PROJECT REPORT

WEB PHISHING DETECTION

Submitted by

Team ID : PNT2022TMID29453

M.MOHAN RAJ(510419104047)

R.ARUNACHALAM(510419104015)

E.RAJESH(510419104055)

R.DHEERATH KUMAR(510419104024)

in partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

ARUNAI ENGINEERING COLLEGE

TIRUVANNAMALAI

ANNA UNIVERSITY: CHENNAI 600 025

NOV-DEC 2022

TABLE OF CONTENTS

CHAPTER1

INTRODUCTION

1.1 Project Overview

Now a days Phishing becomes a main area of concern for security researchers because it is not difficult to create the fake website which looks so close to legitimate website. Experts can identify fake websites but not all the users can identify the fake website and such users become the victim of phishing attack. Main aim of the attacker is to steal banks account credentials. In United States businesses, there is a loss of US$2billion per year because their clients become victim to phishing .

Phishing attacks are becoming successful because lack of user awareness. Since phishing attack exploits the weaknesses found in users, it is very difficult to mitigate them but it is very important to enhance phishing detection techniques. The general method to detect phishing websites by updating blacklisted URLs, Internet Protocol (IP) to the antivirus database which is also known as “blacklist" method. To evade blacklists attackers uses creative techniques to fool users by modifying the URL to appear legitimate via obfuscation and many other simple techniques including: fast-flux, in which proxies are automatically generated to host the web-page; algorithmic generation of new URLs; etc. Major drawback of this method is that, it cannot detect zero-hour phishing attack. Heuristic based detection which includes characteristics that are found to exist in phishing attacks in reality and can detect zero-hour phishing attack, but the characteristics are not guaranteed to always exist in such attacks and false positive rate in detection is very high.

To overcome the drawbacks of blacklist and heuristics based method, many security researchers now focused on machine learning techniques. Machine learning technology consists of a many algorithms which requires past data to make a decision or prediction on future data. Using this technique, algorithm will analyze various blacklisted and legitimate URLs and their features to accurately detect the phishing websites including zero- hour phishing websites.

1.2 Purpose

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

Common threats of web phishing:

- Web phishing aims to steal private information, such as usernames, passwords, and credit card details, by way of impersonating a legitimate entity.
- It will lead to information disclosure and property damage.
- Large organizations may get trapped in different kinds of scams.

This Guided Project mainly focuses on applying a machine-learning algorithm to detect Phishing websites.
In order to detect and predict e-banking phishing websites, we proposed an intelligent, flexible and effective system that is based on using classification algorithms. We implemented classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy. The e-banking phishing website can be detected based on some important characteristics like URL and domain identity, and security and encryption criteria in the final phishing detection rate. Once a user makes a transaction online when he makes payment through an e-banking website our system will use a data mining algorithm to detect whether the ebanking website is a phishing website or not.

CHAPTER2

LITERATURESURVEY

The purpose or goal behind phishing is data, money or personal information stealing through the fake website. The best strategy for avoiding the contact with the phishing web site is to detect real time malicious URL. Phishing websites can be determined on the basis of their domains. They usually are related to URL which needs to be registered (low-level domain and upper-level domain, path, query). Recently acquired status of intra-URL relationship is used to evaluate it using distinctive properties extracted from words that compose a URL based on query data from various search engines such as Google and Yahoo. These properties are further led to the machine-learningbased classification for the identification of phishing URLs from a real dataset. This paper focus on real time URL phishing against phishing content by using phish-STORM. For this a few relationship between the register domain rest of the URL are consider also intra URL relentless is consider which help to dusting wish between phishing or non phishing URL. For detecting a phishing website certain typical blacklisted urls are used, but this technique is unproductive as the duration of phishing websites is very short. Phishing is the name of avenue. It can be defined as the manner of deception of an organization's customer to communicate with their confidential information in an unacceptable behaviour. It can also be defined as intentionally using harsh weapons such as Spasm to automatically target the victims and targeting their private information. As many of the failures being occurred in the SMTP are exploiting vectors for the phishing websites, there is a greater availability of communication for malicious message deliveries. Proposed a novel classification approach that use heuristic based feature extraction approach. In this, they have classified extracted features into different categories such as URL Obfuscation features, Hyperlink-based features. Moreover, proposed technique gives 92.5% accuracy. Also this model is purely depends on the quality and quantity of the training set and Broken links feature extraction.

2.1 Existing problem

Internet has been become an essential component of our everyday social and financial activities. Nevertheless, internet users may be vulnerable to different types of web threats, which may cause financial damages, identify theft,loss of private information, brand reputation damage and loss of customers confidence in e-commerce and online banking. Phishing is considered as a form of web threats that is defined as the art of impersonating a website of an honest enterprise aiming to obtain confidential information such as usernames, passwords and social security number.

So far, there is no single solution that can capture every phishing attack. In this article,we proposed an intelligent model for predicting phishing attacks based on artificial neural network particularly self- structuring neural networks. phishing is continuous problem where features significant in determining the type of web page are constantly changing. Thus,we need to constantly improve the network structure in order to cope with these changes.

Our model solves this problem by automating the process of structuring the network and shows high acceptance for noisy data , fault tolerance and high prediction accuracy. several experiments were conducted in our research,and the number of epochs differs in each experiment

2.2 References

[1] Gunter Ollmann, “The Phishing Guide Understanding & Preventing Phishing Attacks”, IBMInternet Security Systems, 2007.

[2] https://resources.infosecinstitute.com/category/enterprise /phishing/the-phishinglandscape/phishing-data-attack statistics/#gref

[3] Mahmoud Khonji, Youssef Iraqi, "Phishing Detection: A Literature Survey IEEE, and Andrew Jones, 2013

[4] Mohammad R., Thabtah F. McCluskey L., (2015) Phishing websites dataset.
Available: https://archive.ics.uci.edu/ml/datasets/Phishing+Websites Accessed January 2016

[5] http://dataaspirant.com/2017/01/30/how-decision-tree algorithm-works/

[6] http://dataaspirant.com/2017/05/22/random-forest algorithm-machine-learing/

[7] https://www.kdnuggets.com/2016/07/support-vector machines-simple-explanation.html

[8] www.alexa.com

[9] www.phishtank.com

2.3 Problem Statement Definition

Web Phishing is a form of cyber fraud, which implies that fraudsters use various means to impersonate the URL address and page content of a real website or use vulnerabilities in the server program of a real website to insert dangerous HTML code in certain pages of the site.

It is a threat in various aspects of security on the internet, which might involve scams and private information disclosure. Some of the common threats of web phishing are:

- Obtaining personal information from an individual or organization.
- Impersonating as a trustworthy organization to deliver malicious websites.

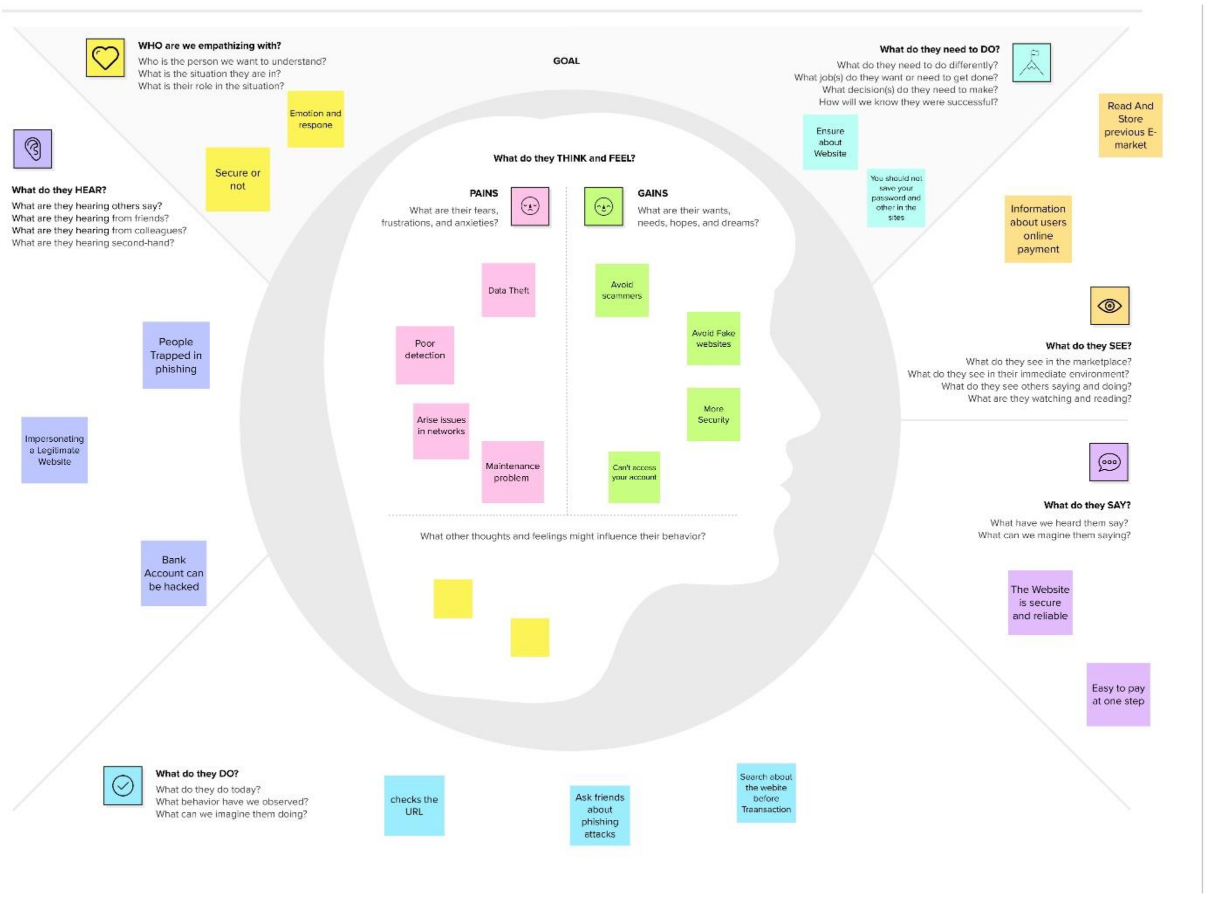
To avoid these threats, we build an efficient and intelligent system to detect such websites using machine-learning algorithms which implements classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy.

CHAPTER3

IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

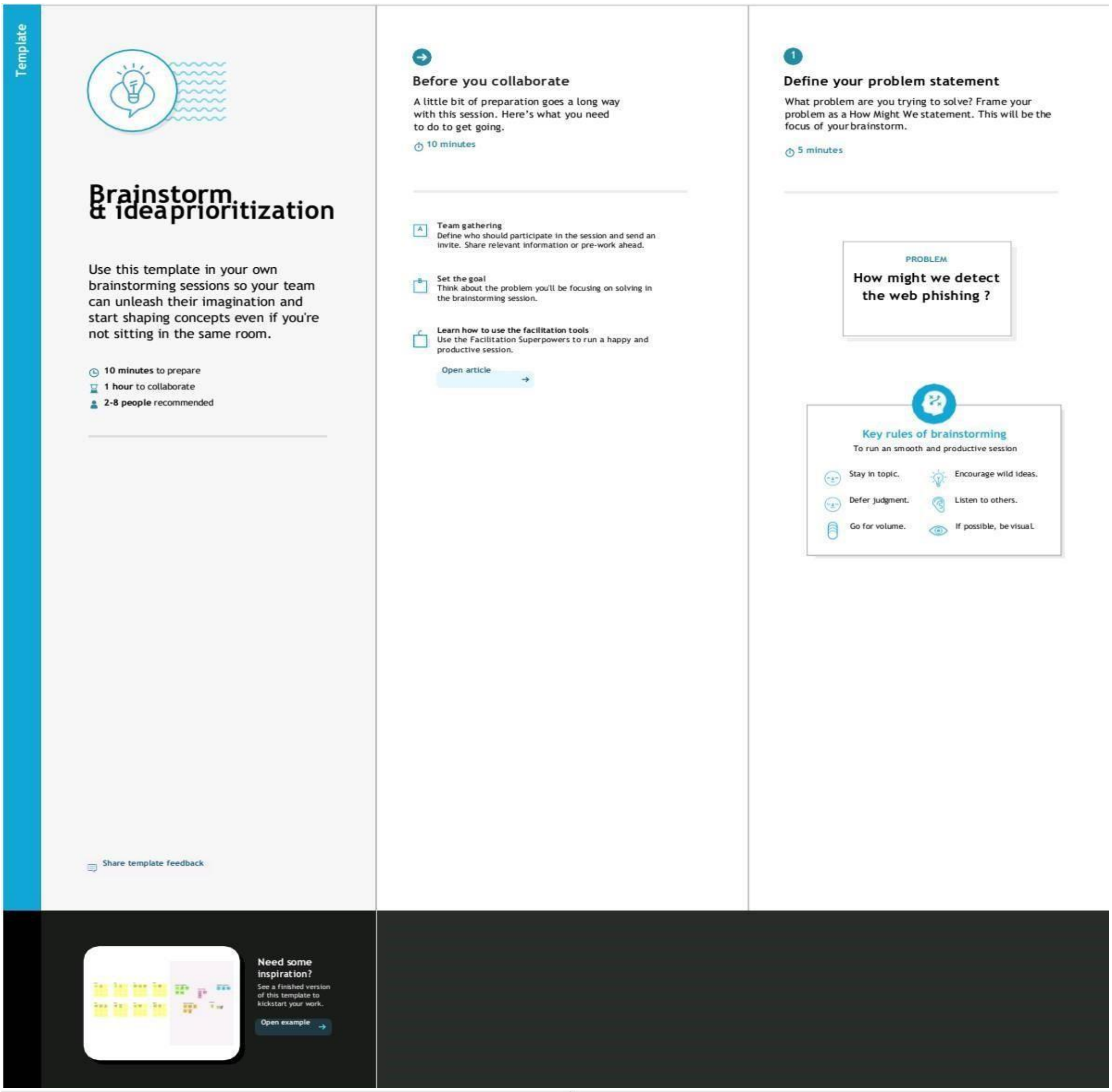
An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment. Empathy maps should be used throughout any UX process to establish common ground among team members and to understand and prioritize user needs. In usercentered design, empathy maps are best used from the very beginning of the design process.


WHO are we empathizing with?
Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?
GOAL
What do they need to DO?
What do they need to do differently?
What job(s) do they want or need to get done?
What decision(s) do they need to make?
How will we know they were successful?
Emotion and respone
Secure or not
Ensure about Website
You should not save your password and other in the sites
Read And Store previous E-market
Information about users online payment
What do they HEAR?
What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?
People Trapped in phishing
Impersonating a Legitimate Website
Bank Account can be hacked
What do they THINK and FEEL?
PAINS
What are their fears, frustrations, and anxieties?
GAINS
What are their wants, needs, hopes, and dreams?
Data Theft
Poor detection
Arise issues in networks
Maintenance problem
Avoid scammers
Avoid Fake websites
More Security
Can't access your account
What other thoughts and feelings might influence their behavior?
What do they SEE?
What do they see in the marketplace?
What do they see in their immediate environment?
What do they see others saying and doing?
What are they watching and reading?
What do they SAY?
What have we heard them say?
What can we magine them saying?
The Website is secure and reliable
Easy to pay at one step
What do they DO?
What do they do today?
What behavior have we observed?
What can we imagine them doing?
checks the URL
Ask friends about phishing attacks
Search about the webite before Traansaction

3.2 Ideation & Brainstorming

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

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



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

Share template feedback

Need some inspiration?
See a finished version of this template to kickstart your work.
Open example →

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

A Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

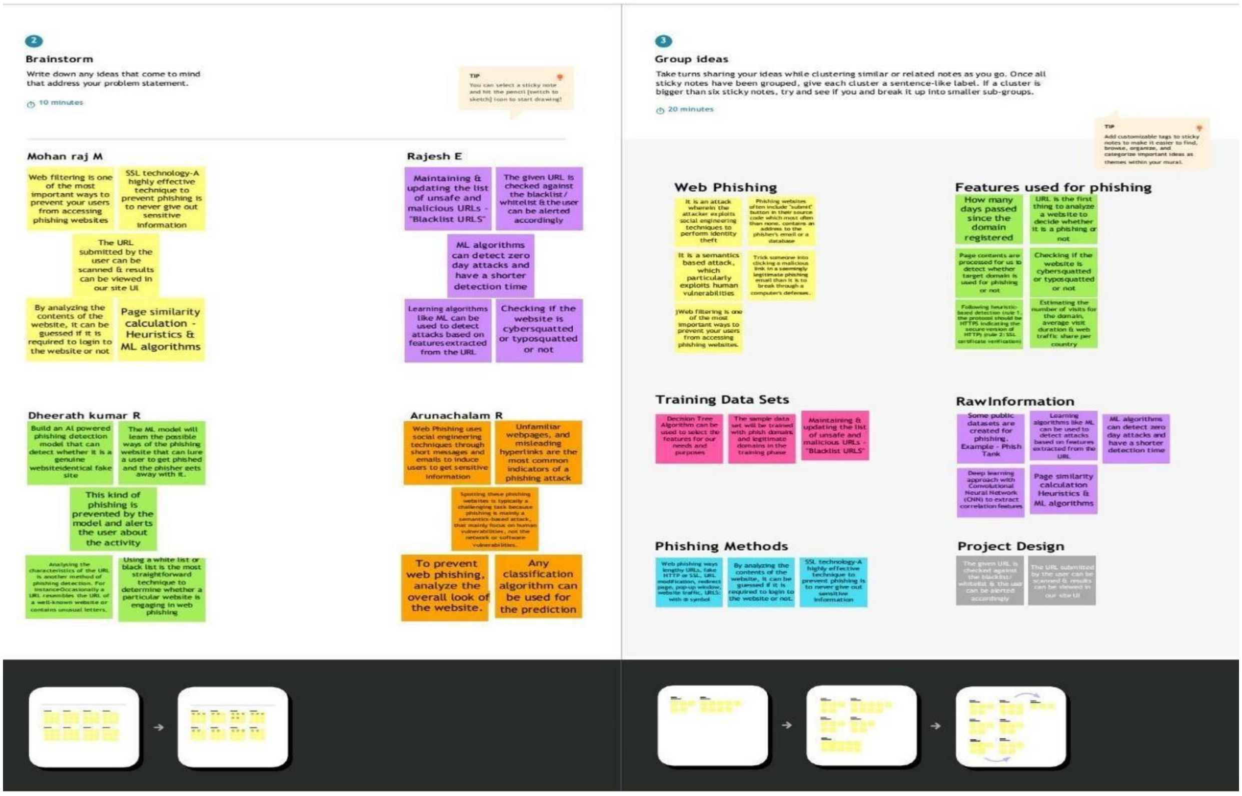
B Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

C 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
How might we detect the web phishing ?

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

TIP
You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Mohan raj M
- Web filtering is one of the most important ways to prevent your users from accessing phishing websites
- SSL technology-A highly effective technique to prevent phishing is to never give out sensitive information
- The URL submitted by the user can be scanned & results can be viewed in our site UI
- By analyzing the contents of the website, it can be guessed if it is required to login to the website or not
- Page similarity calculation - Heuristics & ML algorithms

Rajesh E
- Maintaining & updating the list of unsafe and malicious URLs - "Blacklist URLS"
- The given URL is checked against the blacklist / whitelist & the user can be alerted accordingly
- ML algorithms can detect zero day attacks and have a shorter detection time
- Learning algorithms like ML can be used to detect attacks based on featuresextracted from the URL
- Checking if the website is cybersquatted or typosquatted or not

Dheerath kumar R
- Build an AI powered phishing detection model that can detect whether it is a genuine websiteidentical fake site
- The ML model will learn the possible ways of the phishing website that can lure a user to get phished and the phisher gets away with it.
- This kind of phishing is prevented by the model and alerts the user about the activity
- Analysing the characteristics of the URL is another method of phishing detection. For instanceOccasionally a URL resembles the URL of a well-known website or contains unusual letters.
- Using a white list or black list is the most straightforward technique to determine whether a particular website is engaging in web phishing

Arunachalam R
- Web Phishing uses social engineering techniques through short messages and emails to induce users to get sensitive information
- Unfamiliar webpages, and misleading hyperlinks are the most common indicators of a phishing attack
- Spotting these phishing websites is typically a challenging task because phishing is mainly a semantics-based attack, that mainly focus on human vulnerabilities, not the network or software vulnerabilities.
- To prevent web phishing, analyze the overall look of the website.
- Any classification algorithm can be used for the prediction

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

TIP
Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Web Phishing
- It is an attack wherein the attacker exploits social engineering techniques to perform identity theft
- Phishing websites often include "submit" button in their source code which most often than none, contains an address to the phisher's email or a database
- It is a semantics based attack, which particularly exploits human vulnerabilities
- Trick someone into clicking a malicious link in a seemingly legitimate phishing email than it is to break through a computer's defenses.
-)Web filtering is one of the most important ways to prevent your users from accessing phishing websites.

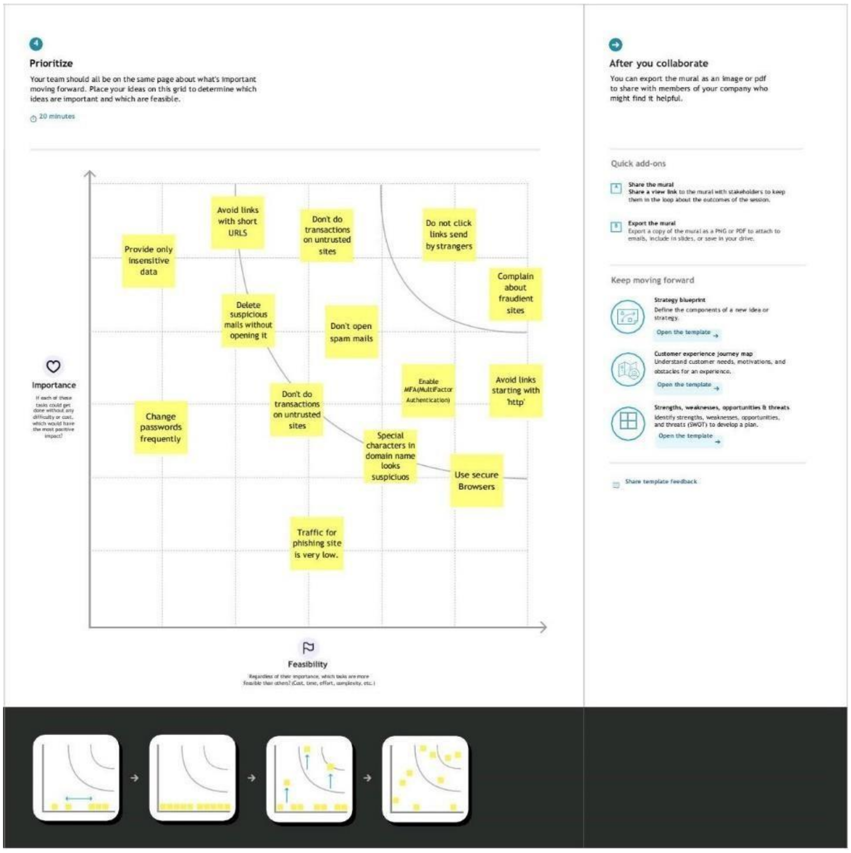
Features used for phishing
- How many days passed since the domain registered
- URL is the first thing to analyze a website to decide whether it is a phishing or not
- Page contents are processed for us to detect whether target domain is used for phishing or not
- Checking if the website is cybersquatted or typosquatted or not
- Following heuristic-based detection (rule 1. the protocol should be HTTPS indicating the secure version of HTTP) (rule 2: SSL certificate verification)
- Estimating the number of visits for the domain, average visit duration & web traffic share per country

Training Data Sets
- Decision Tree Algorithm can be used to select the features for our needs and purposes
- The sample data set will be trained with phish domains and legitimate domains in the training phase
- Maintaining & updating the list of unsafe and malicious URLs - "Blacklist URLS"

RawInformation
- Some public datasets are created for phishing. Example - Phish Tank
- Learning algorithms like ML can be used to detect attacks based on features extracted from the URL
- ML algorithms can detect zero day attacks and have a shorter detection time
- Deep learning approach with Convolutional Neural Network (CNN) to extract correlation features
- Page similarity calculation Heuristics & ML algorithms

Phishing Methods
- Web phishing ways lengthy URLs, fake HTTP or SSL, URL modification, redirect page, pop-up window, website traffic, URLS: with @ symbol
- By analyzing the contents of the website, it can be guessed if it is required to login to the website or not.
- SSL technology-A highly effective technique to prevent phishing is to never give out sensitive information

Project Design
- The given URL is checked against the blacklist/ whitelist & the user can be alerted accordingly
- The URL submitted by the user can be scanned & results can be viewed in our site UI



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

Importance

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

Avoid links with short URLS

Don't do transactions on untrusted sites

Do not click links send by strangers

Provide only insensitive data

Complain about fraudient sites

Delete suspicious mails without opening it

Don't open spam mails

Enable MFA(MultiFactor Authentication)

Avoid links starting with 'http'

Change passwords frequently

Don't do transactions on untrusted sites

Special characters in domain name looks suspiciuos

Use secure Browsers

Traffic for phishing site is very low.

Feasibility

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

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

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

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

Project team shall fill the following information in proposed solution template.

S.No	Parameter	Description
1.	Problem Statement (Problem tobesolved)	**PROBLEM** ➢ Phishing detection techniques suffer low detectionaccuracy &high false alarm speciallyin phishing **SOLUTION** ➢ Use anti-phishing protection & anti-spam software to protect yourself when maliciousmessage slip through to your computer Anti-malware is included to prevent other types ofthreats ➢ Spam software ➢ Anti-malware software These are programmed by security researches to spoteven the stealthiest +malware.
2.	Idea / Solution description	**IDEA** ➢ Phishing assaults are usually detected by experiencedusers however, security is a primaryconcern for system users who are unaware & such situations **SOLUTOIN** ➢ Anti-phishing technology is designed to identityand block phishing emails using a variety of methods certain anti-phishing solution scan the content of inbound & internal emails for any sign of language that suggest a potentialphishing or impersonation attack.
3.	Novelty / Uniqueness	**NOVELTY/UNIQNESS** ➢ Since many phishing sites examines stayed live for atleast 48 hours, we monitored all sites for atleast 2 days. Based on Cyrene's analysis, googlechrome & fire fox did the best job dectection & blocking knows phishing sites with chrome blocking 74% phis site with in 6 hours 20 min on average.
4.	Social Impact / Customer Satisfaction	➢ As a result of this, the organization as well consumers are facing enormous social effectsphishing is causing 2-way damage ➢ In the process of phishing, hackers mainly focuson extraction the details like bank account details, redidcard passwords and Aadhar card verification
5.	Business Model (Revenue Model)	**BUSINESS MODEL** ➢ Linking Aadhar card ➢ Frame work

6.	Scalability of the Solution	**SCALABILITY OF THE SOLUTION** ➢ Avoid falling victims to cyber-attacks including phishing, spear phishing and whaling phishing attempts account for the majority of malware &ransomware attacks. **Highlights:** • CRN Magazines available • Newsletters available

3.4 Problem Solution fit

The Problem-Solution Fit simply means that you have found a problem with your customer andthat the solution you have realized for it actually solves the customer's problem. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would workand why

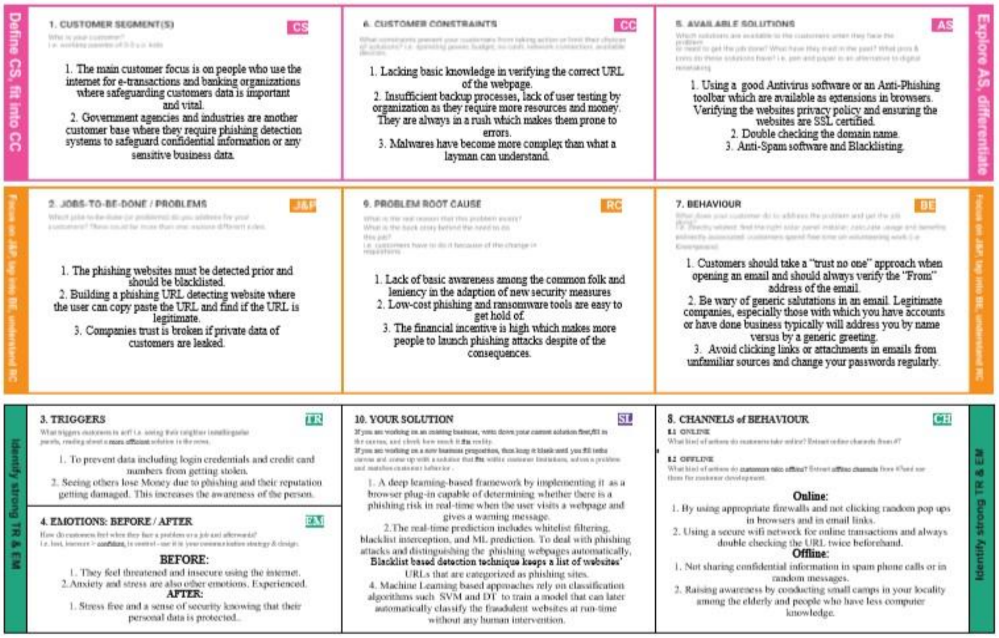
Purpose:

❑ Solve complex problems in a way that fits the state of your customers.

❑ Succeed faster and increase your solution adoption by tapping into existing mediums andchannels of behaviour.

❑ Sharpen your communication and marketing strategy with the right triggers and messaging.

❑ Understand the existing situation in order to improve it for your target group.



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

1. The main customer focus is on people who use the internet for e-transactions and banking organizations where safeguarding customers data is important and vital.
2. Government agencies and industries are another customer base where they require phishing detection systems to safeguard confidential information or any sensitive business data.

6. CUSTOMER CONSTRAINTS CC

1. Lacking basic knowledge in verifying the correct URL of the webpage.
2. Insufficient backup processes, lack of user testing by organization as they require more resources and money. They are always in a rush which makes them prone to errors.
3. Malwares have become more complex than what a layman can understand.

5. AVAILABLE SOLUTIONS AS

1. Using a good Antivirus software or an Anti-Phishing toolbar which are available as extensions in browsers. Verifying the websites privacy policy and ensuring the websites are SSL certified.
2. Double checking the domain name.
3. Anti-Spam software and Blacklisting.

Explore AS, differentiate

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

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

1. The phishing websites must be detected prior and should be blacklisted.
2. Building a phishing URL detecting website where the user can copy paste the URL and find if the URL is legitimate.
3. Companies trust is broken if private data of customers are leaked.

9. PROBLEM ROOT CAUSE RC

1. Lack of basic awareness among the common folk and leniency in the adaption of new security measures
2. Low-cost phishing and ransomware tools are easy to get hold of.
3. The financial incentive is high which makes more people to launch phishing attacks despite of the consequences.

7. BEHAVIOUR BE

1. Customers should take a "trust no one" approach when opening an email and should always verify the "From" address of the email.
2. Be wary of generic salutations in an email. Legitimate companies, especially those with which you have accounts or have done business typically will address you by name versus by a generic greeting.
3. Avoid clicking links or attachments in emails from unfamiliar sources and change your passwords regularly.

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

Identify strong TR & EM

3. TRIGGERS TR

1. To prevent data including login credentials and credit card numbers from getting stolen.
2. Seeing others lose Money due to phishing and their reputation getting damaged. This increases the awareness of the person.

4. EMOTIONS: BEFORE / AFTER EM

BEFORE:

1. They feel threatened and insecure using the internet.
2. Anxiety and stress are also other emotions. Experienced.

AFTER:

1. Stress free and a sense of security knowing that their personal data is protected.

10. YOUR SOLUTION SL

1. A deep learning-based framework by implementing it as a browser plug-in capable of determining whether there is a phishing risk in real-time when the user visits a webpage and gives a warning message.
2. The real-time prediction includes whitelist filtering, blacklist interception, and ML prediction. To deal with phishing attacks and distinguishing the phishing webpages automatically, **Blacklist based detection technique keeps a list of websites'** URLs that are categorized as phishing sites.
4. Machine Learning based approaches rely on classification algorithms such SVM and DT to train a model that can later automatically classify the fraudulent websites at run-time without any human intervention.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

8.2 OFFLINE

Online:

1. By using appropriate firewalls and not clicking random pop ups in browsers and in email links.
2. Using a secure wifi network for online transactions and always double checking the URL twice beforehand.

Offline:

1. Not sharing confidential information in spam phone calls or in random messages.
2. Raising awareness by conducting small camps in your locality among the elderly and people who have less computer knowledge.

Identify strong TR & EM

4.CHAPTER

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

A function of software system is defined in functional requirement and the behavior of the system is evaluated when presented with specific inputs or conditions which may include calculations, data manipulation and processing and other specific functionality.

FR NO.	**Functional** **Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	User Input	User inputs an URL in required field to check its validation.
FR-2	Website Comparison	Model compares the websites using Blacklist andWhite list approach.
FR-3	Feature extraction	After comparing, if none found on comparison then it extracts feature using heuristic and visual similarity approach.
FR-4	Prediction	Model predicts the URL using Machine Learning algorithms such as Logistic Regression, KNN
FR-5	Classifier	Model sends all output to classifier and producesfinal result.
FR-6	Announcement	Model then displays whether website is a legal siteor a phishing site.
FR-7	Events	This model needs the capability of retrieving anddisplaying accurate result for a website

4.2 NON-FUNCTIONAL REQUIREMENTS

FR No.	**Non-Functional Requirement**	**Description**

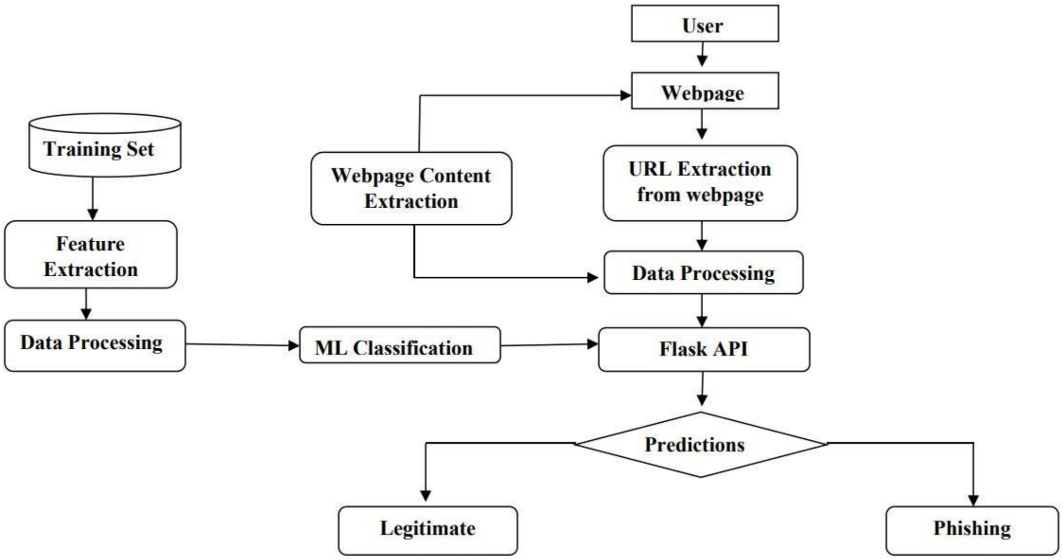
NFR-1	Usability	It is an easy to use and access interface which results in greater efficiency.
NFR-2	Security	It is a secure website which protects the sensitive information of the user and prevents malicious attacks.
NFR-3	Reliability	The system can detect phishing websites with greater accuracy using ML algorithms.
NFR-4	Performance	The system produces responses within seconds and execution is faster.
NFR-5	Availability	Users can access the website via any browser from anywhere at any time.
NFR-6	Scalability	This application can be accessed online without paying. It can detect any web site with high accuracy.

CHAPTER5

PROJECT DESIGN

5.1 Data Flow Diagrams

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

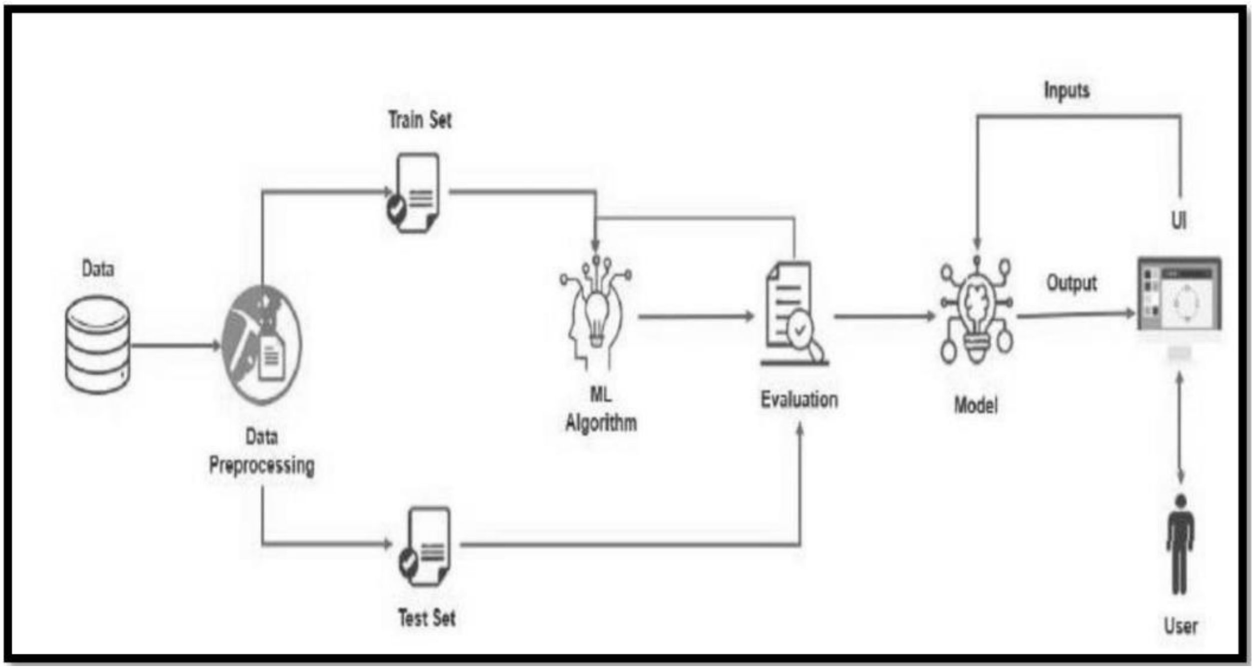


5.2 Solution & Technical Architecture

Our solution is to build an efficient and intelligent system to detect phishing sites by applying a machine learning algorithm which implements classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy by carefully analysing and identifying various factors that could be used to detect a phishing site. These factors fall under the categories of address bar-based features, domain-based features, HTML & JavaScript based features. Using these features, we can identify a phishing site with high accuracy.

Technical Architecture

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



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	I can receive confirmation email & click confirm	High	Sprint-1

			application			
		USN-3	As a user, I can register for the applicationthrough Facebook	I can register & accessthe dashboard with FacebookLogin	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 websitesfor security.	High	Sprint-1
Administrator	Prediction	USN-1	Here the Model will predict the URL websites using Machine Learning algorithms such as LogisticRegression, KNN	In this i can have the correct prediction on	High	Sprint-1

	Classifier	USN-2	Here i will send all the model output to classifier inorder to produce final result.	I this i will find the correct classifier fo	Medium	Sprint-2

CHAPTER6

PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint1	User input	USN-1	User inputs an URL in the required field to check its validation.	1	Medium	M.Mohan raj
Sprint1	Website Comparison	USN-2	Model compares the websites using Blacklistand Whitelist approach.	1	High	R.Arunachalam
Sprint2	Feature Extraction	USN-3	After comparison, if none found on comparison thenit extract feature using heuristic and visual similarity.	2	high	E.Rajesh
Sprint2	Prediction	USN-4	Model predicts the URL using Machine learningalgorithms such as logistic Regression, KNN.	1	Medium	R.Dheerath kumar
Sprint3	Classifier	USN-5	Model sends all the output to the classifier andproduces the final result.	1	Medium	M.Mohan raj
Sprint4	Announcement	USN-6	Model then displays whether the website islegal site or a phishing site.	1	High	R.Arunachalam
Sprint4	Events	USN-7	This model needs the capability of retrieving anddisplaying accurate result for a website.	1	High	R.Dheerath kumar

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA

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 iteration unit(story points per day)

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

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.

https://www.visual-paradigm.com/scrum/scrum-burndown-chart/
https://www.atlassian.com/agile/tutorials/burndown-charts

Reference: https://www.atlassian.com/agile/project-management

https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software

https://www.atlassian.com/agile/tutorials/epics

https://www.atlassian.com/agile/tutorials/sprints

https://www.atlassian.com/agile/project-management/estimation
https://www.atlassian.com/agile/tutorials/burndown-charts

CHAPTER7

CODING & SOLUTIONING

7.1 Feature 1

We define two kinds of features to detect web phishing, and they are an original feature and interactive feature.

Original Feature

There are some features in the phishing URL, such as special characters. We definite these features in URL as an original feature as follows:

- O1:there are special characters in URL, such as @, Unicode, and so on. Those special characters are not allowed in a normal URL.
- O2: there are too many dots or less than four dots in normal URL.
- O3: the age of the domain is too short. For example, the age of the normal domain is more than 3 months.

In order to quantify the above characteristics, all the characteristic values are binary, that is, one of 0 or 1. Intuitively, the more of the 1 appear in the feature, the higher the likelihood that the site will be a phishing site.

Interaction Feature

There are some features in graph G=(V,E), such as access frequency. We define these features through a node relationship as interaction feature as follows:

- I1: in-degree of URL node from REF is very small. In general, the normal websites do not link to phishing sites. The phishing sites are directly accessed.
- I2: out-degree of URL node is very small. In order to get personal private information, the phishing sites are usually terminal websites and do not link to the other sites.
- I3: the frequency of URL from AD is one. In general, one user accesses the phishing site only one time and the user cannot access the phishing site more than one time.
- I4: when AD accesses URL , user browser type UA is not the main browser. Well-known browser vendors often have a built-in filtering phishing site plug-in. A user who uses unknown browsers is more likely to access the phishing sites.
- I5: there is no cookie in user. The phishing site does not leave its cookie in user.

7.2 Feature 2

We have implemented python program to extract featuresfrom URL. Below are the features that we have extracted fordetection of phishing URLs.

1) **Presence of IP address in URL**: If IP address present in URL then the feature is set to 1 else set to 0. Most of the benign sites do not use IP address as an URL to download a webpage.
Use of IP address in URL indicates that attacker is trying to steal sensitive information.

2) **Presence of @ symbol in URL**: If @ symbol present in URL then the feature is set to 1 else set to 0. Phishers add special symbol @ in the URL leads the browser to ignore everything preceding the “@” symbol and the real address often follows the “@” symbol.

3) Number of dots in Hostname: Phishing URLs have many dots in URL. For example http://shop.fun.amazon.phishing.com, in this URL phishing.com is an actual domain name, whereas use of “amazon” word is to trick users to click on it. Average number of dots in benign URLs is 3. If the number of dots in URLs is more than 3 then the feature is set to 1 else to 0.

4) Prefix or Suffix separated by (-) to domain: If domain name separated by dash (-) symbol then feature is set to 1 else to 0. The dash symbol is rarely used I legitimate URLs. Phishers add dash symbol (-) to the domain name so that users feel that they are dealing with a legitimate webpage. For example Actual site is http://www.onlineamazon.com but phisher can create another fake website like http://www.online-amazon.com to confuse the innocent users.

5) URL redirection: If “//” present in URL path then feature is set to 1 else to 0. The existence of “//” within the URL path means that the user will be redirected to another website.

6) HTTPS token in URL: If HTTPS token present in URL then the feature is set to 1 else to 0. Phishers may add the “HTTPS” token to the domain part of a URL in order to trick users. For example, http://https-www-paypal-it-mpp-home.soft-hair.com.

7) Information submission to Email: Phisher might use “mail()” or “mailto:” functions to redirect the user’s information to his personal email[4]. If such functions are present in the URL then feature is set to 1 else to 0.

8) URL Shortening Services “TinyURL”: TinyURL service allows phisher to hide long phishing URL by making it short. The goal is to redirect user to phishing websites. If the URL is crafted using shortening services (like bit.ly) then feature is set to 1 else 0

9) Length of Host name: Average length of the benign URLs is found to be a 25, If URL’s length is greater than25 then the feature is set to 1 else to 0

10) Presence of sensitive words in URL: Phishing sites use sensitive words in its URL so that users feel that they are dealing with a legitimate webpage. Below are the words that found in many phishing URLs :- 'confirm', 'account', 'banking', 'secure', 'ebyisapi', 'webscr', 'signin', 'mail', 'install', 'toolbar', 'backup', 'paypal', 'password', 'username', etc;

11) Number of slash in URL: The number of slashes in benign URLs is found to be a 5; if number of slashes in URL is greater than 5 then the feature is set to 1 else to 0.

12) Presence of Unicode in URL: Phishers can make a use of Unicode characters in URL to trick users to click on it. For example the domain “xn--80ak6aa92e.com” is equivalent to "apple.com". Visible URL to user is "apple.com" but after clicking on this URL, user will visit to “xn--80ak6aa92e.com” which is a phishing site.

13) Age of SSL Certificate: The existence of HTTPS is very important in giving the impression of website legitimacy. But minimum age of the SSL certificate ofbenign website is between 1 year to 2 year.

14) URL of Anchor: We have extracted this feature by crawling the source code oh the URL. URL of the anchor is defined by <a> tag. If the <a> tag has a maximum number of hyperlinks which are from the other domain then the feature is set to 1 else to 0.

15) IFRAME: We have extracted this feature by crawling the source code of the URL. This tag is used to add another web page into existing main webpage. Phishers can make use of the "iframe" tag and make it invisible i.e. without frame borders. Since border of inserted webpage is invisible, user seems that the inserted web page is also the part of the main web page and can enter sensitive information.

16) Website Rank: We extracted the rank of websites and compare it with the first One hundred thousand websites of Alexa database. If rank of the website is greater than 10,0000 then feature

7.3 Database Scheme

MACHINE LEARNING ALGORITHM

Decision Tree Algorithm

One of the most widely used algorithm in machine learning technology. Decision tree algorithm is easy to understand and also easy to implement. Decision tree begins its work by choosing best splitter from the available attributes for classification which is considered as a root of the tree. Algorithm continues to build tree until it finds the leaf node. Decision tree creates training model which is used to predict target value or class in tree representation each internal node of the tree belongs to attribute and each leaf node of the tree belongs to class label. In decision tree algorithm, gini index and information gain methods are used to calculate these nodes.

Random Forest Algorithm

Random forest algorithm is one of the most powerful algorithms in machine learning technology and it is based on concept of decision tree algorithm. Random forest algorithm creates the forest with number of decision trees. High number of tree gives high detection accuracy. Creation of trees are based on bootstrap method. In bootstrap method features and samples of dataset are randomly selected with replacement to construct single tree. Among randomly selected features,random forest algorithm will choose best splitter for the classification and like decision tree algorithm; Random forest algorithm also uses gini index and information gain methods to find the best splitter. This process will get continue until random forest creates n number of trees. Each tree in forest predicts the target value and then algorithm will calculate the votes for each predicted target. Finally random forest algorithm considers high voted predicted target as a final prediction.

Support Vector Machine Algorithm

Support vector machine is another powerful algorithm in machine learning technology. In support vector machine algorithm each data item is plotted as a point in n-dimensional space and support vector machine algorithm constructs separating line for classification of two classes, this separating line is well known as hyper plane. Support vector machine seeks for the closest points called as support vectors and once it finds the closest point it draws a line connecting to them. Support vector machine then construct separating line which bisects and perpendicular to the connecting line. In order to classify data perfectly the margin should be maximum. Here the margin is a distance between hyperplane and support vectors. In real scenario it is not possible to separate complex and non linear data, to solve this problem support vector machine uses kernel trick which transforms lower dimensional space to higher dimensional space.

Logistic Regresssion Algorithm

Logistic regression is an example of supervised learning. It *is* used to calculate or predict the probability of a binary (yes/no) event occurring. An example of logistic regression could be applying machine learning to determine if a person is likely to be infected with COVID-19 or not. Since we have two possible outcomes to this question - yes they are infected, or no they are not infected - this is called *binary* classification.

CHAPTER8

TESTING

8.1 Test Cases

In machine learning systems, however, data and desired behavior are the inputs and the models learn the logic as the outcome of the training and optimization processes. In this case, testing involves validating the consistency of the model's logic and our desired behavior.

We usually write two different classes of tests for Machine Learning systems:

- Pre-train tests
- Post-train tests

Pre-train tests: The intention is to write such tests which can be run without trained parameters so that we can catch implementation errors early on. This helps in avoiding the extra time and effort spent in a wasted training job.

We can test the following in the pre-train test:

- the model predicted output shape is proper or not
- test dataset leakage i.e. checking whether the data in training and testing datasets have no duplication
- temporal data leakage which involves checking whether the dependencies between training and test data do not lead to unrealistic situations in the time domain like training on a future data point and testing on a past data point
- check for the output ranges. In the cases where we are predicting outputs in a certain range (for example when predicting probabilities), we need to ensure the final prediction is not outside the expected range of values.
- Ensuring a gradient step training on a batch of data leads to a decrease in the loss ➢ data profiling assertions

Post-train tests: Post-train tests are aimed at testing the model's behavior. We want to test the learned logic and it could be tested on the following points and more:

- invariance tests which involve testing the model by tweaking only one feature in a data point and checking for consistency in model predictions. For example, if we are working with a loan prediction dataset then change in sex should not affect an individual's eligibility for the loan given all other features are the same or in the case of titanic survivor probability prediction data, change in the passenger's name should not affect their chances of survival.
- Directional expectations wherein we test for a direct relation between feature values and predictions. For example, in the case of a loan prediction problem, having a higher credit score should definitely increase a person's eligibility for a loan.
- Apart from this, you can also write tests for any other failure modes identified for your model.
- Now, let's try a hands-on approach and write tests for the Medical Cost Personal Datasets. Here, we are given a bunch of features and we have to predict the insurance costs

8.2 User Acceptance Testing

Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	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

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	5	0	0	5-
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

CHAPTER9

RESULTS

9.1 Performance Metrics

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

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

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

Thus, accuracy for all the four used models were calculated and ranked. XGBoost performed betterthan other models.

Algorithm	Training Accuracy	Testing Accuracy
Decision Tree	1.0	0.95
Logistic regression	0.92	0.91
RandomForestClassifier	1.0	0.97
GaussionNB	0.887	0.8801

CHAPTER10

ADVANTAGES & DISADVANTAGES

ADVANTAGES

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

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

DISADVANTAGES:

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

CHAPTER11

CONCLUSION

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

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

CHAPTER12

FUTURE SCOPE

In future we intend to build an add-ons for our system and if we get a structured dataset of phishing, we can perform phishing detection much faster than any other technique. We can also use a combination of any two or more classifiers to get maximum accuracy. We plan to explore various phishing techniques which use Network based features, Content based features, Webpage based features and HTML and JavaScript features of web pages which will improve the performance of the system. In particular, we extract features from URLs and pass it through the various classifiers.

CHAPTER13

APPENDIX

13.1 Source Code

App.py

```
import numpy as np

import flask

from flask import Flask, render_template, request, jsonify

import pickle

import inputScript

from werkzeug.exceptions import HTTPException

app = Flask(__name__)

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

@app.route('/predict')

def predict():

   return flask.render_template('final.html')

@app.route('/y_predict', methods = ['POST'])

def y_predict():

   url = request.form['URL']

   checkprediction = inputScript.main(url)

   prediction = model.predict(checkprediction)

   output = prediction[0]

   if(output == 1):

      pred = "You are safe !!  This is a Legitimate Website."
   else:
       pred = "You are on the wrong site. Be caustion!"
   return render_template('final.html',prediction_text='{}'.format(pred),url=url)
@app.route('/predict_api',methods=['POST','GET'])
def predict_api():
   data = request.get_json()
   prediction = model.y_predict([np.array(list(data.values()))])
   output = prediction[0]
   return jsonify(output)
@app.errorhandler(HTTPException)
def handleError(err):
   return render_template("message.html",title=err.name, message=err.description),err.code
if __name__ == '__main__':
   app.run('0.0.0.0',debug=True)
```

inputScript.py

```
import ipaddress
import re
import socket
import time
import urllib.request
from datetime import date, datetime
from tracemalloc import DomainFilter
from urllib.parse import urlencode, urlparse

from xml.dom.xmlbuilder import DOMInputSource

import regex

import fast_pagerank

import requests

import urllib3

import whois

from bs4 import BeautifulSoup

from dateutil.parser import parse as date_parse

from googlesearch import search


 # 1.UsingIp


def UsingIp(url):


    try:


        ipaddress.ip_address(url)


        return 2


    except:
        return 1

# 2.longUrl

def longUrl(url):

    if len(url) < 54:

        return 1

    if len(url) >= 54 and len(url) <= 75:

        return 0

    return 2

# 3.shortUrl

def shortUrl(url):
    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', url)

    if match:

        return 2

    return 1
```

```
  # 4.Symbol@

def symbol(url):

    if "@" in url:

      return 2

    return 1



  # 5.Redirecting//

def redirecting(url):

    if url.rfind('//')>6:

      return 2

    return 1
  # 6.prefixSuffix

def prefixSuffix(url):

    if '-' in urlparse(url).netloc:
      return 2
    else:
      return 1



  # 7.SubDomains

def SubDomains(url):

    dot_count = len(re.findall("\.", url))

    if dot_count == 1:

      return 2
```

```
        elif dot_count == 2:
            return 0
        return 1


    # 8.HTTPS
def Hppts(url):
        try:
            https = url.urlparse.scheme
            if 'https' in https:
                return 1
            return 2
        except:
```

```
      return 2

  # 9.DomainRegLen

def DomainRegLen(url):

    try:

      expiration_date = url.whois_response.expiration_date

      creation_date = url.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 2

    except:
      return 2


  # 10. Favicon

def Favicon(url):

    try:

      for head in url.soup.find_all('head'):

        for head.link in url.soup.find_all('link', href=True):

          dots = [x.start(0) for x in re.finditer('\.', head.link['href'])]

          if url.url in head.link['href'] or len(dots) == 1 or DOMInputSource in head.link['href']:

            return 1

      return 2
```

```
        except:
            return 2



    # 11. NonStdPort
def NonStdPort(url):
        try:
            port = url.domain.split(":")
            if len(port)>1:
                return 2
            return 1
        except:
            return 2
```

```
  # 12. HTTPSDomainURL

def HTTPSDomainURL(url):

    try:

      if 'https' in url.domain:

        return 1

      return 2

    except:

      return 2



  # 13. RequestURL
```

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


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

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

      for iframe in url.soup.find_all('iframe', src=True):
        dots = [x.start(0) for x in re.finditer('\.', iframe['src'])]
        if url.url in iframe['src'] or url.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 2

      except:
```

```
            return 0

        except:

            return 2




    # 14. AnchorURL

def AnchorURL(url):

        try:

            i,unsafe = 0,0

            for a in url.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 url.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 2
except:
    return 2
```

```
        except:

            return 2



    # 15. LinksInScriptTags

def LinksInScriptTags(url):

        try:

            i,success = 0,0



            for link in url.soup.find_all('link', href=True):

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

for script in url.soup.find_all('script', src=True):
    dots = [x.start(0) for x in re.finditer('\.', script['src'])]
    if url.url in script['src'] or url.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 2
    except:
        return 0
except:
    return 2
```

```
  # 16. ServerFormHandler

def ServerFormHandler(url):

    try:

      if len(url.soup.find_all('form', action=True))==0:

        return 1

      else :

        for form in url.soup.find_all('form', action=True):

          if form['action'] == "" or form['action'] == "about:blank":

            return 2

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

            return 0

          else:
            return 1
    except:
      return 2

  # 17. InfoEmail
def InfoEmail(url):
    try:
      if re.findall(r"[mail\(\)|mailto:?]", url.soap):
        return 1
      else:
        return 2
        except:
            return 2


    # 18. AbnormalURL
def AbnormalURL(url):
        try:
            if url.response.text == url.whois_response:
                return 1
            else:
                return 2
        except:
            return 2
    # 19. WebsiteForwarding

def WebsiteForwarding(url):

        try:

            if len(url.response.history) <= 1:

                return 1

            elif len(url.response.history) <= 4:

                return 0

            else:

                return 2

        except:
```

```
      return 2

  # 20. StatusBarCust

def StatusBarCust(url):

    try:

      if re.findall("<script>.+onmouseover.+</script>", url.response.text):

        return 1

      else:

        return 2

    except:

      return 2
    # 21. DisableRightClick

def DisableRightClick(url):

        try:

            if re.findall(r"event.button ?== ?2", url.response.text):

                return 1

            else:

                return 2

        except:

             return 2




    # 22. UsingPopupWindow

def UsingPopupWindow(response):
        try:
            if re.findall(r"alert\(", response.text):
                return 1
            else:
                return 2
        except:
             return 2

    # 23. IframeRedirection
def IframeRedirection(url):
        try:
```

```
      if re.findall(r"[<iframe>|<frameBorder>]", url.response.text):

        return 1

      else:

        return 2

    except:

       return 2



  # 24. AgeofDomain

def AgeofDomain(url):

    try:

      creation_date = url.whois_response.creation_date

      try:
        if(len(creation_date)):
            creation_date = creation_date[0]
    except:
        pass

    today  = date.today()
    age = (today.year-creation_date.year)*12+(today.month-creation_date.month)
    if age >=6:
        return 2
    return 1
except:
      return 2


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

      age = (today.year-creation_date.year)*12+(today.month-creation_date.month)

      if age >=6:

        return 2

      return 1

    except:

      return 2




  # 26. WebsiteTraffic

def WebsiteTraffic(url):

    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 2

            return 1

        except :

            return 2



    # 27. PageRank

def PageRank(url):

        try:

            prank_checker_response = requests.post("https://www.checkpagerank.net/index.php", {"name":
url.domain})
```

```
            global_rank = int(re.findall(r"Global Rank: ([0-9]+)", prank_checker_response.text)[0])

            if global_rank > 0 and global_rank < 100000:

                return 1

            return 2

        except:

            return 2




    # 28. GoogleIndex

def GoogleIndex(url):

        try:
            site = search(url, 5)

            if site:

                return 1

            else:

                return 2

        except:

            return 2




    # 29. LinksPointingToPage

def LinksPointingToPage(url):

        try:
```

```
      number_of_links = len(re.findall(r"<a href=", url.response.text))

      if number_of_links == 0:

        return 1

      elif number_of_links <= 2:

        return 0

      else:

        return 2

    except:

      return 2

# 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', urlencode)

        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\.1
58|181\.174\.165\.13|46\.242\.145\.103|121\.50\.168\.40|83\.125\.22\.219|46\.242\.145\.98|'

'107\.151\.148\.44|107\.151\.148\.107|64\.70\.19\.203|199\.184\.144\.27|107\.151\.148\.108|107\.1
51\.148\.109|119\.28\.52\.61|54\.83\.43\.69|52\.69\.166\.231|216\.58\.192\.225|'

'118\.184\.25\.86|67\.208\.74\.71|23\.253\.126\.58|104\.239\.157\.210|175\.126\.123\.219|141\.8\.
224\.221|10\.10\.10\.10|43\.229\.108\.32|103\.232\.215\.140|69\.172\.201\.153|'

'216\.218\.185\.162|54\.225\.104\.146|103\.243\.24\.98|199\.59\.243\.120|31\.170\.160\.61|213\.19
\.128\.77|62\.113\.226\.131|208\.100\.26\.234|195\.16\.127\.102|195\.16\.127\.157|'

'34\.196\.13\.28|103\.224\.212\.222|172\.217\.4\.225|54\.72\.9\.51|192\.64\.147\.141|198\.200\.56\
.183|23\.253\.164\.103|52\.48\.191\.26|52\.214\.197\.72|87\.98\.255\.18|209\.99\.17\.27|'

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

```
            if url_match:

                return 1

            elif ip_match:

                return 1

            return 2

        except:

            return 2

def Result(url):
    return 0



def main(url):
  check = [[UsingIp(url),longUrl(url), shortUrl(url),symbol(url),redirecting(url), prefixSuffix(url),
            SubDomains(url),Hppts(url), DomainRegLen(url),Favicon(url),NonStdPort(url),
HTTPSDomainURL(url),
            RequestURL(url), AnchorURL(url), LinksInScriptTags(url),ServerFormHandler(url),InfoEmail(url),
            AbnormalURL(url), WebsiteForwarding(url),StatusBarCust(url),DisableRightClick(url),
UsingPopupWindow(url),
            IframeRedirection(url), AgeofDomain(url), DNSRecording(url),WebsiteTraffic(url),PageRank(url),
            GoogleIndex(url),LinksPointingToPage(url),StatsReport(url),Result(url)]]
 return check
```

message.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>{{title}}</title>
  </head>
  <body>
    <h2>{{title}}</h2>
    <h3>{{message}}</h3>
    <p><a href="{{url_for('predict')}}">Click here</a>go to home page</p>
</html>
```

Final.html

```
<!DOCTYPE html>
<html>
```

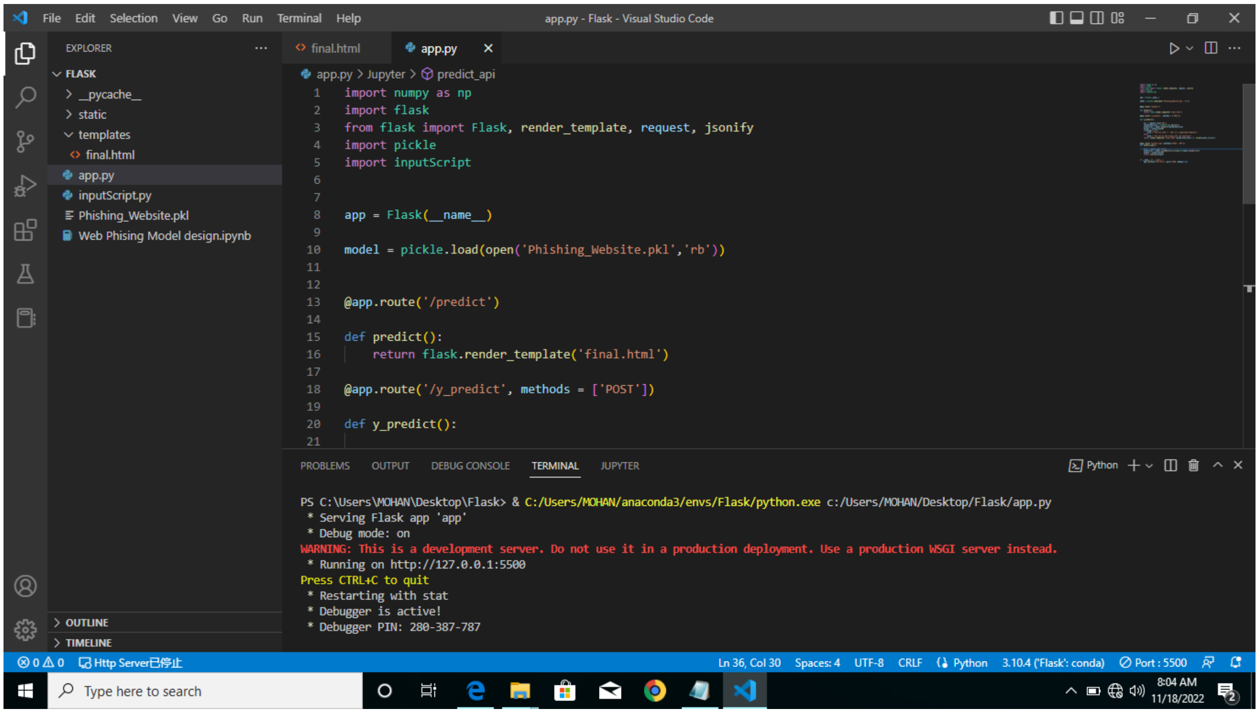
```
   <head>
      <title>Web Phishing Detection</title>
      <style>
*{
   margin:0;
   padding:0;
}

body{
   background-color: rgb(140, 232, 255);
   background-size: 280%;
   background-position: -400px 0px;
}
div.main{
   width:400px;
   margin: 100px auto 0px auto;

}

h2{
   text-align: centre;
   padding: 20px;
   font-family: sans-serif;
```

```
}

div.prediction{

   background-color: rgba(8, 8, 8, 0.5);

   width: 100%;

   font-size: 18px;

   border-radius: 10px;

   border: 1px solid rgba(38, 222, 255, 0.479);

   box-shadow: 2px 2px 15px rgba(0, 0, 0, 0.3);

   color:  rgb(140, 232, 255);

}

form{

   margin: 40px;

}

label{

   font-family: sans-serif;

   font-size: 18px;

   font-style: italic;

}

input#url{

   width: 300px;

   border: 1px solid rgb(14, 5, 5);

   border-radius: 3px;

   outline: 0;
```

```
    padding:7px;

    background-color: #fff;

    box-shadow: insert 1px 1px 5px rgb(0,0,0,0.3);

}

input#submit{

    width: 200px;

    padding: 7px;

    font-size: 16px;

    font-family: sans-serif;

    font-weight: 600;

    border-radius: 3px;

    background-color: rgba(33, 214, 48, 0.979);

    color:rgb(31, 14, 14);

    cursor:pointer;

    border:1px solid rgba(3, 61, 17, 0.719);

    box-shadow: 1px 1px 5px rgb(0,0,0,0.3);

    text-shadow: 1px 1px 5px rgb(0,0,0,0.3);

}

.p{



    text-align: centre;

    padding: 20px;

    font-family: sans-serif;
```

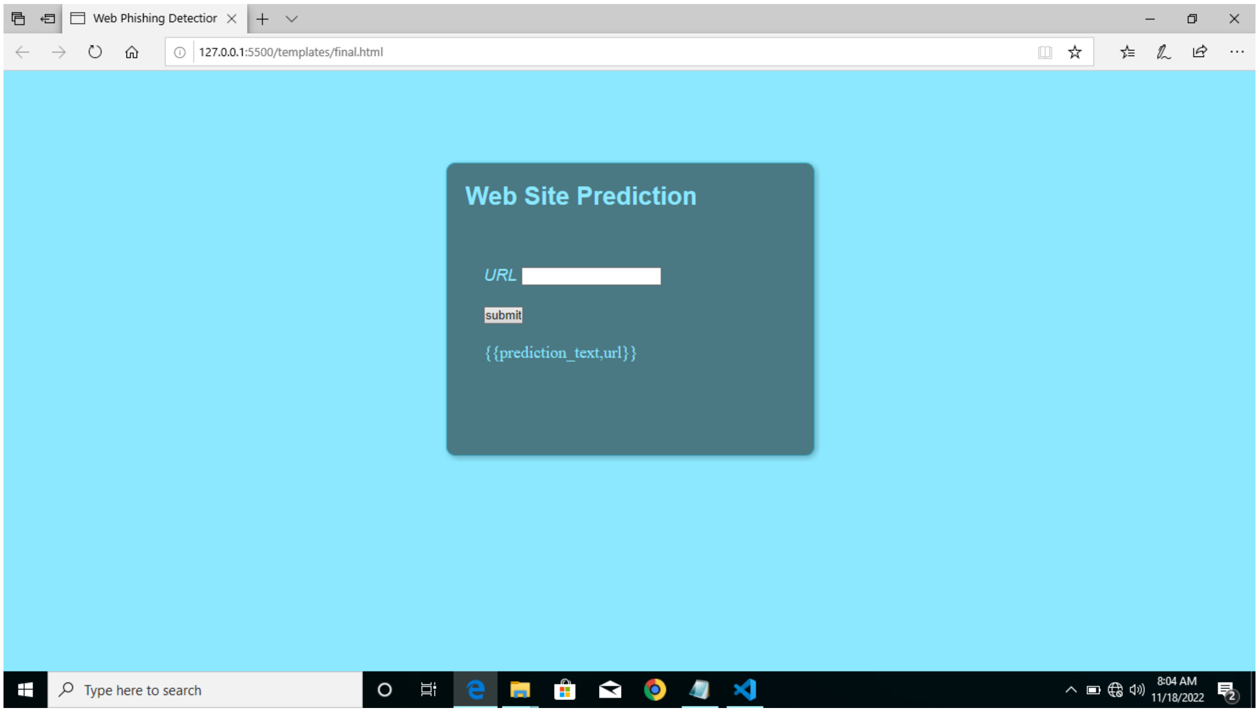
```
   margin-bottom: 40px ;

}

           </style>

   </head>

   <body>

         <div class="main">

         <div class="prediction">

             <h2>Web Site Prediction</h2>

             <form  action="{{url_for('y_predict')}}" method="POST">

                <label for =URL >URL</label>

                <input type="text" name="URL" id="">

                <br>

                <br>

                <input type="submit" value="submit">

                <br><br>

                <p>{{prediction_text,url}}</p>

                          <br><br><br>

                          </form>

         </div>

      </div>

   </body>

</html>
```

OUTPUT SNAP



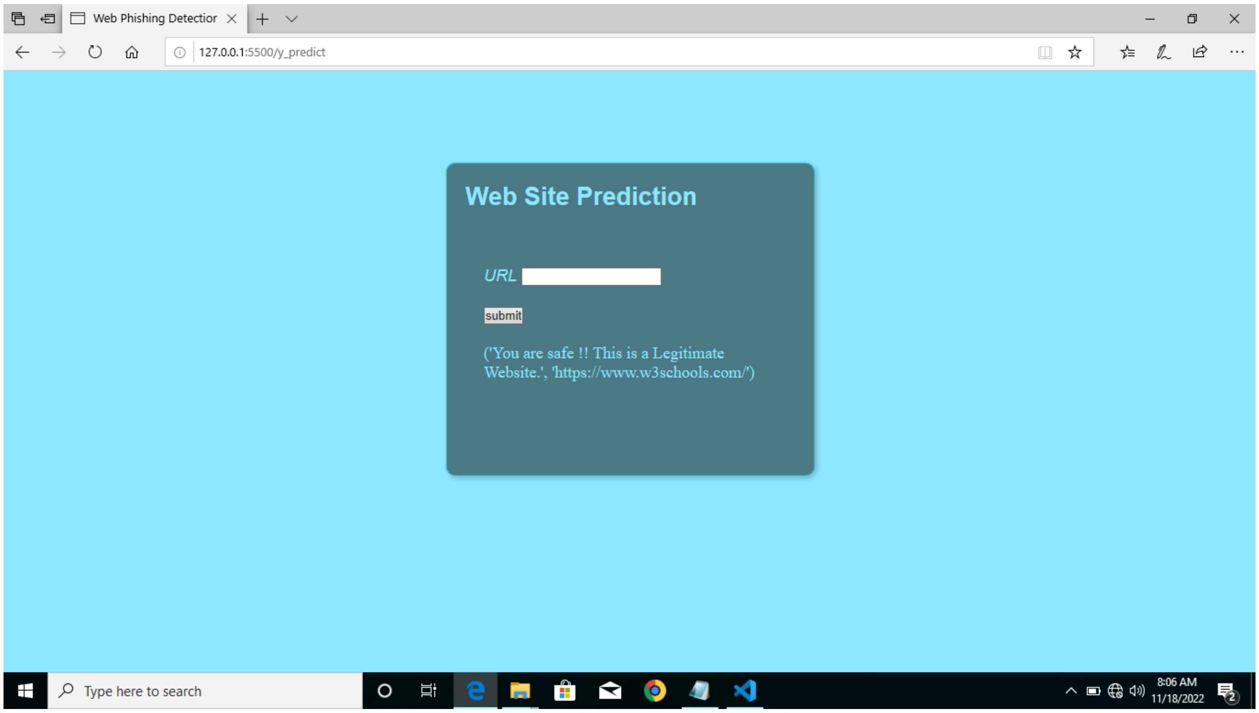
```python
import numpy as np
import flask
from flask import Flask, render_template, request, jsonify
import pickle
import inputScript


app = Flask(__name__)

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


@app.route('/predict')

def predict():
    return flask.render_template('final.html')

@app.route('/y_predict', methods = ['POST'])

def y_predict():
    
```

```
PS C:\Users\MOHAN\Desktop\Flask> & C:/Users/MOHAN/anaconda3/envs/Flask/python.exe c:/Users/MOHAN/Desktop/Flask/app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5500
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 280-387-787
```



Web Site Prediction

{{prediction_text,url}}



Web Site Prediction

('You are safe !! This is a Legitimate Website.', 'https://www.w3schools.com/')

13.2 GitHub & Project Demo Link

GitHub Link:- https://github.com/IBM-EPBL/IBM-Project-52201-1660991191

DemoLink:-
https://drive.google.com/file/d/1nD34KW6YySUOeW096kJRYhrAtyQhLfOw/view?usp=drivesdk