WEB PHISHING DETECTION USING MACHINE LEARNING

Team ID : PNT2022TMID23060

Team Members:

R.Deepthiha

T.Dharshana

P.Kamalika

S.Sharmila

1.**INTRODUCTION**

1.1 Project Overview

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.

1.2 Purpose

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.

2. **LITERATURE SURVEY**

2.1 Existing problem

A)Detecting Phishing Websites Using Machine Learning

Authors:Amani Alswailem, Bashayr Alabdullah, Norah Alrumayh, Dr.Aram Alsedrani.

Phishing website is one of the internet security problems that target the human vulnerabilities rather than software vulnerabilities. It can be described as the process of attracting online users to obtain their sensitive information such as usernames and passwords. In this paper, we offer an intelligent system for detecting phishing websites. The system acts as an additional functionality to an internet browser as an extension that automatically notifies the user when it detects a phishing website. The system is based on a machine learning method, particularly supervised learning. We have selected the Random Forest technique due to its good performance in classification. Our focus is to pursue a higher performance classifier by studying the features of phishing website and choose the better combination of them to train the classifier. As a result, we conclude our paper with accuracy of 98.8% and combination of 26 features.

B)Detection of Phishing Websites by Using Machine Learning-Based URL Analysis

Authors: Mehmet Korkmaz ,Ozgur Koray Sahingoz, Banu Diri.

In recent years, with the increasing use of mobile devices, there is a growing trend to move almost all real-world operations to the cyberworld. Although this makes easy our daily lives, it also brings many security breaches due to the anonymous structure of the Internet. Used antivirus programs and firewall systems can prevent most of the attacks. However, experienced attackers target on the weakness of the computer users by trying to phish them with bogus webpages. These pages imitate some popular banking, social media, e-commerce, etc. sites to steal some sensitive information such as, user-ids, passwords, bank account, credit card numbers, etc. Phishing detection is a challenging problem, and many different solutions are proposed in the market as a blacklist, rule-based detection, anomaly-based detection, etc. In the literature, it is seen that current works tend on the use of machine learning-based anomaly detection due to its dynamic structure, especially for catching the “zero-day” attacks.

C)Machine Learning Techniques for Detection of Website Phishing: A Review for Promises and Challenges

Authors:Ammar Odeh, Eman Abdel fattah, Ismail Keshta.

Attackers fool the users by presenting the masked webpage as legitimate or trustworthy to with better performance for detecting phishing websites compared to the conventional ML techniques. Challenges to ML techniques identified in this work include overfitting, low accuracy, and ML techniques ineffectiveness in case of unavailability of enough training data.

D) Detection of Phishing Websites from URLs by using Classification Techniques on WEKA

Authors: Buket Geyik, Kiiubra Erensoy,Emre Kocyirit

The Internet is getting stronger day by day and it makes our lives easier with many applications that are executed on cyberworld. However, with the development of the internet, cyber-attacks have increased gradually and identity thefts have emerged. It is a type of fraud committed by intruders by using fake web pages to access people's private information such as userid, password, credit card number and bank account numbers, etc. These scammers can also send e-mail from many important institutions and organizations by using phishing attacks which imitate these web pages and acts as if they are original. Traditional security mechanisms can not prevent these attacks because they directly target the weakest part of connection : end-users. Machine learning technology has been used to detect and prevent this type of intrusions. The antiphishing method has been developed by detecting the attacks made with the technologies used. In this paper, we combined the websites used by phishing attacks into a dataset.

E) Phishing Website Detection on Machine Learning Algorithm

Authors: Weiheng Bai

Phishing websites are a means to deceive users personal information by using various means to impersonate the URL address and page content of a real website.This paper analyses the structural ,extracts features of the URL of the phishing website,extracts 12 kinds of features and uses four machine learning algorithms for training. Then, use the best performing algorithm as our model to identity unknown URL's.After the Reconition is completed,a snapshot of the web page is extracted and compared with the regular web pages snapshot to implement the recommendation of the original regular web page of the phishing web page.

2.2 References

[1] Republic of Turkey, “National Cyber Security Strategy, 2016,” Ministry of Transport Martime Affairs and Communications.

[2] R. Loftus, “What cybersecurity trends should you look out for in 2020?,” Daily English Global blogkasperskycom. [Online]. Available: https://www.kaspersky.com/blog/secure-futures-magazine/2020cybersecurity-predictions/32068/. [Accessed: 09-Mar-2020].

[3] E. Buber, Ö. Demir and O. K. Sahingoz, "Feature selections for the machine learning based detection of phishing websites," 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, 2017, pp. 1-5.

[4] pp. 111-122. [16] H. H. Nguyen and D. T. Nguyen, ”Machine Learning Based Phishing Web Sites Detection,” in AETA 2015: Recent Advances in Electrical Engineering and Related Sciences, V. H. Duy, T. T. Dao, I. Zelinka, H.S. Choi, and M. Chadli, Eds. Cham: Springer International Publishing, 2016, pp. 123-131.

[5] M. A. U. H. Tahir, S. Asghar, A. Zafar, and S. Gillani, ”A Hybrid Model to Detect 76 Phishing-Sites Using Supervised Learning Algorithms,” in 2016 International Conference on Computational Science and Computational Intelligence (CSCI), 2016, pp. 1126-1133.

[6] M. S. I. Mamun, M. A. Rathore, A. H. Lashkari, N. Stakhanova, and A. A. Ghorbani, ”Detecting Malicious URLs Using Lexical Analysis,” in Network and System Security: 10th International Conference, NSS 2016, Taipei, Taiwan, September 28-30, 2016, Proceedings, J. Chen, V. Piuri, C. Su, and M. Yung, Eds. Cham: Springer International Publishing, 2016, pp. 467- 482.

[7]M. Weedon, D. Tsaptsinos and J. Denholm-Price, ”Random forest explorations for URL classification,” 2017 International Conference On Cyber Situational Awareness, Data Analytics And Assessment (Cyber SA), London, 2017, pp. 1-4, doi: 10.1109/CyberSA.2017.8073403.

[8] E. Buber, B. Dırı and O. K. Sahingoz, ”Detecting phishing attacks from URL by using NLP techniques,” 2017 International Conference on Computer Science and Engineering (UBMK), Antalya, 2017, pp. 337-342, doi: 10.1109/UBMK.2017.8093406.

[9] C. Liu, L. Wang, B. Lang and Y. Zhou, ”Finding effective classifier for malicious URL detection,” Proceedings of the 2018 2nd International Conference on Management Engineering, Software Engineering and Service Sciences, 2018, pp. 240-244, doi: 10.1145/3180374.3181352.

[10] Rakesh R, Kannan A, Muthurajkumar S, Pandiyaraju V and SaiRamesh L, ”Enhancing the precision of phishing classification accuracy using reduced feature set and boosting algorithm,” 2014 Sixth International Conference on Advanced Computing (ICoAC),

Chennai, 2014, pp. 86-90, doi: 10.1109/ICoAC.2014.7229752.

2.3 Problem Statement Definition

There are a number of users who purchase products online and make payments through e-banking. There are e-banking websites that ask users to provide sensitive data such as username, password & credit card details, etc., often for malicious reasons. This type of e-banking website is known as a phishing website. Web service is one of the key communications software services for the Internet. Web phishing is one of many security threats to web services on the Internet.

Common threats of web phishing:

Web phishing aims to steal private information, such as usernames, passwords, and credit card details, by way of impersonating a legitimate entity.

It will lead to information disclosure and property damage.

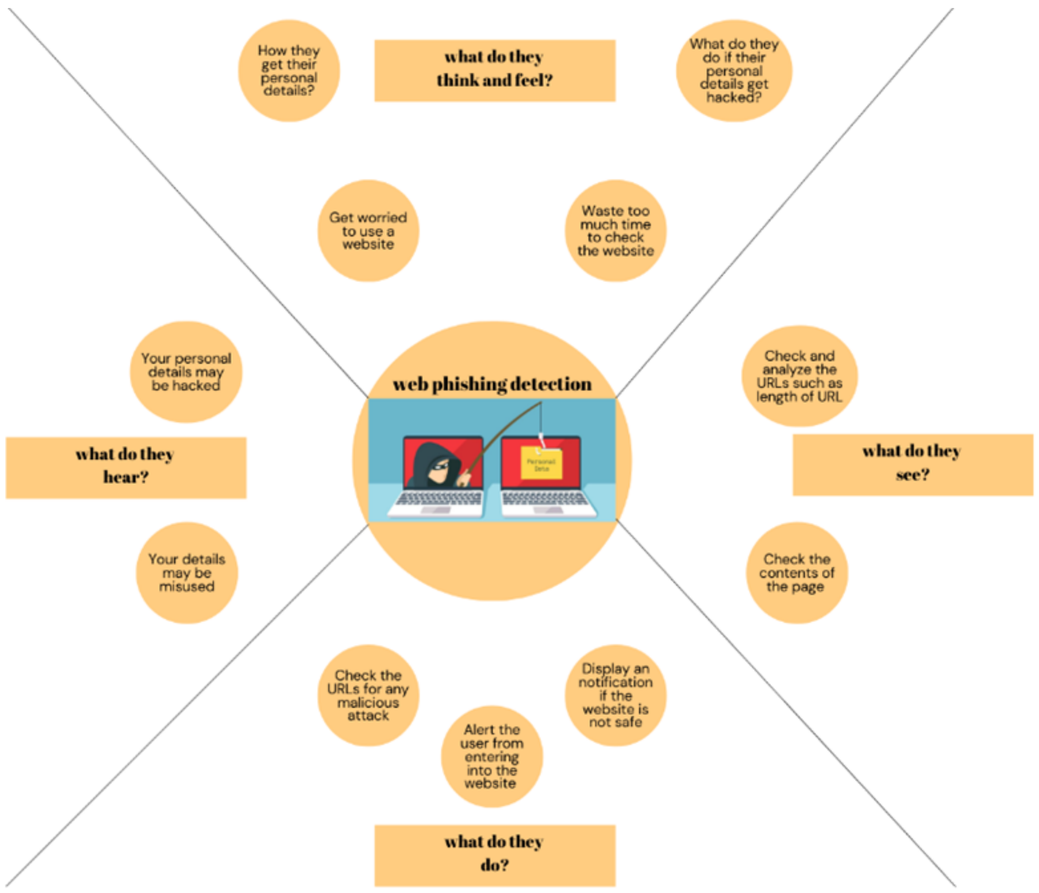
Large organizations may get trapped in different kinds of scams.

This Guided Project mainly focuses on applying a machine-learning algorithm to detect Phishing websites.

In order to detect and predict e-banking phishing websites, we proposed an intelligent, flexible and effective system that is based on using classification algorithms. We implemented classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy. The e-banking phishing website can be detected based on some important characteristics like URL and domain identity, and security and encryption criteria in the final phishing detection rate. Once a user makes a transaction online when he 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 website or not.

3. **IDEATION & PROPOSED SOLUTION**

3.1 Empathy Map Canvas


How they get their personal details?
what do they think and feel?
What do they do if their personal details get hacked?
Get worried to use a website
Waste too much time to check the website
web phishing detection
Personal Data
Your personal details may be hacked
what do they hear?
Your details may be misused
Check and analyze the URLs such as length of URL
what do they see?
Check the contents of the page
Check the URLs for any malicious attack
Alert the user from entering into the website
Display an notification if the website is not safe
what do they do?

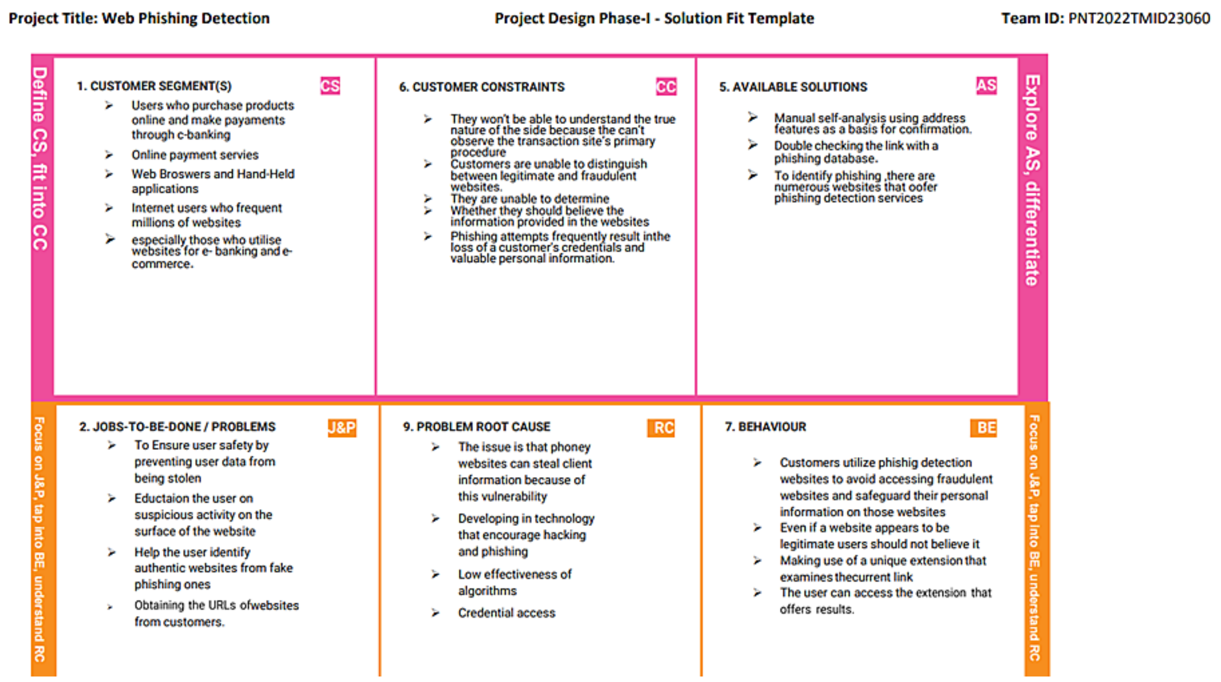
3.2 Ideation & Brainstorming

3.3 Proposed Solution

S.No	Parameter	Description
1.2	Problem Statement (Problem to be solved)	There are a number of users who purchase products online and make payments through e-banking. There are e-banking websites that ask users to provide sensitive data such as username, password & credit card details, etc., often for malicious reasons. This type of e-banking website is known as a phishing website. Web service is one of the key communications software services for the Internet. Web phishing is one of many security threats to web services on the Internet. Common threats of web phishing: • Web phishing aims to steal private information, such as usernames, passwords, and credit card details, by way of impersonating a legitimate entity. • It will lead to information disclosure and property damage. • Large organizations may get trapped in different kinds of scams.
2.	Idea / Solution description	1. In order to detect and predict e-banking phishing websites, we proposed an intelligent, flexible and effective system that is based on using classification algorithms. We implemented classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy. 2. Once a user makes a transaction online when he makes payment through an ebanking website our system will use a data mining algorithm to detect whether the ebanking website is a phishing website or not.

3.	Novelty / Uniqueness	1. Using machine learning and classification algorithms, the criteria for detecting the phishing websites are extracted. 2. The machine is trained with the extracted criteria and when the customer makes payment using e-payment website the system detects thephishing website.
4.	Social Impact / Customer Satisfaction	1. Nowadays e-payment has become most popular way of payment and many online purchases depends on e-payment. 2. The main goal is to detect the phishing epayment website and protect user details from phishingto ensure their privacy.
5.	Business Model (Revenue Model)	1. This system generate revenue from the users and from online product selling platforms.
6.	Scalability of the Solution	1. Machine learning helps people to safely transfer their money without losing their personal detailsby detecting the phishing websites. 2. This makes the user to feel safe and secure to use e-payments for online purchase.

3.4 Problem Solution Fit



Project Title: Web Phishing Detection

Project Design Phase-I - Solution Fit Template

Team ID: PNT2022TMID23060

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

- Users who purchase products online and make payaments through c-banking
- Online payment servies
- Web Broswers and Hand-Held applications
- Internet users who frequent millions of websites
- especially those who utilise websites for e- banking and e-commerce.

6. CUSTOMER CONSTRAINTS CC

- They won't be able to understand the true nature of the side because the can't observe the transaction site's primary procedure
- Customers are unable to distinguish between legitimate and fraudulent websites.
- They are unable to determine
- Whether they should believe the information provided in the websites
- Phishing attempts frequently result inthe loss of a customer's credentials and valuable personal information.

5. AVAILABLE SOLUTIONS AS

- Manual self-analysis using address features as a basis for confirmation.
- Double checking the link with a phishing database.
- To identify phishing ,there are numerous websites that oofer phishing detection services

Explore AS, differentiate

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

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

- To Ensure user safety by preventing user data from being stolen
- Eductaion the user on suspicious activity on the surface of the website
- Help the user identify authentic websites from fake phishing ones
- Obtaining the URLs ofwebsites from customers.

9. PROBLEM ROOT CAUSE RC

- The issue is that phoney websites can steal client information because of this vulnerability
- Developing in technology that encourage hacking and phishing
- Low effectiveness of algorithms
- Credential access

7. BEHAVIOUR BE

- Customers utilize phishig detection websites to avoid accessing fraudulent websites and safeguard their personal information on those websites
- Even if a website appears to be legitimate users should not believe it
- Making use of a unique extension that examines thecurrent link
- The user can access the extension that offers results.

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

4. **REQUIREMENT 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.

1. Our system should be able to load air quality data and preprocess data.
2. It should be able to analyze the air quality data.
3. It should be able to group data based on hidden patterns.
4. It should be able to assign a label based on its data groups.
5. It should be able to split data into trainset and testset.
6. It should be able to train model using trainset.
7. It must validate trained model using testset.

8. It should be able to display the trained model accuracy.
9. It should be able to accurately predict the air quality on unseen data.

4.2 Non-Functional requirement

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" properties that 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:

1. Reliability
2. Maintainability
3. Performance
4. Portability
5. Scalability
6. Flexibility

Some of the quality attributes are as follows:

4.2.1 Accessibility:

Availability is a general term used to depict how much an item, gadget, administration, or condition is open by however many individuals as would be prudent. In our venture individuals who have enrolled with the cloud can get to the cloud to store and recover their information with the assistance of a mystery key sent to their email ids. UI is straightforward and productive and simple to utilize.

4.2.2 Maintainability:

In programming designing, viability is the simplicity with which a product item can be altered so as to:

1. Correct absconds
2. Meet new necessities

New functionalities can be included in the task based the client necessities just by adding the proper documents to existing venture utilizing ASP.net and C# programming dialects. Since the writing computer programs is extremely straightforward, it is simpler to discover and address the imperfections and to roll out the improvements in the undertaking.

4.2.3 Scalability:

Framework is fit for taking care of increment all out throughput under an expanded burden when assets (commonly equipment) are included. Framework can work ordinarily under circumstances, for example, low data transfer capacity and substantial number of clients.

4.2.4 Portability:

Convey ability is one of the key ideas of abnormal state programming. Convenient is the product code base component to have the capacity to reuse the current code as opposed to making new code while moving programming from a domain to another. Venture can be executed under various activity conditions gave it meet its base setups. Just framework records and dependant congregations would need to be designed in such case. The functional requirements for a system describe what the system should do. Those requirments depend on the type of software being developed,the expected users of the software. These are the statement of services the system should provide,how the system should react to particular inputs and how the system should behave in particular situation.

1. Extracting data from CSV files
2. Cleaning the data.
3. Vector Representation.

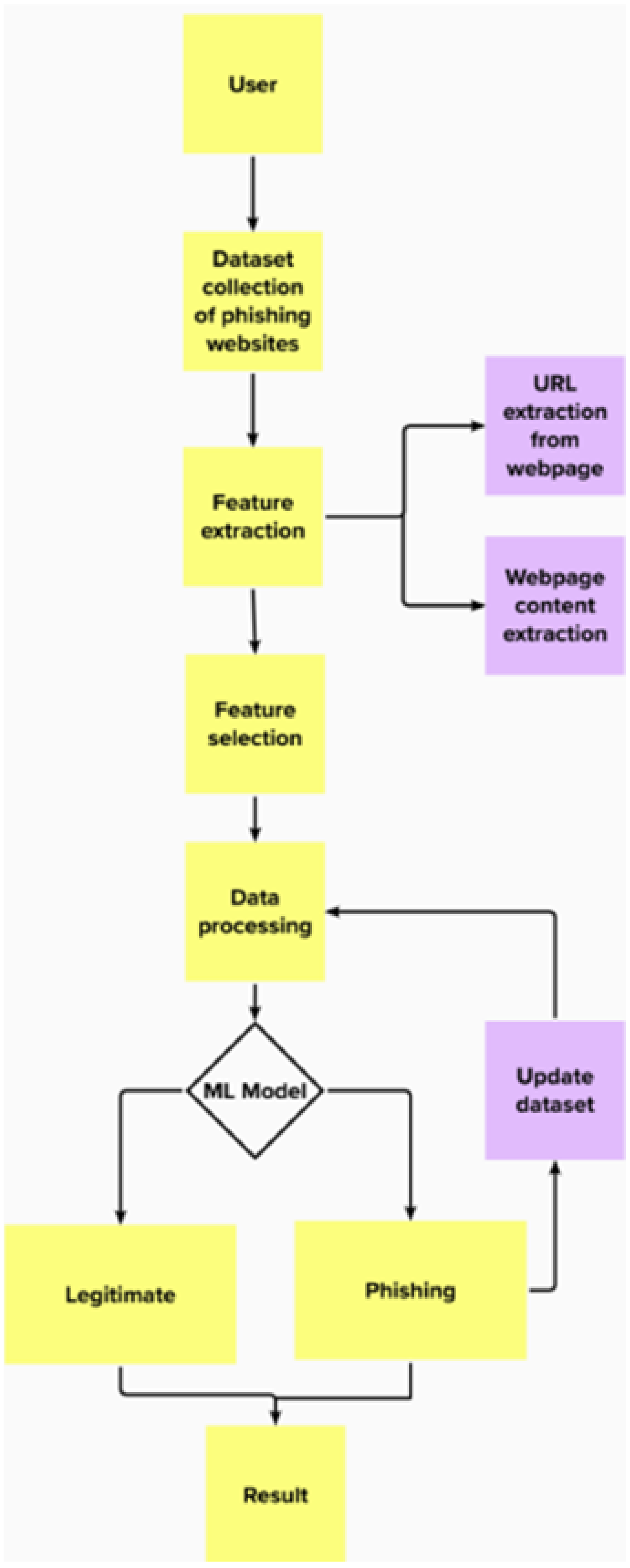
Non-functional requirements is not about functionality or behaviour of system, but rather are used to specify the capacity of a system. They are more related to properties of system such as quality, reliability and quick response time. Non- functional requirements come up via customer needs, because of budget, interoperability need such as software and hardware requirement, organizational policies or due to some external factors such as:-

1. Basic Operational Requirement
2. Organizational Requirement
3. Product Requirement
4. User Requirement

5.**PROJECT DESIGN**

5.1 Data Flow Diagrams

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


User
Dataset collection of phishing websites
Feature extraction
URL extraction from webpage
Webpage content extraction
Feature selection
Data processing
ML Model
Update dataset
Legitimate
Phishing
Result

5.2 Solution & Technical Architecture

Solution Architecture:

Technical Architecture:

5.3 User Stories

6. **PROJECT PLANNING & SCHEDULING**

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

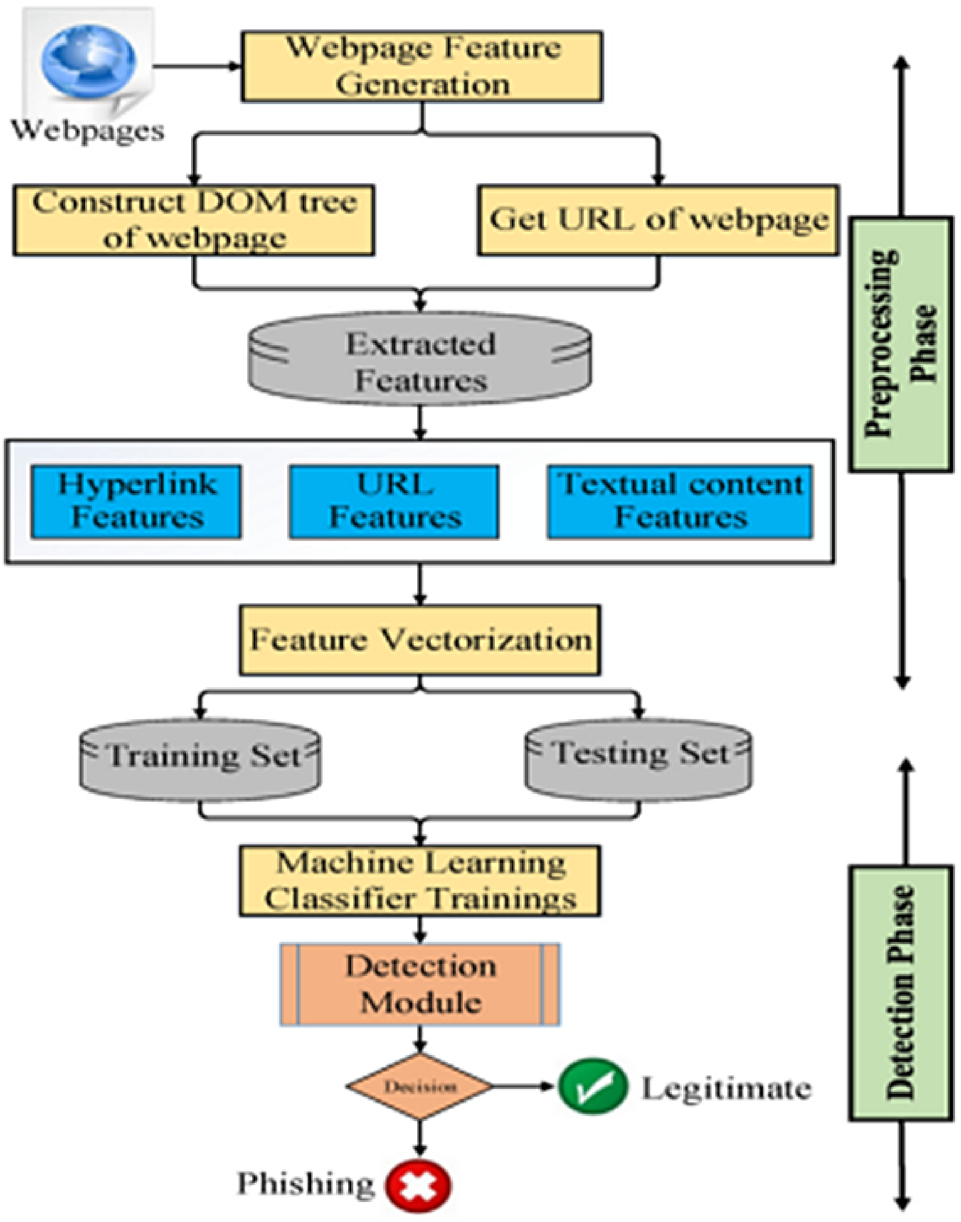
6.3 Reports from JIRA

Roadmap

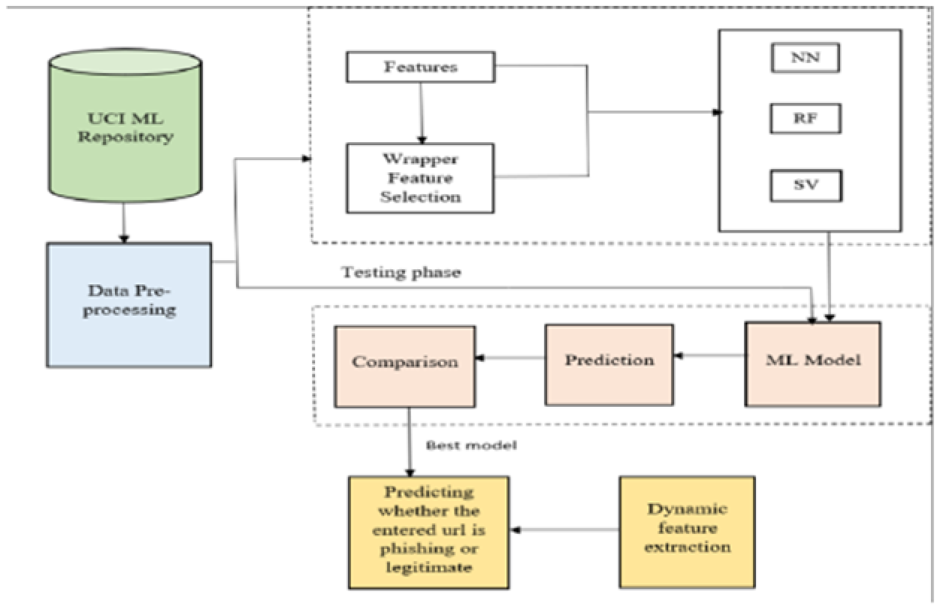
Backlog

5.2 Solution & Technical Architecture

Solution Architecture:



Technical Architecture:


UCI ML Repository
Data Pre-processing
Features
Wrapper Feature Selection
NN
RF
SV
Testing phase
Comparison
Prediction
ML Model
Best model
Predicting whether the entered url is phishing or legitimate
Dynamic feature extraction

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2.

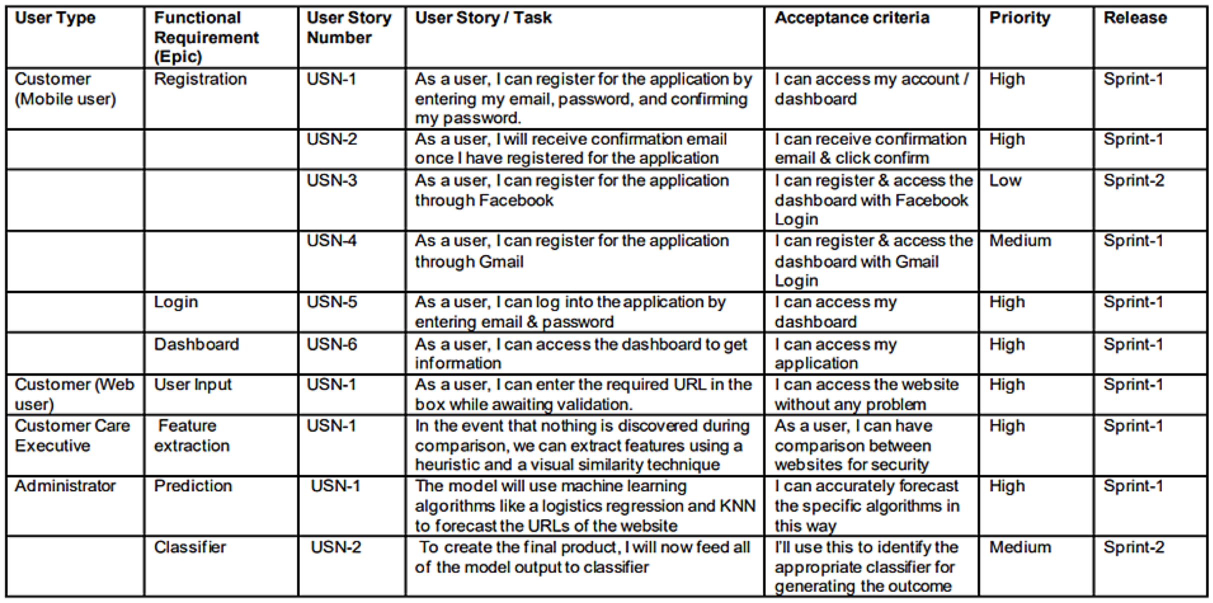
Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript
2.	Application Logic	Logic for a process in the application	Flask login(python)
3.	Cloud Database	Database Service on Cloud	IBM Waston
4.	File Storage	File storage requirements	MongoDB
5.	Machine Learning Model	Purpose of Machine Learning Model	Logistic Regression, Decision Tree
6.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local, Render, IBM Cloud

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Sckit Learn package in python that deals with ML algorithms	Machine learning
2.	Security Implementations	Typosquatting, Cybersquatting	Cybersecurity
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Technology used
4.	Availability	It can balance the load traffic among the servers to help improve uptime. Can scale applications by adding or removing servers, with minimal disruptions to traffic flows.	IBM Cloud Load Balancers
5.	Performance	It provides performance feedback such as page size and how long it takes to load pages, and can show the impact new features have on the performance of the site.	Blacklists/Whitelists, Natural language Processing, Visual similarity, rules, machine learning techniques, etc.

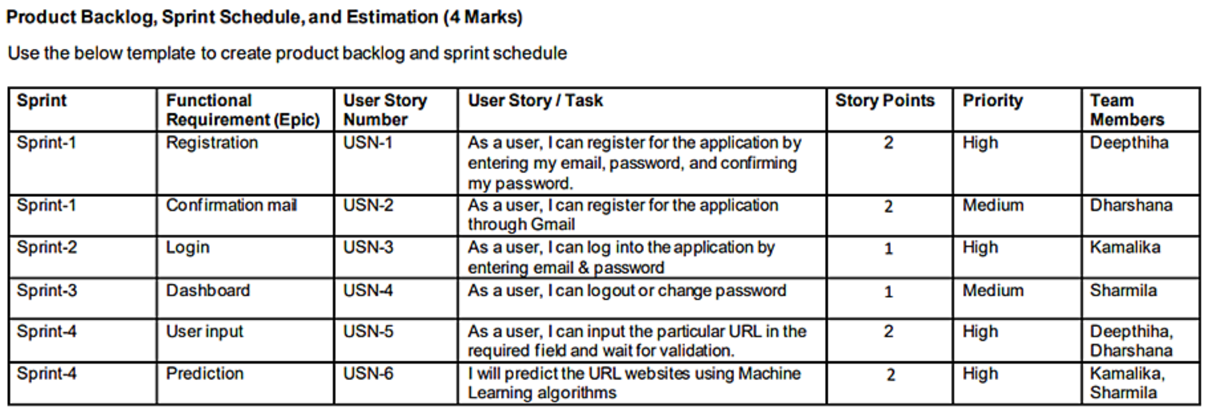
5.3 User Stories



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	I can register & access the dashboard with Gmail Login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can access my dashboard	High	Sprint-1
	Dashboard	USN-6	As a user, I can access the dashboard to get information	I can access my application	High	Sprint-1
Customer (Web user)	User Input	USN-1	As a user, I can enter the required URL in the box while awaiting validation.	I can access the website without any problem	High	Sprint-1
Customer Care Executive	Feature extraction	USN-1	In the event that nothing is discovered during comparison, we can extract features using a heuristic and a visual similarity technique	As a user, I can have comparison between websites for security	High	Sprint-1
Administrator	Prediction	USN-1	The model will use machine learning algorithms like a logistics regression and KNN to forecast the URLs of the website	I can accurately forecast the specific algorithms in this way	High	Sprint-1
	Classifier	USN-2	To create the final product, I will now feed all of the model output to classifier	I'll use this to identify the appropriate classifier for generating the outcome	Medium	Sprint-2

6. **PROJECT PLANNING & SCHEDULING**

6.1 Sprint Planning & Estimation



Product Backlog, Sprint Schedule, and Estimation (4 Marks)

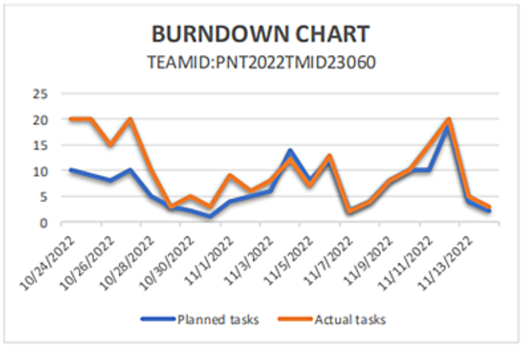
Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Deepthiha
Sprint-1	Confirmation mail	USN-2	As a user, I can register for the application through Gmail	2	Medium	Dharshana
Sprint-2	Login	USN-3	As a user, I can log into the application by entering email & password	1	High	Kamalika
Sprint-3	Dashboard	USN-4	As a user, I can logout or change password	1	Medium	Sharmila
Sprint-4	User input	USN-5	As a user, I can input the particular URL in the required field and wait for validation.	2	High	Deepthiha, Dharshana
Sprint-4	Prediction	USN-6	I will predict the URL websites using Machine Learning algorithms	2	High	Kamalika, Sharmila

6.2 Sprint Delivery Schedule

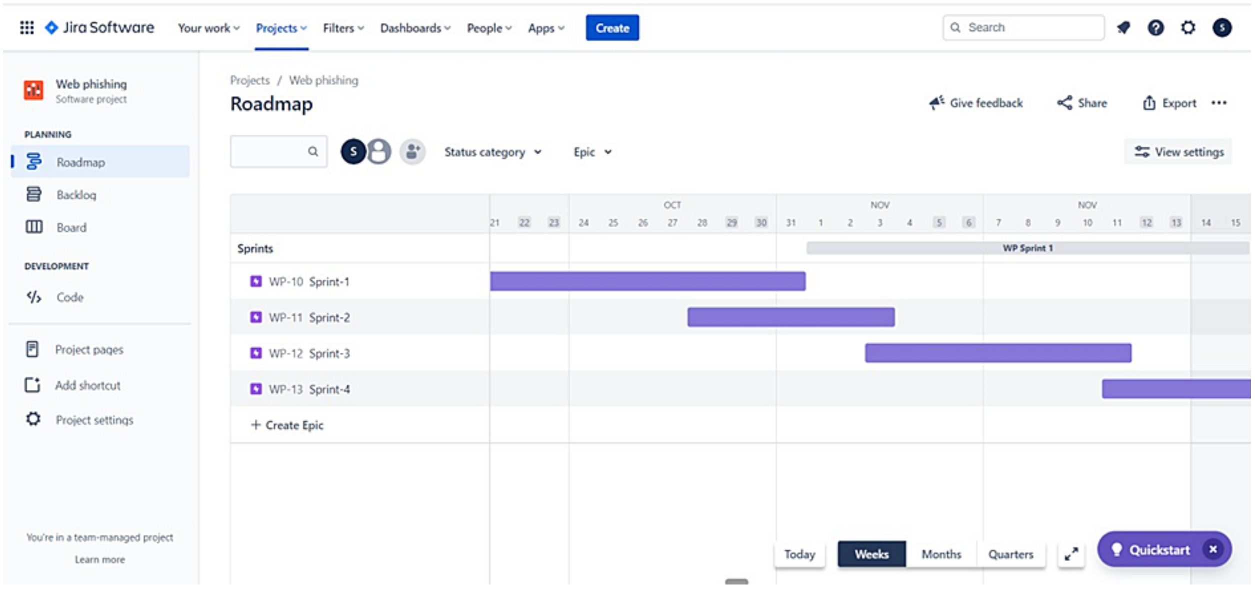
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022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

Burndown Chart:

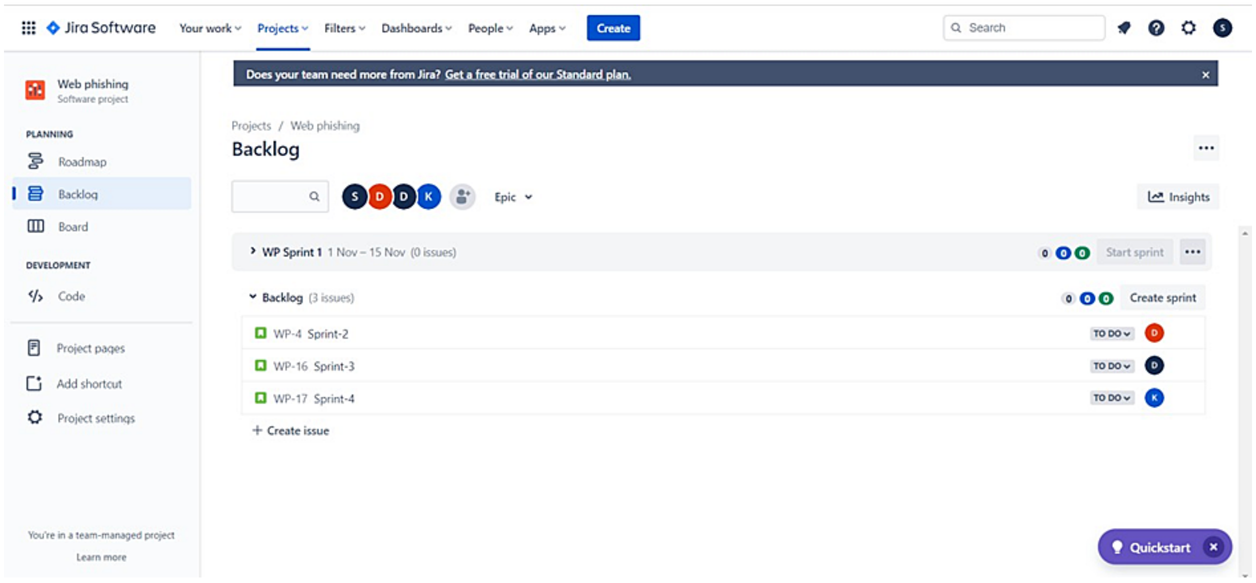


6.3 Reports from JIRA

Roadmap



Backlog



Does your team need more from Jira? Get a free trial of our Standard plan.

Projects / Web phishing

Backlog

WP Sprint 1 1 Nov – 15 Nov (0 issues)

Backlog (3 issues)

WP-4 Sprint-2	TO DO	D
WP-16 Sprint-3	TO DO	D
WP-17 Sprint-4	TO DO	K

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 app.py

```python
import numpy as np
import pandas
from flask import Flask, request, jsonify, render_template
import pickle
import inputscript

app = Flask(__name__)
model = pickle.load(open('Phishing_Website.pkl','rb'))



@app.route("/")
def predict():
    return render_template('index.html')

ans = ""
bns = ""
@app.route('/result_processing_function', methods=['POST','GET'])
```

```
def y_predict():
    url = request.form['url']
    checkprediction = inputscript.main(url)
    prediction = model.predict(checkprediction)
    print(prediction)
    output=prediction[0]
    if(output==1):
        pred="You are safe!!  This is a legitimate Website."
        return render_template('index.html',bns=pred)

    else:
        pred="You are on the wrong site. Be cautious!"
        return render_template('index.html',ans=pred)



@app.route('/predict_api', methods=['POST'])
def predict_api():

    data = request.get_json(force=True)
    prediction = model.y_predict([np.array(list(data.values()))])

    output=prediction[0]
    return jsonify(output)

if __name__ == '__main__':
    app.run()
```

7.2 inputscript.py

```
import regex
from tldextract import extract
import ssl
import socket
from bs4 import BeautifulSoup
import urllib.request

import datetime
import requests

import re




"""
Check if URL contains any IP address. Returns -1 if contains else returns
1
"""
def having_IPhaving_IP_Address(url):
     match=regex.search(
   '(([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-
5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-
5])\\/)|'  #IPv4
                    '((0x[0-9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\.(0x[0-
9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\/)'  #IPv4 in hexadecimal
                    '(?:[a-fA-F0-9]{1,4}:){7}[a-fA-F0-9]{1,4}',url)
    #Ipv6
     if match:
        #print match.group()
        return -1
     else:
        #print 'No matching pattern found'
        return 1
"""
```

```
Check for the URL length. Return 1 (Legitimate) if the URL length is less
than 54 characters
Return 0 if the length is between 54 and 75
Else return -1
"""
def URLURL_Length (url):
    length=len(url)
    if(length<=75):
         if(length<54):
            return 1
         else:
            return 0
    else:
        return -1

"""
Check with the shortened URLs.
Return -1 if any shortened URLs used.
Else return 1
"""
def Shortining_Service (url):

match=regex.search('bit\.ly|goo\.gl|shorte\.st|go2l\.ink|x\.co|ow\.ly|t\.c
o|tinyurl|tr\.im|is\.gd|cli\.gs|'

'yfrog\.com|migre\.me|ff\.im|tiny\.cc|url4\.eu|twit\.ac|su\.pr|twurl\.nl|sn
ipurl\.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',url)
    if match:
        return -1
    else:
        return 1

#Checking for @ symbol. Returns 1 if no @ symbol found. Else returns 0.
def having_At_Symbol(url):
    symbol=regex.findall(r'@',url)
    if(len(symbol)==0):
        return 1
    else:
        return -1

#Checking for Double Slash redirections. Returns -1 if // found. Else
returns 1
def double_slash_redirecting(url):
    for i in range(8,len(url)):
        if(url[i]=='/'):

            if(url[i-1]=='/'):
                return -1
    return 1

#Checking for - in Domain. Returns -1 if '-' is found else returns 1.
def Prefix_Suffix(url):
    subDomain, domain, suffix = extract(url)
    if(domain.count('-')):
        return -1
    else:
        return  1
```

```
"""
Check the Subdomain. Return 1 if the subDomain contains less than 1 '.'
Return 0 if the subDomain contains less than 2 '.'
Return -1 if the subDomain contains more than 2 '.'
"""
def having_Sub_Domain(url):
    subDomain, domain, suffix = extract(url)
    if(subDomain.count('.')<=2):
        if(subDomain.count('.')<=1):
            return 1
        else:
            return 0
    else:
        return -1

#Checking the SSL. Returns 1 if it returns the respomse code and -1 if
exceptions are thrown.
def SSLfinal_State(url):
    try:
        response = requests.get(url)
        return 1
    except Exception as e:
        return -1

#domains expires on ≤ 1 year returns -1, otherwise returns 1

def Domain_registeration_length(url):
    try:
        domain = whois.whois(url)
        exp=domain.expiration_date[0]
        up=domain.updated_date[0]
        domainlen=(exp-up).days
        if(domainlen<=365):
            return -1
        else:
            return 1
    except:
        return -1

#Checking the Favicon. Returns 1 if the domain of the favicon image and
the URL domain match else returns -1.
def Favicon(url):
    subDomain, domain, suffix = extract(url)
    b=domain
    try:
        icons = favicon.get(url)
        icon = icons[0]
        subDomain, domain, suffix =extract(icon.url)
        a=domain
        if(a==b):
            return 1
        else:
            return -1
    except:
        return -1

#Checking the Port of the URL. Returns 1 if the port is available else
returns -1.
def port(url):
    try:
        a_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        location=(url[7:],80)
        result_of_check = a_socket.connect_ex(location)
        if result_of_check == 0:
            return 1
        else:
            return -1
        a_socket.close
    except:
        return -1
```

```python
# HTTPS token in part of domain of URL returns -1, otherwise returns 1
def HTTPS_token(url):
    match=re.search('https://|http://',url)
    if (match and match.start(0)==0):
        url=url[match.end(0):]
    match=re.search('http|https',url)
    if match:
        return -1
    else:
        return 1


#% of request URL<22% returns 1, otherwise returns -1
def Request_URL(url):
    try:
        subDomain, domain, suffix = extract(url)
        websiteDomain = domain

        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        imgs = soup.findAll('img', src=True)
        total = len(imgs)

        linked_to_same = 0
        avg =0
        for image in imgs:
            subDomain, domain, suffix = extract(image['src'])
            imageDomain = domain
            if(websiteDomain==imageDomain or imageDomain==''):
                linked_to_same = linked_to_same + 1
        vids = soup.findAll('video', src=True)
        total = total + len(vids)

        for video in vids:
            subDomain, domain, suffix = extract(video['src'])
            vidDomain = domain
```

```
                if(websiteDomain==vidDomain or vidDomain==''):
                    linked_to_same = linked_to_same + 1
            linked_outside = total-linked_to_same
            if(total!=0):
                avg = linked_outside/total

            if(avg<0.22):
                return 1
            else:
                return -1
        except:
            return -1

#:% of URL of anchor<31% returns 1, % of URL of anchor ≥ 31% and ≤ 67%
returns 0, otherwise returns -1
def URL_of_Anchor(url):
    try:
        subDomain, domain, suffix = extract(url)
        websiteDomain = domain

        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        anchors = soup.findAll('a', href=True)
        total = len(anchors)
        linked_to_same = 0
        avg = 0
        for anchor in anchors:
            subDomain, domain, suffix = extract(anchor['href'])
            anchorDomain = domain
            if(websiteDomain==anchorDomain or anchorDomain==''):
                linked_to_same = linked_to_same + 1
        linked_outside = total-linked_to_same
        if(total!=0):
            avg = linked_outside/total

        if(avg<0.31):
            return 1
        elif(0.31<=avg<=0.67):
            return 0
        else:
            return -1
    except:
        return 0

"""
% of links in <meta>, <script>and<link>tags < 25% returns 1, % of links in
<meta>,
<script> and <link> tags ≥ 25% and ≤ 81% returns 0, otherwise returns -1
"""

def Links_in_tags(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')

        no_of_meta =0
        no_of_link =0
        no_of_script =0
        anchors=0
        avg =0
        for meta in soup.find_all('meta'):
            no_of_meta = no_of_meta+1
        for link in soup.find_all('link'):
            no_of_link = no_of_link +1
        for script in soup.find_all('script'):
            no_of_script = no_of_script+1
        for anchor in soup.find_all('a'):
            anchors = anchors+1
        total = no_of_meta + no_of_link + no_of_script+anchors
        tags = no_of_meta + no_of_link + no_of_script
        if(total!=0):
            avg = tags/total
        if(avg<0.25):
            return -1
        elif(0.25<=avg<=0.81):
            return 0
        else:
            return 1
    except:
        return 0

#Server Form Handling
#SFH is "about: blank" or empty → phishing, SFH refers to a different
domain → suspicious, otherwise → legitimate
def SFH(url):
    #ongoing
    return -1

#:using "mail()" or "mailto:" returning -1, otherwise returns 1
def Submitting_to_email(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        if(soup.find('mailto:','mail():')):
            return -1
        else:
            return 1
    except:
        return -1

#Host name is not in URL returns -1, otherwise returns 1
def Abnormal_URL(url):
    subDomain, domain, suffix = extract(url)
    try:
        domain = whois.whois(url)
        hostname=domain.domain_name[0].lower()
        match=re.search(hostname,url)
        if match:
            return 1
        else:
            return -1
    except:
        return -1

#number of redirect page ≤ 1 returns 1, otherwise returns 0
def Redirect(url):
    try:
        request = requests.get(url)
        a=request.history
        if(len(a)<=1):
            return 1
        else:
            return 0

    except:
        return 0


#onMouseOver changes status bar returns -1, otherwise returns 1
def on_mouseover(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')

        no_of_script =0
        for meta in soup.find_all(onmouseover=True):
            no_of_script = no_of_script+1
        if(no_of_script==0):
            return 1
        else:
            return -1
    except:
        return -1
```

```
#right click disabled returns -1, otherwise returns 1
def RightClick(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        if(soup.find_all('script',mousedown=True)):
            return -1
        else:
            return 1
    except:
        return -1

#popup window contains text field → phishing, otherwise → legitimate
def popUpWidnow(url):
    #ongoing
    return 1

#using iframe returns -1, otherwise returns 1
def Iframe(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        nmeta=0
        for meta in soup.findAll('iframe',src=True):
            nmeta= nmeta+1
        if(nmeta!=0):
            return -1
        else:
            return 1
    except:
        return -1


#:age of domain ≥ 6 months returns 1, otherwise returns -1
def age_of_domain(url):
    try:
        w = whois.whois(url).creation_date[0].year
        if(w<=2018):
            return 1
        else:
            return -1
    except Exception as e:
        return -1

#no DNS record for domain returns -1, otherwise returns 1
def DNSRecord(url):

    subDomain, domain, suffix = extract(url)
    try:
        dns = 0
        domain_name = whois.whois(url)
    except:
        dns = 1

    if(dns == 1):
        return -1
    else:
        return 1

#website rank < 100.000 returns 1, website rank > 100.000 returns 0,
otherwise returns -1
def web_traffic(url):
    try:
        rank =
BeautifulSoup(urllib.request.urlopen("http://data.alexa.com/data?cli=10&da
t=s&url=" + url).read(), "lxml").find("REACH")['RANK']
    except TypeError:
        return -1
    rank= int(rank)
    if (rank<100000):
        return 1
    else:
        return 0

#:PageRank < 0,2 → phishing, otherwise → legitimate
def Page_Rank(url):
    #ongoing
    return 1

#webpage indexed by Google returns 1, otherwise returns -1
def Google_Index(url):
    try:
        subDomain, domain, suffix = extract(url)
        a=domain + '.' + suffix
        query = url
        for j in search(query, tld="co.in", num=5, stop=5, pause=2):
            subDomain, domain, suffix = extract(j)
            b=domain + '.' + suffix
        if(a==b):
            return 1
        else:
            return -1
    except:
        return -1


#:number of links pointing to webpage = 0 returns 1, number of links
pointing to webpage> 0
#and ≤ 2 returns 0, otherwise returns -1

def Links_pointing_to_page (url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        count = 0
        for link in soup.find_all('a'):
            count += 1
        if(count>=2):
            return 1
        else:
            return 0
    except:
        return -1

#:host in top 10 phishing IPs or domains returns -1, otherwise returns 1
def Statistical_report (url):
    hostname = url
    h = [(x.start(0), x.end(0)) for x in
regex.finditer('https://|http://|www.|https://www.|http://www.', hostname)]
    z = int(len(h))
    if z != 0:
        y = h[0][1]
        hostname = hostname[y:]
        h = [(x.start(0), x.end(0)) for x in regex.finditer('/', hostname)]
        z = int(len(h))
        if z != 0:
            hostname = hostname[:h[0][0]]

url_match=regex.search('at\.ua|usa\.cc|baltazarpresentes\.com\.br|pe\.hu|e
sy\.es|hol\.es|sweddy\.com|myjino\.ru|96\.lt|ow\.ly',url)
    try:
        ip_address = socket.gethostbyname(hostname)

ip_match=regex.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\.1
51\.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\.2
32\.215\.140|69\.172\.201\.153|216\.218\.185\.162|54\.225\.104\.146|103\.2
43\.24\.98|199\.59\.243\.120|31\.170\.160\.61|213\.19\.128\.77|62\.113\.22
6\.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\.1
1\.56\.48|110\.34\.231\.42',ip_address)
    except:
        return -1

    if url_match:
        return -1
    else:
        return 1

#returning scrapped data to calling function in app.py
def main(url):



    check = [[having_IPhaving_IP_Address
(url),URLURL_Length(url),Shortining_Service(url),having_At_Symbol(url),

double_slash_redirecting(url),Prefix_Suffix(url),having_Sub_Domain(url),SSLfi
nal_State(url),

Domain_registeration_length(url),Favicon(url),port(url),HTTPS_token(url),R
equest_URL(url),

URL_of_Anchor(url),Links_in_tags(url),SFH(url),Submitting_to_email(url),Ab
normal_URL(url),

Redirect(url),on_mouseover(url),RightClick(url),popUpWidnow(url),Iframe(ur
l),

age_of_domain(url),DNSRecord(url),web_traffic(url),Page_Rank(url),Google_In
dex(url),
              Links_pointing_to_page(url),Statistical_report(url)]]
    print(check)
    return check
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <center> <h1 style="color: white;"> PHISHING WEBSITE DETECTION </h1> </center>
  <style>
    body{
      background-image: url('AnatomyOfPhishing.webp');
      background-repeat: no-repeat;
      background-attachment: fixed;
      background-size: 100% 100%;
    }
  </style>
</head>

<body>

<div class=" container">
  <div class="row">
    <div class="form col-md" id="form1">
      <center><h2 style="color: white; margin-top: 15%;">PHISHING URL
DETECTION</h2></center>

      <br>
      <form  method ="post" style="margin-top: 2%; text-align: center;">
        <input type="text" class="form__input" name ='url' id="url" placeholder="Enter
the URL" required="" /><br>
        <button  onclick=othername()>Check here</button><br>
      </form><br>

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

```
</body>
</html>
```

8.**TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product it is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.1 Types of Testing

8.1.1UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.1.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

8.1.3 VALIDATION TESTING

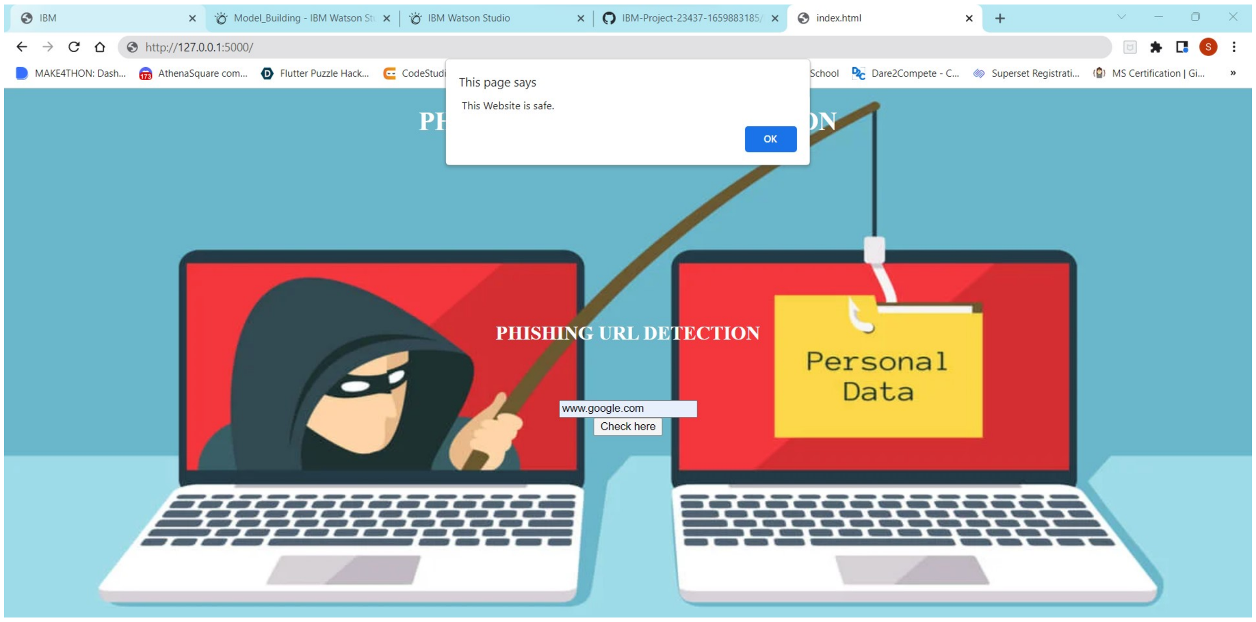
An engineering validation test (EVT) is performed on first engineering prototypes, to ensure that the basic unit performs to design goals and specifications. It is important in identifyingdesign problems, and solving them as early in the design cycle as possible, is the key to keeping projects on time and within budget. Too often, product design and performance problems are not detected until late in the product development cycle — when the product is ready to be shipped. The old adage holds true: It costs a penny to make a change in engineering, a dime in production and a dollar after a product is in the field. Verification is a Quality control process that is used to evaluate whether or not a product, service, or system complies with regulations, specifications, or conditions imposed at the start of a development phase. Verification can be in development, scale-up, or production. This is often an internal process. Validation is a Quality assurance process of establishing evidence that provides a high degree of assurance that a product, service, or system accomplishes its intended requirements. This often involves acceptance of fitness for purpose with end users and other product stakeholders.

8.1.4 SYSTEM TESTING

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have successfully passed integration testing and also the software system itself integrated with any applicable hardware system(s). System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole. System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification(s)

9.**RESULTS**

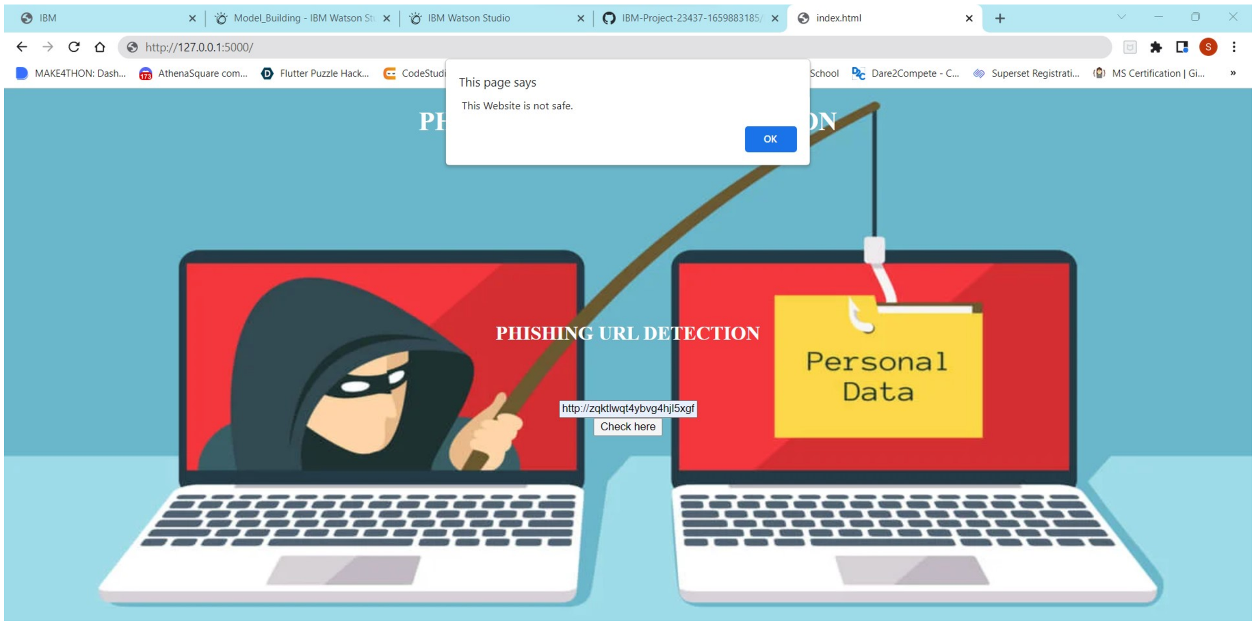
9.1 Performance Metrics



This page says

This Website is safe.

www.google.com



This page says

This Website is not safe.

http://zqktlwqt4ybvg4hjl5xgf

10. **ADVANTAGES & DISADVANTAGES**

Advantages

• This system can be used by many E-commerce or other websites in order to have good customer relationship.

• User can make online payment securely.

• Data mining algorithm used in this system provides better performance as compared to other traditional classifications algorithms.

• With the help of this system user can also purchase products online without any hesitation.

Disadvantages

• If Internet connection fails, this system won't work.

• All websites related data will be stored in one place.

11. **CONCLUSION**

It is outstanding that a decent enemy of phishing apparatus ought to anticipate the phishing assaults in a decent timescale. We accept that the accessibility of a decent enemy of phishing device at a decent time scale is additionally imperative to build the extent of anticipating phishing sites. This apparatus ought to be improved continually through consistent retraining. As a matter of fact, the accessibility of crisp and cutting-edge preparing dataset which may gained utilizing our very own device [30, 32] will help us to retrain our model consistently and handle any adjustments in the highlights, which are influential in deciding the site class. Albeit neural system demonstrates its capacity to tackle a wide assortment of classification issues, the procedure of finding the ideal structure is very difficult, and much of the time, this structure is controlled by experimentation.Our model takes care of this issue via computerizing the way toward organizing a neural system conspire; hence, on the off chance that we construct an enemy of phishing model and for any reasons we have to refresh it, at that point our model will encourage this procedure, that is, since our model will mechanize the organizing procedure and will request scarcely any client defined parameters.

12. **FUTURE SCOPE**

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. Measure the degrees of corporate and employee vulnerability.Eliminate the cyber threat risk level.Increase user alertness to phishing risks.Instill a cyber security culture and create cyber security heroes.Change behavior to eliminate the automatic trust response.Deploy targeted anti-phishing solutions.Protect valuable corporate and personal data.Meet industry compliance obligations.Assess the impacts of cyber security awareness training. Segment phishing simulation. Phishing is one of the most common and most dangerous attacks among cybercrimes. The aim of these attacks is to steal the information used by individuals and organizations to conduct transactions. When a user opens a fake webpage and enters the username and protected password, the credentials of the user are acquired by the attacker which can be used for malicious purposes. Phishing websites look very similar in appearance to their corresponding legitimate websites to attract large number of Internet users.

13. **APPENDIX**

Links:

GitHub link : https://github.com/IBM-EPBL/IBM-Project-23437-1659883185

Project Demo Link :
https://drive.google.com/file/d/1pXRjmJvyGQYOi7FZw4NqjA_SZvfpfRbI/view?usp=share_link

(The video is aslo uploaded on GitHub, incase of technical error while accessing the video.)