url).netlocif 'https' in domain:
        return -1else:
        return 1
```

Status Bar Customization (on mouse over):

Phishers may use JavaScript to show a fake URL in the status bar to users. To extract this feature, wemust dig-out the webpage source code, particularly the “onMouseOver” event, and check if it makesany changeson the status bar.If the response is empty or onmouseover is found then, the value assigned to this feature is -1(phishing) or else 1 (legitimate).

Presence of Popup Window:

Pop up windows are another option used by phishers to redirect users to other pages. They display attractive ads to lure the user to click the link. Nonetheless, for this feature, we will search for event“alert” in the webpage source code and check if it is present.If the response is empty or alert is not found then, the value assigned to this feature is -1 (phishing)or else 1 (legitimate).

```
def popUpWidnow(self):try:
        if re.findall(r"alert\(", self.response.text):return 1
        else:
            return -1except:
         return -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 thisregard, phishers make use of the “frame Border” 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 1 (legitimate).

```
def Iframe(self):try:
     if re.findall(r"[<iframe>|<frameBorder>]", self.response.text):return 1
     else:
        return -1except:
      return -1
```

Age of Domain:

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

```
def age_of_domain(self):
      creation_date =
      self.domain_name.creation_date expiration_date = self.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 -1else:
       ageofdomain = abs((expiration_date - creation_date).days)if ((ageofdomain/30) < 6):
          return -1else:
          return 1
```

If age of domain > 12 months,the vlaue of this featureis -1 (phishing) else 1 (legitimate).

DNS Record:

For phishing websites, either the claimed identity is not recognized by the WHOIS database or norecords founded for the hostname.

If the DNS record is empty or not found then, the value assigned to this feature is -1 (phishing) orelse1 (legitimate).

```
dns = -1try:self.domain_name =
whois.whois(urlparse(url).netloc) except:
        dns = 1
```

CHAPTER 8TESTING

Test Cases:

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute		Expected Result	Actual Result	Status
DashBoard_TC_OO1	Functional	Home Page	Verify user is able to enter the URL in the form	1. Open HookPhish website 2. Enter a URL and click submit		Result of classification willbe displayed	Working as expected	Pass
DashBoard_TC_OO2	UI	Home Page	Verify the UIelements in the form	1. Enter URL and click go 2. The services and teams' sections are visible 3. Enter a URL and click submit		Application should show below UI elements: a. input form b. submit button c. services d. team	Working as expected	Pass
DashBoard_TC_OO3	Functional	Home page	Verify user is able to see an alert when nothing is entered in the textbox	1. Enter URL and click go 2. Enter nothing and click submit 3. An alert is displayed to provide proper input		Alert of incomplete input	Working as expected	Pass
DashBoard_TC_OO4	Functional	Home page	Verify user is able to see the result when URL is enteredin the textbox	1. Enter URL and click go 2. Enter any URL and click submit 3. The result of the classification is displayed.		Result of classification willbe displayed	Working as expected	Pass
Report_TC_OO1	Functional	Report page	Verify user is able to enter their name, email and query message in the form	1. Enter URL and click go 2. Click on report button 3. Enter Valid name, email and query in the form 4. Click on submit button	—	Details are storedin the database	Working as expected	Pass

Acceptance Testing:

Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they wereresolved

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

Test Case Analysis:

This reportshows the numberof test cases that have passed, failed,and untested

Section	TotalCases	Not Tested	Fail	Pass
Print Engine	5	0	0	5-
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

CHAPTER 9

RESULTS

9.1 Performance metrics:

The median efficiency is used to assess each categorization model's effectiveness. The final item will appear in the way it was envisioned. Graphical representations are used to depict information duringclassification. The percentage of predictions made using the testing dataset is used to gauge accuracy.By dividing the entirenumber of forecasts even by properlypredicted estimates, it is simple to calculate. The difference between actual and anticipated output is used to calculate accuracy.

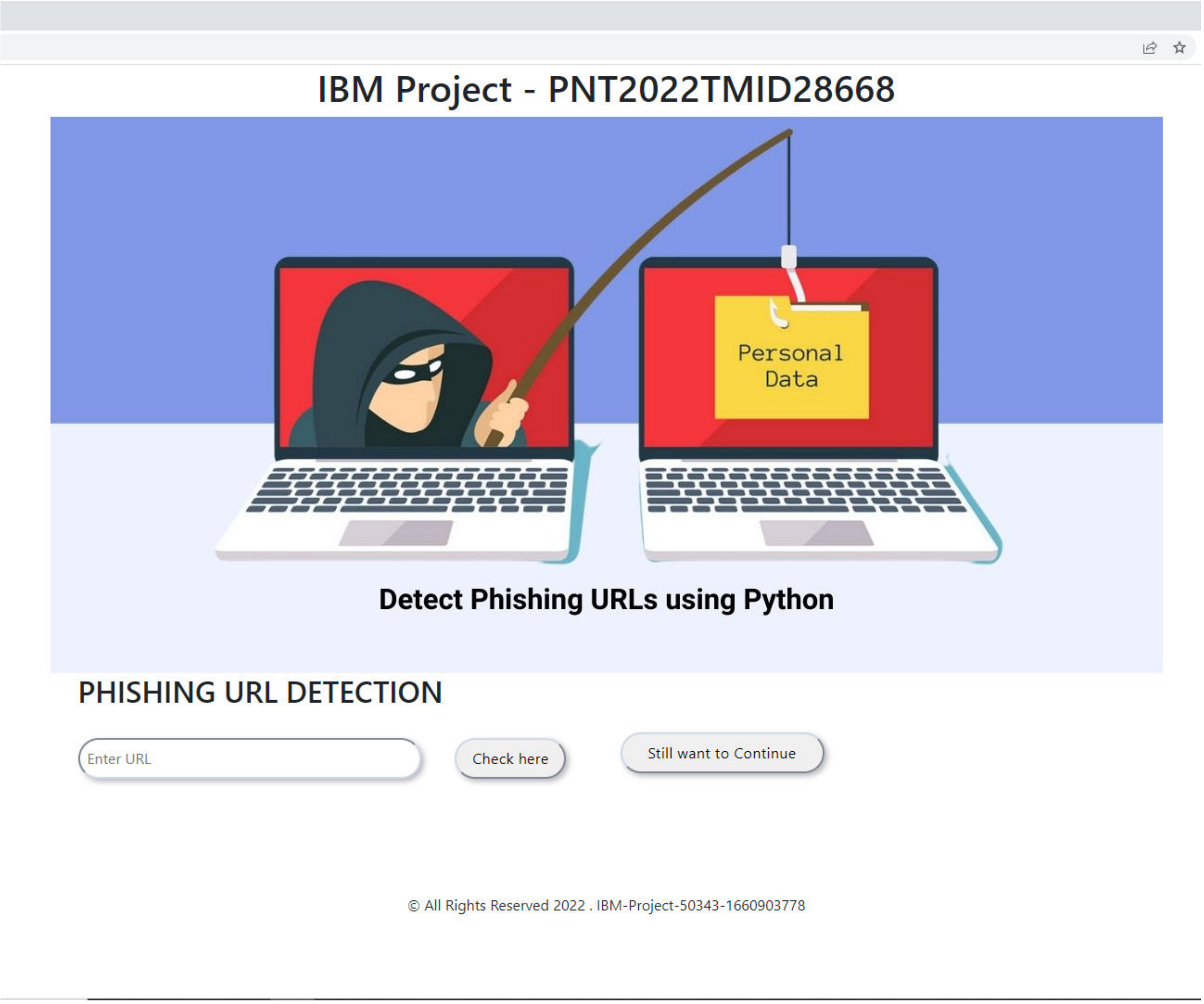
$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

here TP = True Positives, TN = True Negatives, FN = False Negatives and FP = False Positives.

Thus,

accuracy foall the four used models were calculated and ranked. XGBoost performed betterthan other models.

OUTPUT

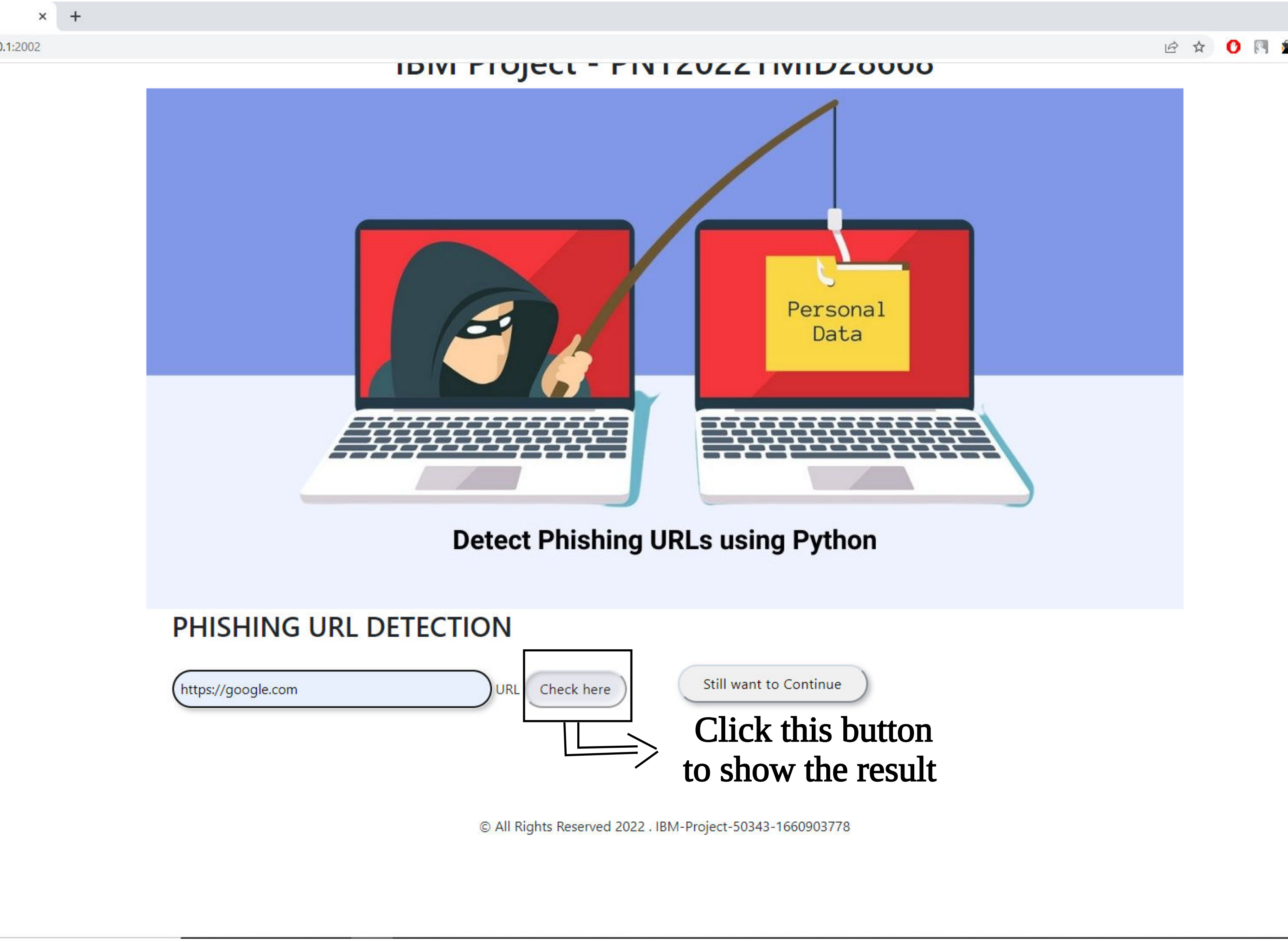


IBM Project - PNT2022TMID28668

PHISHING URL DETECTION

© All Rights Reserved 2022 . IBM-Project-50343-1660903778

After Entering the URL



IBM Project - PNT2022TMID28668

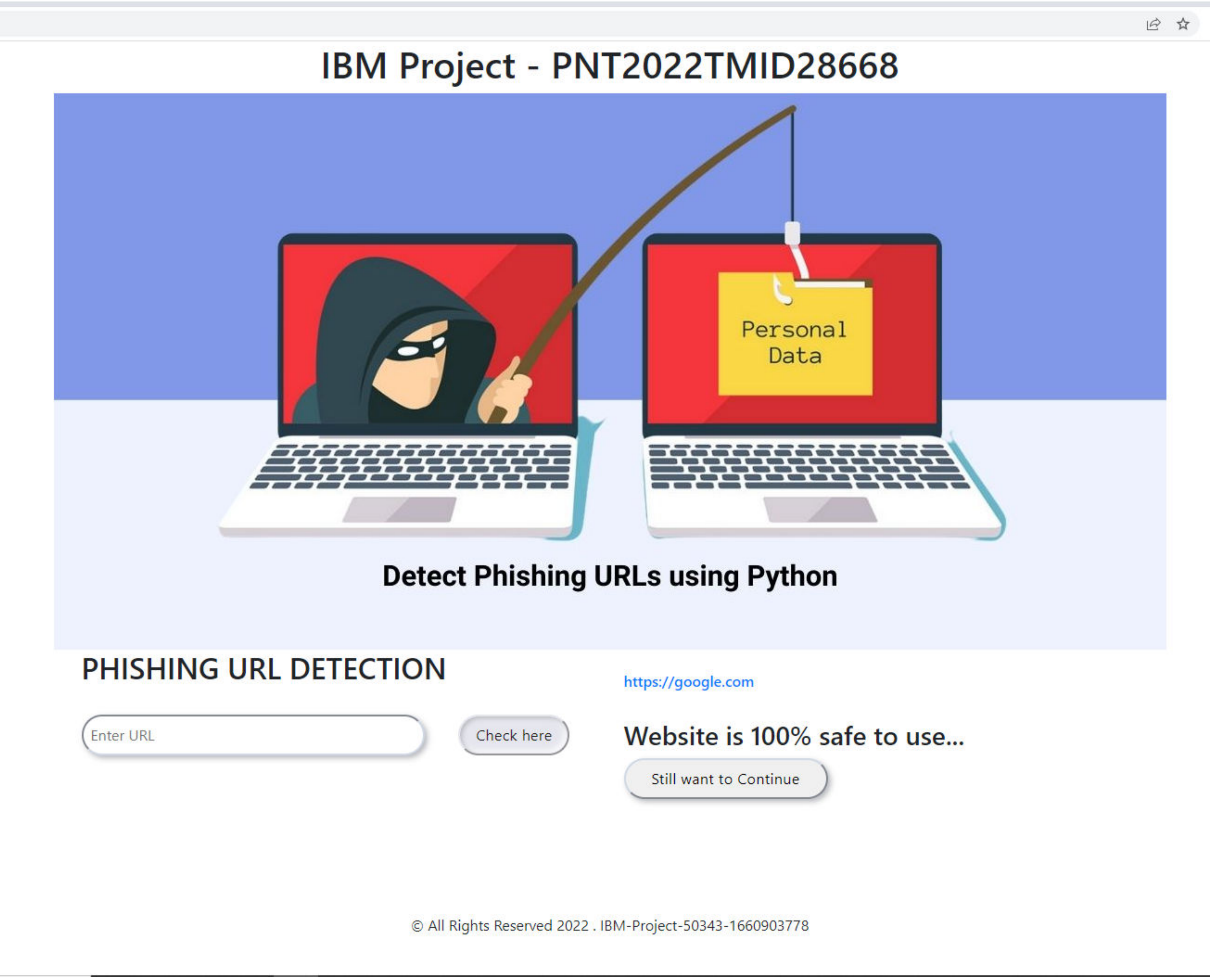
Detect Phishing URLs using Python

PHISHING URL DETECTION

https://google.com

Click this button
to show the result

© All Rights Reserved 2022 . IBM-Project-50343-1660903778



IBM Project - PNT2022TMID28668

PHISHING URL DETECTION

https://google.com

Website is 100% safe to use...

© All Rights Reserved 2022 . IBM-Project-50343-1660903778

Result : The website is not a phishing site

CHAPTER 10

ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- **Increases user alertness to phishing risks**Whenever the user navigates into the website and provide the URL of the website that needs to be verified for legitimacy, the system detects phishing sites by applying a machine learning algorithm which implements classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy which in turn helps the customers to eliminate the risks of cyber threat and protect their valuable corporate or personal data.

- **Users will also be able to pose any query to the admin through the report page designed** Our system is also provided with an option for the clients to report to the administrator which helps them to ask their questions significantlyimproving their experience on our site.

DISADVANTAGES:

- Not a generalized model
- Huge number of rules
- Needs feed continuously

CHAPTER 11

CONCLUSION

Phishing detection is now an area of great interest among the researchers due to its significance in protecting privacy and providing security. There are many methods to perform phishing detection. Our system aims to enhance the detection method to detect phishing websites using machine learning technology. We achieved a high detection accuracy, and the results show that the classifiers givebetter performance when we use more data as training data.

In future, hybrid technology willbe implemented to detect phishing websites more accurately.

CHAPTER 12

FUTURE SCOPE

In future we intend to build an add -ons for our system and if we get a structureddataset ofphishing,we can perform phishing detection much faster than any othertechnique. We can alsouse acombination of any two or more classifiers to getmaximum accuracy. We plan to explorevarious phishing techniques which use Network based features, Content based features,Webpagebasedfeatures and HTML and JavaScript features of web pages which willimprove theperformance of thesystem. In particular, we extract features from URLsandpass it throughthevarious classifiers.

APPENDIX

Source code:

app.py

```
# importing required libraries


from feature import FeatureExtraction

from flask import Flask, request, render_template

import numpy as np
import pandas as pd
from sklearn import metrics
import warnings
import pickle

warnings.filterwarnings('ignore')

file = open("model.pkl", "rb")

gbc = pickle.load(file)

file.close()

app = Flask(__name__)
@app.route("/", methods=["GET", "POST"])

def index():

    if request.method == "POST":


        url = request.form["url"]

        obj = FeatureExtraction(url)

        x = np.array(obj.getFeaturesList()).reshape(1, 30)

        y_pred = gbc.predict(x)[0]

        #1 is safe

        #-1 is unsafe

        y_pro_phishing = gbc.predict_proba(x)[0, 0]

y_pro_non_phishing = gbc.predict_proba(x)[0, 1]

        # if(y_pred ==1 ):

        pred = "It is {0:.2f} % safe to go ".format
(y_pro_phishing*100)

        return render_template('index.html', xx=round
(y_pro_non_phishing, 2), url=url)

    return render_template("index.html", xx=-1)

if __name__ == "__main__":

    app.run(debug=True, port=2002)
```

Feature.py

```
import ipaddress
import re
import urllib.request
from bs4 import BeautifulSoup
import socket
import requests
from googlesearch import search
import whois
from datetime import date, datetime
import time
from dateutil.parser import parse as date_parse
from urllib.parse import urlparse


class FeatureExtraction:
    features = []

    def __init__(self, url):
        self.features = []
        self.url = url
        self.domain = ""
        self.whois_response = ""
        self.urlparse = ""
        self.response = ""
        self.soup = ""

        try:
            self.response = requests.get(url)
            self.soup = BeautifulSoup(response.text,
'html.parser')
        except:
            pass

        try:
            self.urlparse = urlparse(url)
            self.domain = self.urlparse.netloc
        except:
            pass

        try:
            self.whois_response = whois.whois(self.domain)
        except:
            pass

        self.features.append(self.UsingIp())
        self.features.append(self.longUrl())
        self.features.append(self.shortUrl())
        self.features.append(self.symbol())
        self.features.append(self.redirecting())
        self.features.append(self.prefixSuffix())
        self.features.append(self.SubDomains())
        self.features.append(self.Hppts())
        self.features.append(self.DomainRegLen())
        self.features.append(self.Favicon())
  self.features.append(self.RequestURL())
        self.features.append(self.AnchorURL())
        self.features.append(self.LinksInScriptTags())
        self.features.append(self.ServerFormHandler())
        self.features.append(self.InfoEmail())
        self.features.append(self.AbnormalURL())
        self.features.append(self.WebsiteForwarding())
        self.features.append(self.StatusBarCust())
self.features.append(self.DisableRightClick())
          self.features.append(self.UsingPopupWindow())
          self.features.append(self.IframeRedirection())
          self.features.append(self.AgeofDomain())
          self.features.append(self.DNSRecording())
          self.features.append(self.WebsiteTraffic())
          self.features.append(self.PageRank())
          self.features.append(self.GoogleIndex())
          self.features.append(self.LinksPointingToPage())
          self.features.append(self.StatsReport())

       # 1.UsingIp

      def UsingIp(self):
          try:
              ipaddress.ip_address(self.url)
              return -1
          except:
              return 1

      # 2.longUrl
      def longUrl(self):
          if len(self.url) < 54:
              return 1
          if len(self.url) >= 54 and len(self.url) <= 75:
              return 0
          return -1

      # 3.shortUrl
      def shortUrl(self):
          match = re.search('bit\.ly|goo\.gl|shorte\.st|go2l\.ink|
  x\.co|ow\.ly|t\.co|tinyurl|tr\.im|is\.gd|cli\.gs|'
                            'yfrog\.com|migre\.me|ff\.im|tiny\.cc|
  url4\.eu|twit\.ac|su\.pr|twurl\.nl|snipurl\.com|'
                            'short\.to|BudURL\.com|ping\.fm|post
  \.ly|Just\.as|bkite\.com|snipr\.com|fic\.kr|loopt\.us|'
                            'doiop\.com|short\.ie|kl\.am|wp\.me|
  rubyurl\.com|om\.ly|to\.ly|bit\.do|t\.co|lnkd\.in|'
                            'db\.tt|qr\.ae|adf\.ly|goo\.gl|bitly
  \.com|cur\.lv|tinyurl\.com|ow\.ly|bit\.ly|ity\.im|'
                            'q\.gs|is\.gd|po\.st|bc\.vc|twitthis
  \.com|u\.to|j\.mp|buzurl\.com|cutt\.us|u\.bb|yourls\.org|'
                            'x\.co|prettylinkpro\.com|scrnch\.me|
  filoops\.info|vzturl\.com|qr\.net|1url\.com|tweez\.me|v\.gd|tr
  \.im|link\.zip\.net', self.url)
          if match:
              return -1
          return 1
```

```
# 4.Symbol@
        def symbol(self):
            if re.findall("@", self.url
    ):
                return -1
            return 1
        # 5.Redirecting//
        def redirecting(self):
            if self.url.rfind('//') >
    6:
            return -1
        return 1
    # 6.prefixSuffix
    def prefixSuffix(self):
        try:
            match = re.findall('\-'
, self.domain)
            if match:
                return -1
            return 1
        except:
            return -1
    # 7.SubDomains
    def SubDomains(self):
        dot_count = len(re.findall
("\.", self.url))
        if dot_count == 1:
            return 1
        elif dot_count == 2:
            return 0
        return -1
    # 8.HTTPS
    def Hppts(self):
        try:
            https =
self.urlparse.scheme
            if 'https' in https:
             return 1
         return -1
     except:
         return 1
         # 9.DomainRegLen
             def DomainRegLen
         (self):
                 try:
         expiration_date =
         self.whois_response.exp
         iration_date
         creation_date =
         self.whois_response.cre
         ation_date
                     try:
                         if(len
         (expiration_date)):
         expiration_date =
         expiration_date[0]
                     except:
                         pass
```

```
        # 10. Favicon
    def Favicon(self):
        try:
            for head in
self.soup.find_all('head'):
                for head.link in
self.soup.find_all('link', href=True):
                    dots = [x.start(0)
                            for x in
re.finditer('\.', head.link['href'])]
                    if self.url in
head.link['href'] or len(dots) == 1 or
domain in head.link['href']:
                        return 1
            return -1
        except:
            return -1

    # 11. NonStdPort
    def NonStdPort(self):
        try:
            port = self.domain.split(":"
)
            if len(port) > 1:
                return -1
            return 1
        except:
            return -1

    # 12. HTTPSDomainURL
    def HTTPSDomainURL(self):
        try:
            if 'https' in self.domain:
                return -1
            return 1
        except:
            return -1

    # 13. RequestURL
    def RequestURL(self):
        try:
            for img in
self.soup.find_all('img', src=True):
                dots = [x.start(0) for x
in re.finditer('\.', img['src'])]
                if self.url in img['src'
] or self.domain in img['src'] or len
(dots) == 1:
                    success = success +
1
                i = i+1


            for audio in
self.soup.find_all('audio', src=True):
                dots = [x.start(0) for x
in re.finditer('\.', audio['src'])]
                if self.url in audio
['src'] or self.domain in audio['src']
or len(dots) == 1:
                    success = success +
1
                i = i+1
```

```
# 15. LinksInScriptTags
        def LinksInScriptTags(self):
            try:
                i, success = 0, 0

                for link in
    self.soup.find_all('link', href=
    True):
                    dots = [x.start(0
    ) for x in re.finditer('\.', link
    ['href'])]
                    if self.url in
    link['href'] or self.domain in
    link['href'] or len(dots) == 1:
                        success =
    success + 1
                    i = i+1

                for script in
    self.soup.find_all('script', src=
    True):
                    dots = [x.start(0
    ) for x in re.finditer('\.',
    script['src'])]
                    if self.url in
    script['src'] or self.domain in
    script['src'] or len(dots) == 1:
                        success =
    success + 1
                    i = i+1

                try:
                    percentage =
    success / float(i) * 100
                    if percentage
    < 17.0:
                        return 1
                    elif((percentage
    >= 17.0) and (percentage < 81.0)
    ):
              _         return 0
                    else:
                        return -1
                except:
                    return 0
            except:
                return -1

        # 16. ServerFormHandler
        def ServerFormHandler(self):
            try:
                if len
    (self.soup.find_all('form',
    action=True)) == 0:
                    return 1
                else:
                    for form in
    self.soup.find_all('form', action
    =True):
                        if form
    ['action'] == "" or form['action
```

```
    # 17. InfoEmail
    def InfoEmail(self):
        try:
            if re.findall(r"
[mail\(\)|mailto:?]", self.soap
):
                return -1
            else:
                return 1
        except:
            return -1

    # 18. AbnormalURL
    def AbnormalURL(self):
        try:
            if
self.response.text ==
self.whois_response:
                return 1
            else:
                return -1
        except:
            return -1

    # 19. WebsiteForwarding
    def WebsiteForwarding(self):
        try:
            if len
(self.response.history) <= 1:
                return 1
            elif len
(self.response.history) <= 4:
                return 0
            else:
                return -1
        except:
            return -1

    # 20. StatusBarCust
    def StatusBarCust(self):
        try:
            if re.findall("
<script>.+onmouseover.+</script>
", self.response.text):
                return 1
            else:
                return -1
        except:
            return -1

    # 21. DisableRightClick
    def DisableRightClick(self):
        try:
            if re.findall(r"
event.button ?== ?2",
self.response.text):
                return 1
            else:
                return -1
        except:
            return -1
```

```
# 24. AgeofDomain
    def AgeofDomain(self):
        try:
            creation_date = self.whois_response.creation_date
            try:
                if(len(creation_date)):
                    creation_date = creation_date[0]
            except:
                pass

            today = date.today()
            age = (today.year-creation_date.year) * \
                12+(today.month-creation_date.month)
            if age >= 6:
                return 1
            return -1
        except:
            return -1

    # 25. DNSRecording
    def DNSRecording(self):
        try:
            creation_date = self.whois_response.creation_date
            try:
                if(len(creation_date)):
                    creation_date = creation_date[0]
            except:
                pass

            today = date.today()
            age = (today.year-creation_date.year) * \
                12+(today.month-creation_date.month)
            if age >= 6:
                return 1
            return -1
        except:
            return -1

    # 26. WebsiteTraffic
    def WebsiteTraffic(self):
        try:
            rank = BeautifulSoup(urllib.request.urlopen(
                "http://data.alexa.com/data?cli=10&dat=s&url=" + url).read(),
"xml").find("REACH")['RANK']
            if (int(rank) < 100000):
                return 1
            return 0
        except:
            return -1

    # 27. PageRank
    def PageRank(self):
        try:
            prank_checker_response = requests.post(
                "https://www.checkpagerank.net/index.php", {"name":
self.domain})

            global_rank = int(re.findall(
                r"Global Rank: ([0-9]+)", rank_checker_response.text)[0])
            if global_rank > 0 and global_rank < 100000:
                return 1
            return -1
        except:
            return -1

    # 28. GoogleIndex

    def GoogleIndex(self):
        try:
            site = search(self.url, 5)
            if site:
```

```
# 29. LinksPointingToPage
    def LinksPointingToPage(self):
        try:
            number_of_links = len(re.findall(r"<a href=", self.response.text))
            if number_of_links == 0:
                return 1
            elif number_of_links <= 2:
                return 0
            else:
                return -1
        except:
            return -1
    # 30. StatsReport
    def StatsReport(self):
        try:
            url_match = re.search(
                'at\.ua|usa\.cc|baltazarpresentes\.com\.br|pe\.hu|esy\.es|hol
\.es|sweddy\.com|myjino\.ru|96\.lt|ow\.ly', url)
            ip_address = socket.gethostbyname(self.domain)
            ip_match = re.search('146\.112\.61\.108|213\.174\.157\.151|121\.50
\.168\.88|192\.185\.217\.116|78\.46\.211\.158|181\.174\.165\.13|46\.242\.145
\.103|121\.50\.168\.40|83\.125\.22\.219|46\.242\.145\.98|'
                                 '107\.151\.148\.44|107\.151\.148\.107|64\.70
\.19\.203|199\.184\.144\.27|107\.151\.148\.108|107\.151\.148\.109|119\.28\.52
\.61|54\.83\.43\.69|52\.69\.166\.231|216\.58\.192\.225|'
                                 '118\.184\.25\.86|67\.208\.74\.71|23\.253\.126
\.58|104\.239\.157\.210|175\.126\.123\.219|141\.8\.224\.221|10\.10\.10\.10|43
\.229\.108\.32|103\.232\.215\.140|69\.172\.201\.153|'
                                 '216\.218\.185\.162|54\.225\.104\.146|103\.243
\.24\.98|199\.59\.243\.120|31\.170\.160\.61|213\.19\.128\.77|62\.113\.226\.131|
208\.100\.26\.234|195\.16\.127\.102|195\.16\.127\.157|'
                                 '34\.196\.13\.28|103\.224\.212\.222|172\.217\.4
\.225|54\.72\.9\.51|192\.64\.147\.141|198\.200\.56\.183|23\.253\.164\.103|52\.48
\.191\.26|52\.214\.197\.72|87\.98\.255\.18|209\.99\.17\.27|'
                                 '216\.38\.62\.18|104\.130\.124\.96|47\.89\.58
\.141|78\.46\.211\.158|54\.86\.225\.156|54\.82\.156\.19|37\.157\.192\.102|204
\.11\.56\.48|110\.34\.231\.42', ip_address)
            if url_match:
                return -1
            elif ip_match:
                return -1
            return 1
        except:
            return 1
    def getFeaturesList(self):
        return self.features
```

index.html

```
<!DOCTYPE html>
<html lang="en">


<head>
    <center>
        <h1> IBM Project - PNT2022TMID28668</h1>
    </center>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="This website is develop for identify the safety of 
url.">
    <meta name="keywords" content="phishing url,phishing,cyber security,machine learning,
classifier,python">


    <!-- BootStrap -->
    <link rel="stylesheet" href="
https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"
        integrity="sha384-9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk"
crossorigin="anonymous">

    <link href="https://drive.google.com/uc?export=download&id=1s673-
y2tlCRny0qx50Wiyi8HqUJ1o4bV" rel="stylesheet">
    <title>URL detection12</title>
    <link rel="shortcut icon" href="{{ url_for('static', filename='favicon.ico') }}">

</head>

<body>
    <center> <img class="image image-contain"
            src="https://cdn.activestate.com/wp-content/uploads/2021/02/phishing-detection-
with-Python.jpg"
            alt="MDN logo" /> </center>

    <div class=" container">
        <div class="row">
            <div class="form col-md" id="form1">
                <h2>
                    PHISHING URL DETECTION
                </h2>

                <br>
                <form action="/" method="post">
                    <input type="text" class="form__input" name='url' id="url" placeholder=
"Enter URL" required="" />
                    <label for="url" class="form__label">URL</label>
                    <button class="button" role="button">Check here</button>
                </form>
       <div class="col-md" id="form2">

                <br>
                <h6 class="right "><a href={{ url }} target="_blank">{{ url }}</a></h6>

                <br>
                <h3 id="prediction"></h3>
                <button class="button2" id="button2" role="button" onclick="window.open('
{{url}}')"
                    target="_blank">Still want to Continue</button>

            </div>
        </div>
        <br>
    </div>

    <!-- JavaScript -->
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"
        integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+
OrCXaRkfj"
        crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
        integrity="sha384-Q6E9RHvbIyZFJoft+
2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
        crossorigin="anonymous"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"
        integrity="sha384-OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+
6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI"
        crossorigin="anonymous"></script>


    <script>

        let x = '{{xx}}';
        let num = x * 100;
        if (0 <= x && x < 0.50) {
            num = 100 - num;
        }
        let txtx = num.toString();
        if (x <= 1 && x >= 0.50) {
            var label = "Website is " + txtx + "% safe to use...";
            document.getElementById("prediction").innerHTML = label;
            document.getElementById("button1").style.display = "block";
        }
        else if (0 <= x && x < 0.50) {
            var label = "Website is " + txtx + "% unsafe to use..."
            document.getElementById("prediction").innerHTML = label;
            document.getElementById("button2").style.display = "block";
        }

    </script>

</body>
<footer>
    <center>
        <p>© All Rights Reserved 2022 . IBM-Project-50343-1660903778</p>
    </center>
</footer>

</html>
```

```
  position: relative;

  transition: all 0.2s ease;

  border-color: #D1D9E6;

  box-shadow: 3px 3px 6px #b8b9be, -3px -3px 6px #ffffff;

  padding: 8px;

  border-radius: 25px;

  width: 350px;


}


.form__input:placeholder-shown + .form__label {

  opacity: 0;                                      -1

  visibility: hidden;

  -webkit-transform: translateY(+4rem);

  transform: translateY(+4rem);

}




button {

  position: relative;

  transition: all 0.2s ease;

  letter-spacing: 0.025em;

  font-size: 1rem;

  border-color: #D1D9E6;

  box-shadow: 3px 3px 6px #b8b9be, -3px -3px 6px #ffffff;

  padding: 8px;

  border-radius: 25px;
  width: 100px;
}

button:hover {
  color: #161616;
    background-color: #e6e7ee;
    border-color: #e6e7ee;
    box-shadow: inset 2px 2px 5px #b8b9be, inset -3px -
3px 7px #ffffff;
}
```

```
button:active {
          transform: translateY(0.125rem);
          border: none;
        }

        .button:focus {
          color: #44476A;
            background-color: #e6e7ee;
            border-color: #D1D9E6;
            outline: 0;
            box-shadow: inset 2px 2px 5px #b8b9be, inset -3px -3px 7px #FFFFFF, none;
        }


        .main-body{
          display: flex;
          flex-direction: row;
          width: 75%;
          justify-content:space-around;
        }

        .button2{
          width: 200px;

        }
        .button1{
          width: 100px;

        }

        .right {
          right: 0px;
          width: 300px;
        }
        footer{
         padding: 100px;
        }
        @media (max-width: 576px) {
          .form {
            width: 100%;
          }
         }
        .abc{
          width: 50%;
        }
```

LINKS

GitHub link: https://github.com/IBM-EPBL/IBM-Project-50343-1660903778

Demo link: https://drive.google.com/file/d/1EJMFAUPgiLtATfK4KQDWMBxVtadAWrka/view?usp=share_link