NALAIYA THIRAN - IBM PROJECT REPORT

(19IT410T Professional Readiness for Innovation, Employability and Entrepreneurship)

ON

WEB PHISHING DETECTION

Submitted by

TEAM ID: PNT2022TMID23587

THARIKA JAYARAJ (113219071045)

HARIPRIYA P (113219071012)

MIRUDHULA S V (113219071020)

SAHITHYA V (113219071034)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATIONTECHNOLOGY

VELAMMAL ENGINEERING COLLEGE, CHENNAI-66.

(An Autonomous Institution, Affiliated to Anna University, Chennai)

2022-2023

VELAMMAL ENGINEERING COLLEGE
CHENNAI -66

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this NALAIYA THIRAN – IBM PROJECT REPORT **“WEB PHISHING DETECTION”** is the Bonafide work of “THARIKA JAYARAJ (113219071045), HARIPRIYA P(113219071012), MIRUDHULA S V (113219071020), and SAHITHYA V (113219071034)” carried out in “PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP (NALAIYA THIRAN-IBM PROJECT)**”** during the Academic Year 2022-2023.

FACULTY EVALUATOR	**HEAD OF THE DEPARTMENT**
DR. J. SATHYA PRIYA	**DR. JEEVAA KATIRAVAN**
Associate Professor	**Professor and Head**
Dept. of Information Technology	Dept. of Information Technology
Velammal Engineering College	Velammal Engineering College
Chennai-600 066	Chennai-600 066

CONTENTS

1. **INTRODUCTION**
   Project Overview
   Purpose
2. **LITERATURE SURVEY**
   Existing problem
   References
   Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
   Empathy Map Canvas
   Ideation & Brainstorming
   Proposed Solution
   Problem Solution fit
4. **REQUIREMENT ANALYSIS**
   Functional requirement
   Non-Functional requirements
5. **PROJECT DESIGN**
   Data Flow Diagrams
   Solution & Technical Architecture
   User Stories
6. **PROJECT PLANNING & SCHEDULING**
   Sprint Planning & Estimation
   Sprint Delivery Schedule
   Reports from JIRA
7. **CODING & SOLUTIONING:**
   Feature 1 - building html pages
   Feature 2 - That data contains several factors should be considered when deciding whether a website URL is phishing.
8. **TESTING**
   Test Cases
   User Acceptance Testing
9. **RESULTS**
   Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX**
    Source Code
    GitHub & Project Demo Link

ABSTRACT

Phishing is the most commonly used social engineering and cyber-attack. Through such attacks, the phisher targets naive online users by tricking them into revealing confidential information, with the purpose of using it fraudulently. In order to avoid getting phished, Users should have awareness of phishing websites. Have a blacklist of phishing websites which requires the knowledge of website being detected as phishing. Detect them in their early appearance, using machine learning and deep neural network algorithms. Of the above three, the machine learning based method is proven to be most effective than the other methods. A phishing website is a common social engineering method that mimics trustful uniform resource locators (URLs) and web pages. 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 measured and compared.

Keywords: Deep learning, Machine learning, Phishing website attack, Phishing website detection, Anti-phishing website, Legitimate website, Phishing website datasets, Phishing website features.

PRE-REQUISITES TOOLS: JUPYTER NOTEBOOK

OPERATING SYSTEM: WINDOWS 11

LANGUAGE: PYTHON

INSTALLINGLIBRARIES

In this first step, we have to import the most common libraries used in python for machine learning such as

- Pandas
- NumPy
- Seaborn
- Matplotlib

IMPORTING DATA:

In this project, we have used the URL preprocessed data

1 . Introduction

1.1 Project Overview:

Phishing is a social engineering attack that aims at exploiting the weakness found in system processes as caused by system users. For example, a system can be technically secure enough against password theft, however unaware end users may leak their passwords if an attacker asked them to update their passwords via a given Hypertext Transfer Protocol (HTTP) link, which ultimately threatens the overall security of the system or

over, technical vulnerabilities (e.g., Domain Name System (DNS) cache poisoning) can be used by attackers to construct far more persuading socially- engineered messages (i.e., use of legitimate, but spoofed, domain names can be far more persuading than using different domain names). This makes phishing attacks a layered problem, and an effective mitigation would require addressing issues at the technical and human layers.

Since phishing attacks aim at exploiting weaknesses found in humans (i.e.,system end-users), it is difficult to mitigate them. For example, as evaluated in end-users failed to detect 29% of phishing attacks even when trained with the best performing user awareness program. On the other hand, software phishing detection techniques are evaluated against bulk Phishing attacks, which makes their performance practically unknown with regards to targeted forms of phishing attacks. These limitations in phishing mitigation techniques have practically resulted in security breaches against several organizations including leading information security providers.

In order to address the limitations of the previous definitions above, we consider phishing attacks as semantic attacks which use electronic communication channels (such as Email's, HTTP, SMS, VoIP, etc...) to communicate socially engineered messages to persuade victims to perform certain actions (without restricting the actions) for an attacker's benefit (without restricting the benefits) Phishing is a type of computer attack that communicates socially engineered messages to humans via electronic communication channels in order to persuade them to perform certain actions for the attacker's benefit. For example, the performed action (which the attacker persuades the victim to perform it) for a PayPal user is submitting his/her login credentials to a fake website that looks similar to PayPal. As a perquisite, this also implies that the attack should create a need for the end-user to perform such action, such as informing him that his/her account would be suspended unless he logs in to update certain pieces of information.

1.2 PURPOSE:

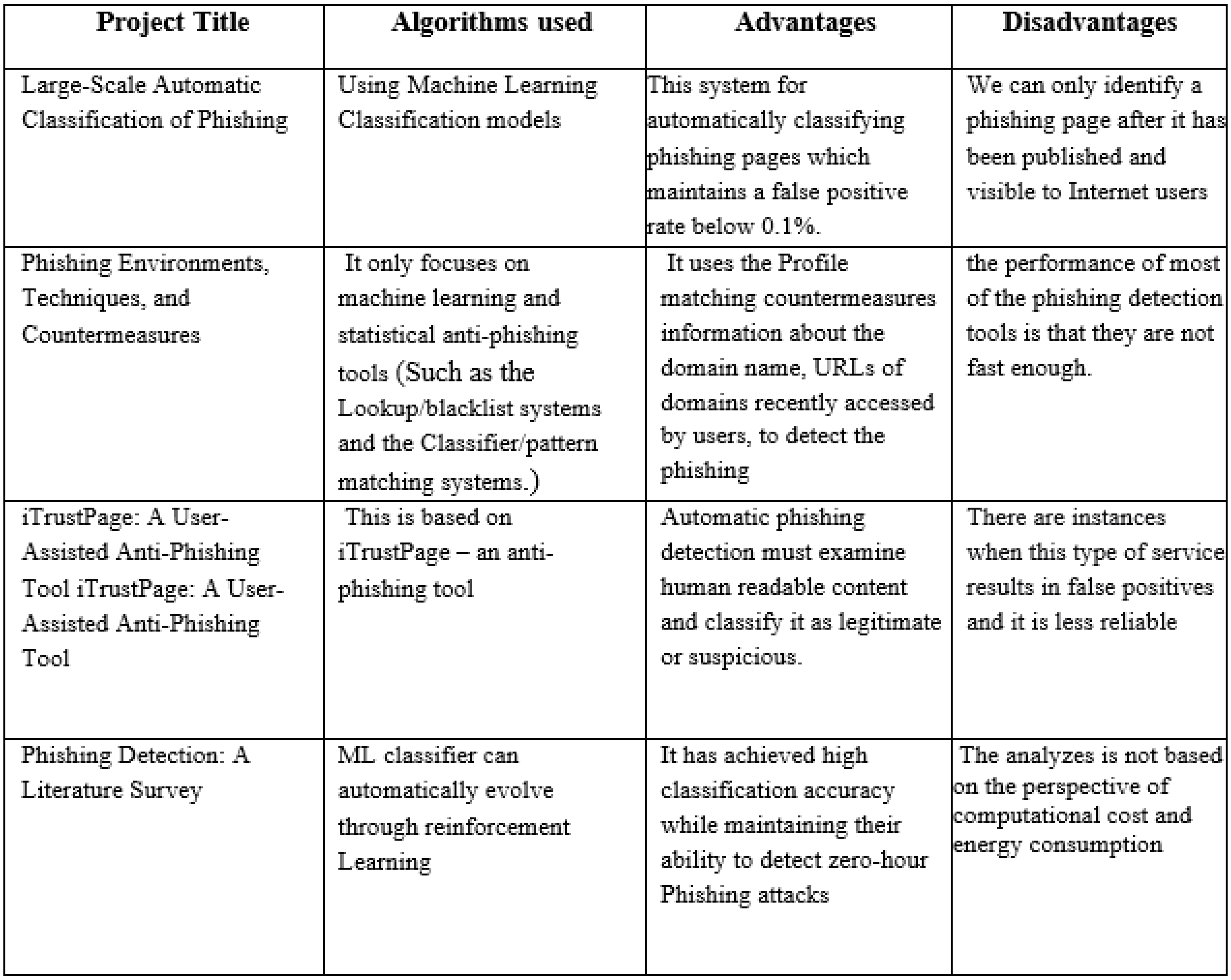
Web Phishing Detection Category: Machine Learning Objective A phishing website is a common social engineering method that mimics trustful uniform resource locators (URLs) and web pages. The objective of this project is to train machine learning models on the dataset given to predict phishing websites.

There have been several recent studies against phishing based on the characteristics of a domain, such as website URLs, website content, incorporating both the website URLs and content, the source code of the website and the screenshot of the website. However, there is a lack of useful anti-phishing tools to detect malicious URL in an organization to protect its users. In the event of malicious code being implanted on the website, hackers may steal user information and install malware, which poses a serious risk to cyber security and user privacy. Malicious URLs on the Internet can be easily identified by analyzing it through Machine Learning (ML) technique.

Phishing detection schemes which detect phishing on the server side are better than phishing prevention strategies and user training systems. These systems can be used either via a web browser on the client or through specific host-site software. Presents the classification of Phishing detection approaches. Heuristic and ML based approach is based on supervised and unsupervised learning techniques. It requires features or labels for learning an environment to make a prediction. Proactive phishing URL detection is similar to ML approach. However, URLs are processed and support a system to predict a URL as a legitimate or malicious.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM:



Project Title	Algorithms used	Advantages	Disadvantages
Large-Scale Automatic Classification of Phishing	Using Machine Learning Classification models	This system for automatically classifying phishing pages which maintains a false positive rate below 0.1%.	We can only identify a phishing page after it has been published and visible to Internet users
Phishing Environments, Techniques, and Countermeasures	It only focuses on machine learning and statistical anti-phishing tools (Such as the Lookup/blacklist systems and the Classifier/pattern matching systems.)	It uses the Profile matching countermeasures information about the domain name, URLs of domains recently accessed by users, to detect the phishing	the performance of most of the phishing detection tools is that they are not fast enough.
iTrustPage: A User-Assisted Anti-Phishing Tool iTrustPage: A User-Assisted Anti-Phishing Tool	This is based on iTrustPage – an anti-phishing tool	Automatic phishing detection must examine human readable content and classify it as legitimate or suspicious.	There are instances when this type of service results in false positives and it is less reliable
Phishing Detection: A Literature Survey	ML classifier can automatically evolve through reinforcement Learning	It has achieved high classification accuracy while maintaining their ability to detect zero-hour Phishing attacks	The analyzes is not based on the perspective of computational cost and energy consumption

2.2 REFERENCES:

[1] S. Sheng, M. Holbrook, P. Kumaraguru, L. F. Cranor, and J. Downs, “Who falls for phish?: a demographic analysis of phishing susceptibility and effectiveness of interventions,” in Proceedings of the 28th international conference on Human factors in computing systems, ser.CHI’10. NewYork,NY, USA: ACM, 2010,pp. 373–382.

[2] B. Krebs, “H. B. Gary Federal hacked ,”http://krebsonsecurity.com/2011/02/hbgary-federal-hacked-by-anonymous/,2011, accessed December 2011.

[3] B. Schneier, “Lockheed Martin hack linked to RSA’s SecurID breach”, http://www.schneier.com/blog/archives/2011/05/lockheedmartin.html, 2011 ,accessed December 2011.

[4] C.Whittaker, B.Ryner, and M.Nazif, “Large-scale automatic classification of phishing pages,” in NDSS’10 ,2010.

2.3 PROBLEM STATEMENT DEFINISION:

The goal of our project is to implement a machine learning solution to the problem of detecting phishing and malicious web links. The end result of our project will be a software product which uses machine learning algorithm to detect malicious URLs. Phishing is the technique of extracting user credentials and sensitive data from users by masquerading as a genuine website. In phishing, the user is provided with a mirror website which is identical to the legitimate one but with malicious code to extract and send user credentials to phishers.

Phishing attacks can lead to huge financial losses for customers of banking and financial services. The traditional approach to phishing detection has been to either to use a blacklist of known phishing links or heuristically evaluate the attributes in a suspected phishing page to detect the presence of malicious codes.

The heuristic function relies on trial and error to define the threshold which is used to classify malicious links from benign ones. The drawback to this approach is poor accuracy and low adaptability to new phishing links. We plan to use machine learning to overcome these drawbacks by implementing some classification algorithms and comparing the performance of these algorithms on our dataset. We will test algorithms such as Logistic Regression, SVM, Decision Trees and Neural Networks on a dataset of phishing links from UCI Machine Learning repository and pick the best model to develop a browser plug in, which can be published as chrome extension.

3. IDEATION AND PROPOSED SOLUTION:

3.1. EMPATHY MAPCANVAS:

An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment. The empathy map was originally created by Dave Gray and has gained much popularity within the agile community.

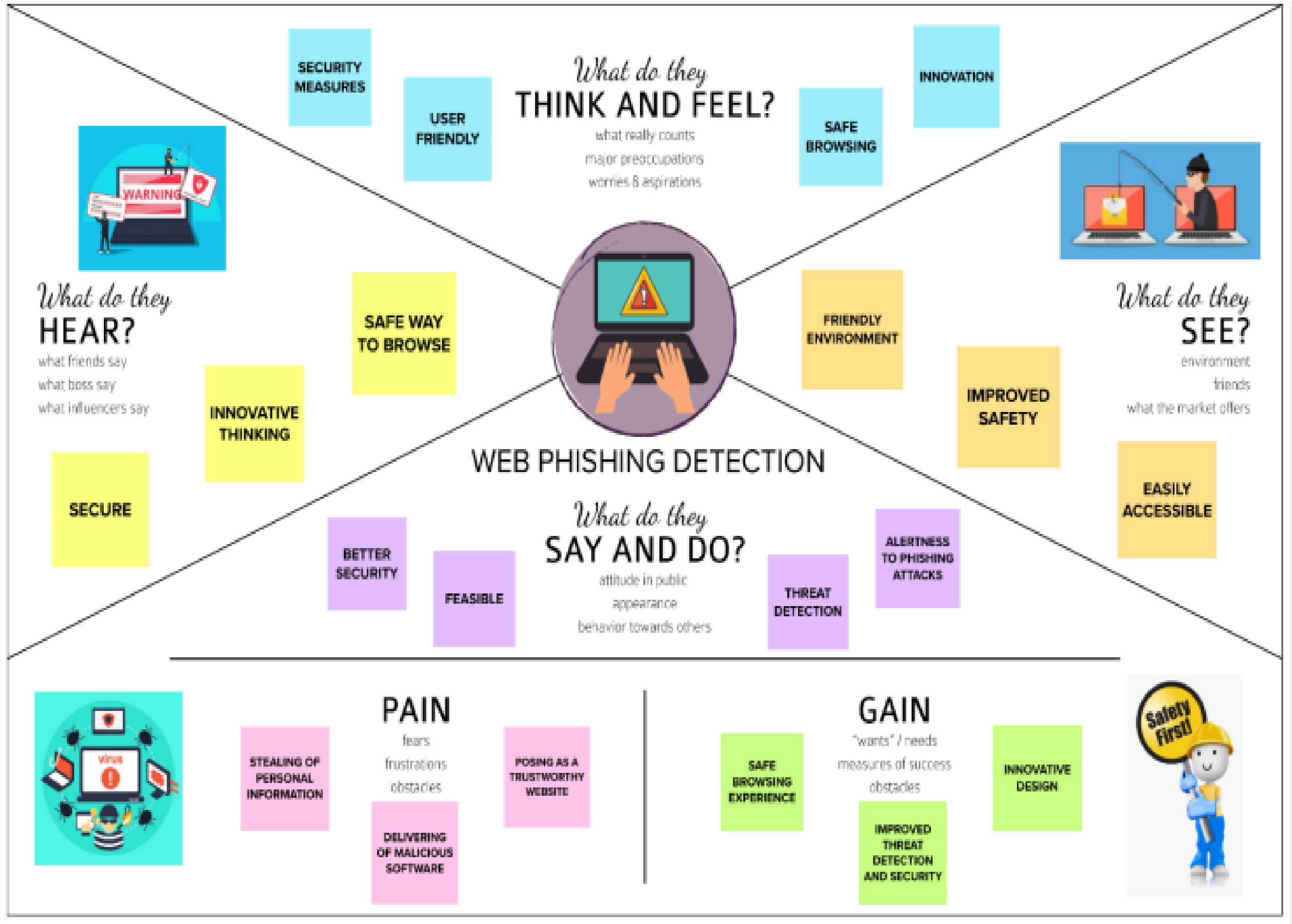


FIG: EMPATHY MAP

Ideation & Brainstorming

Ideas laid out by each Team Member

HARI PRIYA P:

- Idea 1: To create a friendly environment for the users
- Idea 2: Makes sure that the threat is detected.
- Idea 3: Ensure user's safety.
- Idea 4: To create a safe way to browse.

MIRUDHULA S V:

- Idea 1: To make sure the safety of the user.
- Idea 2: The malicious should be easily detected.
- Idea 3: The security for user is important.
- Idea 4: the browser should be Feasible.

SAHITHYA V:

- Idea 1: The web page should be accessible by everyone.
- Idea 2: Should satisfy the user's expectation.
- Idea 3: The results should be accurate.
- Idea 4: To make the browser flexible.

THARIKA JAYARAJ:

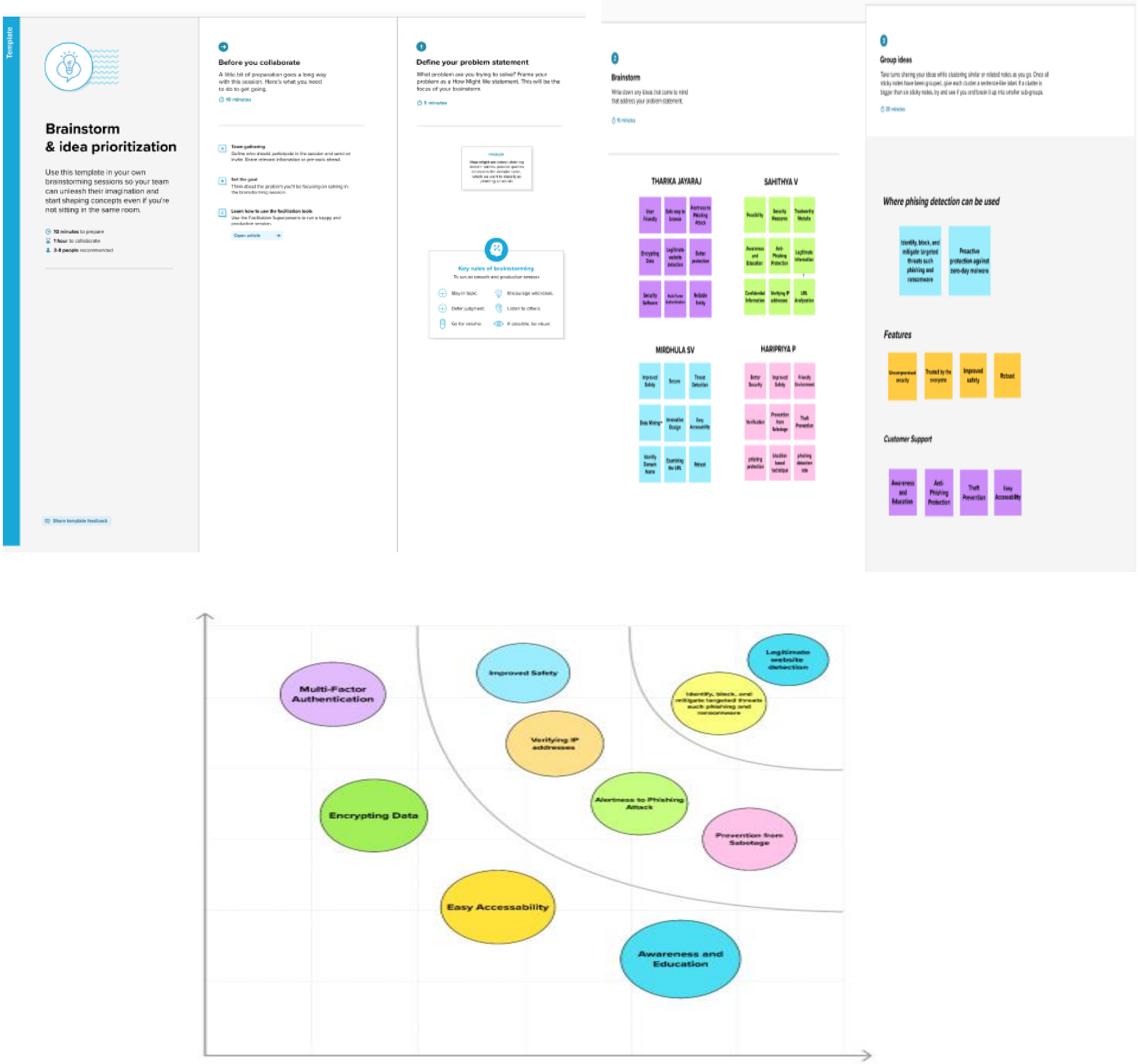
- Idea 1: Should alert the phishing attack.
- Idea 2: Innovative measures to prevent the phishing.
- Idea 3: To detect whether the browser is safe or not.
- Idea 4: Safest way to access the web browser.
-

SHORTLISTED IDEAS:

- Idea 1: Makes sure that the threat is detected.
- Idea 2: The results should be accurate.
- Idea 3: Safest way to access the web browser.

3.2. IDEATION & BRAINSTORMING

Brainstorming is a group problem-solving method that involves the spontaneous contribution of creative ideas and solutions. This technique requires intensive, freewheeling discussion in which every member of the group is encouraged to think aloud and suggest as many ideas as possible based on their diverse knowledge.



Template

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- 10 minutes to prepare
- 1 hour to collaborate
- 2-8 people recommended

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

- **Team gathering** Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.
- **Set the goal** Think about the problem you'll be focusing on solving in the brainstorming session.
- **Learn how to use the facilitation tools** Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

[illegible]

Key rules of brainstorming

To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

2 Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

3 Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

FIG: BRAINSTORMING

3.3 PROPOSED SOLUTION:

- Phishing is a fraudulent technique that is used over the internet to manipulate user to extract their personal information such as Username, Passwords, Credit Cards, Bank Account information etc.
- Phishing use multiple methods, including E-mail, Uniform Resource Locators (URL's), Instant messages, Form posting, Telephone calls and Text messages to steal user information.
- Many cypher infiltrations are accomplished through phishing attacks where user is tricked into interacting with web pages that appear to be legitimate.

Idea / Solution Description:

- This project aims to develop these methods of defense utilizing various approaches to categorizing Websites and narrow them down to the best Machine Learning algorithm by comparing the accuracy rate, false positive and false negative rate of each algorithm.
- To find unknown malicious URLs compared to the blacklist approach. iii. And use
- anti-phishing protection and anti- spam software to protect yourself.

Novelty/Uniqueness:

- Our model uses the power of Machine learning to detect phishing sites.
- Python serves as a powerful tool to execute the application with Low false positives, High accuracy.
- Uses the latest techniques that gives an efficient and great performance.
- It can easily differentiate the fake and safe URL's. If it's fake means, a warning message will be intimate to the user.

Feasibility Of Ideas:

- Using data visualization and machine learning algorithm, we safeguard the user’s data by detecting malicious websites.
- This application is easy to be built we have a lot of existing software tools that aid us in creating a web phishing detector.
- Faster, easier and seamless performance can be obtained. Business Model:
- Our model can be used by all users to secure their data from malicious websites.

Social Impact:

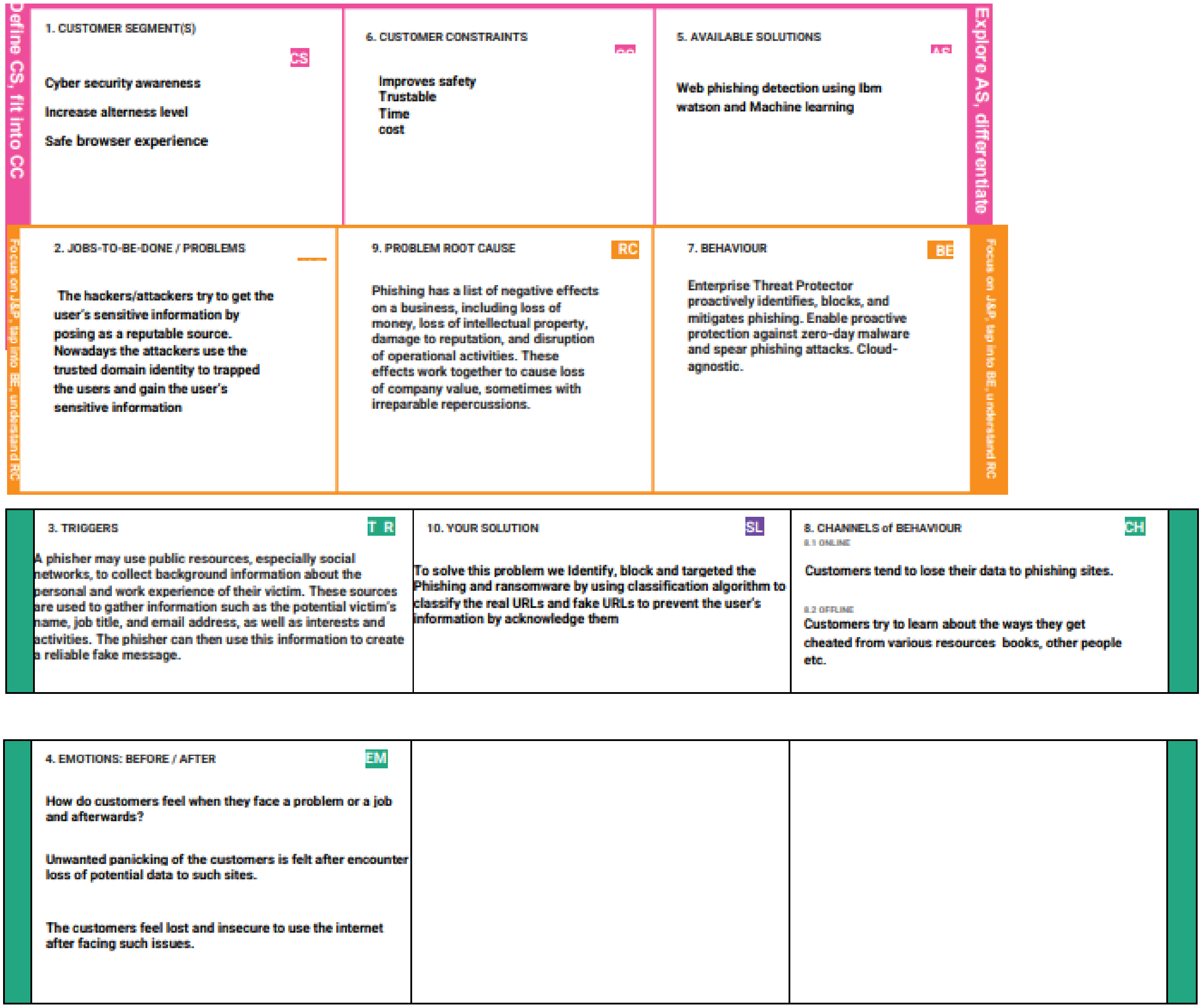
- According to recent research by Google, these was a 4505 increase in phishing websites from January to March 2021.
- Phishing has a list of negative effects on a business, including loss of money, loss of intellectual property, damage to reputation, and disruption of operational activities
- As an impact of this model, people can able to find fraudulent websites of fake ones.
- So that, they can avoid sharing sensitive data with unrecognized websites.

Scalability of the Solution:

- A-part from E-Banking sector the idea proposed can be developed into platform independent model.
- Adapts to all sort of web application and ease of preventing users from scam

3.4 PROBLEM SOLUTION FIT:

- ➢ This occurs when you have evidence that customers care about certain jobs, pains, and gains.
- ➢ At this stage you've proved the existence of a problem and have designed a value proposition that addresses your customers' jobs, pains and gains.



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Cyber security awareness

Increase alterness level

Safe browser experience

6. CUSTOMER CONSTRAINTS CC

Improves safety
Trustable
Time
cost

5. AVAILABLE SOLUTIONS AS

Web phishing detection using Ibm watson and Machine learning

Explore AS, differentiate

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

2. JOBS-TO-BE-DONE / PROBLEMS

The hackers/attackers try to get the user's sensitive information by posing as a reputable source. Nowadays the attackers use the trusted domain identity to trapped the users and gain the user's sensitive information

9. PROBLEM ROOT CAUSE RC

Phishing has a list of negative effects on a business, including loss of money, loss of intellectual property, damage to reputation, and disruption of operational activities. These effects work together to cause loss of company value, sometimes with irreparable repercussions.

7. BEHAVIOUR BE

Enterprise Threat Protector proactively identifies, blocks, and mitigates phishing. Enable proactive protection against zero-day malware and spear phishing attacks. Cloud-agnostic.

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

3. TRIGGERS TR

A phisher may use public resources, especially social networks, to collect background information about the personal and work experience of their victim. These sources are used to gather information such as the potential victim's name, job title, and email address, as well as interests and activities. The phisher can then use this information to create a reliable fake message.

10. YOUR SOLUTION SL

To solve this problem we Identify, block and targeted the Phishing and ransomware by using classification algorithm to classify the real URLs and fake URLs to prevent the user's information by acknowledge them

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

Customers tend to lose their data to phishing sites.

8.2 OFFLINE

Customers try to learn about the ways they get cheated from various resources books, other people etc.

4. EMOTIONS: BEFORE / AFTER EM

How do customers feel when they face a problem or a job and afterwards?

Unwanted panicking of the customers is felt after encounter loss of potential data to such sites.

The customers feel lost and insecure to use the internet after facing such issues.

4. REQUIREMENTANALYSIS

There are two types of requirements, such as

- Functional requirement.
- Non-functional requirement.

4.1 Functional requirement:

Functional requirements are the desired operations of a program, or system as defined in software development and systems engineering. The systems in systems engineering can be either software electronic hardware or combination software-driven electronics.

Following are the functional requirements of the proposed solution.

FR No.	**Functional Requirement(Epic)**	**Sub Requirement(Story/Sub-Task)**
FR-1	UserRegistration	Registration through Form Registration through Gmail Registration through LinkedIn Registration through Mobile Number
FR-2	UserConfirmation	Confirmation via Email Confirmation via OTP
FR-3	WebsiteComparison	Blacklist filtering and White list filtering techniques are used to compare the website URL.
FR-4	FeatureSelection	Based on the length of an URL, number of dots in URL and check for the correct spelling and grammar.
FR-5	Prediction	Model predicts the URL using Machine Learning Algorithms.
FR-6	Classifier	Send all the output to the classifier and produces the final output results.
FR-7	Detection	The model developed should have the capability to retrieve and display the correct accuracy of the website.

Non-functional requirement:

A non-functional requirement defines the quality attribute of a software system. It specifies "What should the software system do?". It places constraints on "How should the software system fulfill the functional requirements?"

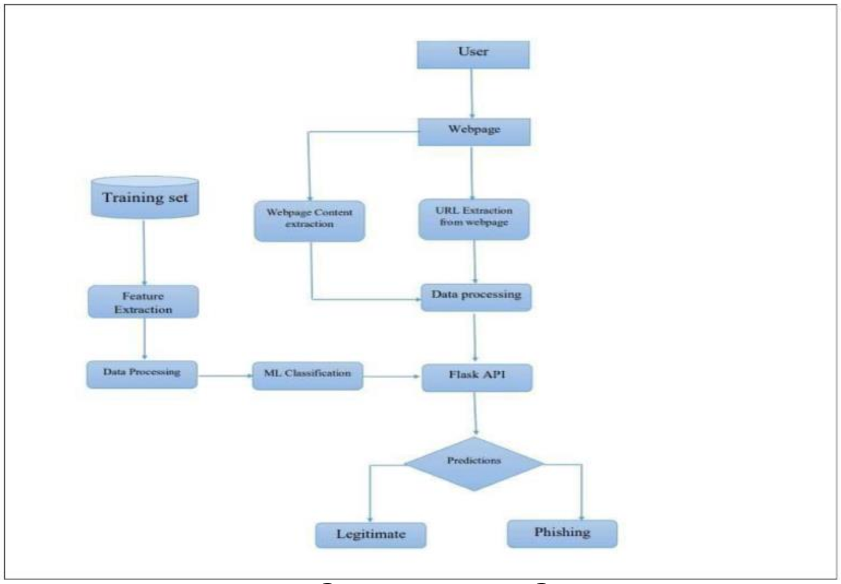
FR No.	**Non-FunctionalRequirement**	**Description**
NFR-1	**Usability**	The user should have the full access to login into website without asking the personal details like password, credit card and debit card number.
NFR-2	**Security**	To find that whether the particular website is secure or not we can either send a mail or we can either notify it by showing a warning message while using the websites.
NFR-3	**Reliability**	The website should be more trustworthy to the user when they are using it.
NFR-4	**Performance**	The performance of the website is mainly based on the how quickly the site content loads and displays the screen and how well it responds to the user interaction.
NFR-5	**Availability**	The website should be available to all the users to access the resources.
NFR-6	**Scalability**	The website should be able to handle the large number of users at a same time without getting hanged or disrupting the users.

Table: Non-functional Requirement

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAM:

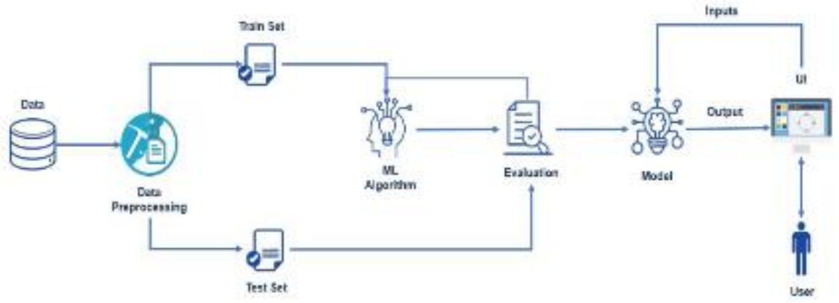
It's easy to understand the flow of data through systems with the right data flow diagram software. This guide provides everything you need to know about data flow diagrams, including definitions, history, and symbols and notations. You'll learn the different levels of a DFD, the difference between a logical and a physical DFD and tips for making a DFD.



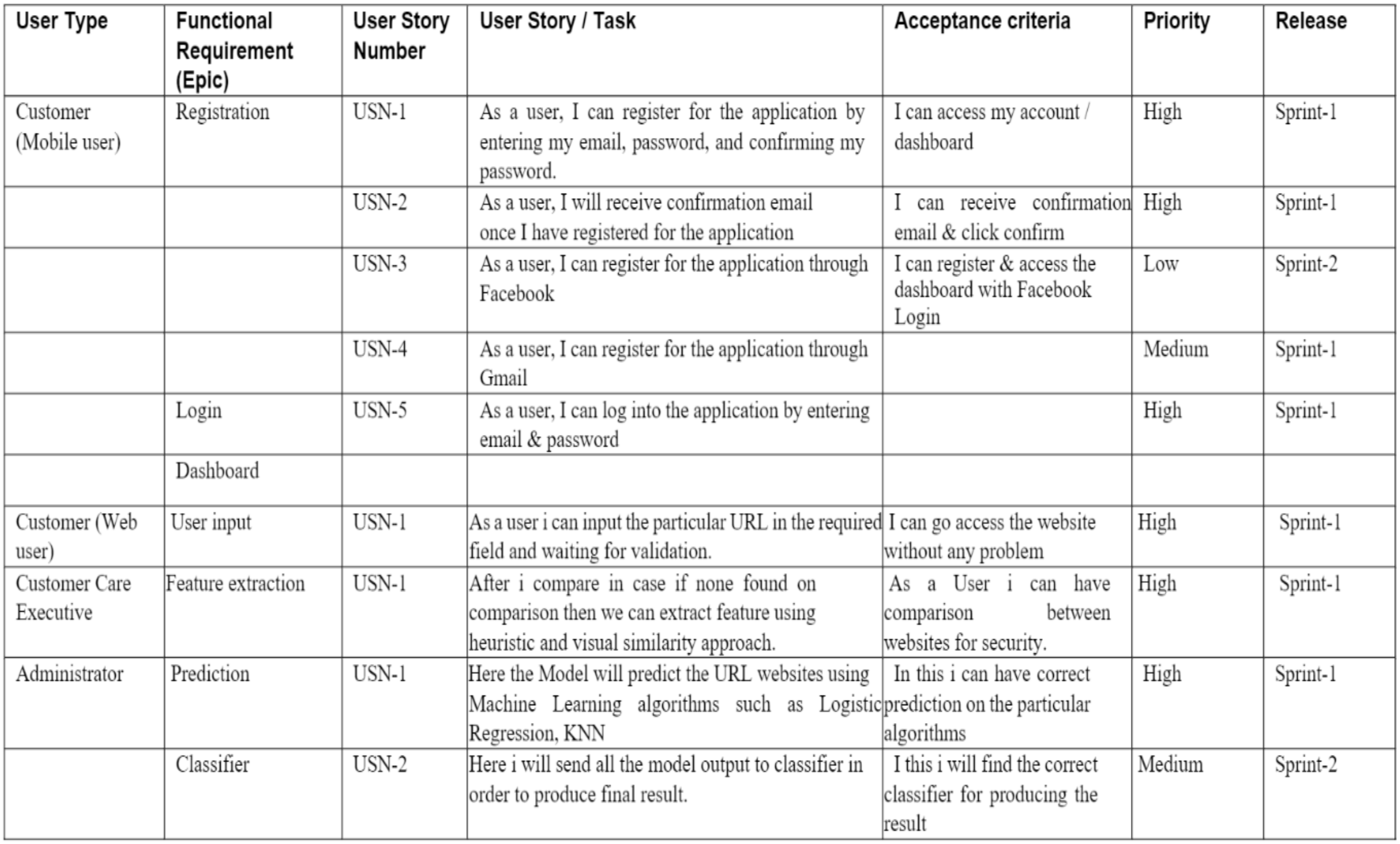
5.2 SOLUTION AND TECHNICAL ARCHITECTURE:

Solution architects oversee these tasks and activities and monitor a team's progress to keep the project on schedule. In contrast, technical architects complete the tasks to implement IT strategies. They ensure the solutions identified by other architects’ function correctly with the company's existing infrastructure.

TECHNOLOGY ARCHITECTURE:



5.3 USER STORIES:

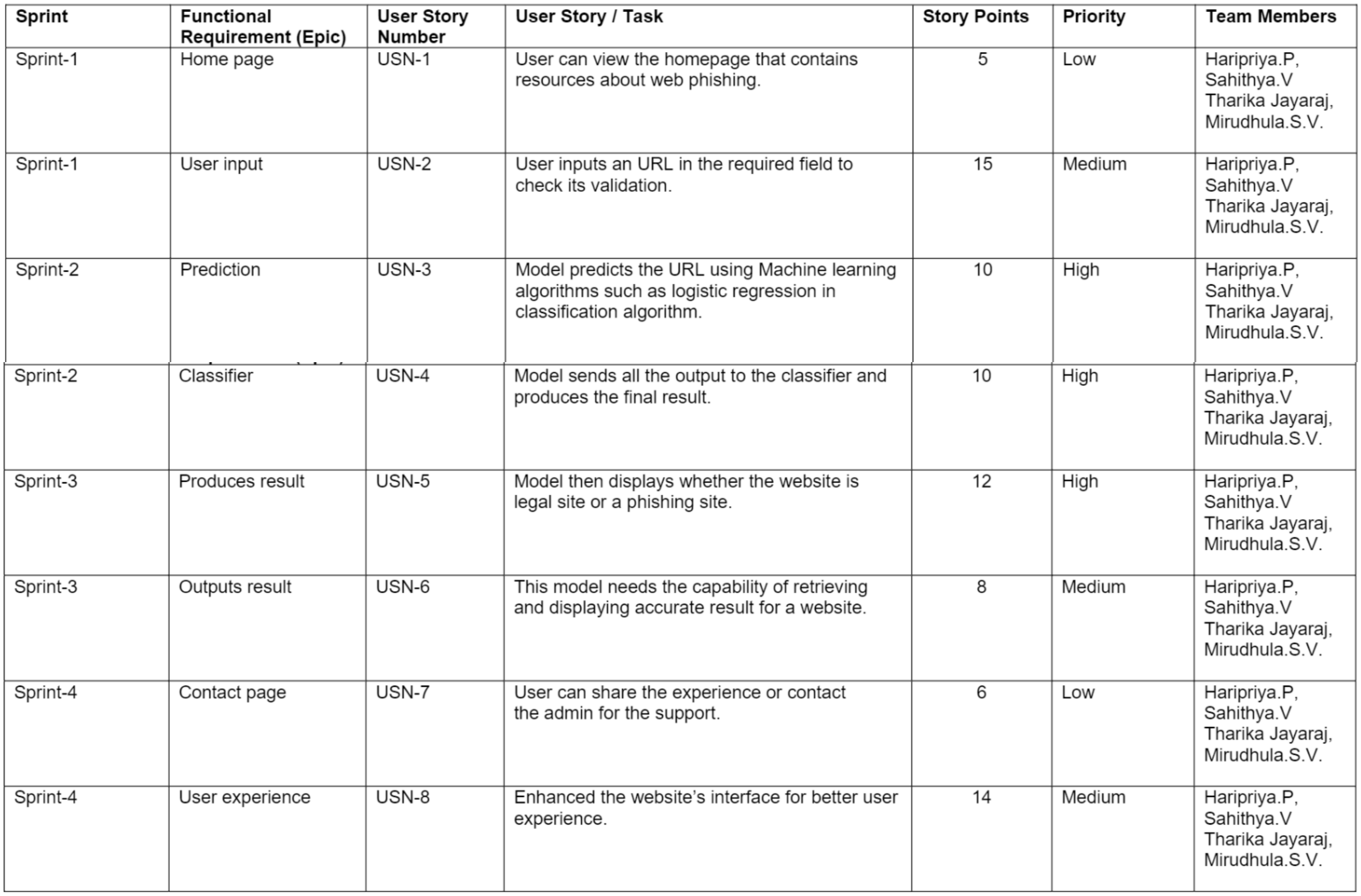


User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	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		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

The deliverable shall include the architectural diagram as below and the information as per the table.

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation



Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Home page	USN-1	User can view the homepage that contains resources about web phishing.	5	Low	Haripriya.P, Sahithya.V Tharika Jayaraj, Mirudhula.S.V.
Sprint-1	User input	USN-2	User inputs an URL in the required field to check its validation.	15	Medium	Haripriya.P, Sahithya.V Tharika Jayaraj, Mirudhula.S.V.
Sprint-2	Prediction	USN-3	Model predicts the URL using Machine learning algorithms such as logistic regression in classification algorithm.	10	High	Haripriya.P, Sahithya.V Tharika Jayaraj, Mirudhula.S.V.
Sprint-2	Classifier	USN-4	Model sends all the output to the classifier and produces the final result.	10	High	Haripriya.P, Sahithya.V Tharika Jayaraj, Mirudhula.S.V.
Sprint-3	Produces result	USN-5	Model then displays whether the website is legal site or a phishing site.	12	High	Haripriya.P, Sahithya.V Tharika Jayaraj, Mirudhula.S.V.
Sprint-3	Outputs result	USN-6	This model needs the capability of retrieving and displaying accurate result for a website.	8	Medium	Haripriya.P, Sahithya.V Tharika Jayaraj, Mirudhula.S.V.
Sprint-4	Contact page	USN-7	User can share the experience or contact the admin for the support.	6	Low	Haripriya.P, Sahithya.V Tharika Jayaraj, Mirudhula.S.V.
Sprint-4	User experience	USN-8	Enhanced the website's interface for better user experience.	14	Medium	Haripriya.P, Sahithya.V Tharika Jayaraj, Mirudhula.S.V.

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 ReleaseDate(Actual)
Sprint-1	20	6Days	24Oct2022	29Oct2022	20	29Oct2022
Sprint-2	20	6Days	31Oct2022	05Nov2022	20	05Nov2022
Sprint-3	20	6Days	07Nov2022	12Nov2022	20	12Nov2022
Sprint-4	20	6Days	14Nov2022	19Nov2022	20	19Nov2022

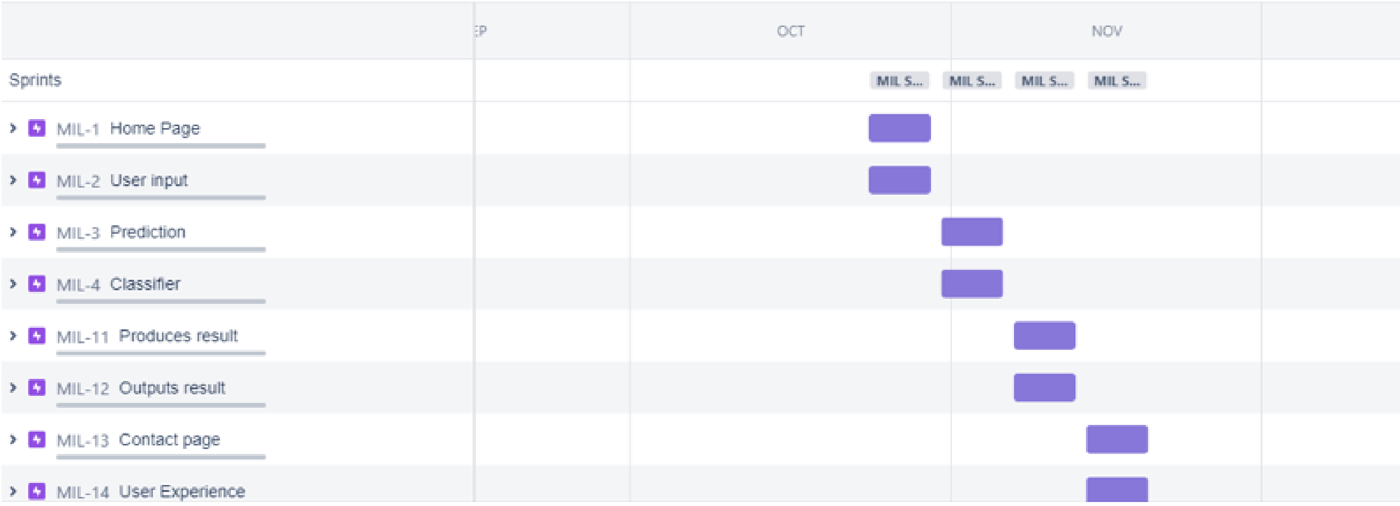
Velocity:
Imagine we have a10-days print duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day).

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

We have a 6-day sprint duration, and the velocity of the team is 20 (points per sprint). So, our team’s average velocity (AV) per iteration unit (story points per day).

AV= (Sprint Duration /Velocity)=20/6=3.34

6.3 REPORTS FROM JIRA:



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

7.1 **Feature 1 –** Building the HTML Pages

#app.py

```
 import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle
#importing the inputScript file used to analyze the URL
import inputScript


#load model
app = Flask(__name__)
model = pickle.load(open(r"C:\Users\Tharika\JSCODE\Web phishing detection\Flask\
Phishing_website.pkl", 'rb'))

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

 @app.route('/final')
 def final():
 return render_template("Final.html")

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

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

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

#Redirects to the page to give the user input URL.
@app.route('/predict')
def predict():
 return render_template('Final.html')

#Fetches the URL given by the URL and passes to inputScript
@app.route('/y_predict',methods=['POST'])
def y_predict():
'''
For rendering results on HTML GUI
url = request.form['URL']
checkprediction = inputScript.main(url)
print(checkprediction)
```

```
prediction = model.predict(checkprediction)
print(prediction)
output=prediction[0]
 if(output==1):
 pred="Your are safe!!  This is a Legitimate Website."
else:
 pred="You are on the wrong site. Be cautious!"
 return render_template('Final.html', prediction_text='{}'.format(pred),url=url)

#Takes the input parameters fetched from the URL by inputScript and returns the predictions
@app.route('/predict_api',methods=['POST'])
 def predict_api():
 '''
 For direct API calls trought request
 '''
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( debug=True)


 #index.html
 <!DOCTYPE html>

     <html lang="en">
     <head>
     <meta charset="UTF-8">
     <meta http-equiv="X-UA-Compatible" content="IE=edge">
     <meta name="viewport" content="width=device-width, initial-scale=1.0">
     <title>SecureSurfers</title>
     <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/style1.css') }}">
     <link rel="stylesheet" href="style1.css">
     </head>
     <body>
     <nav>
     <ul>
     <li id="websiteName">SecureSurfers.</li>
     <li><a href="{{ url_for('contact') }}">Contact</a></li>
     <li><a href="{{ url_for('about')+'#about' }}">About</a></li>
     <li><a href="#">Home</a></li>

     </ul>
     </nav>
     <div class="sectionOne" id="home">
     <section id="section1">
     <imgsrc="{{ url_for('static', filename='images/img1.jpg') }}" id="homepageImg">
     </section>
     <section id="section2">
```

```
<div id="section2Div">
<span id="line1">Are you browsing on a safe website?!<br></span>
<br><br>
<span id="line2">Check the website now to know if the URL is valid and save yourself from phishing
attacks.</span>
<br><br><br><br><br>
<a href="{{ url_for('final')}}"><span id="checkWebsiteBtn" style="padding: 20px 40px;">Check your
website now</span></a>
</div>
</section>
</div>
<div class="sectionTwo" id="about">
<h1></h1>
<h2>About</h2>
<hr>
<br>
<section id="para">
<p id="paraOne" class="para">
Web service is one of the key communications software services for the internet. Web phishing is one
of the many security threats to web services on the Internet. Web phishing aims to steal private
information such as usernames, passwords, and credit card details, by the way of impersonating a
legitimate entity.
</p>
<p id="paraTwo" class="para">
The recipient is tricked into clicking a malicious link, which can lead to the installation of malware, the
freezing of the system as a part pf a randsomware attack or the stealing of sensitive information. It will
 lead to information disclosure and property damage.
</p>
</section>
</div>
</body>
</html>

   #final.html
   <!DOCTYPE html>
  <html lang="en">
  <head>
  <!-- meta tags-->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!--Css Attachment-->
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/style1.css') }}">
  <link rel="stylesheet" href="style1.css">
  <title>Final page</title>
  </head>
  <body>
  <nav>
  <ul>
  <li id="websiteName">SecureSurfers.</li>
  <li><a href="{{ url_for('contact') }}">Contact</a></li>
  <li><a href="{{ url_for('about')+'#about' }}">About</a></li>
```

```
<li><a href="{{ url_for('index') }}">Home</a></li>
</ul>
</nav>
<form action="{{ url_for('y_predict')}}"method="post">
<div id="phishingSection" class="boxContainer">
<h1>Phishing website detection using Machine Learning</h1>
<br><br>


<input type="text" id="urlInput" name="URL" placeholder="Enter the URL to be verified :"
style="padding:10px; width: 1000px;">
<br><br>
<button type="submit" class="btn" value="Check the URL" id="predictButton">Predict</button>

<div style="text-align: center ;">
<div id='result', style="color: rgb(233, 146, 96);padding-top: 3rem;font-size: 2.2rem;font-weight: 600
;" font-size:30px;>{{ prediction_text }}</div>
<br><a href=" {{ url }} "> {{ url }} </a>
</div>
</div>
</form>
</body>
</html>


#contact.html
<!DOCTYPE html>

          <html lang="en">
          <head>
          <meta charset="UTF-8">
          <meta http-equiv="X-UA-Compatible" content="IE=edge">
          <meta name="viewport" content="width=device-width, initial-scale=1.0">
          <title>SecureSurfers</title>
          <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/style1.css') }}">
          <link rel="stylesheet" href="style1.css">
          </head>
          <body>
          <nav>
          <ul>
          <li id="websiteName">SecureSurfers.</li>
          <li><a href="#">Contact</a></li>
          <li><a href="{{ url_for('about')+'#about' }}">About</a></li>
          <li><a href="{{ url_for('index') }}">Home</a></li>
          </ul>
          </nav>
          <div id="phishingSection">
          <section id="contact">
          <h1>Contact Us</h1>
          <form name="EmailForm" id="formid">
          <br><br>
          <p>Email-ID :</p><br>
```

```
<input type="text" id="mailId">

<br><br>
<p>Message :</p><br>
<textarea id="message"></textarea>
<br><br>
<button type="submit" value="Submit" id="predictButton">Submit</button>
</form>
<p id="output"></p>
</section>
</div>

</body>
<script>
document.getElementById('formid').onsubmit= function(){submit()};
function submit(){
alert("Thank you for your contacting us. Will get back to you soon.");
}
</script>
</html>
```

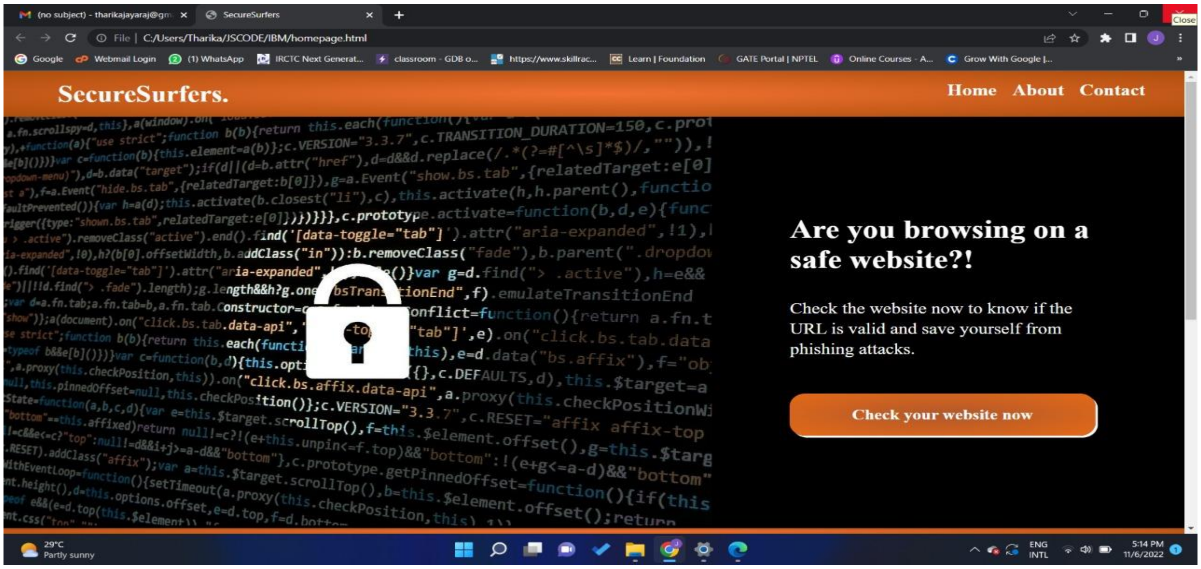
#style.css

```
*{
    margin:0 0;
    }
    body{
    background-color: black;
    height:100%;
    }
    nav{
    /* background-image:linear-gradient(to bottom,#380742,#731077); */
    /* background-color:#e66a17; */
    background-image:radial-gradient(#f27232 ,#ae4e0d);

    /* background-image:linear-gradient(to bottom,#7d2d08,#ad5013); */

    /* background-image:linear-gradient(to bottom,#0f0845,#202a65); */
    color: white;
    padding:15px 50px;
    }
    ul{
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    }

    li{
    float: right;
    text-decoration: none;
    list-style: none;
```

```
padding-left: 20px;
font-size: 25px;
font-weight: 600;

}
a{
text-decoration: none;
color: white;
}
li>a:hover{
color: rgb(46, 45, 45);
font-size: 30px;
}
#websiteName{
float: left;
font-size: 35px;
font-weight: 900;
}
.sectionOne{
color: white;
margin-bottom: 0%;
}
#section1{
width:60%;
float: left;
}
#section2{
width: 40%;
float:right;
}
#homepageImg{
width:100%;
height: 630px;
}
#section2Div{
padding: 150px 100px;
}
#line1{
font-size: 2.5em;
font-weight:600;
}
#line2{
font-size: 1.5em;
line-height: 1.3em;
}
#checkWebsiteBtn{
/* background-color: white; */
background-image:radial-gradient(#b1592d,#e66a17);
color: white;
padding: 0;
border-radius: 20px;
font-size: 1.4em;
```

```css
font-weight:800;
box-shadow:3px 3px white;
}
#checkWebsiteBtn:hover{
background-image:radial-gradient(#5c2e18,#964610);

box-shadow:3px 3px rgb(175, 173, 173);
/* box-shadow:none; */
}


.sectionTwo{
clear:both;
font-size: 25px;
font-weight: 550;
line-height: 40px;
color: black;
}
.sectionTwo h1{
background-image:radial-gradient(#f27232 ,#ae4e0d);
padding: 5px;
}
.sectionTwo h2{
color:white;
text-align:center;
padding:30px 80px;;
font-size: 50px;
}
#para{
padding:0px 40px;
padding-bottom: 3%;
display:flex;
}
#paraOne{
width:40%;
margin-right:5%;
margin-left: 3%;
float:left;
background-image:radial-gradient(#f27232 ,#893e0d);
padding:5%;
border-radius: 30px;
box-shadow:3px 3px white;


}
#paraTwo{
width:40%;
float:right;
background-image:radial-gradient(#f27232 ,#ae4e0d);
padding:5%;
border-radius: 30px;
box-shadow:3px 3px white;
```

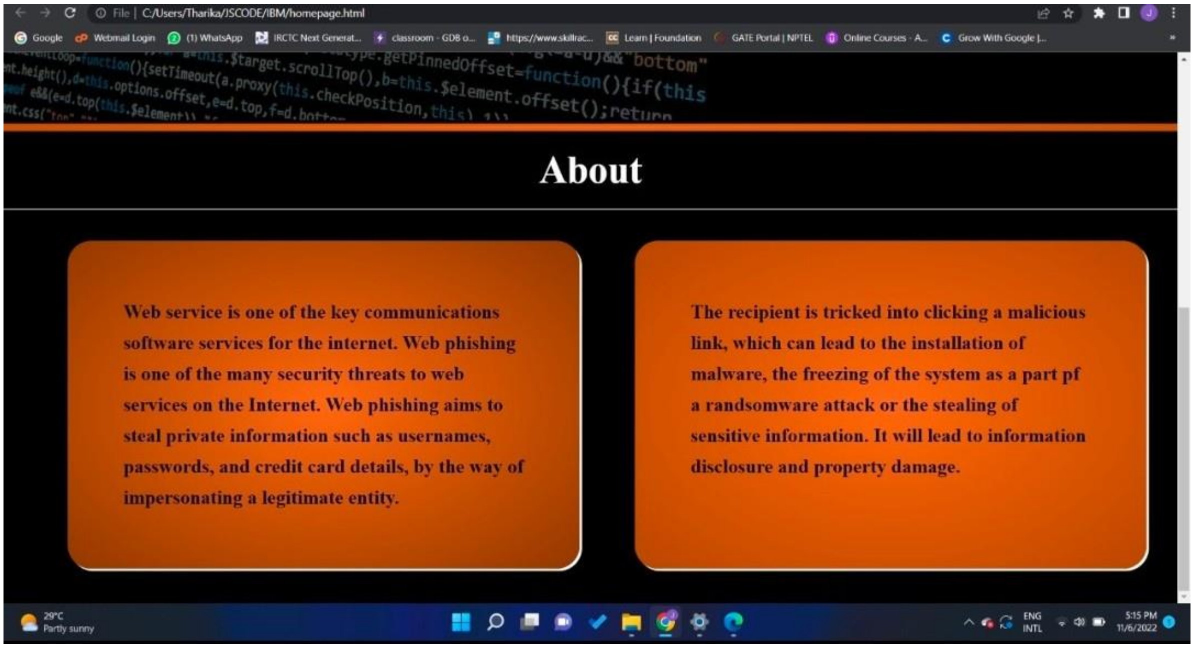
```
}

#phishingSection{
color: white;
text-align:center;
padding: 5%;
font-size: 20px;
}
#phishingSection>p{
font-size: 30px;
}
#urlInput{
padding: 10px;
font-size: 25px;
border-radius: 10px;
}
#predictButton{
font-size: 25px;
padding: 10px 50px;
background-image:radial-gradient(#b1592d,#e66a17);
color: white;
font-weight:500;
border-radius: 20px;
box-shadow:3px 3px white;
}
#predictButton:hover
{
background-image:radial-gradient(#5c2e18,#964610);
box-shadow:3px 3px rgb(175, 173, 173);
}
#mailId, #message{
padding: 5px 100px;
width:30%;
font-size: 25px;
border-radius: 20px;
}

#message{
padding: 30px 100px;
}
```



SecureSurfers.

Are you browsing on a safe website?!

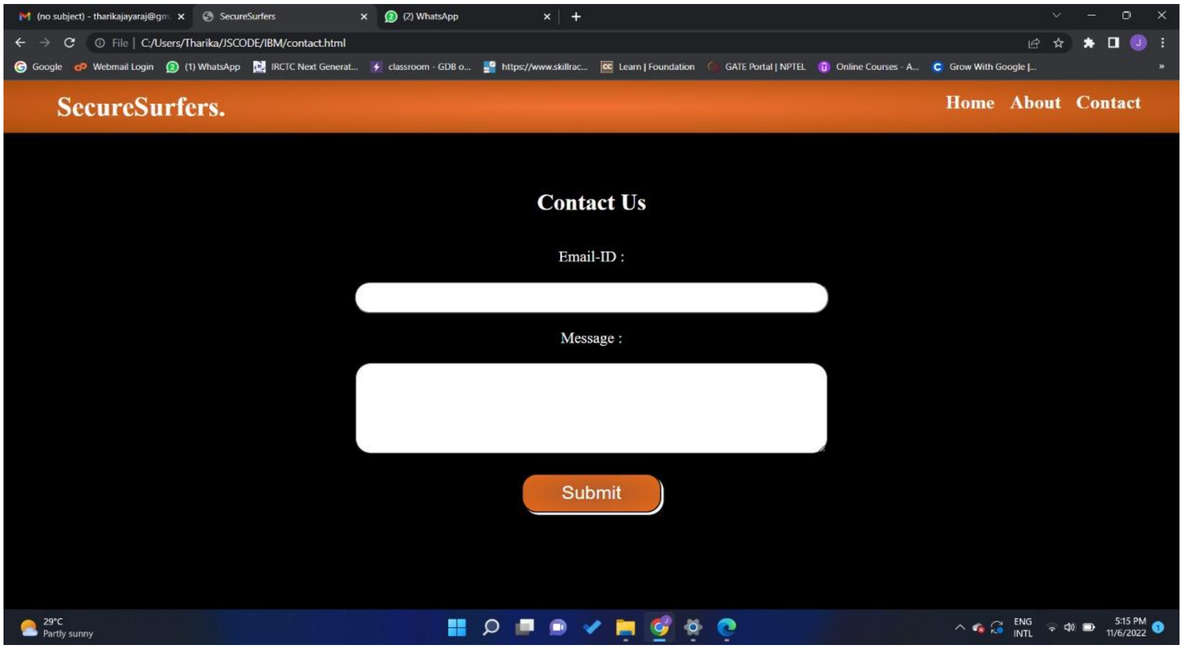
Check the website now to know if the URL is valid and save yourself from phishing attacks.



About

Web service is one of the key communications software services for the internet. Web phishing is one of the many security threats to web services on the Internet. Web phishing aims to steal private information such as usernames, passwords, and credit card details, by the way of impersonating a legitimate entity.

The recipient is tricked into clicking a malicious link, which can lead to the installation of malware, the freezing of the system as a part pf a randsomware attack or the stealing of sensitive information. It will lead to information disclosure and property damage.



SecureSurfers.

Contact Us

Email-ID :

Message :

7.2 Feature 2

That data contains several factors that should be considered when deciding whether a website URL is phishing.

Address Bar based Features:

Using the IP address, If the URL has an IP address rather than a sphere name, such as 125.96.2.121, a person can practically be assured that his private detail are being stolen.

The Suspicious Part is hidden by a long URL. By selecting a long URL, phishers can hide the suspect portion of the URL inside the URL bar.

Applying URL shortening services. The URL is very short URL shortening is a mechanism on the Internet that allows a URL to be drastically reduced in length while still directing to the desired webpage.

```
#inputscript.py

 import regex
from tldextract import extract
import ssl
import socket
from bs4 import BeautifulSoup
import urllib.request
import whois
import datetime
import requests
import favicon
import re
from googlesearch import search
import parser
#checking 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[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

#Checking for the URL length. Returns 1 (Legitimate) if the URL length is less than 54 characters
#Returns 0 if the length is between 54 and 75
#Else returns -1;
```

```
def URLURL_Length (url):
length=len(url)
if(length<=75):
if(length<54):
return 1
else:
return 0
else:
return -1

#Checking with the shortening URLs.
#Returns -1 if any shortening URLs used.
#Else returns 1
def Shortining_Service (url):
match=regex.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',
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

#checking the Subdomain. Returns 1 if the subDomain contains less than 1 '.'
#Returns 0 if the subDomain contains less than 2 '.'
#Returns -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

# 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.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
```

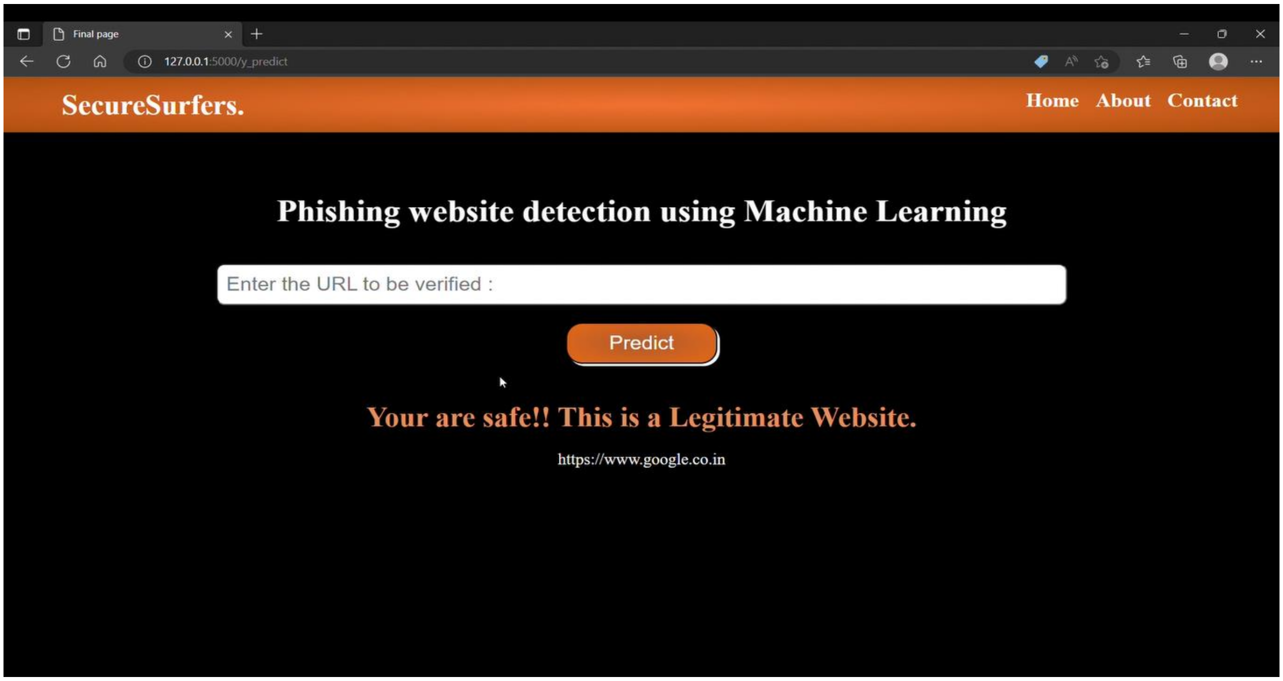
```
no_of_link =0
no_of_script =0
anchors=0
avg =0
for meta in soup.find_all('meta'):
no_of_meta = no_of_meta+1
for link in soup.find_all('link'):
no_of_link = no_of_link +1
for script in soup.find_all('script'):
no_of_script = no_of_script+1
for anchor in soup.find_all('a'):
anchors = anchors+1
total = 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

#usingiframe 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&dat=s&url=" + url).read(),
"xml").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|esy\.es|hol\.es|sweddy\.com|myjino
\.ru|96\.l
t|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\.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\.2 25\.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\.51192\.64\.147\.141|198\.200\.56\.183|23\.253\.164\.103|52\.48\.191\.26|52\.214\.197\.72|87\.98\.255\.18|20
9\.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)
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),SSLfinal_State(url),
Domain_registeration_length(url),Favicon(url),port(url),HTTPS_token(url),Request_URL(url),
URL_of_Anchor(url),Links_in_tags(url),SFH(url),Submitting_to_email(url),Abnormal_URL(url),
Redirect(url),on_mouseover(url),RightClick(url),popUpWidnow(url),Iframe(url),
age_of_domain(url),DNSRecord(url),web_traffic(url),Page_Rank(url),Google_Index(url),
Links_pointing_to_page(url),Statistical_report(url)]]
print(check)
return check
```

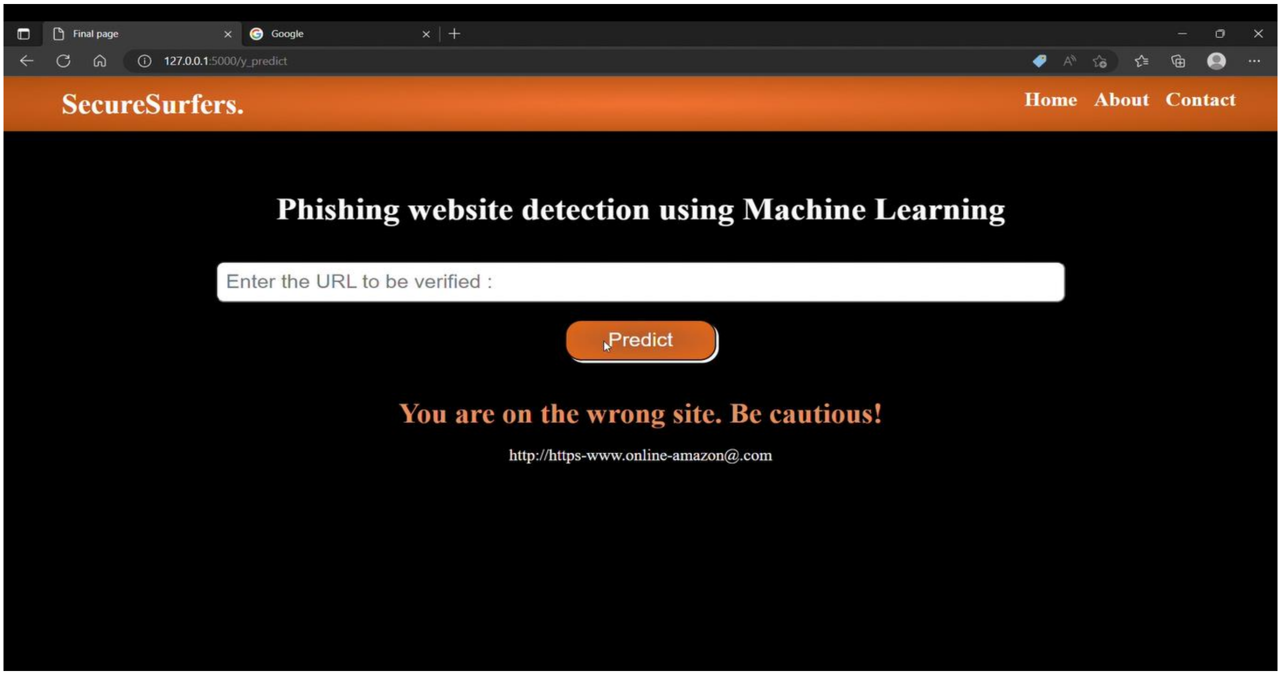


Phishing website detection using Machine Learning

Enter the URL to be verified :

Your are safe!! This is a Legitimate Website.

https://www.google.co.in



Phishing website detection using Machine Learning

Enter the URL to be verified :

You are on the wrong site. Be cautious!

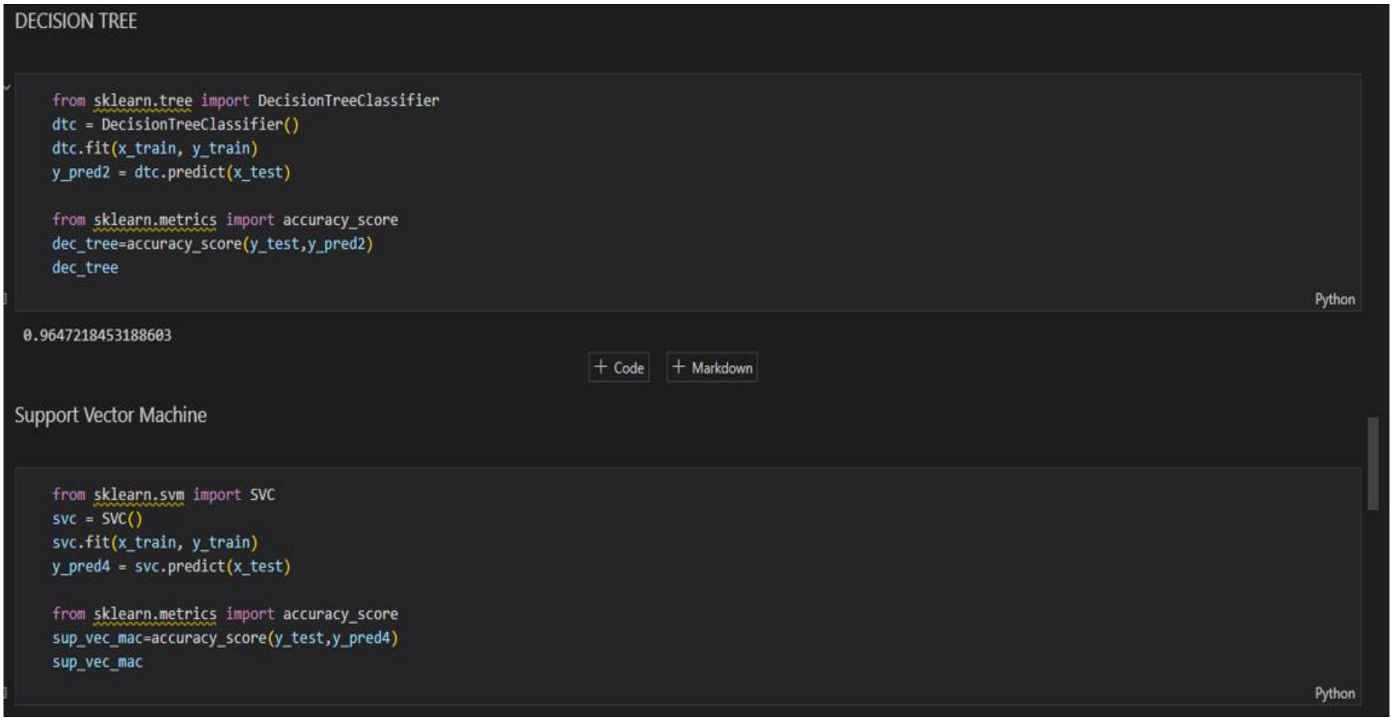
http://https-www.online-amazon@.com

8. TESTING:

8.1.1 Test Cases:

Test CaseID	Test Case Description	Test Steps
TC01	Check Predict button is Rooted to Prediction page	In home page, Click Prediction URL button.

TC02	In Prediction Page, Check prediction of URL is done or not.	In prediction page, Enter Url Then press Prediction Button to predict URL
TC03	In Prediction output page, Check the “Predict another URL” button.	In result page, press Predict another URL button.
TC04	In Prediction Page, Check Prediction is done in positive and negative.	In prediction page, 1.Enter URL for good site and bad site. 2.then press Predict button.
TC05	Check User experience form is submitted in google form or not.	In add URL page, 1.Enter the required fields. 2.press submit button.
TC06	Check About button root to About page.	Press about button.
TC07	Check project Details Button root’s to Project details button.	Press Project details button
TC08	Check all buttons are Working properly or not	Press all button and check it root’s to corresponding Page or not.



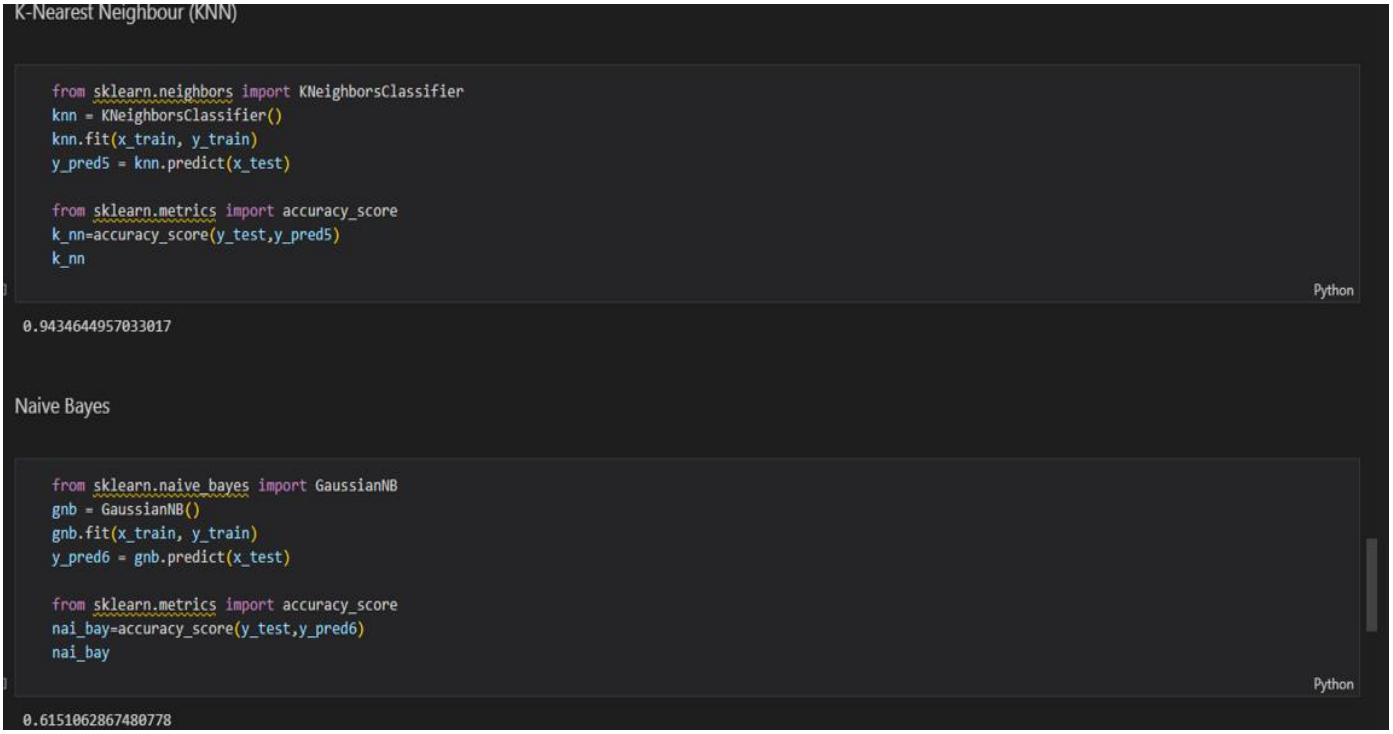
DECISION TREE

```
from sklearn.tree import DecisionTreeClassifier
dtc = DecisionTreeClassifier()
dtc.fit(x_train, y_train)
y_pred2 = dtc.predict(x_test)

from sklearn.metrics import accuracy_score
dec_tree=accuracy_score(y_test,y_pred2)
dec_tree
```

```
0.9647218453188603
```

Support Vector Machine

```
from sklearn.svm import SVC
svc = SVC()
svc.fit(x_train, y_train)
y_pred4 = svc.predict(x_test)

from sklearn.metrics import accuracy_score
sup_vec_mac=accuracy_score(y_test,y_pred4)
sup_vec_mac
```



K-Nearest Neighbour (KNN)

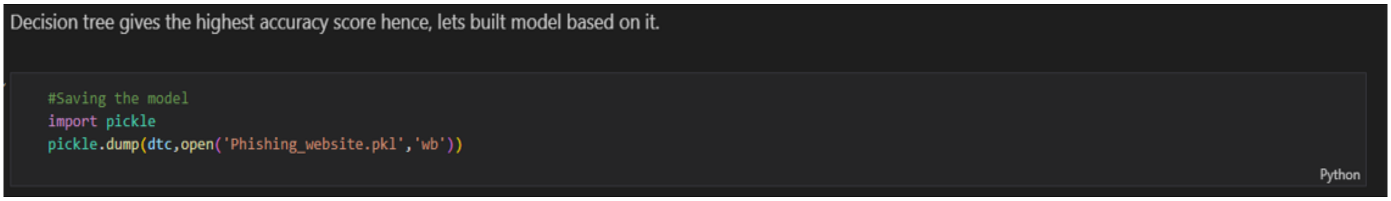
```python
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier()
knn.fit(x_train, y_train)
y_pred5 = knn.predict(x_test)

from sklearn.metrics import accuracy_score
k_nn=accuracy_score(y_test,y_pred5)
k_nn
```

```
0.9434644957033017
```

Naive Bayes

```python
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(x_train, y_train)
y_pred6 = gnb.predict(x_test)

from sklearn.metrics import accuracy_score
nai_bay=accuracy_score(y_test,y_pred6)
nai_bay
```

```
0.6151062867480778
```



Decision tree gives the highest accuracy score hence, lets built model based on it.

```python
#Saving the model
import pickle
pickle.dump(dtc,open('Phishing_website.pkl','wb'))
```

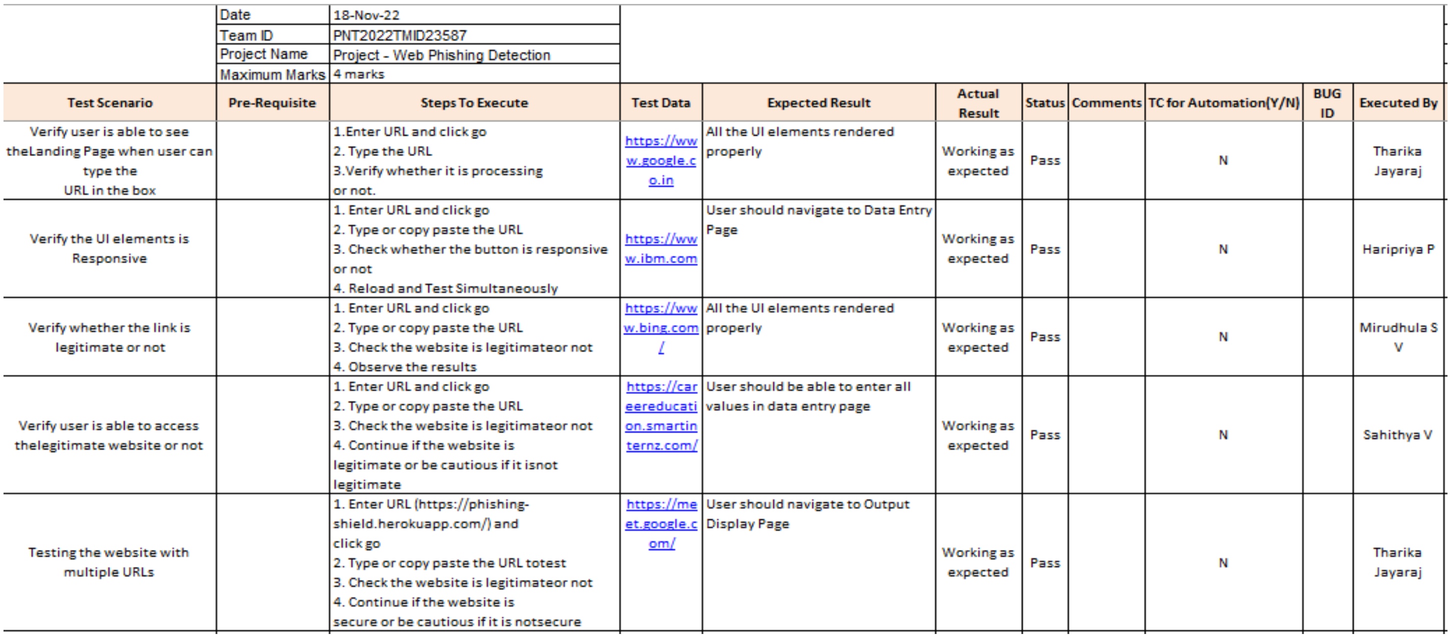
8.1.2 User Acceptance Testing:

Purpose of Document

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).

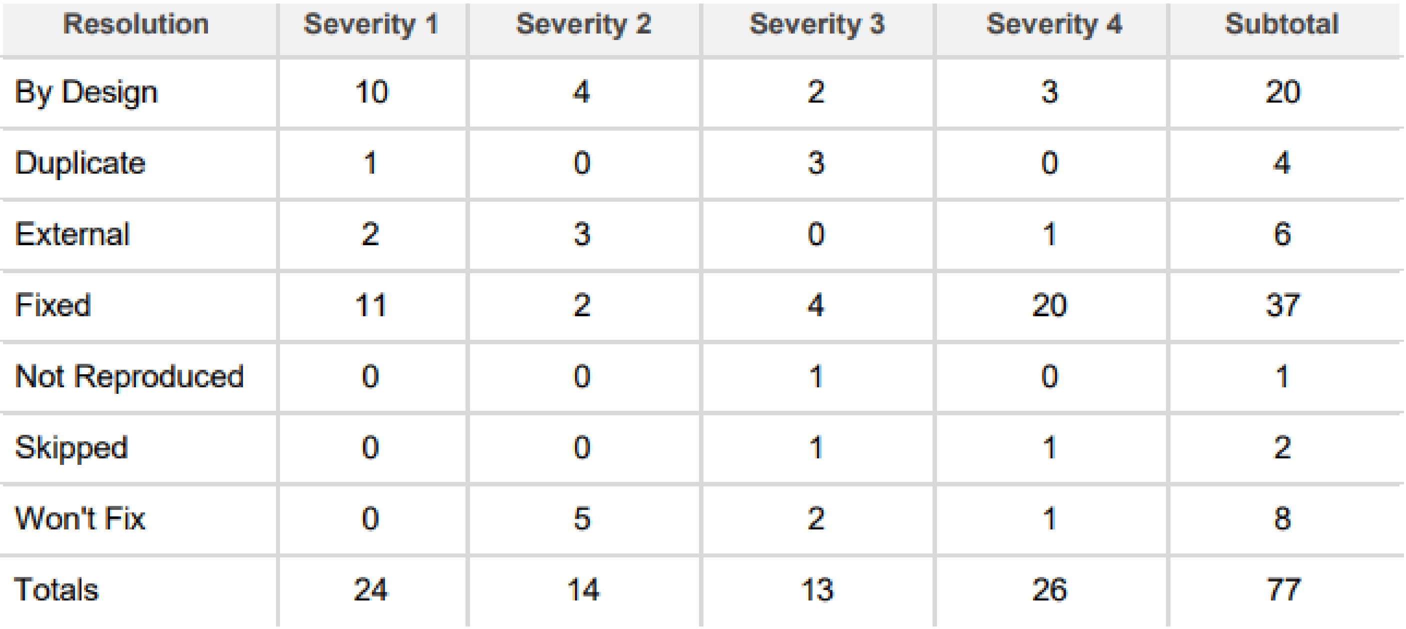
Defect Analysis

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



Date	18-Nov-22
Team ID	PNT2022TMID23587
Project Name	Project - Web Phishing Detection
Maximum Marks	4 marks

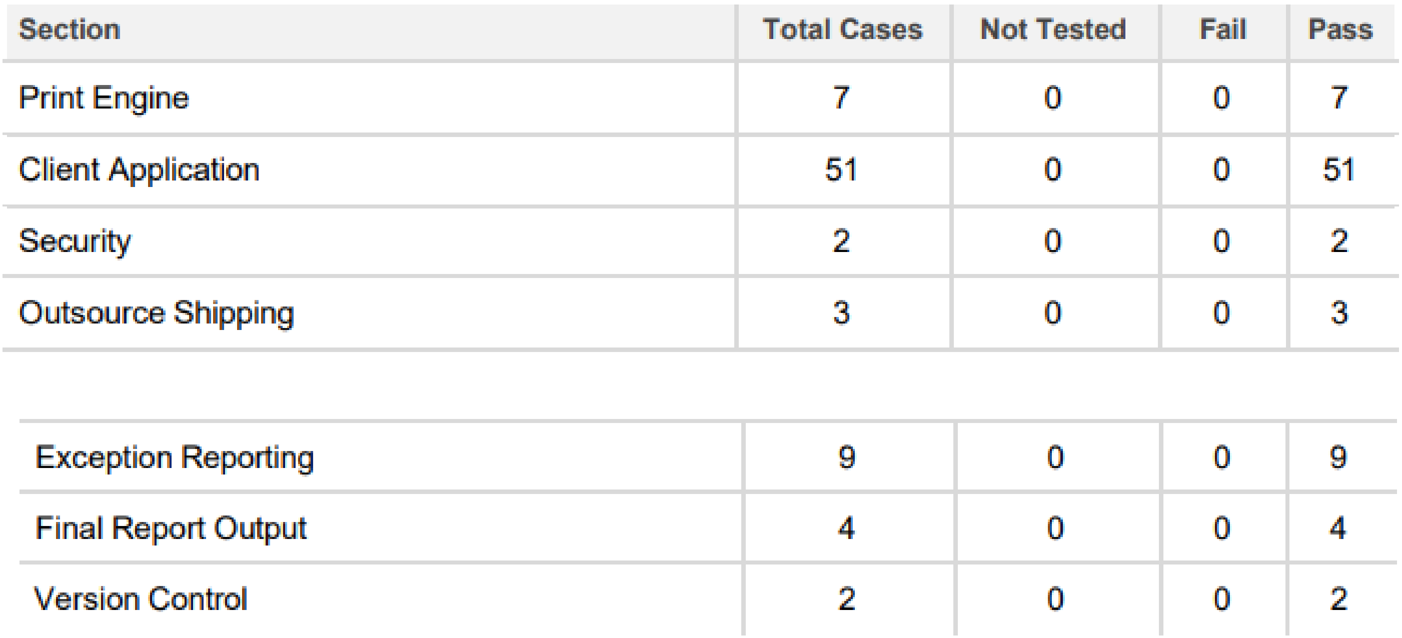
Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
Verify user is able to see theLanding Page when user can type the URL in the box		1.Enter URL and click go 2. Type the URL 3.Verify whether it is processing or not.	https://www.google.co.in	All the UI elements rendered properly	Working as expected	Pass		N		Tharika Jayaraj
Verify the UI elements is Responsive		1. Enter URL and click go 2. Type or copy paste the URL 3. Check whether the button is responsive or not 4. Reload and Test Simultaneously	https://www.ibm.com	User should navigate to Data Entry Page	Working as expected	Pass		N		Haripriya P
Verify whether the link is legitimate or not		1. Enter URL and click go 2. Type or copy paste the URL 3. Check the website is legitimateor not 4. Observe the results	https://www.bing.com/	All the UI elements rendered properly	Working as expected	Pass		N		Mirudhula S V
Verify user is able to access thelegitimate website or not		1. Enter URL and click go 2. Type or copy paste the URL 3. Check the website is legitimateor not 4. Continue if the website is legitimate or be cautious if it isnot legitimate	https://careereducation.smartinternz.com/	User should be able to enter all values in data entry page	Working as expected	Pass		N		Sahithya V
Testing the website with multiple URLs		1. Enter URL (https://phishing-shield.herokuapp.com/) and click go 2. Type or copy paste the URL totest 3. Check the website is legitimateor not 4. Continue if the website is secure or be cautious if it is notsecure	https://meet.google.com/	User should navigate to Output Display Page	Working as expected	Pass		N		Tharika Jayaraj



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	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

8.1.3 Test Case Analysis

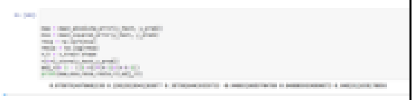
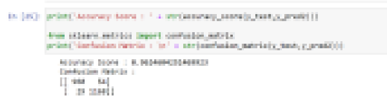
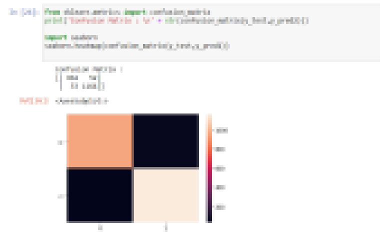
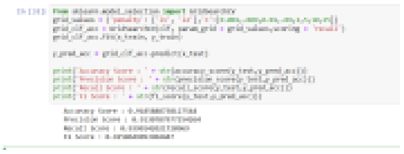
This report shows the number of test cases that have passed, failed, and untested.



Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

9.RESULTS

Performance Metrics:

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** Decision tree MAE – 0.075 MSE - 0.15 RMSE – 0.387 R2 score – 0.84 **Classification Model:** Confusion Matrix Accuracy Score-0.96	  
2.	Tune the Model	Hyperparameter Tuning - Grid Search Cross Validation	

```
In [[illegible]]:
    mae = mean_absolute_error(y_test, y_pred)
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)
    rmsle = np.log(rmse)
    r2 = [illegible]
    [illegible]
    [illegible]
    print(mae,mse,rmse,rmsle,r2,adj_r2)
    [illegible]
```

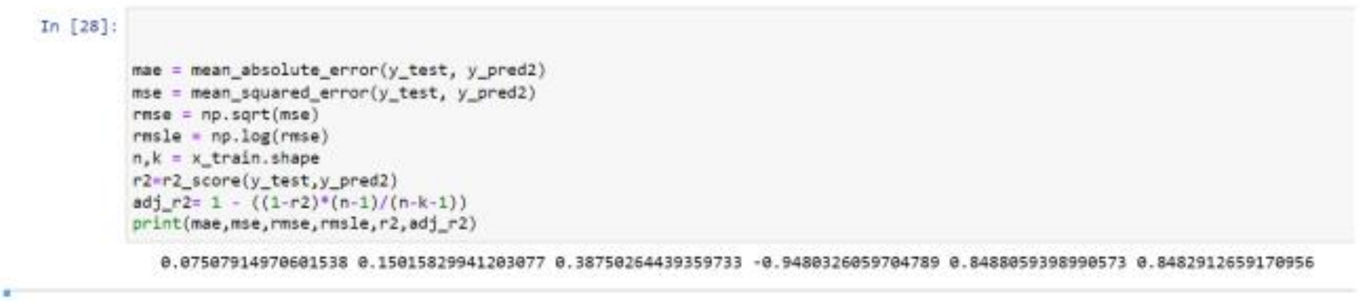
```
In [25]: print('Accuracy Score : ' + str(accuracy_score(y_test,y_pred)))

         from sklearn.metrics import confusion_matrix
         print('Confusion Matrix : \n' + str(confusion_matrix(y_test,y_pred2)))

         Accuracy Score : 0.96[illegible]
         Confusion Matrix :
         [[ [illegible]   [illegible]]
          [  [illegible] [illegible]]]
```

```
In [26]: from sklearn.metrics import confusion_matrix
         print('Confusion Matrix : \n' + str(confusion_matrix(y_test,y_pred2)))

         import seaborn
         seaborn.heatmap(confusion_matrix(y_test,y_pred2))

         Confusion Matrix :
         [[ [illegible]   [illegible]]
          [  [illegible] [illegible]]]
Out[26]: <AxesSubplot:>
```

```
In [[illegible]]: from sklearn.model_selection import GridSearchCV
         grid_values = {'penalty': ['l1', 'l2'],'C':[0.001,.009,0.01,.09,1,5,10,25]}
         grid_clf_acc = GridSearchCV(clf, param_grid = grid_values,scoring = 'recall')
         grid_clf_acc.fit(X_train, y_train)

         y_pred_acc = grid_clf_acc.predict(X_test)

         print('Accuracy Score : ' + str(accuracy_score(y_test,y_pred_acc)))
         print('Precision Score : ' + str(precision_score(y_test,y_pred_acc)))
         print('Recall Score : ' + str(recall_score(y_test,y_pred_acc)))
         print('F1 Score : ' + str(f1_score(y_test,y_pred_acc)))

         Accuracy Score : 0.[illegible]
         Precision Score : 0.[illegible]
         Recall Score : 0.[illegible]
         F1 Score : 0.[illegible]
```

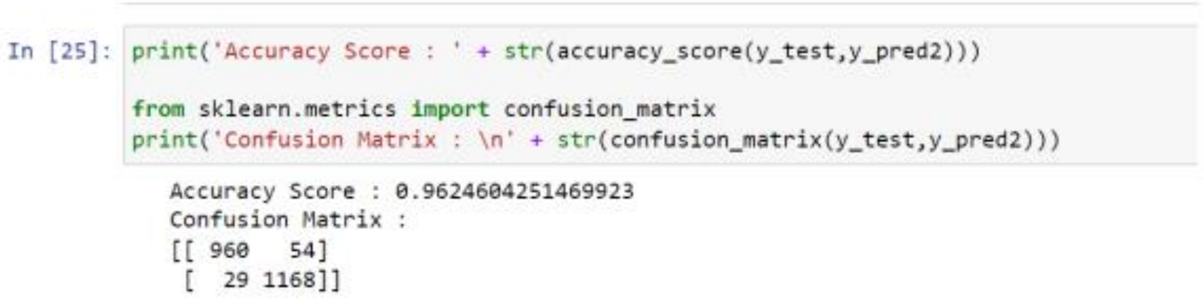
Metrics :

1.Regression Model:

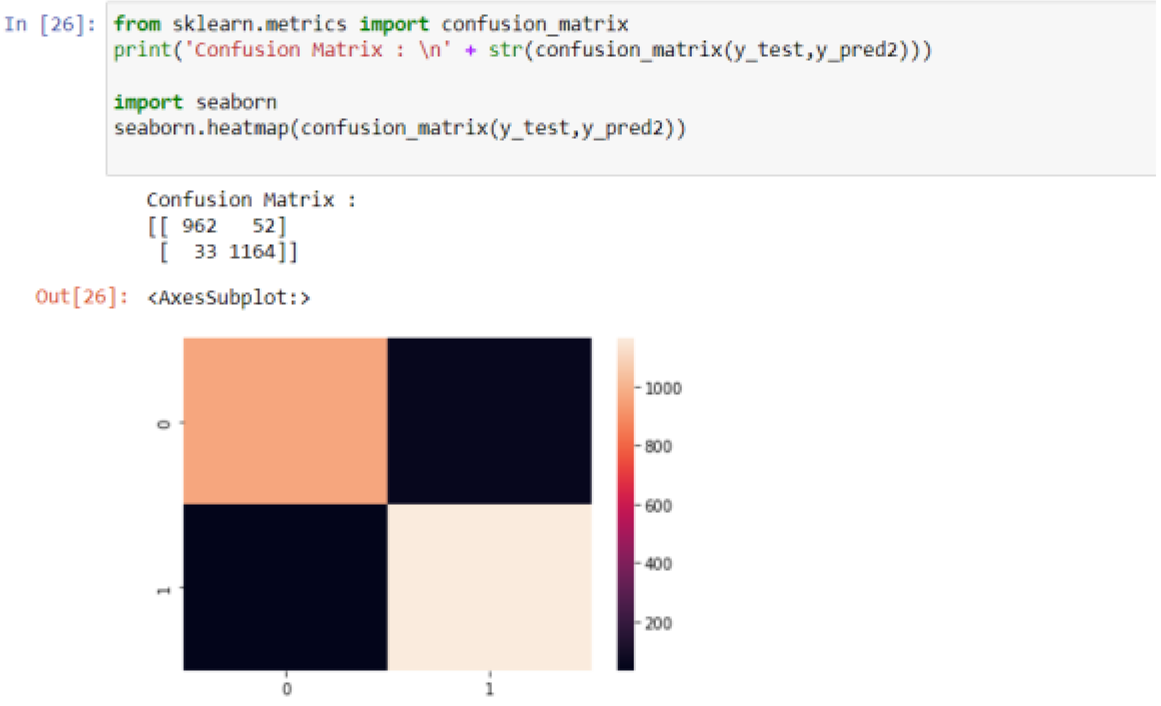


```
In [28]:

mae = mean_absolute_error(y_test, y_pred2)
mse = mean_squared_error(y_test, y_pred2)
rmse = np.sqrt(mse)
rmsle = np.log(rmse)
n,k = x_train.shape
r2=r2_score(y_test,y_pred2)
adj_r2= 1 - ((1-r2)*(n-1)/(n-k-1))
print(mae,mse,rmse,rmsle,r2,adj_r2)

0.07507914970601538 0.15015829941203077 0.38750264439359733 -0.9480326059704789 0.8488059398990573 0.8482912659170956
```

2.Classification Model:

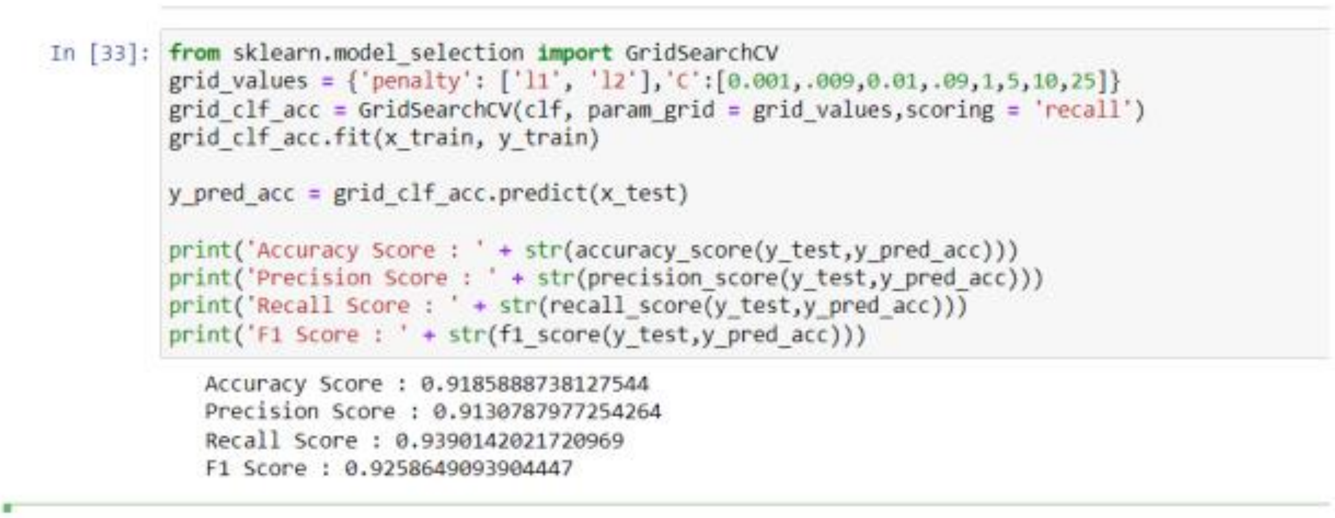


```
In [25]: print('Accuracy Score : ' + str(accuracy_score(y_test,y_pred2)))

         from sklearn.metrics import confusion_matrix
         print('Confusion Matrix : \n' + str(confusion_matrix(y_test,y_pred2)))

            Accuracy Score : 0.9624604251469923
            Confusion Matrix :
            [[ 960   54]
             [  29 1168]]
```



```
In [26]: from sklearn.metrics import confusion_matrix
         print('Confusion Matrix : \n' + str(confusion_matrix(y_test,y_pred2)))

         import seaborn
         seaborn.heatmap(confusion_matrix(y_test,y_pred2))

           Confusion Matrix :
           [[ 962   52]
            [  33 1164]]

Out[26]: <AxesSubplot:>
```

3.Tuning the Model:



```
In [33]: from sklearn.model_selection import GridSearchCV
         grid_values = {'penalty': ['l1', 'l2'],'C':[0.001,.009,0.01,.09,1,5,10,25]}
         grid_clf_acc = GridSearchCV(clf, param_grid = grid_values,scoring = 'recall')
         grid_clf_acc.fit(x_train, y_train)

         y_pred_acc = grid_clf_acc.predict(x_test)

         print('Accuracy Score : ' + str(accuracy_score(y_test,y_pred_acc)))
         print('Precision Score : ' + str(precision_score(y_test,y_pred_acc)))
         print('Recall Score : ' + str(recall_score(y_test,y_pred_acc)))
         print('F1 Score : ' + str(f1_score(y_test,y_pred_acc)))
```

```
Accuracy Score : 0.9185888738127544
Precision Score : 0.9130787977254264
Recall Score : 0.9390142021720969
F1 Score : 0.9258649093904447
```

10 . ADVANTAGES:

- Measure the degrees of corporate and employee vulnerability.
- Eliminate the cyber threat risk level.
- Increase user alertness stop his hing risks.
- Instill a cyber security culture and create cyber security heroes.
- Improve on Inefficiencies of SEG and Phishing Awareness Training
- It Takes a Load off the Security Team.
- It Offers a Solution, not a Tool.
- Separate You from Your Competitors.
- This system can be used by many e-commerce websites in order to have good customer relationships.
- If internet connection fails this system will work.

DISADVANTAGES:

Phishing has a list of negative effects on a business, including loss of money, loss of intellectual property, damage to reputation, and disruption of operational activities. These effects work together to cause loss of company value, sometimes with irreparable repercussions.

1. All website related data will be stored in one place.
2. It is a very time-consuming process.

11. CONCLUSION:

Our execution confirms that we had successfully implemented our project work and we had also tested the min different cases in the given timeline. Our project is distributes the work of design, implementation, testing and documentation in different levels so that we can complete our project on time. The results generated are upto the expected marks from which we concluded that our project is accomplised effectively, As a proof of completion we had produce the Demo video link and Coding of the project in our Documentation.

12. FUTURESCOPE

There is a scope for future development of this project. We will implement this using advanced deep learning method to improve the accuracy and precision. Enhancements can be done in an efficient manner. Thus, the project is flexible and can be enhanced at any time with more advanced features.

13. APPENDIX

1. **Application Building:**
   https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Project%20Development%20Phase/4.%20Application%20building

2. **Collection of Dataset:**
   https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Project%20Development%20Phase/1.%20Collection%20of%20dataset

3. **Data Pre-processing:**
   https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Project%20Development%20Phase/2.%20Data%20pre-processing

4. **Integration of Flask App with IBM Cloud:**
   https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Final%20deliverables

5. **Model Building:**
   https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Project%20Development%20Phase/3.%20Model%20Building

6. **Performance Testing:**
   https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Project%20Development%20Phase/Testing/Performance%20Testing

7. **Training the model on ibm :**
   https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Project%20Development%20Phase/5.%20Training%20the%20model%20on%20ibm

8. **User Acceptance Testing:**
https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Project%20Development%20Phase/Testing/User%20Acceptance%20Testing

9. **Ideation Phase:**
https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Project%20Design%20and%20Planning/ideation%20phase

10. **Preparation Phase:**
https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Project%20Development%20Phase

11. **Project Planning:**
https://github.com/IBM-EPBL/IBM-Project-23025-1659864591/tree/main/Project%20Design%20and%20Planning/Project%20Planning

GITHUB LINK: https://github.com/IBM-EPBL/IBM-Project-23025-1659864591

DEMO LINK: https://github.com/IBM-EPBL/IBM-Project-23025-1659864591
/tree/main/Final%20deliverables/Demo%20link