TEAM ID	**PNT2022TMID26156**
PROJECT NAME	**WEB PHISHING DETECTION**

1. INTRODUCTION

Phishing scams trick victims into divulging sensitive data, downloading malware, and exposing themselves or their organizations to cybercrime.

Phishing attacks are fraudulent emails, text messages, phone calls or web sites designed to manipulate people into downloading malware, sharing sensitive information (e.g., Social Security and credit card numbers, bank account numbers, login credentials), or taking other actions that expose themselves or their organizations to cybercrime.

Successful phishing attacks often lead to identity theft, credit card fraud, ransomware attacks, data breaches, and huge financial losses for individuals and corporations.

Phishing is the most common form of social engineering, the practice of deceiving, pressuring or manipulating people into sending information or assets to the wrong people. Social engineering attacks rely on human error and pressure tactics for success. The attacker typically masquerades as a person or organization the victim trusts—e.g., a coworker, a boss, a company the victim or victim's employer does business with—and creates a sense of urgency that drives the victim to act rashly. Hackers use these tactics because it's easier and less expensive to trick people than it is to hack into a computer or network.

1.1 Project Overview

The project report has been prepared based on available data, forecasts provided by experts and other project management tools. The real life situation can be little different depending on the circumstances. The project is considered as not for profit. The members working in the team will get fixed amount at the end of the project. Any inclusion or deduction is possible as we have enough buffer time. Risks has been estimated based on common issues

faced by this type of project. CPM method has been used to estimate time and creating Gantt chart. Spreadsheet (EXCEL) analysis has been used to do CPM. A detailed network diagram has been drawn to describe every step clearly. A register form is included in the appendix section to monitor the project through its lifespan. Risk register should be updated at every meeting. Full effort has been given to complete each and every pros and cons, so that they are taken into account. However, the report isn't full proof. There is always room for improvement.

1.2 Purpose

The purpose of Phishing Domain Detection is detecting phishing domain names. Therefore, passive queries related to the domain name, which we want to classify as phishing or not, provide useful information to us. Some useful Domain-Based Features are given below.

- Its domain name or its IP address in blacklists of well-known reputation services?
- How many days passed since the domain was registered?
- Is the registrant name hidden?

Page-Based Features

Page-Based Features are using information about pages which are calculated reputation ranking services. Some of these features give information about how much reliable a web site is. Some of Page-Based Features are given below.

- Global Pagerank
- Country Pagerank
- Position at the Alexa Top 1 Million Site

Some Page-Based Features give us information about user activity on target site. Some

of these features are given below. Obtaining these types of features is not easy. There are some paid services for obtaining these types of features.

- Estimated Number of Visits for the domain on a daily, weekly, or monthly basis
- Average Pageviews per visit
- Average Visit Duration
- Web traffic share per country
- Count of reference from Social Networks to the given domain
- Category of the domain
- Similar websites etc.

2. LITERATURE SURVEY

S. No	**Topic**	Year	Description	**Author**	Merits	Demerits
1	**Mitigation of Phishing Attacks**	15 April 2013	This paper aims at a detection of phishing attacks. A high-level overview of various categories of phishing mitigation techniques are also presented, such as: detection, offensive defence, correction, and prevention, which we belief is critical to present where the phishing detection techniques fit in the overall mitigation process.	**Mahmoud Khonji, Youssef Iraqi, Andy Jones**	1.It adds great value to the overall security to an organisatio n 2. Use of different defence approaches.	1.Increased bandwidth demand. 2.The empirical effectiveness of this solution is bot accurately measured.

2	**Phishing websited detection(volume3)**	02 February 2014	Phishing is a attempt to steal user's personal information through emails and other messaging services. Various researches have been done to prevent this phishing attack. They include fire walls , black listing certain domain and fake website detection.	**Feon Jaison , Seeni a Francis**	1. Web browsers have integrated an anticipating filter into browser itself. 2. Atleast one brand of security software has integrated anti phishing filter.	1. Phishing attacks possess the detection of combination of customer reportage, pots in addition to technique.
3	**Comparison of Phishing Detection Techniques (volume 03)**	20 March 2014	Email has popular topic of discussion in today's world. Each month, more & more attacks are launched at the purpose of making web-users believe that they are dealing with a trusted & reliable entity for the purpose of stealing logon credentials, account information and	**Parth Parmar, Kalpesh Patel**	1. It constructs classification n models. 2.Mitigate zero hour attacks.	1. High computational cost. 2.Higher fp rate than blacklists.
4	**Phishing**	July	Phishing possesses the	**FadiThabtah ,**	1.Effective when	1.Mitigation of zero -

	detection: A recent intelligent machine learning comparison based on model content and features.	2017	characteristic of a singular fraud framework that uses a singular mixture possessed by designing what objective identify is additional advancement to sensitive in addition to data. Phishing attacks are becoming successful possessed by user awareness.	**Neda Abdelhamid , Hussein Abdel-Jaber**	minimal fp rates are required.	hour phishing attacks. 2.Excessive queries with heavily loaded servers.
5	**Detection of URL based phishing attacks using machine learning(volume-08)**	27 November 2019	This proposed system predicts the URL based phishing attacks with maximum accuracy. Different machine learning algorithms are used in the proposed system to detect URL based phishing attacks. The hybrid algorithm approach by combining the algorithms will increase accuracy.	**Ms. Sophia Shikargar, Dr.S.D. Sawarkar, Mrs. Swati Narwane**	1. Accuracy obtained by using different classifiers in the histogram graphicalnr epresentation 2. More secured than previous systems.	1.Use of many classifiers give in accurate result.
6	**A Survey of URL based PHISHING detection**	2019	This paper emphasize on URL based phishing detection techniques. It aims to understand the structure of URL based features and surveying their diverse detection techniques and mechanisms. It consist of summary of findings to promote better URL based phishing detection systems.	**Eint Sandi Aung , Chaw ThetZan and Hayato Yamana**	12 Use of more than one algorithm ensures accuracy. 2.Effective phishing detection is achieved using different machine learning algorithm.	1.Classification of structured and unstructured dataset is difficult.
7	**Phishing Detection using Machine Learning based URL Analysis**	02August 2021	This paper tells that we are exposed to greater risks in the form of cybercrimes .URL based phishing attacks are one of the most common threats to the internet users. The goal is to	**Arathi Krishna v, Anusree A, Blessy Jose, Karthika Anil Kumar, Ojus Thomas Lee**	1.Uses performanc eevaluation metrics and confusion matrix adds value to the accuracy.	1.Choosing the right approach best suited for the specific dataset or application is a challenging task.

	(Volume 09 – Issue 13)		create a survey resources for researchers to learn and contribute in making phishing detection model that yields more results.		2.Effective ness is ensured byvarious performanc e metrics.	
8	**Survey on Phishing Websites Detection using Machine Learning (volume 10)1**	May 2022	Machine Learning is an effective method for combating phishing assaults. This paper examines the features utilised in detection as well as machine learning based detecti on approaches.	**B. Ravi Raju , Sai Likitha , N Deepa , S Sushma**	1.Uses zero-hour attack detection, Language independency and accuracy rate ensures phishing detection.	1.It lags in feature selection mechanism.
9	**Applicatio ns of deep learning for phishing detection(volume-64)**	23 May 2022	Deep neural network and hybrid deep learning provides best performance . This paper aims at phishing detection approaches were develop among which deep learning algorithms provided promising results. This paper address how deep learning algorithms have been used for phishing detection.	**Cagatay Catal, Gorkem Giray, Bedir Tekinerdoga n, Sandeep Kumar& amp, Suyash Shukla**	1.Effective deep learning methods are used inprevention of phishing attacks. 2.Various methods such as Deep Neural Network and Hybrid deep learning.	1.Challenges in calculation of datasets. 2.Model interpretability is difficult.

2.1 Existing problem

Web Security Problem

The web security problem consists of three major parts:

- Securing the web server and the data that is on it. You need to be sure that the server can continue its operation, the information on the server is not modified without authorization, and the information is only distributed to those individuals to whom you

want it to be distributed.

- Securing information that travels between the web server and the user. You would like to assure that information the user supplies to the web server (usernames, passwords, financial information, etc.) cannot be read, modified, or destroyed by others. Many network technologies are especially susceptible to eavesdropping, because information is broadcast to every computer that is on the local area network.

- Securing the user's own computer. You would like to have a way of assuring users that information, data, or programs downloaded to their systems will not cause damage—otherwise, they will be reluctant to use the service. You would also like to havea way of assuring that information downloaded is controlled thereafter, in accordance with the user's license agreement and/or copyright.

Along with all of these considerations, we may also have other requirements. For instance, in some cases, we have the challenges of:

- Verifying the identity of the user to the server
- Verifying the identity of the server to the user
- Ensuring that messages get passed between client and server in a timely fashion, reliably, and without replay
- Logging and auditing information about the transaction for purposes of billing, conflict resolution, "nonrepudiation," and investigation of misuse
- Balancing the load among multiple servers

To properly address these concerns requires the interaction of several of our three main components, along with the underlying network and OS fabric.

2.2 References

1. M. Blythe, H. Petrie, and J. A. Clark. F for fake: Four studies on how we fall for phish. In CHI, 2011.
2. S. A. Chatzichristofis, A. Arampatzi, and Y. S. Boutalis. Investigating the behavior of compact composite descriptors in early fusion, late fusion and distributed image retrieval. Radioengineering, 2010.
3. S. A. Chatzichristofis, K. Zagoris, Y. S. Boutalis, and N. Papamarkos. Accurate image retrieval based on compact composite descriptors and relevance feedback information. IJPRAI, 2010.
4. T. Chen, S. Dick, and J. Miller. Detecting visually similar web pages. Transactions on Internet Technology, 2010.
5. L. Cieplinski. Mpeg-7 color descriptors and their applications. In CAIP, 2001.
6. R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. In CHI, 2006.
7. S. Egelman, L. F. Cranor, and J. Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In CHI, 2008.
8. Google Inc. Google safe browsing. www.google.com/tools/firefox/safebrowsing
9. M. Lux and S. A. Chatzichristofis. Lire: lucene image retrieval: an extensible java cbir library. In MM, 2008.
10. E. Medvet, E. Kirda, and C. Kruegel. Visual-similarity-based phishing detection. In SecureComm, 2008.
11. OpenDNS. Web content filtering and phishing. OpenDNS 2010 Report, 2010.
12. T. Roessler and A. Saldhana. W3C: Web security context: User interface guidelines. www.w3.org/TR/wsc-ui/2010.
13. L. Wenyin, G. Huang, L. Xiaoyue, Z. Min, and X. Deng. Detection of phishing webpages based on visual similarity. In WWW, 2005.

2.3 Problem Statement Definition

Phishing detection techniques do suffer low detection accuracy and high false alarm especially when novel phishing approaches are introduced. Besides, the most common technique used, blacklist-based method, is inefficient in responding to emanating phishing attacks since registering a new domain has become easier, no comprehensive blacklist can ensure a perfect up-to-date database. Furthermore, page content inspection has been used by some strategies to overcome the false negative problems and complement the vulnerabilities of the stale lists. The inconsistent nature of attack behaviors and continuously changing URL phish patterns require timely updating of the reference model. Therefore, it requires an effective technique to regulate retraining to enable machine learning algorithms to actively adapt to the changes in phish patterns.

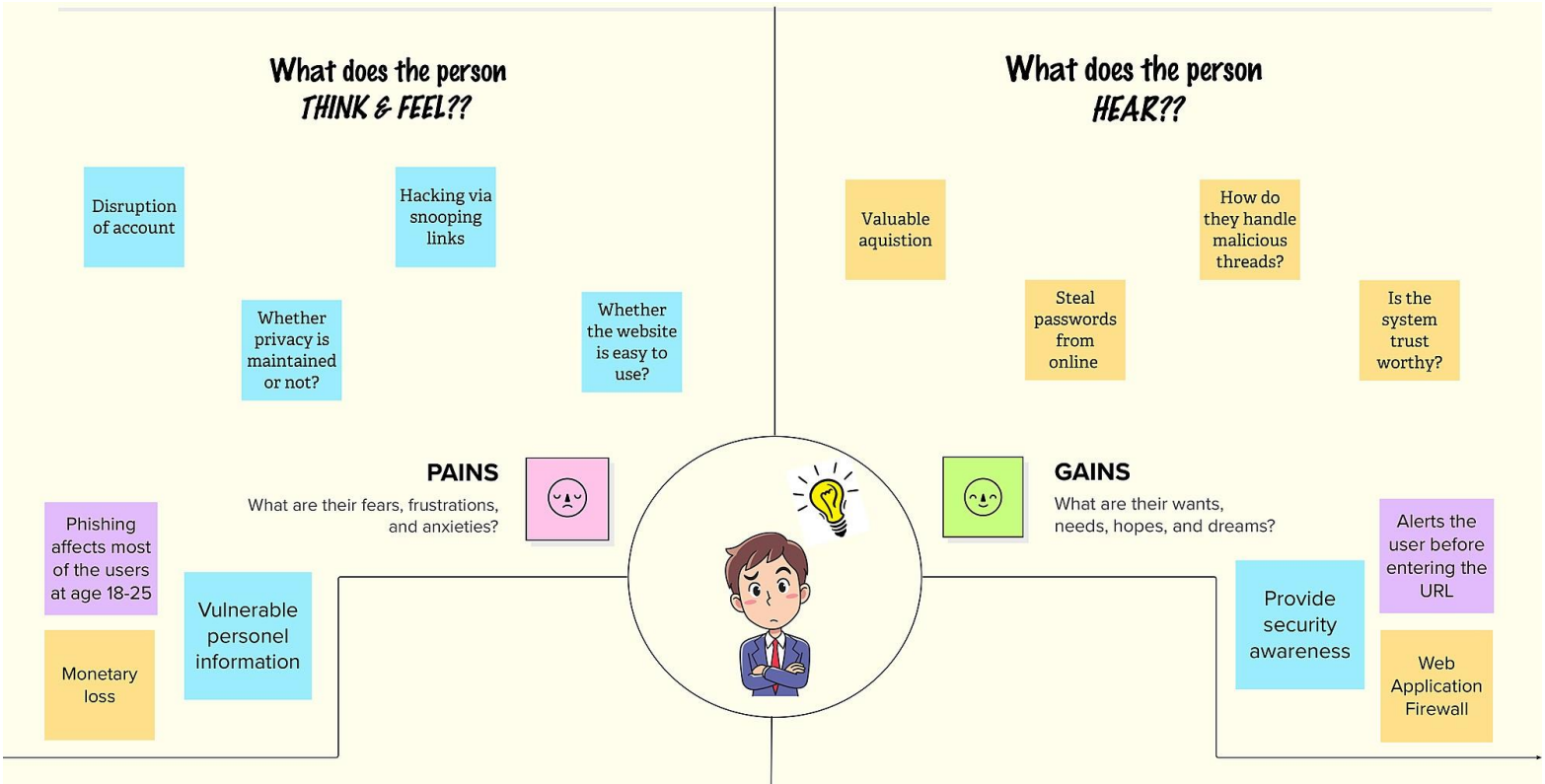
3. IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

Web Phishing Detection:

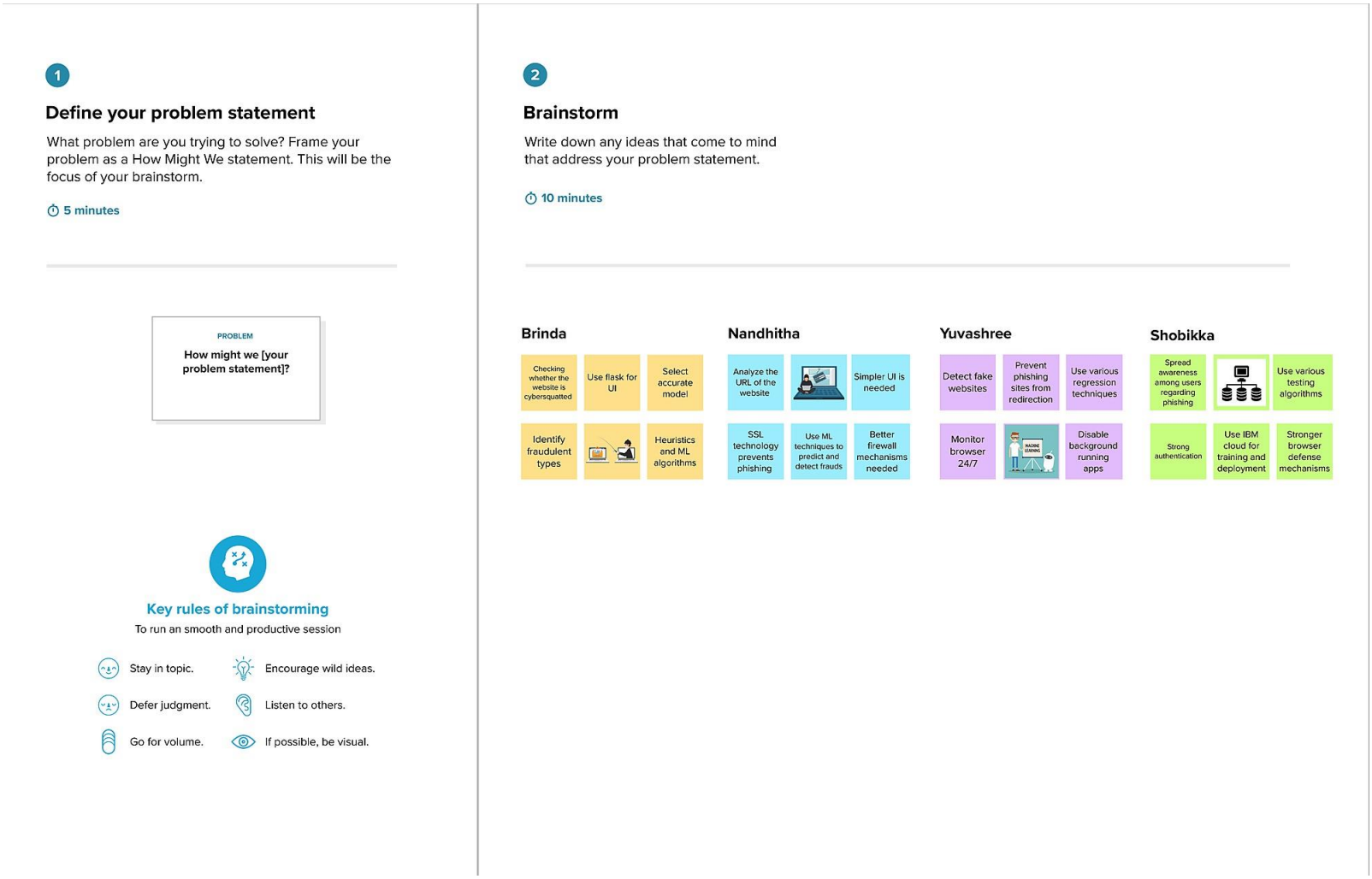


3.2 IDEATION AND BRAINSTORMING

Brainstorm & Idea Prioritization :

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Brainstorm:



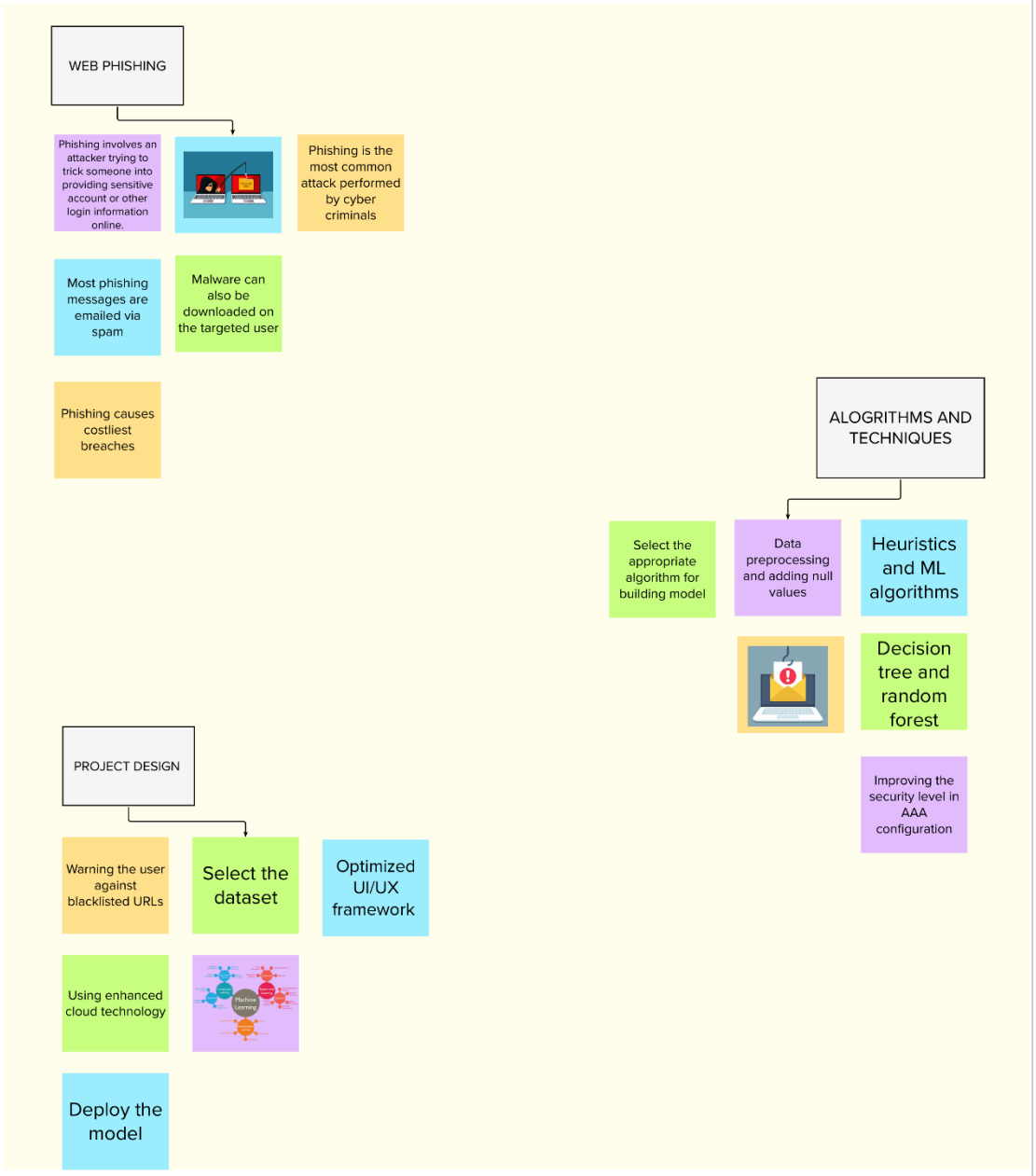
Ideation:

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



Prioritization:

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

A **Share the mural**
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

B **Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

Share template feedback

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Web phishing tends to steal a lots of information from the user during online transaction like username, password, important documents that has been attached to that websites. There are Multiple Types of Attacks happens here every day, but there is no auto detection Process through Machine Learning is achieved
2.	Idea / Solution description	Through ML and data miningtechniques like classification algorithm user can able to attain a warning signal to notify these phishing websites which helps the user to safeguard their identities and their login credentials etc. python is the language thathelps to enablethese techniques for theonline users
3.	Novelty / Uniqueness	This project not only able to identify the malicious websites it also has the ability to automatically block these kind of websites completely in the future when it has been identified and also blocks some various mails /ads from these malicious websites
4.	Social Impact / Customer Satisfaction	This web phishing detection project attains the customer satisfaction by discarding various kinds of malicious websites to protect their privacy. This project is not only capable of using by an single individual ,a large social community and a organisation can use this web phishing detection to protect their privacy. This project helps to block various malicious websites simultaneously.
6.	Scalability of the Solution	This project's performance rate will be high and it also providemany capabilities to the user without reducing its efficiency to detect the malicious websites. Thus scalability of this project will be high.

3.4 PROBLEM SOLUTION FIT:

Define CS, Fit into CC	1. **CUSTOMER SEGMENT(S)** • An Individual trying to Buy a product online, • A Professional surfing through Internet for work purpose, • And any person who wants to access any internet service CS	**6. Customer Constraints** Customers have very little awareness on phishing websites. They don't know what to do after losing data. CC	**5. AVAILABLE SOLUTIONS** The already available solutions are blocking such phishing sites and by triggering a message to the customer about dangerous nature of the website. AS
Focus on J&P, Tap into BE, Understand RC	**2. JOBS-TO-BE-DONE / PROBLEMS** The phishing websites must be detected in a earlier stage . The user can be blocked from entering such sites for the prevention of such issues. JP	**9. PROBLEM ROOT CAUSE** The hackers use new ways to cheat the naïve users. Very limited research is performed on this part of the internet. RC	**7. BEHAVIOUR** The option to check the legitimacy of the Websites is provided. Users get an idea what to do and more importantly what not to do. BE
Identify strong TR & EM	**3. TRIGGERS** A trigger message can be popped warning the user about the site. Phishing sites can be blocked by the ISP and can show a "site is blocked" or "phishing site detected" message. TR **4. EMOTIONS: BEFORE / AFTER** How do customers feel when they face a problem or a job and afterwards? The customers feel lost and insecure to use the internet after facing such issues. Unwanted panicking of the customers is felt after encounter loss of potential data to such sites. EM	**10. YOUR SOLUTION** An option for the users to check the legitimacy of the websites is provided. This increases the awareness among users and prevents misuse of data, data theft etc., SL	**8. CHANNELS of BEHAVIOUR** 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 viz., books, other people etc., CH

4.REQUIREMENT ANALYSIS

4.1 Functional Requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Cinfirmation	Confirmation via Email Confirmation via OTP
FR-3	Website Evaluation	The model evaluates the website that has been entered by the user to check whether it is malicious or not.
FR-4	Prediction	The model predicts the malicious website using machine learning algorithms.
FR-5	Authentication-Results	The model predicts the website based on the evaluation results and alerts the user before providing any confidential information

4.2 Non-Functional Requirement

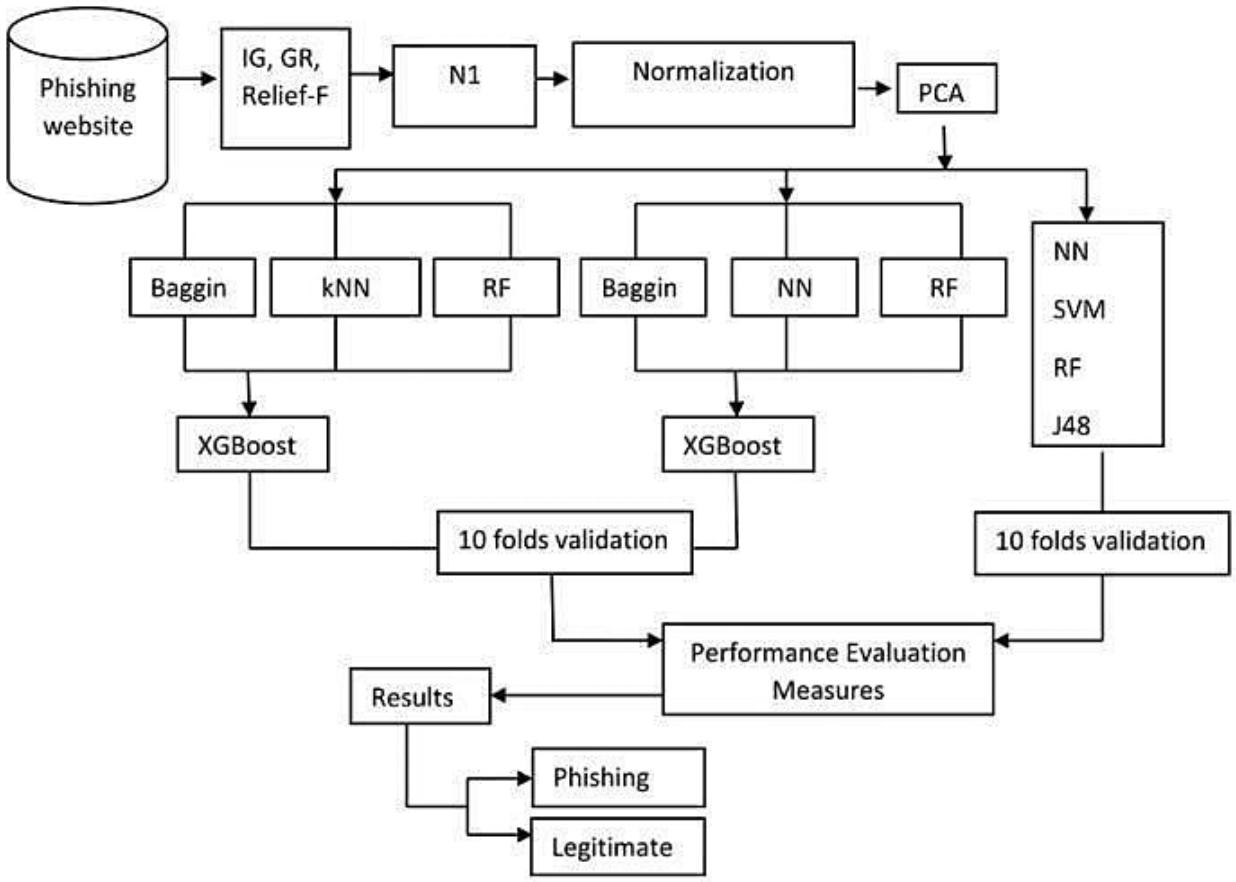
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Usability is a quality attribute that assesses how easy user interfaces are to use. In web phishing, users can use the website without any fear of losing their own credentials
NFR-2	Security	Security refers to protecting and securing users'a, networks, and software, from unauthorized access, misuse, theft, information loss, and other security issues. Here, users will be able to access the website without losing confidential data to an unauthorized person.
NFR-3	Reliabiity	Reliability is the probability that a product, system,or service will perform its intended function adequately for a specified period or will operate in

		a defined environment without failure. The website should detect phishing websites accurately without confusion.
NFR-4	Performance	Performance defines how fast a software system or a particular piece of it responds to certain users' actions under a certain workload. In most cases, this metric explains how long a user must wait before the target operation happens given the overall number of users now.
NFR-5	Availability	Availability describes how likely the system is accessible to a user at a given point in time. The phishing detection application must be readily available to detect the websites and intimate the user any time. There shouldn't be any delay in terms of responsiveness of web application.
NFR-6	Scalability	Scalability is the ability of the application to handle an increase in workload without performance degradation, or its ability to quickly enlarge. It is the ability to enlarge the architecture to accommodate more users, more processes, more transactions, and additional nodes and services as the business requirements change and as the system evolves to meet the future needs of the business. In web phishing detection, the increase in end users should not lead to decrease in performance. It must also diversify different sources of phishing (emails, websites) from vast number of users.

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 DFD 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.



5.2 Solution & Technical Architecture

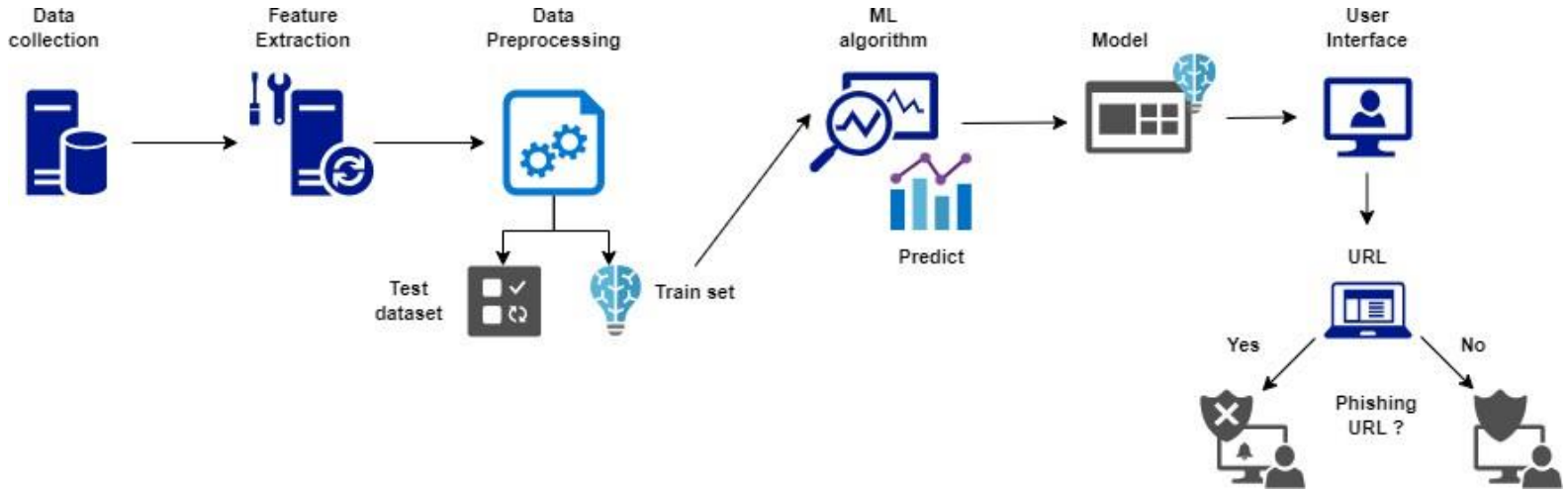


Table-1: Components & Technologies

S. No	Component	Description	Technology
1.	User Interface	The user interacts with application For example:Web UI	HTML, CSS, JavaScript
2.	Application Logic	Predict if the given URL isgenuine or not.	Python, Flask API
3.	Database	Stores user input in a storage device called database.	MySQL
4.	Cloud Database	Database Service on Cloud	IBM DB2 orIBM Cloudant
5.	File Storage	Store training and testing datasets.	Local Filesystem
6.	Machine Learning Model	Classify genuine and phishingURLs.	Classification model
7.	Infrastructure(Server or Cloud)	Application Deployment on Local System or Cloud	Local, Cloud

Table-2: Application Characteristics

S. No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Open-source frameworks usedis deep learning.	PYTORCH
2.			Spoofing detection,

	Security Implementations	User launches a web browser and opens email. The backend phishing detection engine will check the email before it is opened.	fraud detection, filtering/blocking technology.
3.	Scalable Architecture	We consider creating a self-management architecture that will allow ISPs to safeguard their customers from phishing scams.	Machine learningalgorithm
4.	Availability	Laptops, tablets, and mobile devices will all be compatible with this service.	Evaluation training dataset,Data pre-processing.
5.	Performance	The system needs to be quick and precise to handle all potential mistakes in a way that prevents data loss and extended periods of outage. Without any errors, the system should be able to handle many photographs, a lot of data, and many users.	Deep learning and cloudstorage

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story/ Task	Acceptance criteria	Priority	Release
Customer(Mobile user)	Registraqtion	USN-1	As a user, I can register my personal detailsonlyin official websites.	I can access my account / dashboard	Medium	Sprint-1
		USN-	As a user, I should	I can	High	Sprint-1

		2	create strong passwords.	access my account securely		
		USN-3	As a user, I can register in websites which doesn’tnavigate me to anyother websites.	I can store the datain legitimate website	Low	Sprint-2
	Login	USN-4	As a user, I can login into required websites.	I can access my account	Low	Sprint-1
Customer (Mobile user)	Registration	USN-5	As a user, I can register with verification code.	Authorized Login	High	Sprint-1
		USN-6	As a user, I should not register at unknown orrandom calls.	I can be prevented from Cyber Attacks	Medium	Sprint-1
		USN-7	As a user, I should not register in otherdevices.	I can access in my authorized device.	Low	Sprint-2
Administrator		USN-8	Admin shouldmaintain his/her database securely.	Prevented fromPhishingAttacks	High	Sprint-2
Customer Care		USN-9	As a user, If my account is Phished or Attacked.	I can report / Complain	High	Sprint-1
		USN-10	As a user, I should not take othersinformation	I can be punished for it.	Medium	Sprint-1

6.PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Product backlog and sprint schedule:

Sprint	Functional Requirement(Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Homepage	USN-1	As a user, I can explore the resources of the homepage for the functioning	10	Low	S.Yuvashree, N.Nandhitha
Sprint-1		USN-2	As a user, I can learn aboutthe various sidesof the web phishing and be awareof the scams	5	High	N.Nandhitha, Brinda.P
Sprint-2	Final page	USN-3	As a user, I can explore the resources of thefinalpage for the	15	Low	Shobikka.Y, Brinda.P

			functioning			
Sprint-3	Prediction	USN-4	As a user, I can predict the URL easily for detecting whether the website is legitimate or not	10	High	S.Yuvashree, N.Nandhitha, Brinda.P, Shobikka.Y
Sprint-4	Chat	USN-5	As a user, I can share the experience orcontact the admin for the support	10	High	S.Yuvashree, N.Nandhitha, Brinda.P, Shobikka.Y
Sprint-1	Homepage	USN-6	As a admin, we candesign interface andmaintain the functioning of the website	5	High	N.Nandhitha, Brinda.P
Sprint-2	Final page	USN-7	As a admin, we candesign	5	Medium	S.Yuvashree, Shobikka.Y

			the complexity ofthe websitefor making it user-friendly			
Sprint-3	Prediction	USN-8	As a admin, we can use various ML classifier modelfor the accurate result for thedetect ion of URL	10	High	S.Yuvashree, N.Nandhitha, Brinda.P, Shobikka.Y
Sprint-4	Chat	USN-9	As a admin, we can respon se to the user messa ge for improve ment of the website	10	Medi um	S.Yuvashree, Brinda.P

6.2 Sprint Delivery Schedule

Project Tracker, Velocity & Burndown Chart

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

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iterationunit (story points per day)

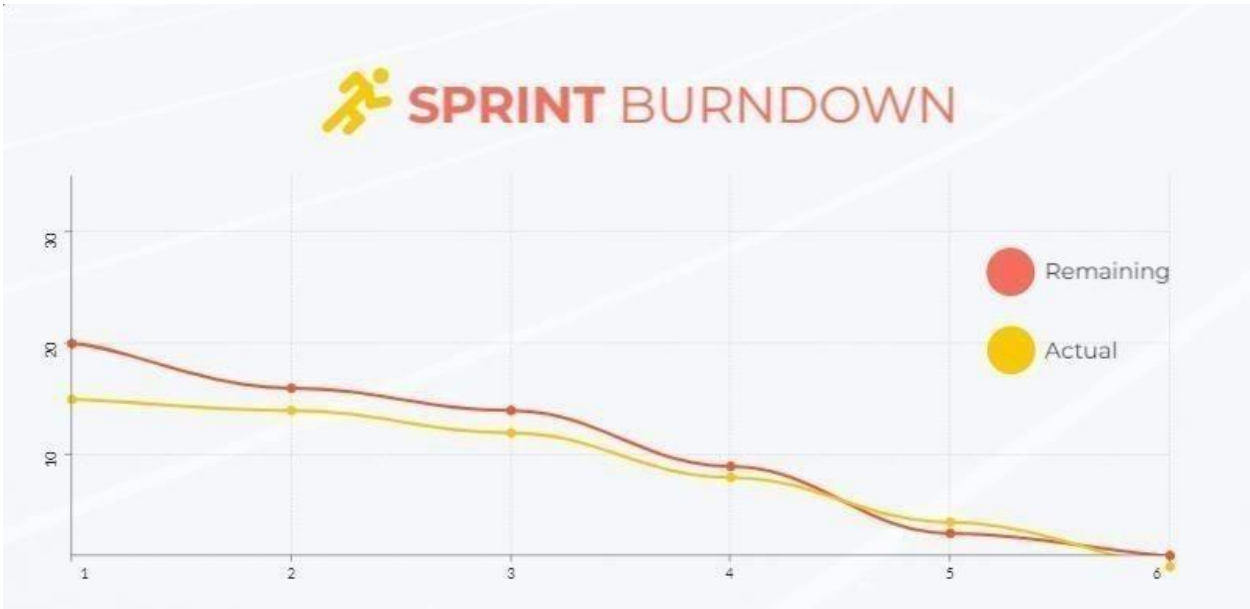
AV= sprint duration= 20 = 2
velocity 10

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

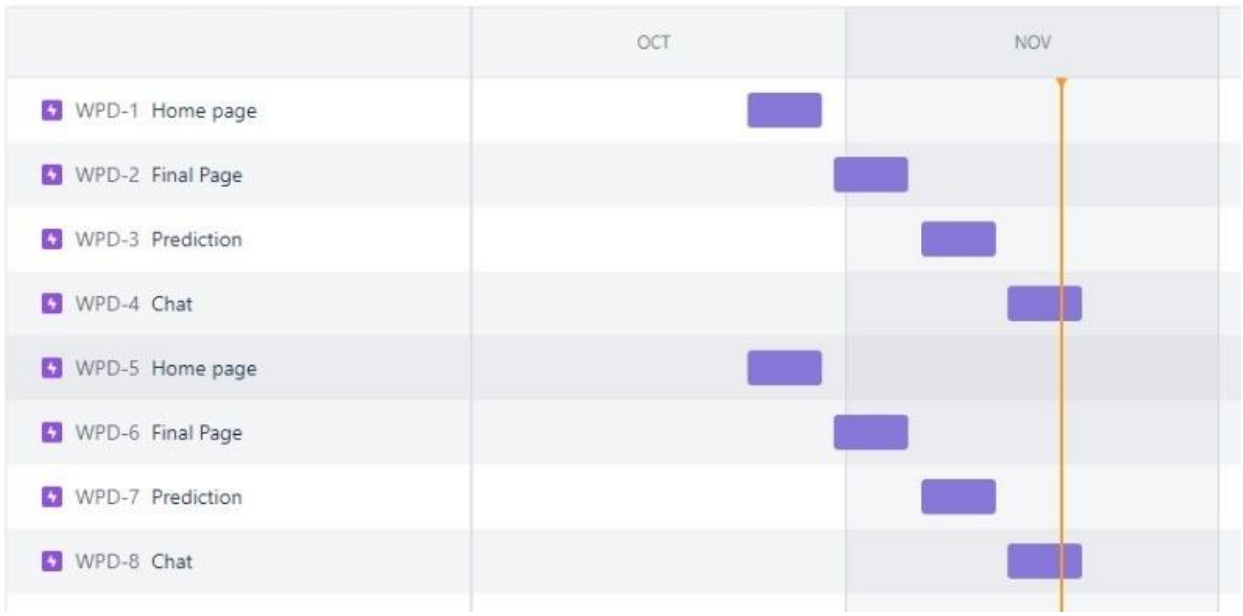
AV = (Sprint Duration / Velocity) = 20 /6 = 3.33

Burndown Chart:

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



6.3 Reports from JIRA



7.CODING & SOLUTIONING

CODE

7.1 feature.py

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

```
class FeatureExtraction:
    features = []
    def __init_(self,url):
        self.features = []
        self.url = url
        self.domain = ""
        self.whois_response = ""
        self.urlparse = ""
        self.response = ""
        self.soup = ""

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

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

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




         self.features.append(self.UsingIp())
        self.features.append(self.longUrl())
        self.features.append(self.shortUrl())
        self.features.append(self.symbol())
        self.features.append(self.redirecting())
        self.features.append(self.prefixSuffix())
        self.features.append(self.SubDomains())
        self.features.append(self.Hppts())
        self.features.append(self.DomainRegLen())
```

```
    self.features.append(self.Favicon())


    self.features.append(self.NonStdPort())
    self.features.append(self.HTTPSDomainURL())
    self.features.append(self.RequestURL())
    self.features.append(self.AnchorURL())
    self.features.append(self.LinksInScriptTags())
    self.features.append(self.ServerFormHandler())
    self.features.append(self.InfoEmail())
    self.features.append(self.AbnormalURL())
    self.features.append(self.WebsiteForwarding())
    self.features.append(self.StatusBarCust())

    self.features.append(self.DisableRightClick())
    self.features.append(self.UsingPopupWindow())
    self.features.append(self.IframeRedirection())
    self.features.append(self.AgeofDomain())
    self.features.append(self.DNSRecording())
    self.features.append(self.WebsiteTraffic())
    self.features.append(self.PageRank())
    self.features.append(self.GoogleIndex())
    self.features.append(self.LinksPointingToPage())
     self.features.append(self.StatsReport())
```

1.UsingIp

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

2.longUrl

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

```
        return -1
```

3.shortUrl

```
    def shortUrl(self):
        match =
re.search('bit\.ly|goo\.gl|shorte\.st|go2l\.ink|x\.co|ow\.ly|t\.co|tinyurl|tr\.im|is\.gd|cli\.gs|'
                  'yfrog\.com|migre\.me|ff\.im|tiny\.cc|url4\.eu|twit\.ac|su\.pr|twurl\.nl|snipurl\.com|'

'short\.to|BudURL\.com|ping\.fm|post\.ly|Just\.as|bkite\.com|snipr\.com|fic\.kr|loopt\.us|'
                  'doiop\.com|short\.ie|kl\.am|wp\.me|rubyurl\.com|om\.ly|to\.ly|bit\.do|t\.co|lnkd\.in|'
                    'db\.tt|qr\.ae|adf\.ly|goo\.gl|bitly\.com|cur\.lv|tinyurl\.com|ow\.ly|bit\.ly|ity\.im|'

'q\.gs|is\.gd|po\.st|bc\.vc|twitthis\.com|u\.to|j\.mp|buzurl\.com|cutt\.us|u\.bb|yourls\.org|'

'x\.co|prettylinkpro\.com|scrnch\.me|filoops\.info|vzturl\.com|qr\.net|1url\.com|tweez\.me|v\.gd
|tr\.im|link\.zip\.net', self.url)
        if match:
            return -1
        return 1
```

4.Symbol@

```
    def symbol(self):
        if re.findall("@",self.url):
            return -1
        return 1
```

5. Redirecting//

```
    def redirecting(self):
        if self.url.rfind('/ ')>6:
            return -1
        return 1
```

6.prefixSuffix

```
    def prefixSuffix(self):
        try:
            match = re.findall('\-', self.domain)if
            match:
                return -1
            return 1
        except:
```

```
        return -1

    # 7.SubDomains
    def SubDomains(self):
        dot_count = len(re.findall("\.", self.url))
        if dot_count == 1:
            return 1
        elif dot_count == 2:
            return 0
        return -1

    # 8.HTTPS
    def Hppts(self):
        try:
            https = self.urlparse.schemeif
            'https' in https:
                return 1
            return -1
        except:
            return 1

    # 9.DomainRegLen
    def DomainRegLen(self):
        try:
            expiration_date = self.whois_response.expiration_date
            creation_date = self.whois_response.creation_date
            try:
                if(len(expiration_date)):
                    expiration_date = expiration_date[0]
            except:
                pass
            try:
                if(len(creation_date)):
                    creation_date = creation_date[0]
            except:
                pass

            age = (expiration_date.year-creation_date.year)*12+ (expiration_date.month-
creation_date.month)
              if age >=12:
```

```
            return 1
        return -1
    except:
        return -1

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

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

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

# 13. RequestURL
def RequestURL(self):
    try:
        for img in self.soup.find_all('img', src=True):
```

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

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

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

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


          try:
            percentage = success/float(i) * 100
            if percentage < 22.0:
                return 1
            elif((percentage >= 22.0) and (percentage < 61.0)):
                return 0
            else:
                    return -1
        except:
            return 0
    except:
        return -1

# 14. AnchorURL
def AnchorURL(self):
    try:
        i,unsafe = 0,0
```

```
        for a in self.soup.find_all('a', href=True):
            if "#" in a['href'] or "javascript" in a['href'].lower() or "mailto" in a['href'].lower() or not (url in
a['href'] or self.domain in a['href']):
                unsafe = unsafe + 1
            i = i + 1

        try:
            percentage = unsafe / float(i) * 100
            if percentage < 31.0:
                return 1
            elif ((percentage >= 31.0) and (percentage < 67.0)):
                return 0
            else:
                return -1
        except:
            return -1

    except:
        return -1

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

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

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

        try:
            percentage = success / float(i) * 100
            if percentage < 17.0:
                return 1
```

```
            elif((percentage >= 17.0) and (percentage < 81.0)):
                return 0
            else:
                return -1
        except:
            return 0
    except:
        return -1

# 16. ServerFormHandler
def ServerFormHandler(self):
    try:
        if len(self.soup.find_all('form', action=True))==0:return
            1
        else :
            for form in self.soup.find_all('form', action=True):
                if form['action'] == "" or form['action'] == "about:blank":return
                    -1
                elif self.url not in form['action'] and self.domain not in form['action']:
                    return 0
                else:
                        return 1
    except:
        return -1

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

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

```
        else:
            return -1
    except:
        return -1

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

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

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

# 22. UsingPopupWindow
def UsingPopupWindow(self):
    try:
```

```
        if re.findall(r"alert\(", self.response.text):
            return 1
        else:
            return -1
    except:
        return -1

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

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

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

# 25. DNSRecording
def DNSRecording(self):
    try:
        creation_date = self.whois_response.creation_datetry:
            if(len(creation_date)):
```

```
            creation_date = creation_date[0]
        except:
            pass

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

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

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

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


   # 28. GoogleIndex
   def GoogleIndex(self):
```

```
        try:
            site = search(self.url, 5)
            if site:
                return 1
            else:
                return -1
        except:
            return 1

    # 29. LinksPointingToPage
    def LinksPointingToPage(self):
        try:
            number_of_links = len(re.findall(r"<a href=", self.response.text))
            if number_of_links == 0:
                return 1
            elif number_of_links <= 2:
                return 0
            else:
                return -1
        except:
            return -1

    # 30. StatsReport
    def StatsReport(self):
        try:
            url_match = re.search(

'at\.ua|usa\.cc|baltazarpresentes\.com\.br|pe\.hu|esy\.es|hol\.es|sweddy\.com|myjino\.ru|96\.lt
|ow\.ly', url)
            ip_address = socket.gethostbyname(self.domain)
            ip_match =
re.search('146\.112\.61\.108|213\.174\.157\.151|121\.50\.168\.88|192\.185\.217\.116|78\.46\.
211\.158|181\.174\.165\.13|46\.242\.145\.103|121\.50\.168\.40|83\.125\.22\.219|46\.242\.14
5\.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|14
1\.8\.224\.221|10\.10\.10\.10|43\.229\.108\.32|103\.232\.215\.140|69\.172\.201\.153|'
```

```
'216\.218\.185\.162|54\.225\.104\.146|103\.243\.24\.98|199\.59\.243\.120|31\.170\.160\.61|2
13\.19\.128\.77|62\.113\.226\.131|208\.100\.26\.234|195\.16\.127\.102|195\.16\.127\.157|'

'34\.196\.13\.28|103\.224\.212\.222|172\.217\.4\.225|54\.72\.9\.51|192\.64\.147\.141|198\.20
0\.56\.183|23\.253\.164\.103|52\.48\.191\.26|52\.214\.197\.72|87\.98\.255\.18|209\.99\.17\.2
7|'

'216\.38\.62\.18|104\.130\.124\.96|47\.89\.58\.141|78\.46\.211\.158|54\.86\.225\.156|54\.82\
.156\.19|37\.157\.192\.102|204\.11\.56\.48|110\.34\.231\.42',     ip_address)
            if url_match:
                return -1
            elif ip_match:
                return -1
            return 1
        except:
            return 1

    def getFeaturesList(self):
        return self.features
```

7.2 app.py

```
#importing required libraries

from flask import Flask, request, render_template
import numpy as np
import pandas as pd
from sklearn import metrics
import warnings
import pickle
warnings.filterwarnings('ignore')
from feature import FeatureExtraction

file = open("pickle/model.pkl","rb")gbc
= pickle.load(file)
file.close()
```

```
app = Flask(__name__)

@app.route("/", methods=["GET", "POST"])
def index():
    if request.method == "POST":url
        = request.form["url"]
        obj = FeatureExtraction(url)
        x = np.array(obj.getFeaturesList()).reshape(1,30)y_pred
        =gbc.predict(x)[0]
        #1 is safe
        #-1 is unsafe
        y_pro_phishing = gbc.predict_proba(x)[0,0]
        y_pro_non_phishing = gbc.predict_proba(x)[0,1]
        # if(y_pred ==1 ):
        pred = "It is {0:.2f} % safe to go ".format(y_pro_phishing*100)
        return render_template('index.html',xx =round(y_pro_non_phishing,2),url=url )
    return render_template("index.html", xx =-1)


if__name___== "_main_":
    app.run(debug=True)
```

7.3 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">
    <meta name="description" content="This website is develop for identify the safety of url.">
    <meta name="keywords" content="phishing url,phishing,cyber security,machine
learning,classifier,python">
    <meta name="author" content="VAIBHAV BICHAVE">

    <!-- BootStrap -->
    <link rel="stylesheet"
href="https:/ stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"
        integrity="sha384-
```

```
9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk"
crossorigin="anonymous">

   <link href="static/styles.css" rel="stylesheet">
   <title>URL detection</title>

</head>

<body>

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

         <br>
         <form action="/" method ="post">
            <input type="text" class="form_input" name ='url' id="url" placeholder="Enter URL"
required="" />
            <label for="url" class="form_label">URL</label>
            <button class="button" role="button" >Check here</button>
         </form>

   </div>

   <div class="col-md" id="form2">

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

      <br>
      <h3 id="prediction"></h3>
      <button class="button2" id="button2" role="button" onclick="window.open('{{url}}')"
target="_blank" >Still want to Continue</button>
      <button class="button1" id="button1" role="button" onclick="window.open('{{url}}')"
target="_blank">Continue</button>
   </div>
</div>
<br>
 <h1>GitRepo Team ID : PNT2022TMID26156</h1>
```

```
</div>

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


    <script>

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

    </script>


</body>
```

```
</html>
```

7.4 style.css

```
*,
*::after,
*::before {
  margin: 0;
  padding: 0;
  box-sizing: inherit;
  font-size: 62,5%;
}

body {
  padding: 10% 5%;
  background: #c31432;  /* fallback for old browsers */
  background: -webkit-linear-gradient(to right, #240b36, #c31432); /* Chrome 10-25, Safari 5.1-6
*/
  background: linear-gradient(to right, #240b36, #c31432); /* W3C, IE 10+/ Edge, Firefox 16+,
Chrome 26+, Opera 12+, Safari 7+ */
  justify-content: center;
  align-items: center;
  height: 100vh;
  color: #fff;
}

.form_label {
  font-family: 'Roboto', sans-serif;
  font-size: 1.2rem;
  margin-left: 2rem;
  margin-top: 0.7rem;
  display: block;
  transition: all 0.3s;
  transform: translateY(0rem);
}

.form_input {
  top: -24px;
  font-family: 'Roboto', sans-serif;
```

```
  color: #333;
  font-size: 1.2rem;
  padding: 1.5rem 2rem;
  border-radius: 0.2rem;
  background-color: rgb(255, 255, 255);
  border: none;
  width: 75%;
  display: block;
  border-bottom: 0.3rem solid transparent;
  transition: all 0.3s;
}

.form_input:placeholder-shown + .form_label {
  opacity: 0;
  visibility: hidden;
  -webkit-transform: translateY(+4rem);transform:
  translateY(+4rem);
}


.button  {
  appearance: button;
  background-color: transparent;
  background-image: linear-gradient(to bottom, #fff, #f8eedb);
  border: 0 solid #e5e7eb;
  border-radius: .5rem;
  box-sizing: border-box;
  color: #073e39;
  column-gap: 1rem;
  cursor: pointer;
  display: flex;
  font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica
Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto
Color Emoji";
  font-size: 100%;
  font-weight:  700;
  line-height:  24px;
  margin: 0;
  outline: 2px solid transparent;
  padding: 1rem 1.5rem;
  text-align: center;
```

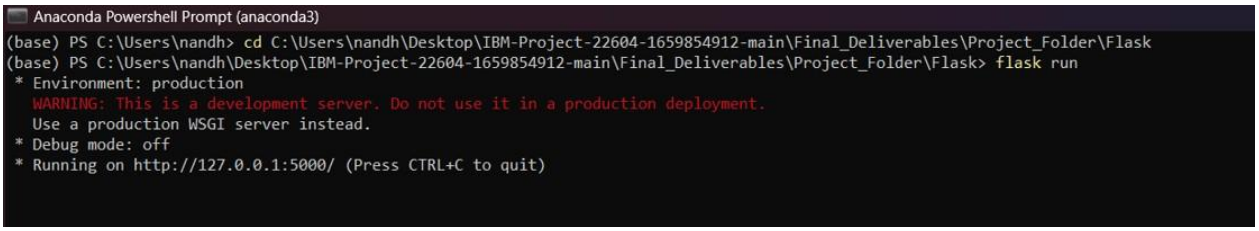
```
  text-transform: none;
  transition: all .1s cubic-bezier(.4, 0, .2, 1);
  user-select: none;
  -webkit-user-select: none;
  touch-action: manipulation;
  box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);
}

.button:active {
  background-color: #f3f4f6;
  box-shadow: -1px 2px 5px rgba(81,41,10,0.15),0px 1px 1px rgba(81,41,10,0.15);transform:
  translateY(0.125rem);
}

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


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

.button1{
  appearance: button;
  background-color: transparent;
  background-image: linear-gradient(to bottom, rgb(160, 245, 174), #37ee65);
  border: 0 solid #e5e7eb;
  border-radius: .5rem;
  box-sizing: border-box;
  color: #482307;
  column-gap: 1rem;
  cursor: pointer;
  display: flex;
  font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica
Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto
Color Emoji";
```

```
  font-size: 100%;
  font-weight:  700;
  line-height:  24px;
  margin: 0;
  outline: 2px solid transparent;
  padding: 1rem 1.5rem;
  text-align: center;
  text-transform: none;
  transition: all .1s cubic-bezier(.4, 0, .2, 1);
  user-select: none;
  -webkit-user-select: none;
  touch-action: manipulation;
  box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);
  display: none;
}

.button2{
  appearance: button;
  background-color: transparent;
  background-image: linear-gradient(to bottom, rgb(252, 162, 162), #ee3737);
  border: 0 solid #e5e7eb;
  border-radius: .5rem;
  box-sizing: border-box;
  color: #482307;
  column-gap: 1rem;
  cursor: pointer;
  display: flex;
  font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica
Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto
Color Emoji";
  font-size: 100%;
  font-weight:  700;
  line-height:  24px;
  margin: 0;
  outline: 2px solid transparent;
  padding: 1rem 1.5rem;
  text-align: center;
  text-transform: none;
  transition: all .1s cubic-bezier(.4, 0, .2, 1);
  user-select: none;
  -webkit-user-select: none;
```

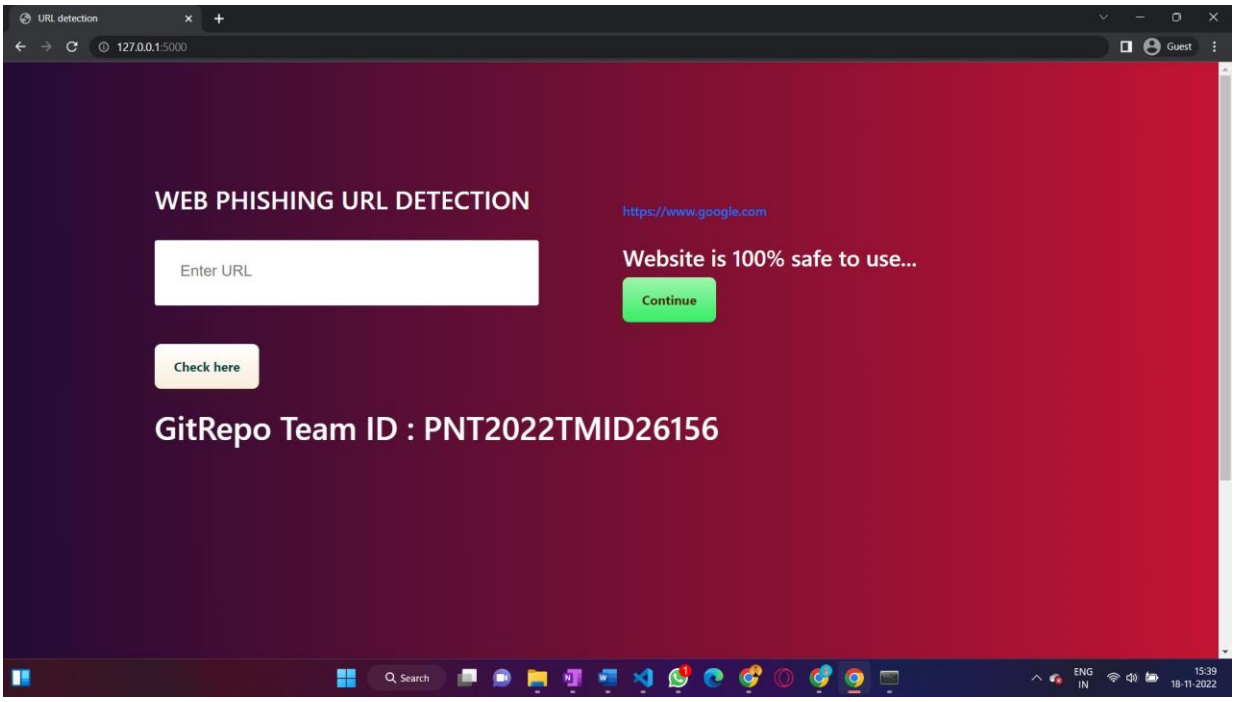
```
  touch-action: manipulation;
  box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);
  display: none;
}

.right {
  right: 0px;
  width: 300px;
}

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

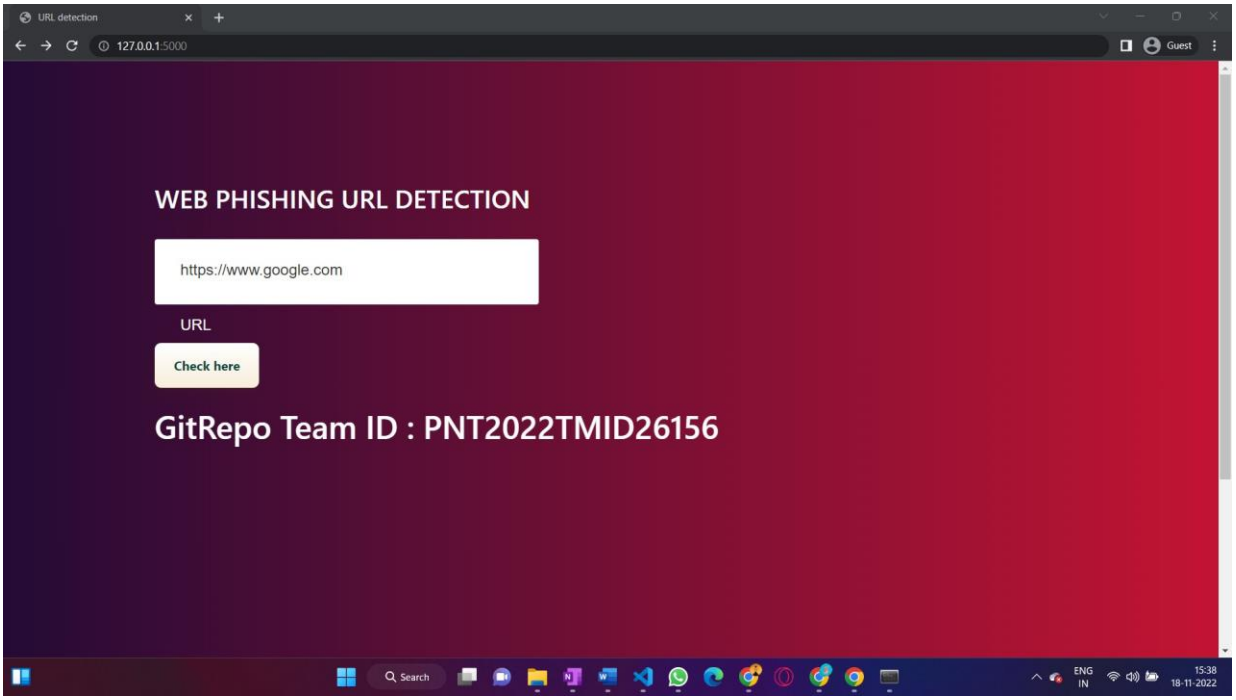
7.5 SOULTIONING

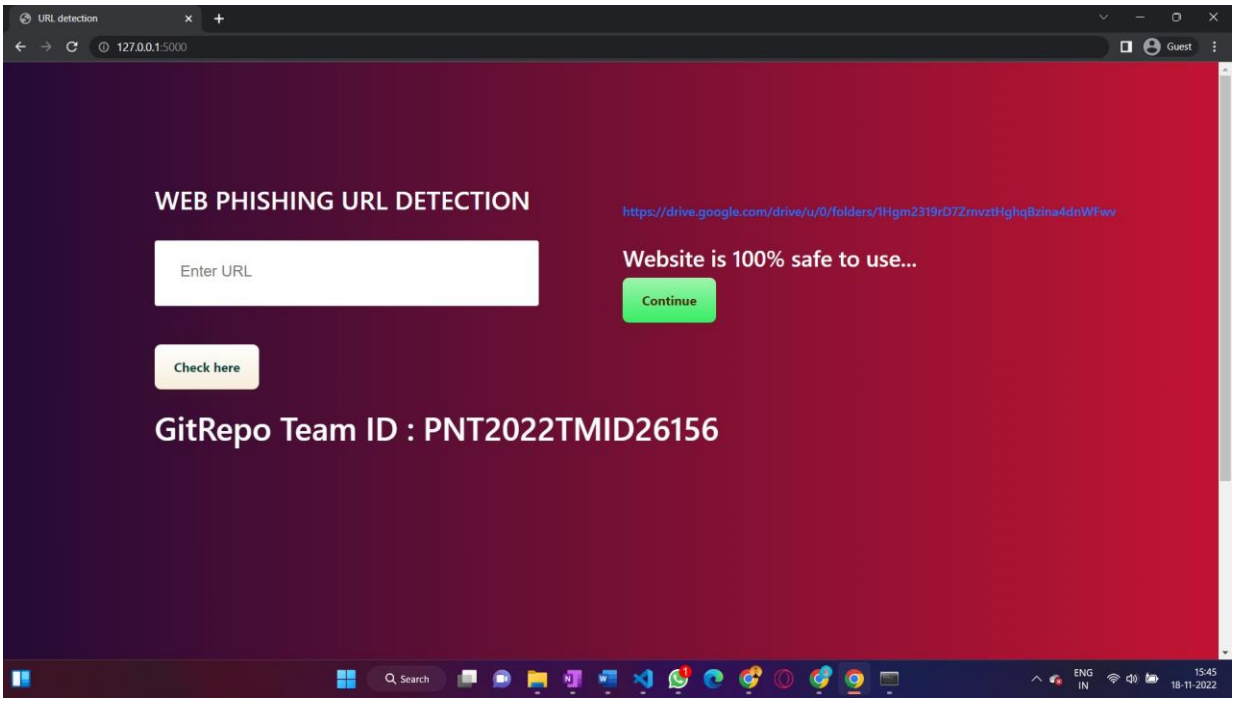
Running **app.py** in Anaconda PowerShell Prompt

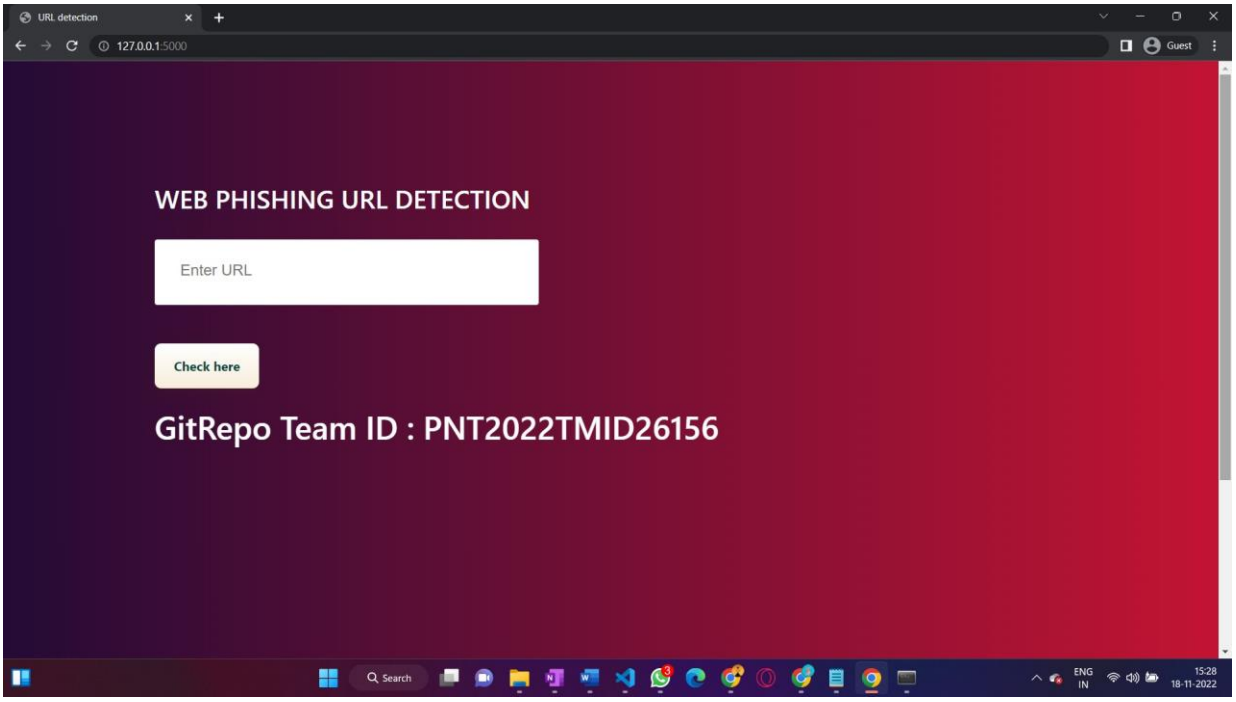


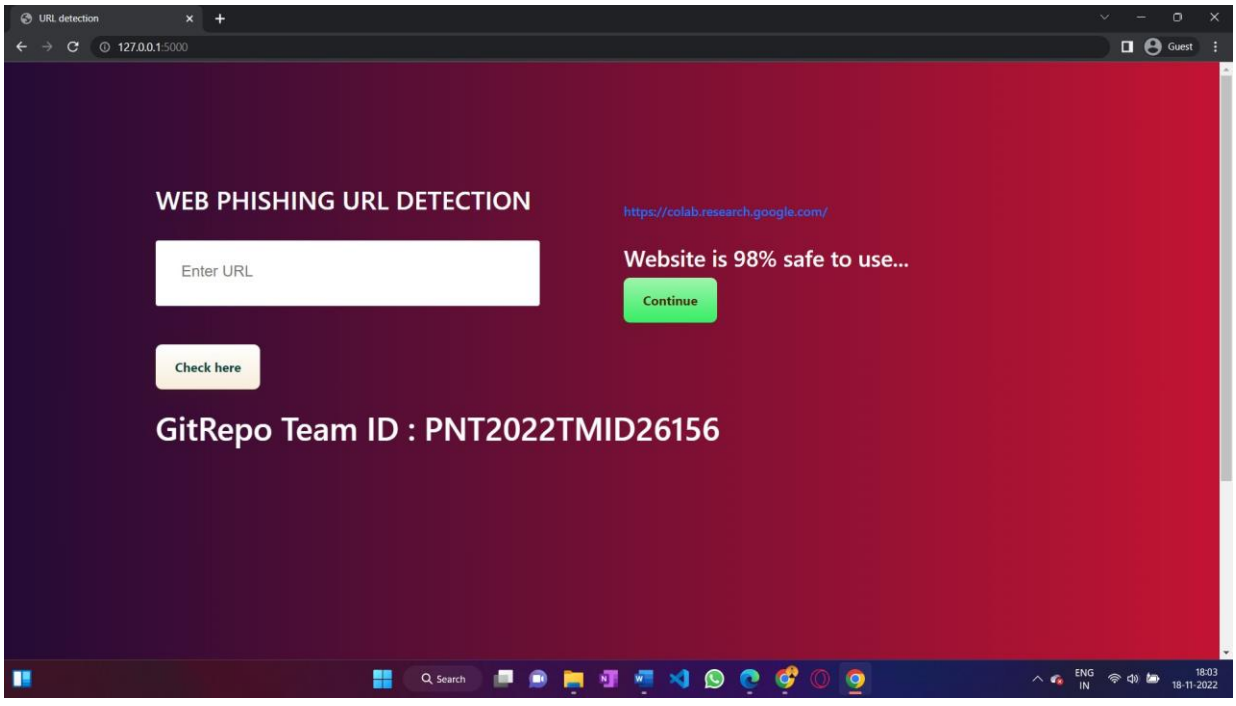
index.html (Home page of the web app is executed)




URL detection
127.0.0.1:5000
Guest
WEB PHISHING URL DETECTION
https://www.google.com
URL
Check here
GitRepo Team ID : PNT2022TMID26156
Search
ENG
IN
15:38
18-11-2022


URL detection
127.0.0.1:5000
Guest
WEB PHISHING URL DETECTION
https://drive.google.com/drive/u/0/folders/1Hgm2319rD7ZrnvztHghqBzina4dnWFwv
Enter URL
Website is 100% safe to use...
Continue
Check here
GitRepo Team ID : PNT2022TMID26156
Search
ENG
IN
15:45
18-11-2022


URL detection
127.0.0.1:5000
Guest
WEB PHISHING URL DETECTION
Enter URL
Check here
GitRepo Team ID : PNT2022TMID26156
Search
ENG
IN
15:28
18-11-2022


URL detection
127.0.0.1:5000
Guest
WEB PHISHING URL DETECTION
Enter URL
Check here
https://colab.research.google.com/
Website is 98% safe to use...
Continue
GitRepo Team ID : PNT2022TMID26156
Search
ENG
IN
18:03
18-11-2022

8. TESTING

8.1 Test Cases

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps ToExecute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUGID	Executed By
LoginPage_TC_OO1	Functional	Home Page	Verify user is able to see the Landing Page when user can typethe URLin the box		1.Enter URL and click go2.Typethe URL 3.Verify whether it is processing or not.	https://phishing- shield.herokuapp.com/	Should Display the Webpage	Working as expected	Pass		N		Brinda P
LoginPage_TC_OO2	UI	Home Page	Verify the UI elements isResponsive		1. Enter URL and click go 2. Type or copy paste the URL 3. Check whether the button isresponsive or not 4. Repeat Last Test	https://phishing- shield.herokuapp.com/	Should Wait for Response andthengets Acknowledge	Working as expected	Pass		N		Nandhitha N
LoginPage_TC_OO3	Functional	Home page	Verify whether the link islegitimate or not		1. Enter URL and click go 2. Type or copy paste the URL 3. Check the website is legitimate or not 4. Observe the results	https://phishing- shield.herokuapp.com/	User should observe whether thewebsite is legitimate or not.	Working as expected	Pass		N		Yuvashree S
LoginPage_TC_OO4	Functional	Home Page	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 websiteis legitimateor not 4. Continue if the website is legitimate or be cautious if it isnot legitimate.	https://phishing- shield.herokuapp.com/	Application should showthat SafeWebpage or Unsafe.	Working as expected	Pass		N		Shobikka Y
LoginPage_TC_OO5	Functional	Home Page	Testing the website withmultiple URLs		1. Enter URL(https://phishing-shield.herokuapp.com/) and click go 2. Type or copy paste the URL totest 3. Check the websiteis legitimateor not 4. Continue if thewebsite is secure or be cautious if it is notsecure	1.https://drive.google.com/drive/u/0/my-drive2.https://www.google.com/ 3. https://www.photopea.com/ 4.https://dictionary.cambridge.org/	User can able to identify the websites whetherit is secureornot	Working as expected	Pass		N		Brinda P

8.2 User Acceptance Testing

Purpose of Document

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

Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	10	2	4	20	36
Not Reproduced	0	0	1	0	1
Skipped	0	0	0	0	0
Won't Fix	0	0	2	1	3
Totals	23	9	12	25	70

Test Case Analysis

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

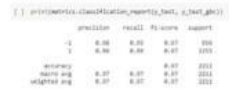
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	10	0	0	10
Client Application	50	0	0	50
Security	5	0	0	4
Outsource Shipping	3	0	0	3
Exception Reporting	10	0	0	9
Final Report Output	10	0	0	10
Version Control	4	0	0	4

9. RESULTS

9.1 Performance Metrics

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Classification Model: Gradient Boosting Classification** Accuray Score- 97.1%	
2.	Tune the Model	Hyperparameter Tuning - 97% Validation Method – KFOLD & Cross Validation Method	

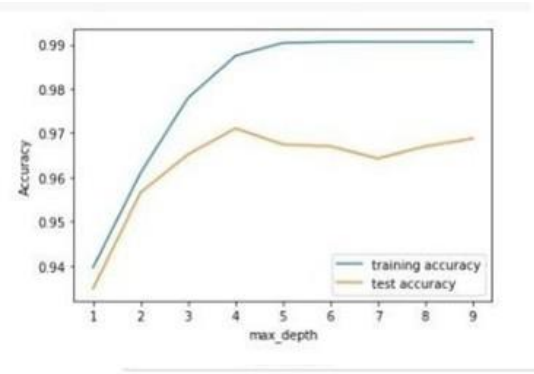
METRICS:

CLASSIFICATION REPORT:

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

	precision	recall	f1-score	support
-1	0.98	0.95	0.97	956
1	0.96	0.99	0.97	1255
accuracy			0.97	2211
macro avg	0.97	0.97	0.97	2211
weighted avg	0.97	0.97	0.97	2211

PERFORMANCE



	ML Model	Accuracy	f1_score	Recall	Precision
0	Support Vector Machine	0.957	0.963	0.982	0.966
1	Logistic Regression	0.924	0.933	0.947	0.927
2	K-Nearest Neighbors	0.953	0.959	0.990	0.989
3	Decision Tree	0.958	0.963	0.992	0.991
4	Gradient Boosting Classifier	0.971	0.975	0.992	0.985
5	Random Forest	0.964	0.969	0.992	0.989

TUNE THE MODEL – HYPERPARAMETER TUNING

```
gbc.fit(X_train,y_train)
```

```
GradientBoostingClassifier(learning_rate=0.7, max_depth=4)
```

Out[58]:

GridSearchCV

```
GridSearchCV(cv=5,
             estimator=GradientBoostingClassifier(learning_rate=0.7,
                                                  max_depth=4),
             param_grid={'max_features': array([1, 2, 3, 4, 5]),
                         'n_estimators': array([ 10,  20,  30,  40,  50,  60,  70,  80,  90, 100, 110, 120, 130,
       140, 150, 160, 170, 180, 190, 200])})
```

estimator: GradientBoostingClassifier

```
GradientBoostingClassifier(learning_rate=0.7, max_depth=4)
```

GradientBoostingClassifier

```
GradientBoostingClassifier(learning_rate=0.7, max_depth=4)
```

VALIDATION METHODS: KFOLD & Cross Folding

Wilcoxon signed-rank test

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

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

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

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

Out[78]: 95.0

5x2CV combined F test

In [89]:
```
from mlxtend.evaluate import combined_ftest_5x2cv
from sklearn.tree import DecisionTreeClassifier, ExtraTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier
from mlxtend.data import iris_data

# Prepare data and clfs
X, y = iris_data()
clf1 = GradientBoostingClassifier()
clf2 = DecisionTreeClassifier()

# Calculate p-value
f, p = combined_ftest_5x2cv(estimator1=clf1,
                            estimator2=clf2,
                            X=X, y=y,
                            random_seed=1)

print('f-value:', f)
print('p-value:', p)
```

```
f-value: 1.727272727272733
p-value: 0.2840135734291782
```

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

● In this system , we have used Gradient boosting algorithm which has better performance when compared to other traditional classifications algorithms.

● User can purchase products online and make payments securely without any hesitation.

DISADVANTAGES:

● This system won't work, if the Internet connection lost.

11. CONCLUSION

The demonstration of phishing is turning into an advanced danger to this quickly developing universe of innovation. The project was carried out in Anaconda IDE and was written in Python. The proposed method uses four machine learning classifiers to achieve this and a comparative study of the six algorithms was made. A good accuracy score was also achieved. The six algorithms used are K-Nearest neighbor, Support vector Algorithm, Logistic regression Decision Tree ,Gradient Boosting Algorithm & Random Forest Classifier. All the six classifiers gave promising results with the best being Gradient Boosting Classifier with an accuracy score of 97.1%. The accuracy score might vary with datasets and other algorithms. Gradient Boosting Algorithm is an ensemble classifier and hence the high accuracy. This model can be deployed in real time to detect the URLs as phishing or legitimate.

12. FUTURE SCOPE

Further work can be done to enhance the model by using ensembling models to get greater accuracy score. Ensemble methods is a machine learning technique that combines many base models to generate an optimal predictive model. Further reaching future work would be combining multiple classifiers, trained on different aspects of the same training set, into a single classifier that may provide a more robust prediction than any of the single classifiers on their own. The project can also include other variants of phishing like smishing, vishing, etc. to complete the system. The collections will ideally grow incrementally over time so there will need to be a way to apply a classifier incrementally to the new data, but also potentially have this classifier receive feedback that might modify it over time.

13. APPENDIX

Git hub link: https:/ github.com/IBM-EPBL/IBM-Project-4073-1658682850

Project demo link:
https://github.com/IBM-EPBL/IBM-Project40731658682850/blob/main/Final%20Deliverables/Demo_video.mp4