**HINDUSTHAN INSTITUTE OF TECHNOLOGY**

**(An Autonomous Institution, Approved by AICTE, New Delhi, Affiliated to Anna University, Chennai, Accredited with “A” Grade by NAAC) Valley Campus, Pollachi Main Road, Coimbatore 641 032.**

**DEPARTMENT COMPUETR SCIENCE AND ENGINEERING**

**REPORT ON**

**HX 8001 PROFESSIONAL READINESS FOR INNOVATION,**

**EMPLOYABILITY AND ENTREPRENEURSHIP**

**(Naalaiya Thiran Program)**

**PROJECT TITLE**

**WEB PHISHING DETECTION**

**TEAM ID: PNT2022TMID10436**

**TEAM MEMBERS MENTOR**

1. M SIVANESAN (TEAM LEADER) Mr. SURESHKUMAR P

2. MOHAMMED NISHAD S **EVALUATOR**

3. RUSSEL GABRIEL THOMAS Mr. BIJU BALAKRISHNAN

4. PRANAV P

1. **INTRODUCTION**

**1.1 Project Overview:**

Phishing costs Internet users billions of dollars per year. It refers to luring techniques used by identity thieves to fish for personal information in a pond of unsuspecting Internet users. Phishers use spoofed e-mail, phishing software to steal personal information and financial account details such as usernames and passwords. This paper deals with methods for detecting phishing Web sites by analyzing various features of benign and phishing URLs by Machine learning techniques. We discuss the methods used for detection of phishing Websites based on lexical features, host properties and page importance properties. We consider various machine learning algorithms for evaluation of the features in order to get a better understanding of the structure of URLs that spread phishing. The fine-tuned

parameters are useful in selecting the apt machine learning algorithm for separating the phishing sites from benign sites. The criminals, who want to obtain sensitive data, first create unauthorized replicas of a real website and e-mail, usually from a financial institution or another company that deals with financial information. The e-mail will be created using logos and slogans of a legitimate company. The nature of website creation is one of the reasons that the Internet has grown so rapidly as a communication medium, it also permits the abuse of trademarks, trade names, and other corporate identifiers upon which consumers have come to rely as mechanisms for authentication. Phisher then send the "spoofed" e-mails to as many people

as possible in an attempt to lure them in to the scheme. When these e-mails are opened or when a link in the mail is clicked, the consumers are redirected to a spoofed website, appearing to be from the legitimate entity.

The main purpose of the project is to detect the fake or phishing websites who are trying to get access to the sensitive data or by creating the fake websites and trying to get access of the user personal credentials. We are using machine learning algorithms to safeguard the sensitive data and to detect the phishing websites who are trying to gain access on sensitive data.

1. **LITERATURE SURVEY:**

**2.1 Existing Problem:**

The purpose or goal behind phishing is data, money or personal information stealing through the fake website. The best strategy for avoiding the contact with the phishing website is to detect real time malicious URL. Phishing websites can be determined on the basis of their domains. They usually are related to URL which needs to be registered (low-level domain and upper-level domain, path, query). Recently acquired status of intra-URL relationship is used to evaluate it using distinctive properties extracted from words that compose a URL based on query data from various search engines such as Google and Yahoo. These properties are further led to the machine-learning based classification for the identification of phishing URLs from a real dataset. This paper focus on real time URL phishing against phishing content by using phish-STORM. For this a few relationship between the register domain rest of the URL are consider also intra URL relentless is

consider which help to dusting wish between phishing or non-phishing URL. For detecting a phishing website certain typical blacklisted urls are used, but this technique is unproductive as the duration of phishing websites is very short. Phishing is the name of avenue. It can be defined as the manner of deception of an organization's customer to communicate with their confidential information in an unacceptable behavior. It can also be defined as intentionally using harsh weapons such as Spasm to automatically target the victims and targeting their private information. As many of the failures being occurred in the SMTP are exploiting vectors for the phishing websites, there is a greater availability of communication for malicious message deliveries. Proposed a novel classification approach that use heuristic based feature extraction approach. In this, they have classified extracted features into different categories such as URL Obfuscation features, Hyperlink-based features. Moreover, proposed technique gives 92.5% accuracy. Also this model is purely depends on the quality and quantity of the training set and Broken links feature extraction.

**2.2 References:**

• Liu J, Ye Y (2001) Introduction to E-business operators: commercial center arrangements,

security issues, and market interest. In: E-business specialists, commercial

centerarrangements, security issues, and market interest, London, UK

• APWG, Aaron G, Manning R (2013) APWG phishing reports. APWG, 1 February 2013.

[Online]. Accessible: [http://www](http://www/).antiphishing.org/assets/apwg-reports/. Gotten to 8

Feb2013

• Kaspersky Lab (2013) Spam in January 2012: love, governmental issues and game.

[Online].

Available:http://www.kaspersky.com/about/news/spam/2012

Spam\_in\_January\_2012\_Love\_Politics\_and\_ Sport. Gotten to 11 Feb 2013

• Seogod (2011) Black Hat SEO. Search engine optimization Tools. [Online].

Accessible:http://www.seobesttools.com/dark cap website optimization/. Gotten to 8 Jan

2013

• Dhamija R, Tygar JD, Hearst M (2006) Why phishing works. In: Proceedings of the

SIGCHImeeting on human factors in figuring frameworks, Cosmopolitan Montre &#39;al,

Canada

• Cranor LF (2008) A system for thinking about the human tuned in. In: UPSEC&#39;08

Proceedings of the first meeting on ease of use, brain science, and security, Berkeley,

CA,USA

• Miyamoto D, Hazeyama H, Kadobayashi Y (2008) An assessment of AI based techniquesfor

recognition of phishing destinations. Aust J Intell Inf Process Syst 10(2):54–6

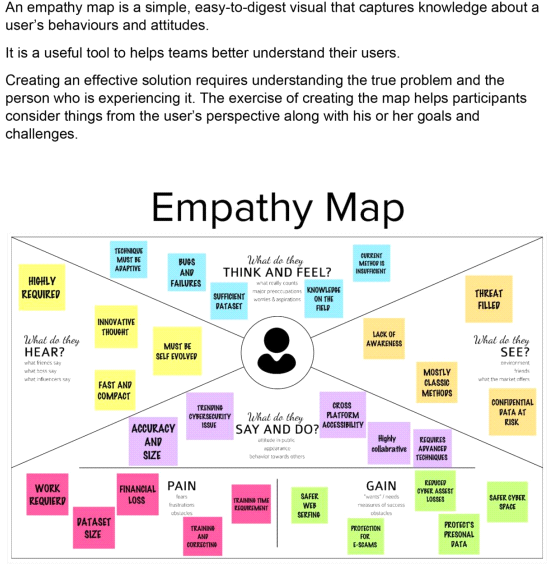
• Xiang G, Hong J, Rose CP, Cranor L (2011) CANTINA?:a include rich AI structure for identifying phishing sites. ACM Trans Inf Syst Secur 14(2):1–28

* 1. **Problem Statement Definition:**

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 tunneling, 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.

1. **IDEATION & PROPOSED SOLUTION:**

**3.1 Empathy Map Canvas:**



1. **Ideation & Brainstorming:**

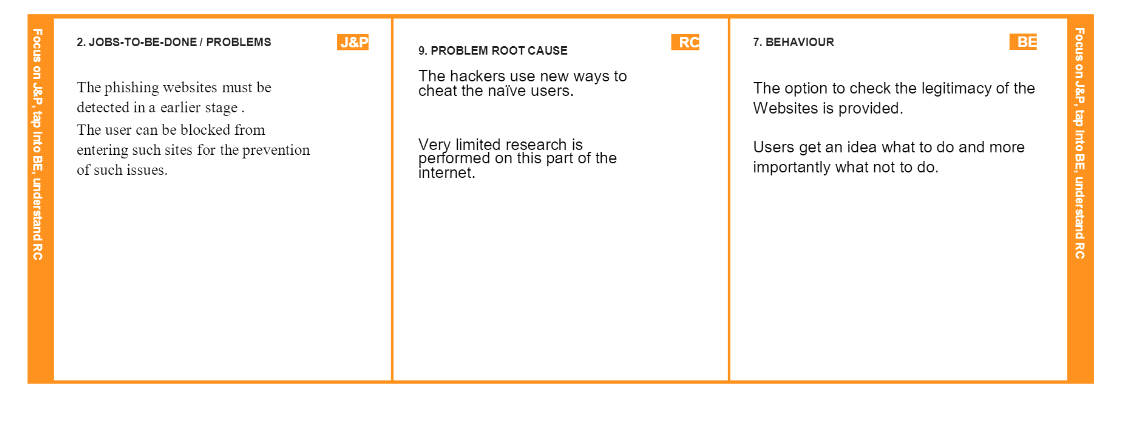


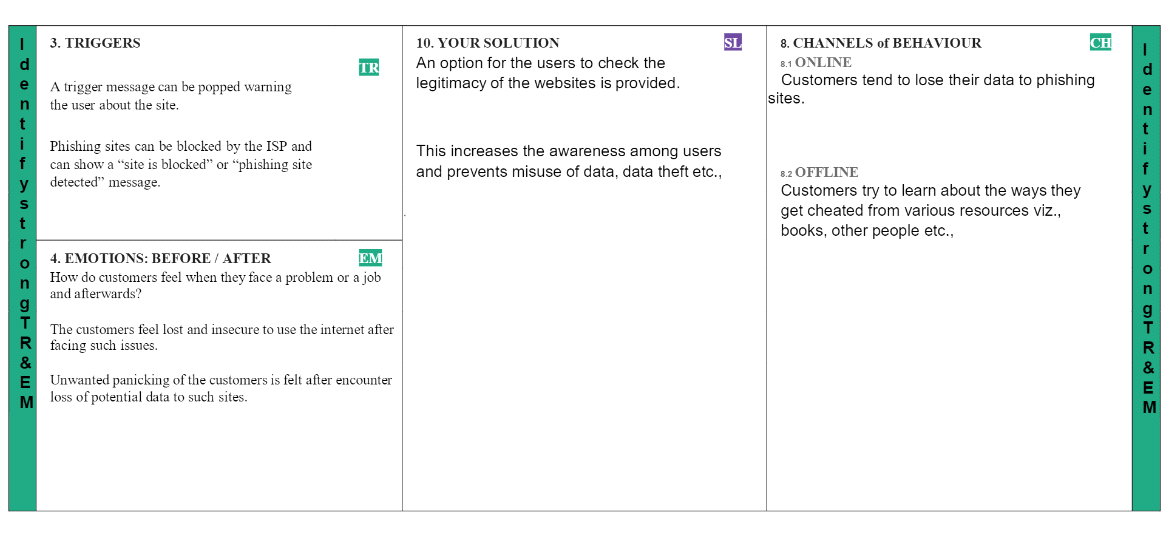
1. **Proposed Solution:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S no.** | **Parameter** | | **Description** | |
| 1. | Problem Statement | | An online user needed to purchase something through an online. So he entered into the online website through internet. It takes some time to display the product. He started to see all the products. He search the necessary things in online website. At last he found the needed products. After that he entered all the credit card details, username and password for purchasing the things through online. Then he received the message "Your order is placed and transaction is successfully completed. You will receive the ordered product within 2 days”. After that within 24 hours he got a message in mobile and the bank account was empty then the customer shocked . Then only he realized that was a fake website and his bank account details was stolen by hacker .To avoid this scenario. We need to solve this problem by using the Web Phishing Detection. | |
| 2. | Solution description | | To overcome the problem of phishing website whenever we are clicking on one website it must show an alert box like it is a secure website or it is not a secure website Then another way is that we can scan the website in order to prevent our system or mobile from the phishing attack. Even though technologies are there we as the user have to be aware of the websites whether it is secure or not. We should not click any unwanted websites. | |
| 3. | Uniqueness | | The proposed approach has divided the hyperlink specific features into 12 different categories and used these features to train the machine learning algorithms. We have evaluated the performance of our proposed phishing detection approach on various classification algorithms using the phishing and non-phishing websites dataset. | |
| 4. | | Customer Satisfaction | | As we are using some websites but while clicking that website it display an alert box which leads to an aware of the customer which results in satisfaction of the user while using the websites, And another way is that we can scan the website in order to prevent the hacking of the information which makes even more satisfaction to the customer. | |

**3 Problem Solution Fit:**







1. **REQUIRED ANALYSIS :**

**4.1 Functional Requirement:**

A function of software system is defined in functional requirement and the behavior of the

system is evaluated when presented with specific inputs or conditions which may include

calculations, data manipulation and processing and other specific functionality.

• Our system should be able to load air quality data and preprocess data.

• It should be able to analyze the air quality data.

• It should be able to group data based on hidden patterns.

• It should be able to assign a label based on its data groups.

• It should be able to split data into trainset and testset.

• It should be able to train model using trainset.

• It must validate trained model using testset.

• It should be able to display the trained model accuracy.

• It should be able to accurately predict the air quality on unseen data.

* 1. **Non-Functional Requirements:**

Nonfunctional requirements describe how a system must behave and establish constraints of its functionality. This type of requirements is also known as the system’s quality attributes. Attributes such as performance, security, usability, compatibility are not the feature of the system, they are a required characteristic. They are "developing" propertiesthat emerge from the whole arrangement and hence we can't compose a particular line of code to execute them. Any attributes required by the customer are described by the specification. We must include only those requirements that are appropriate for our project.

Some Non-Functional Requirements are as follows:

• Reliability

• Maintainability

• Performance

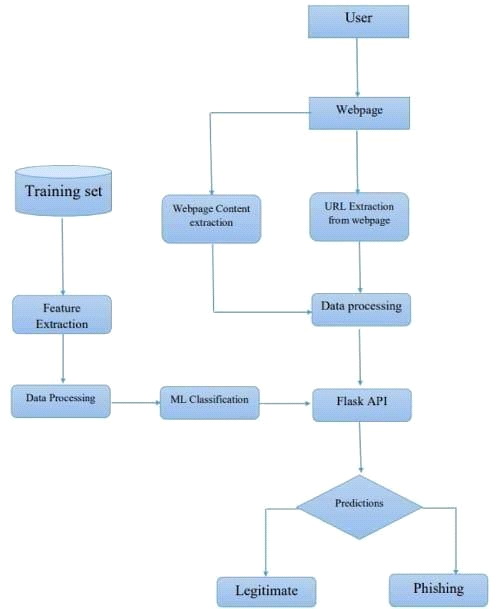
• Portability

• Scalability

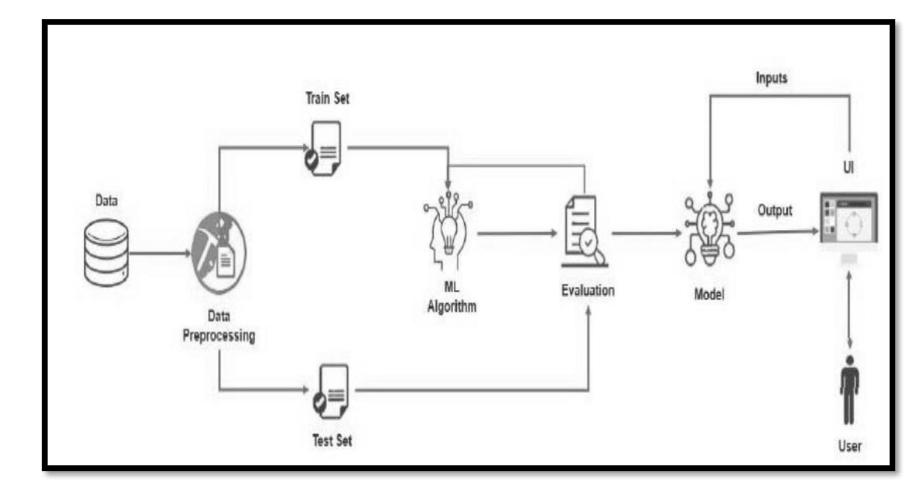
• Flexibility

1. **PROJECT DESIGN :**

**5.1 Data Flow Diagrams :**



* 1. **Solution & Technical Architecture :**



* 1. **User Stories :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| Customer (Mobile user) | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | I can access my account/ dashboard | High | Sprint-1 |
|  |  | USN-2 | As a user, I will receive confirmation email once I have registered for the application | I can receive confirmation email & click confirm | High | Sprint-1 |
|  |  | USN-3 | As a user, I can register for the application through Facebook | I can register & access the dashboard with Facebook  Login | Low | Sprint-2 |
|  |  | USN-4 | As a user, I can register for the application  through Gmail |  | Medium | Sprint-1 |
|  | Login | USN-5 | As a user, I can log into the application by entering email & password |  | High | Sprint-1 |
|  | Dashboard |  |  |  |  |  |
| Customer (Web  user) | User input | USN-1 | As a user i can input the particular URL in the required field and waiting for validation. | I can go access the website without any problem | High | Sprint-1 |
| Customer Care Executive | Feature extraction | USN-1 | After i compare in case if none found on comparison then we can extract feature using  heuristic and visual similarity approach. | As a User i can have comparison between websites  for security. | High | Sprint-1 |
| Administrator | Prediction | USN-1 | Here the Model will predict the URL websites using  Machine Learning algorithms such as Logistic Regression, KNN | In this i can have correct  prediction on the particular algorithms | High | Sprint-1 |
|  | Classifier | USN-2 | Here i will send all the model output to classifier in order to produce final result. | I this i will find the correct classifier for producing the result | Medium | Sprint-2 |

1. **PROJECT PLANNING & SCHEDULING :**
   1. **Sprint Planning & Delivery :**

Product backlog and sprint schedule:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement (Epic)** | **User**  **Story Number** | | **User Story / Task** | **Story Points** | **Priority** | **Team Members** | |  |  |
| Sprint-1 | Homepage | USN-1 | | As a user, I can explore the resources of thehomepage for the functioning | 10 | Low | Sivanesan, Mohammed Nishad | |  |  |
| Sprint-1 |  | USN-2 | | As a user, I can learn about the various sides  of the web phishing and be aware of the scams | 5 | High | Sivanesan , Russel | |  |  |
| Sprint-2 | Final page | USN-3 | | As a user, I can explore the resources of the final page for the functioning | 15 | Low | Mohammed Nishad, Pranav | |  |  |
| Sprint-3 | Prediction | USN-4 | | As a user, I can predict the URL easily for detecting whether the website is legitimate or not | 10 | High | Russel,Pranav | |  |  |
| Sprint-4 | Chat | USN-5 | | As a user, I can share the experience or contact the admin for the support | 10 | High | Sivanesan, Mohammed Nishad, Russel | |  |  |
| Sprint-1 | Homepage | USN-6 | | As a admin, we can design interface and maintain the functioning of the  website | 5 | High | Pranav,Russel | |  |  |
| Sprint-2 | Final page | USN-7 | | As a admin, we can design the complexity of the website for making it user-friendly | 5 | Medium | Mohammed Nishad, Sivanesan | |  |  |
| Sprint-3 | Prediction | USN-8 | | As a admin, we can use various ML classifier model for the accurate result for the detection of URL | 10 | High | Mohammed Nishad, Sivanesan, Russel, Pranav | |  |  |
| Sprint-4 |  | USN-9 | As a admin, we can response to the usermessage for improvement of the website | | 10 | Medium | | Sivanesan, Mohammed Nishad | |

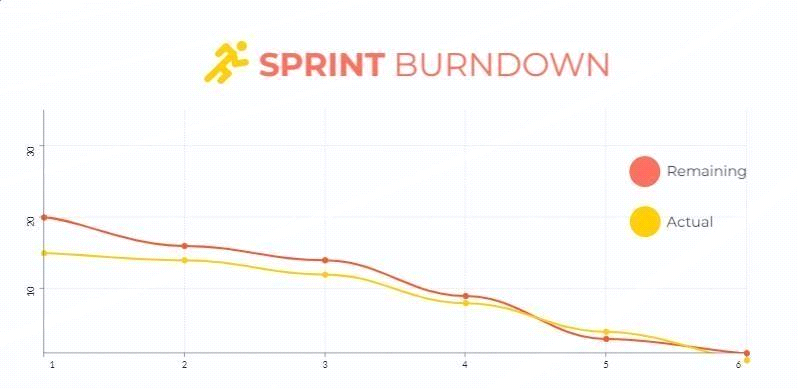
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date(Planned)** | **Story Points Completed (as on Planned End**  **Date)** | **Sprint Release Date(Actual)** |
| Sprint-1 | 20 | 6 Days | 24 Oct 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 |

**Project Tracker, Velocity & Burn down Chart**

**6.2 Reports from JIRA**

**Burn-down Chart**

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time



**7 CODING & SOLUTION :**

**7.1 Feature 1:**

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.NonStdPort())

self.features.append(self.HTTPSDomainURL())

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.expiration\_date

creation\_date = self.whois\_response.creation\_date

try:

if(len(expiration\_date)):

expiration\_date = expiration\_date[0]

except:

pass

try:

if(len(creation\_date)):

creation\_date = creation\_date[0]

except:

pass

age = (expiration\_date.year-creation\_date.year)\*12+ (expiration\_date.month-creation\_date.month)

if age >=12:

return 1

return -1

except:

return -1

# 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

for embed in self.soup.find\_all('embed', src=True):

dots = [x.start(0) for x in re.finditer('\.', embed['src'])]

if self.url in embed['src'] or self.domain in embed['src'] or len(dots) == 1:

success = success + 1

i = i+1

for iframe in self.soup.find\_all('iframe', src=True):

dots = [x.start(0) for x in re.finditer('\.', iframe['src'])]

if self.url in iframe['src'] or self.domain in iframe['src'] or len(dots) == 1:

success = success + 1

i = i+1

try:

percentage = success/float(i) \* 100

if percentage < 22.0:

return 1

elif((percentage >= 22.0) and (percentage < 61.0)):

return 0

else:

return -1

except:

return 0

except:

return -1

# 14. AnchorURL

def AnchorURL(self):

try:

i,unsafe = 0,0

for a in self.soup.find\_all('a', href=True):

if "#" in a['href'] or "javascript" in a['href'].lower() or "mailto" in a['href'].lower() or not (url in a['href'] or self.domain in a['href']):

unsafe = unsafe + 1

i = i + 1

try:

percentage = unsafe / float(i) \* 100

if percentage < 31.0:

return 1

elif ((percentage >= 31.0) and (percentage < 67.0)):

return 0

else:

return -1

except:

return -1

except:

return -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'] == "about:blank":

return -1

elif self.url not in form['action'] and self.domain not in form['action']:

return 0

else:

return 1

except:

return -1

# 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

# 22. UsingPopupWindow

def UsingPopupWindow(self):

try:

if re.findall(r"alert\(", self.response.text):

return 1

else:

return -1

except:

return -1

# 23. IframeRedirection

def IframeRedirection(self):

try:

if re.findall(r"[<iframe>|<frameBorder>]", 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:

return 1

else:

return -1

except:

return 1

# 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

**8 Testing:**

**8.1 User Acceptance Testing:**

* **Purpose of Document**

The purpose of this document is to briefly explain the test coverage and open issues of the [Web Phishing Detection] project at the time of the release to User Acceptance Testing (UAT).

* **Defect Analysis**

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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**

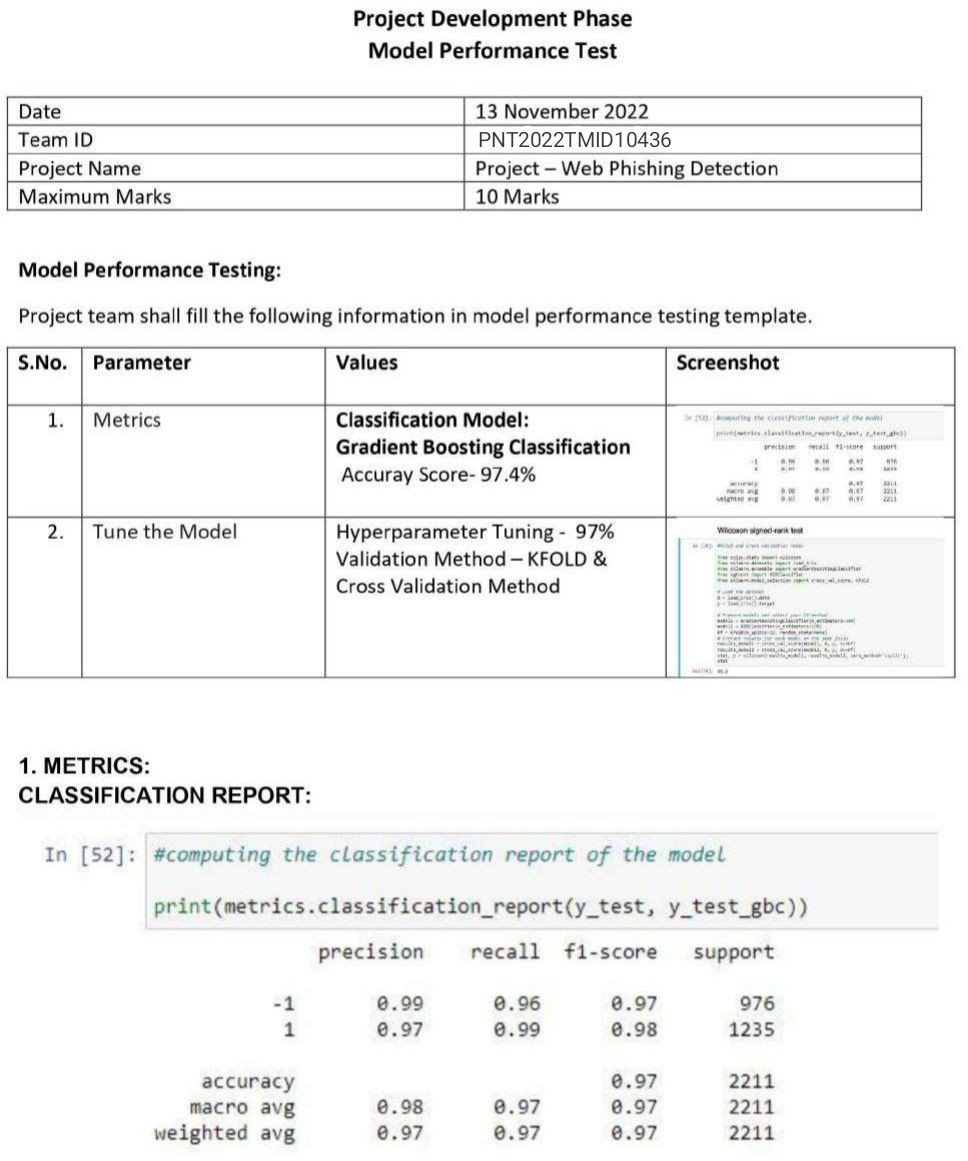
This report shows the number of test cases that have passed, failed, and untested

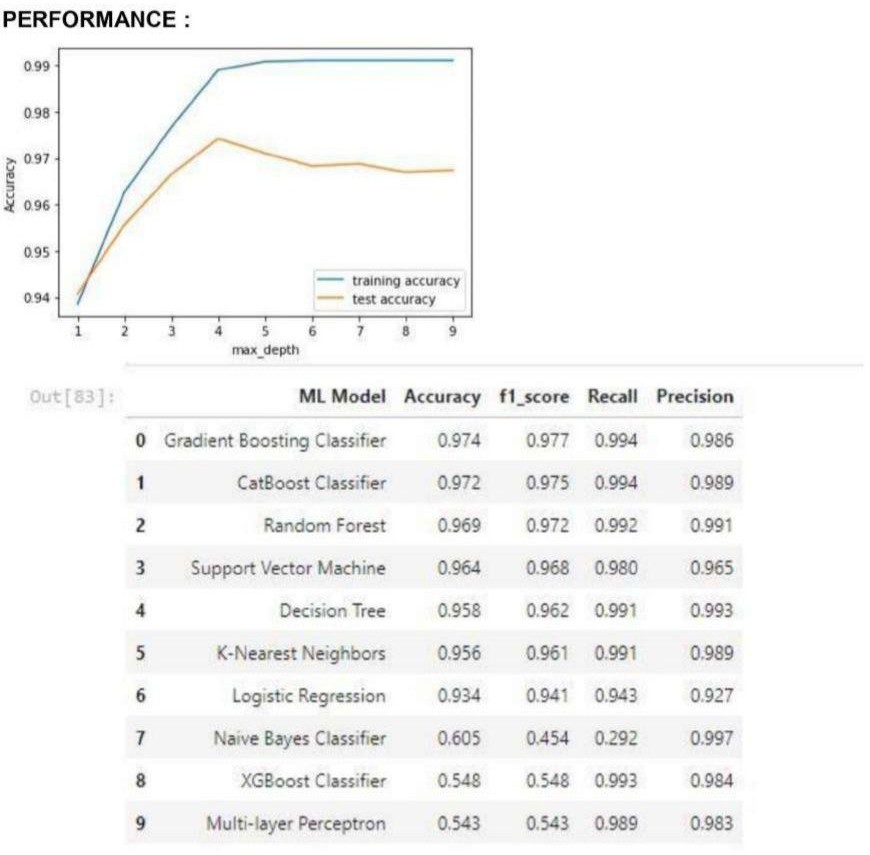
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Section** | **Total Cases** | **Not Tested** | **Fail** | **Pass** |
| Print Engine | 10 | 0 | 0 | 10 |
| Client Application | 50 | 0 | 0 | 50 |
| Security | 5 | 0 | 0 | 4 |
| Outsource Shipping | 3 | 0 | 0 | 3 |

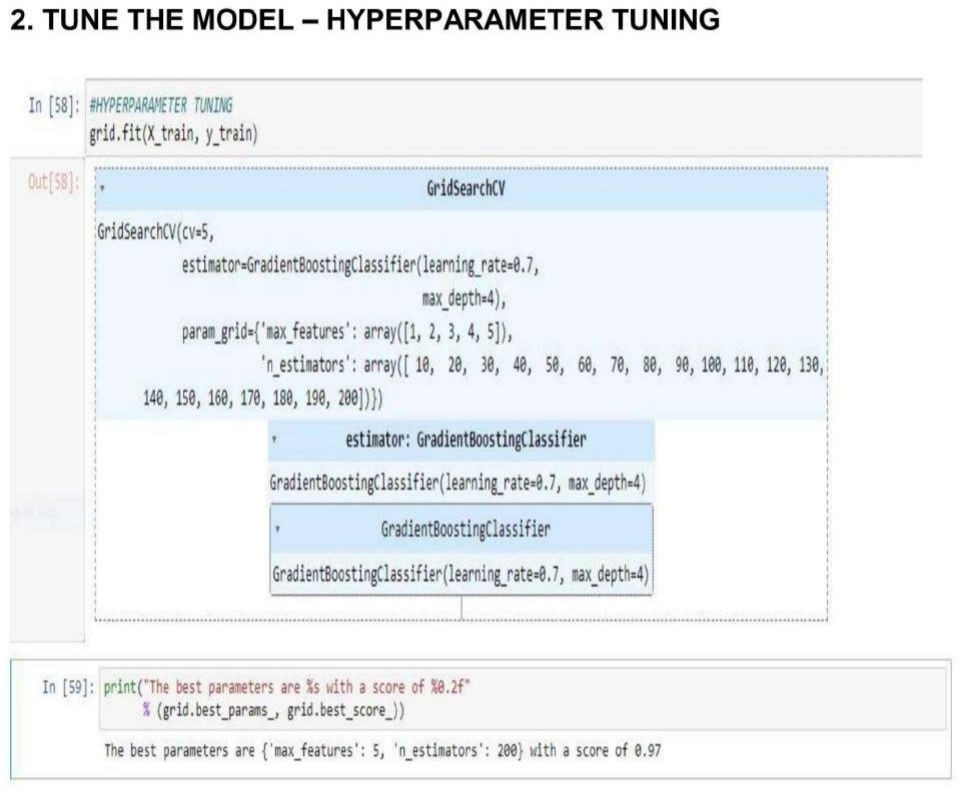
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Exception Reporting | 10 | 0 | 0 | 9 |
| Final Report Output | 10 | 0 | 0 | 10 |

**9 RESULTS**

**9.1 Performance Metrics:**







**10 ADVANTAGES & DISADVANTAGES**

**Advantages**

1. Improve on Inefficiencies of SEG and Phishing Awareness Training.
2. It Takes a Load off the Security Team.

###### It Offers a Solution, Not a Tool.

###### Separate You from Your Competitors.

1. Eliminate the cyber threat risk level.
2. Increase user alertness to phishing risks.
3. Instill a cyber security culture and create cyber security heroes.

###### Disadvantages

1. Scammers use “Bayesian poisoning” technique to circumvent Bayesian content filtering.
2. The Accuracy is not 100% correct because it is depended on unsupervised learning.
3. Sometimes it detects the wrong webpage.
4. Time consuming.
5. Non-Standard classifier.
6. Need feed continuously.

**11 CONCLUSION**

As we have seen, phishing is an increasing security concern, which due to a lack of knowledge on the primary way it is spread, impacts all users including those with extensive training experience on the subject. To tackle this problem, my project was to develop a phishing learning and detection system, to answer the question: How might we develop a phishing learning and detection tool that will protect from, and inform users about, malicious URLs? I chose to develop the system as a browser extension to filter all of a user’s encountered URLs and chose Chrome as the browser to develop for due to its dominant market share.  
The system was developed for above-average technical users since these are shown to be the most beneficial group to teach about computer security matters. To better understand how these users would like to interact with such a system, I conducted interviews with 17 participants. As a result of these interviews and a prior literature review, the following requirements were able to be defined:  
1. Classifying every URL according to one of three states as part of a traffic light system   
2. Using the browser to prevent the user from visiting any URLs that have been classified as malicious without explicit user consent  
3. Presenting the information on each URL to the user in an understandable way, both at appropriate points of intervention and on user request  
4. That the end built system includes the best practice of software engineering: maximizing efficiency and accuracy  
From this basis, I outlined the design of a system, including the design of a URL analysis algorithm, which was subsequently evaluated by an expert in the field. This allowed me to develop the user interface and infrastructure of the system and complete the goals outlined for the project this year.  
The system was evaluated with regards to its usability, and how well it met the system requirements. The goal was to assess how suitable and easy to use this system is for analyzing the details of malicious URLs and preventing users from visiting them. To do this, I conducted a survey of 43 participants who all filled out the SUS survey, eight Think Aloud and a further expert evaluation. In conclusion, it would seem that my system is usable and not overly complicated, which fulfills the goals that were intended to be achieved in the implementation of the user interaction and system design.

**12 FUTURE SCOPE**

Phishing is a considerable problem differs from the other security threats such as intrusions and Malware which are based on the technical security holes of the network systems. The weakness point of any network system is its Users. Phishing attacks are targeting these users depending on the trikes of social engineering. Despite there are several ways to carry out these attacks, unfortunately the current phishing detection techniques cover some attack vectors like email and fake websites. Therefore, building a specific limited scope detection system will not provide complete protection from the wide phishing attack vectors. This paper develops detection system with a wide protection scope using URL features only which is relying on the fact that users directly deal with URLs to surf the internet and provides a good approach to detect malicious URLs as proved by previous studies. Additionally, Anti-phishing solutions can be positioned at different levels of attack flow where most researchers are focusing on client side solutions which turn to add more processing overhead at the client side and lead to losing the trust and satisfaction of the users. Nowadays many organizations make centralized protection of spam filtering. This paper proposes a system which can be integrated into such process in order to increase the detection performance in a real time. The simulation results of the proposed system showed a phishing URLs detection accuracy with 93% and provided online process of a single URL in average time of 0.12 second.

**13. APPENDIX**

**SOURCE CODE:**

#importing required libraries

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')

from feature import FeatureExtraction

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)

**HTML Code:**

<!DOCTYPE html>

<html lang="en">

<head>

<center> <h1> IBM Project Based Learning </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">

<meta name="author" content="Balajee A V">

<!-- 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="static/styles.css" rel="stylesheet">

<title>URL detection</title>

</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>

<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>

<button class="button1" id="button1" role="button" onclick="window.open('{{url}}')" target="\_blank">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>© Sivanesan M</p> </center>

</footer>

</html>

**CSS code:**

\*,

\*::after,

\*::before {

margin: 0;

padding: 0;

box-sizing: inherit;

font-size: 62,5%;

}

.image {

width: 500px;

height: 500px;

}

.image-contain {

object-fit: contain;

object-position: center;

}

.image-cover {

object-fit: cover;

object-position: center;

}

body {

padding: 10% 5%;

background: #0f2027;

background: linear-gradient(to right,#2c5364, #203a43, ##55FFFF);

justify-content: center;

align-items: center;

height: 100vh;

color: #fff;

}

.form\_\_label {

font-family: 'Roboto', sans-serif;

font-size: 1.2rem;

margin-left: 2rem;

margin-top: 0.7rem;

display: block;

transition: all 0.3s;

transform: translateY(0rem);

}

.form\_\_input {

top: -24px;

font-family: 'Roboto', sans-serif;

color: #333;

font-size: 1.2rem;

padding: 1.5rem 2rem;

border-radius: 0.2rem;

background-color: rgb(255, 255, 255);

border: none;

width: 75%;

display: block;

border-bottom: 0.3rem solid transparent;

transition: all 0.3s;

}

.form\_\_input:placeholder-shown + .form\_\_label {

opacity: 0;

visibility: hidden;

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

transform: translateY(+4rem);

}

.button {

appearance: button;

background-color: transparent;

background-image: linear-gradient(to bottom, #fff, #f8eedb);

border: 0 solid #e5e7eb;

border-radius: .5rem;

box-sizing: border-box;

color: #482307;

column-gap: 1rem;

cursor: pointer;

display: flex;

font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";

font-size: 100%;

font-weight: 700;

line-height: 24px;

margin: 0;

outline: 2px solid transparent;

padding: 1rem 1.5rem;

text-align: center;

text-transform: none;

transition: all .1s cubic-bezier(.4, 0, .2, 1);

user-select: none;

-webkit-user-select: none;

touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);

}

.button:active {

background-color: #f3f4f6;

box-shadow: -1px 2px 5px rgba(81,41,10,0.15),0px 1px 1px rgba(81,41,10,0.15);

transform: translateY(0.125rem);

}

.button:focus {

box-shadow: rgba(72, 35, 7, .46) 0 0 0 4px, -6px 8px 10px rgba(81,41,10,0.1), 0px 2px 2px rgba(81,41,10,0.2);

}

.main-body{

display: flex;

flex-direction: row;

width: 75%;

justify-content:space-around;

}

.button1{

appearance: button;

background-color: transparent;

background-image: linear-gradient(to bottom, rgb(160, 245, 174), #37ee65);

border: 0 solid #e5e7eb;

border-radius: .5rem;

box-sizing: border-box;

color: #482307;

column-gap: 1rem;

cursor: pointer;

display: flex;

font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";

font-size: 100%;

font-weight: 700;

line-height: 24px;

margin: 0;

outline: 2px solid transparent;

padding: 1rem 1.5rem;

text-align: center;

text-transform: none;

transition: all .1s cubic-bezier(.4, 0, .2, 1);

user-select: none;

-webkit-user-select: none;

touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);

display: none;

}

.button2{

appearance: button;

background-color: transparent;

background-image: linear-gradient(to bottom, rgb(252, 162, 162), #ee3737);

border: 0 solid #e5e7eb;

border-radius: .5rem;

box-sizing: border-box;

color: #482307;

column-gap: 1rem;

cursor: pointer;

display: flex;

font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";

font-size: 100%;

font-weight: 700;

line-height: 24px;

margin: 0;

outline: 2px solid transparent;

padding: 1rem 1.5rem;

text-align: center;

text-transform: none;

transition: all .1s cubic-bezier(.4, 0, .2, 1);

user-select: none;

-webkit-user-select: none;

touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);

display: none;

}

.right {

right: 0px;

width: 300px;

}

@media (max-width: 576px) {

.form {

width: 100%;

}

}

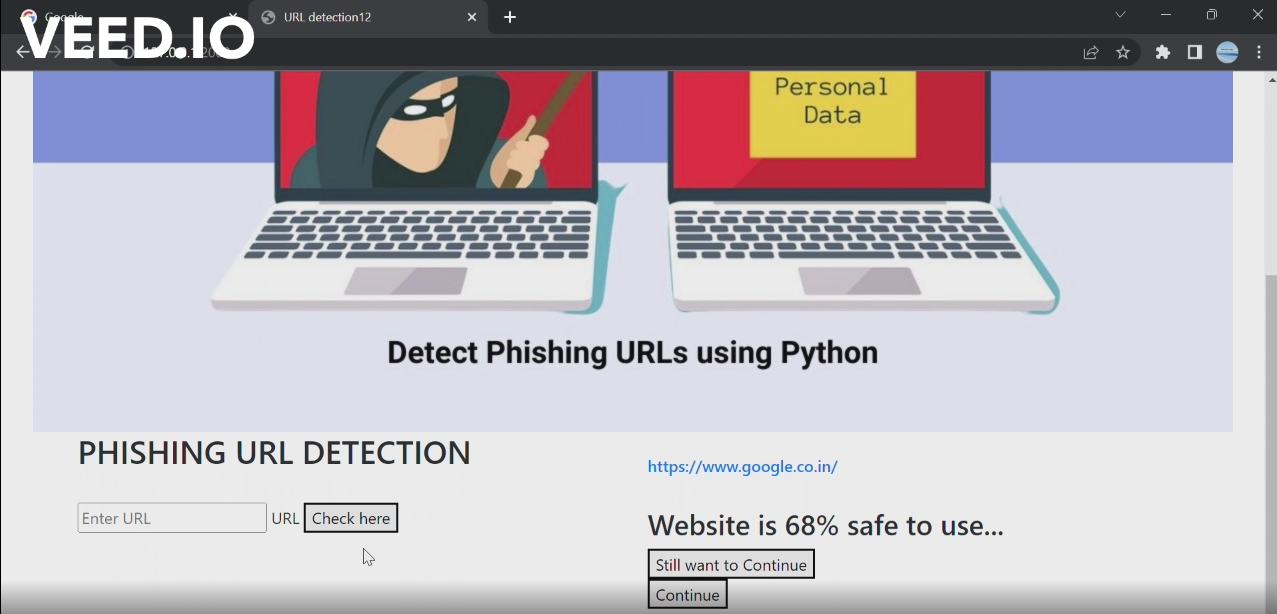
.abc{

width: 50%;

}

**OUTPUT:**

****

****

**GITHUB LINK**

[IBM-EPBL](https://github.com/IBM-EPBL)/[**IBM-Project-35596-1660286590**](https://github.com/IBM-EPBL/IBM-Project-35596-1660286590)

**PROJECT DEMO LINK**

[**https://drive.google.com/file/d/1f2C1QTejBoxtHdHQtv7mvE\_jdUvdb30\_/view?usp=drivesdk**](https://drive.google.com/file/d/1f2C1QTejBoxtHdHQtv7mvE_jdUvdb30_/view?usp=drivesdk)

**TEAM MEMBERS**

1. **SIVANESAN M**
2. **RUSSEL GABRIEL THOMAS**
3. **MOHAMMED NISHAD S**
4. **PRANAV P**