PROJECT REPORT

WEB PHISHING DETECTION

TEAM ID: PNT2022TMID24896

Submitted By:

SHARON ROSELINE S – 210419205043
SRUTHI S – 210419205048
SOWMIKA D – 210419205047
ANUMESH R – 210419205004

Under the guidance of:
Prof. DR A R Kavitha

Industry Mentor:
Sandesh P

Department of
INFORMATION TECHNOLGY

CHENNAI INSTITUTE OF TECHNOLOGY

CHENNAI – 600069

NOVEMBER 2022

TABLE OF CONTENT

CHAPTER 1

1. INTRODUCTION

1.1 Project Overview

Nowadays 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. Phishing emails are a routine occurrence for anyone with email in the Internet age. 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. In 3rd Microsoft Computing Safer Index Report released in February 2014, it was estimated that the annual worldwide impact of phishing could be as high as $5 billion. 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 analyse various blacklisted and legitimate URLs and their features to accurately detect the phishing websites including zero- hour phishing websites.

1.2 Purpose

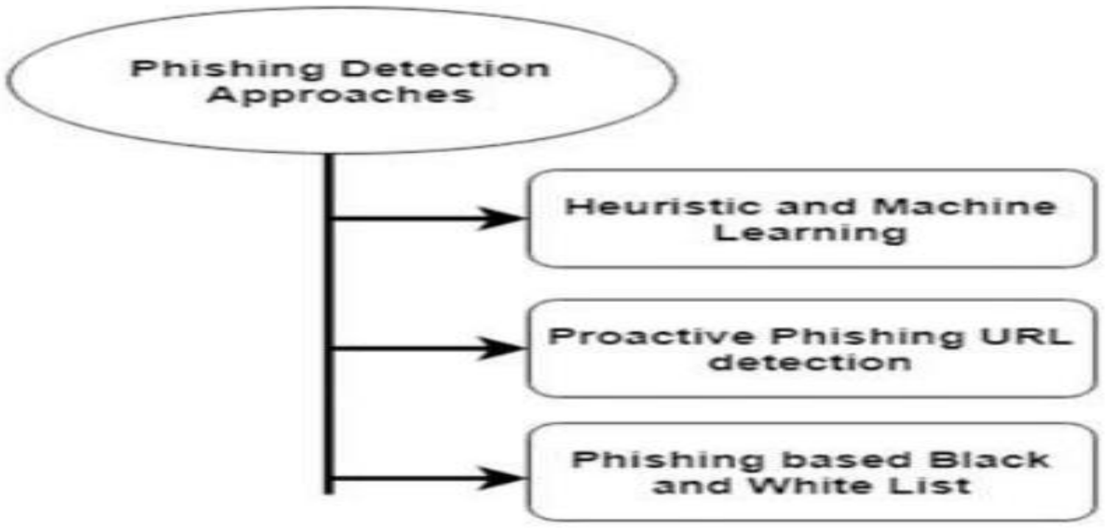
The main purpose of the project is to detect the fake or phishing websites who are trying to get access to the sensitive data or by creating the fake websites and trying to get access of the user personal credentials. We are using machine learning algorithms to safeguard the sensitive data and to detect the phishing websites who are trying to gain access on sensitive data. It helps to train them in URL safety by presenting this information at critical points of intervention. The primary motivation behind this project is the lack of tools which purposefully prevent users from visiting malicious URLs and more crucially inform them of why they have been prevented. Machine learning techniques are very successful in matching established patterns of URLs but not as successful at identifying new URL variations, which means malicious URLs can go sometime before being detected. The lack of user knowledge of phishing also encourages users to ignore warnings when presented to them. In the UK for example, only 72% of technology users had heard of phishing as a term despite 95% of organizations saying that theytrain end users. For these reasons, it is important to train users to detected phishing themselves.

CHAPTER 2

2. LITERATURE SURVEY

2.1 Existing Problem

Phishing detection schemes which detect phishing on the server side are better than phishing prevention strategies and user training systems. These systems can be used either via a web browser on the client or through specific host-site software presents the classification of Phishing detection approaches. Heuristic and ML based approach is based on supervised and unsupervised learning techniques. It requires features or labels for learning an environment to make a prediction. Proactive phishing URL detection is similar to ML approach. However, URLs are processed and support a system to predict a URL as a legitimate or malicious. Blacklist and Whitelist approaches are the traditional methods to identify the phishing sites. The exponential growth of web domains reduces the performance of the traditional method. The existing methods rely on new internet users to a minimum. Once they identify phishing website, the site is not accessible, or the user is informed of the probability that the website is not genuine. This approach requires minimum user training and requires no modifications to existing website authentication systems.


Phishing Detection Approaches
Heuristic and Machine Learning
Proactive Phishing URL detection
Phishing based Black and White List

2.2 References

1. Detecting Phishing Websites Using Machine Learning by Sagar Patil,Yogesh Shetye, Nilesh Shendage published in the year 2020.

2. Phishing Website Classification and Detection Using Machine Learning by Jitendra Kumar, A. Santhanavijayan, B. Janet, Balaji Rajendran, B.S. Bindhumadhava was published in the year 2020.

3. Adebowale MA, Lwin KT, Hossain MA (2020) Intelligent phishing detection scheme using deep learning algorithms. J Enterp Inf Manag. https://doi.org/10.1108/JEIM-01-2020-0036

4. Alom MZ, Taha TM (2017) Network intrusion detection for cyber security using unsupervised deep learning approaches. In: 2017 IEEE national aerospace and electronics conference (NAECON). IEEE, pp 63–69

5. Benavides E, Fuertes W, Sanchez S, Sanchez M (2020) Classification of phishing attack solutions by employing deep learning techniques: a systematic literature review. In: Developments and advances in defense and security. Springer, Singapore, pp 51–64

2.3 Problem Statement Definition

Phishing is a major problem, which uses both social engineering and technical deception to get users' important information such as financial data, emails, and other private information. Phishing exploits human vulnerabilities; therefore, most protection protocols cannot prevent the whole phishing attacks.

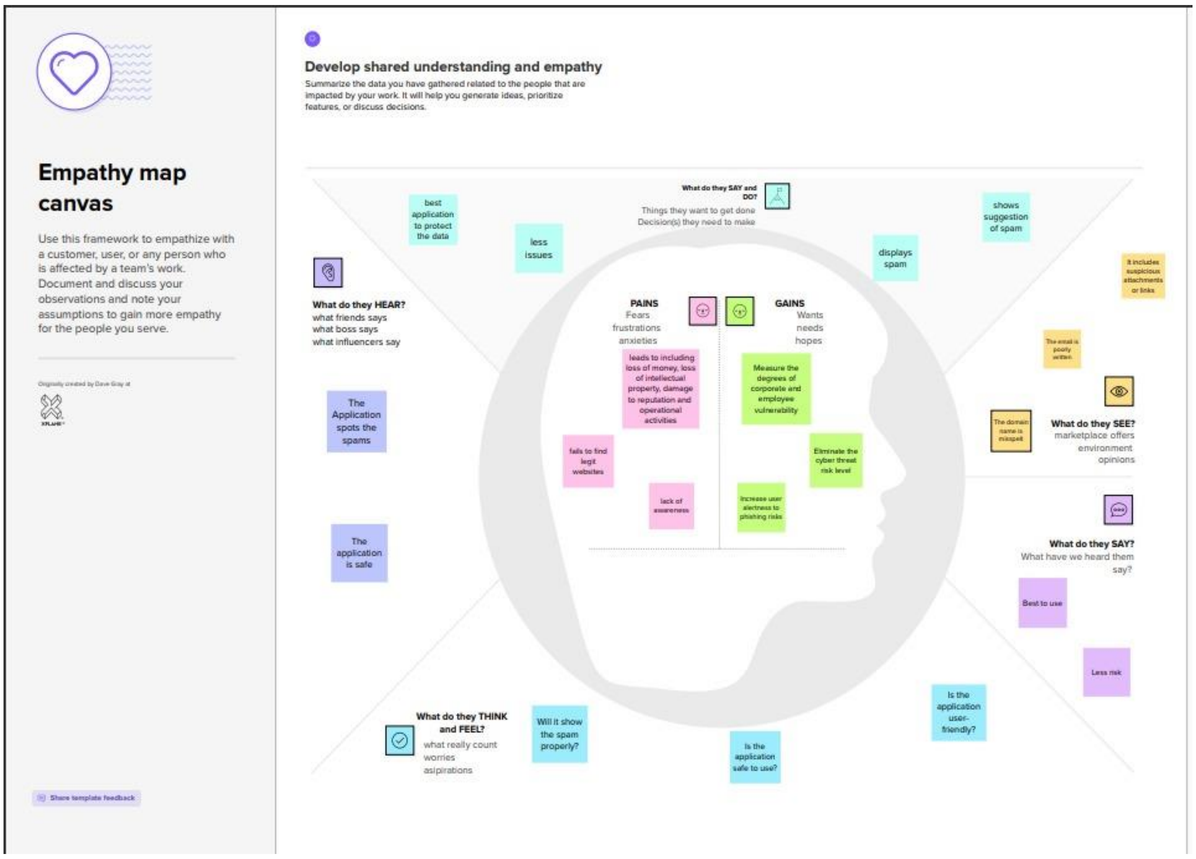
Phishing sites are malicious websites that imitate as legitimate websites or web pages and aim to steal user’s personal credentials like user id, password, and financial information. Spotting these phishing websites is typically a challenging task because phishing is mainly a semantics-based attack, that mainly focus on human vulnerabilities, not the network or software vulnerabilities. Phishing can be elaborated as the process of charming users in order to gain their personal credentials like user-id’s and passwords. In this paper, we come up with an intelligent system that can spot the phishing sites. This intelligent system is based on a machine learning model. Our aim through this paper is to stalk a better performance classifier by examining the features of the phishing site and choose appropriate combination of systems for the training of the classifier. Phishing is a non-ethical method comprising both social engineering and technical tricks to capture user’s information and sensitive credentials like financial credentials. Some of the social engineering techniques use spam mails, pretending as a legitimate company or organization, that are specially designed to forefront users to knock- off websites that moreover recipients to fall into the trap which steal financial credentials like user-ids and passwords.

CHAPTER 3

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

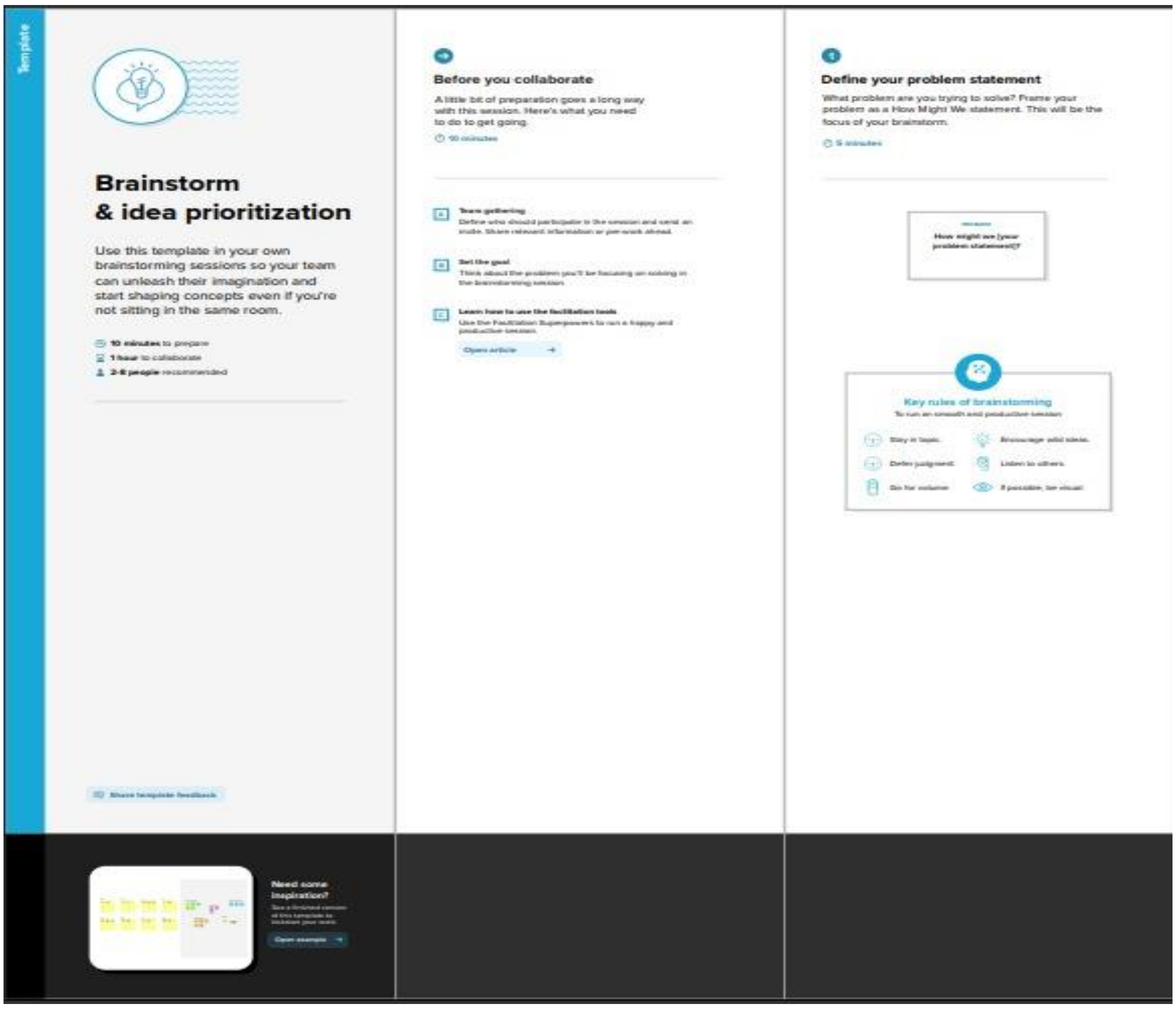
An empathy map is a collaborative visualization used to articulate what we know about a particular type of user. It externalizes knowledge about users in order to 1) create a shared understanding of user needs, and 2) aid in decision making. Traditional empathy maps are split into 4 quadrants (Says, Thinks, Does, and Feels), with the user or persona in the middle. Empathy maps provide a glance into who a user is as a whole and are not chronological or sequential. Empathy maps should be used throughout any UX process to establish common ground among team members and to understand and prioritize user needs. In user-centred design, empathy maps are best used from the very beginning of the design process.



Empathy map canvas
Use this framework to empathize with a customer, user, or any person who is affected by a team's work. Document and discuss your observations and note your assumptions to gain more empathy for the people you serve.
Originally created by Dave Gray at
XPLANE
Share template feedback
Develop shared understanding and empathy
Summarize the data you have gathered related to the people that are impacted by your work. It will help you generate ideas, prioritize features, or discuss decisions.
What do they SAY and DO?
Things they want to get done
Decision(s) they need to make
best application to protect the data
less issues
displays spam
shows suggestion of spam
It includes suspicious attachments or links
What do they HEAR?
what friends says
what boss says
what influencers say
The Application spots the spams
The application is safe
PAINS
Fears
frustrations
anxieties
leads to including loss of money, loss of intellectual property, damage to reputation and operational activities
fails to find legit websites
lack of awareness
GAINS
Wants
needs
hopes
Measure the degrees of corporate and employee vulnerability
Eliminate the cyber threat risk level
Increase user alertness to phishing risks
The email is poorly written
The domain name is misspelt
What do they SEE?
marketplace offers
environment
opinions
What do they SAY?
What have we heard them say?
Best to use
Less risk
What do they THINK and FEEL?
what really count
worries
asipirations
Will it show the spam properly?
Is the application safe to use?
Is the application user-friendly?

3.2 Ideation & Brainstorming

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. Brainstorming is usually conducted by getting a group of people together to come up with either general new ideas or ideas for solving a specific problem or dealing with a specific situation. For example, a major corporation that recently learned it is the object of a major lawsuit may want to gather together top executives for a brainstorming session on how to publicly respond to the lawsuit being filed.



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 →

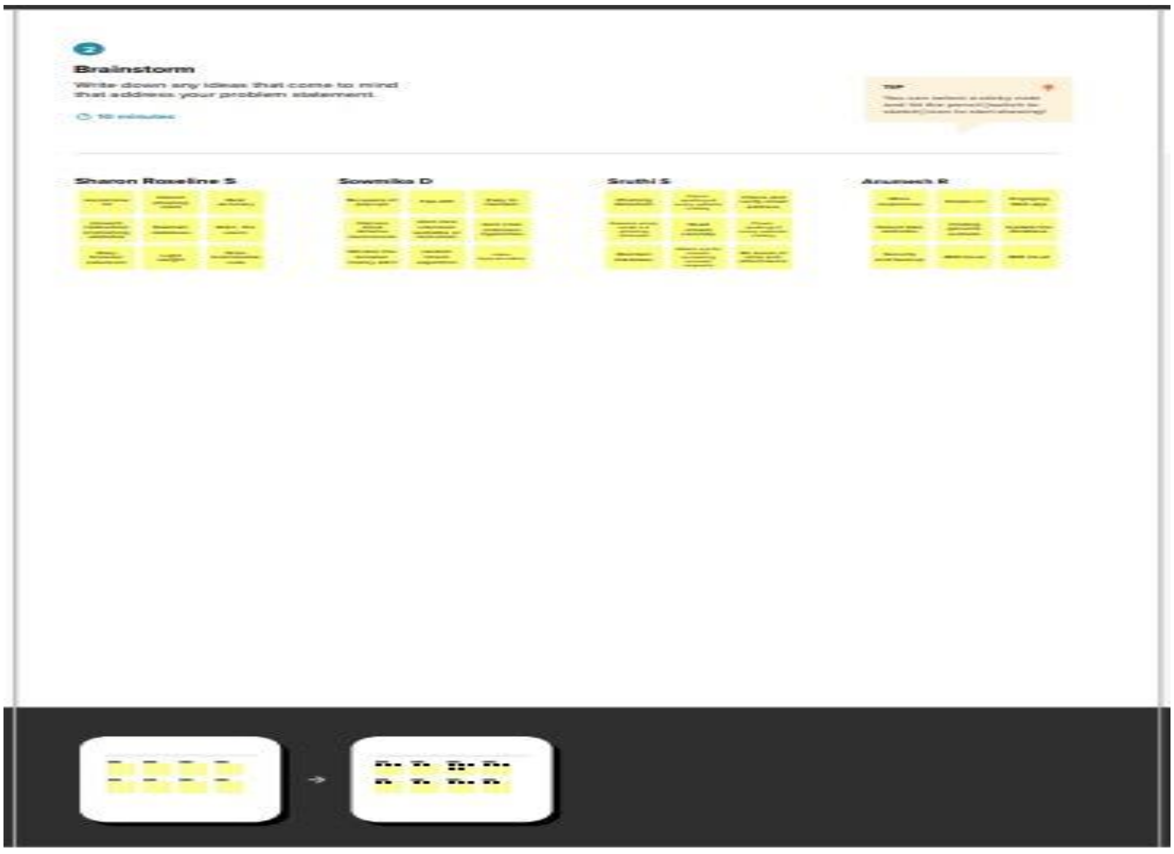
1
Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
5 minutes

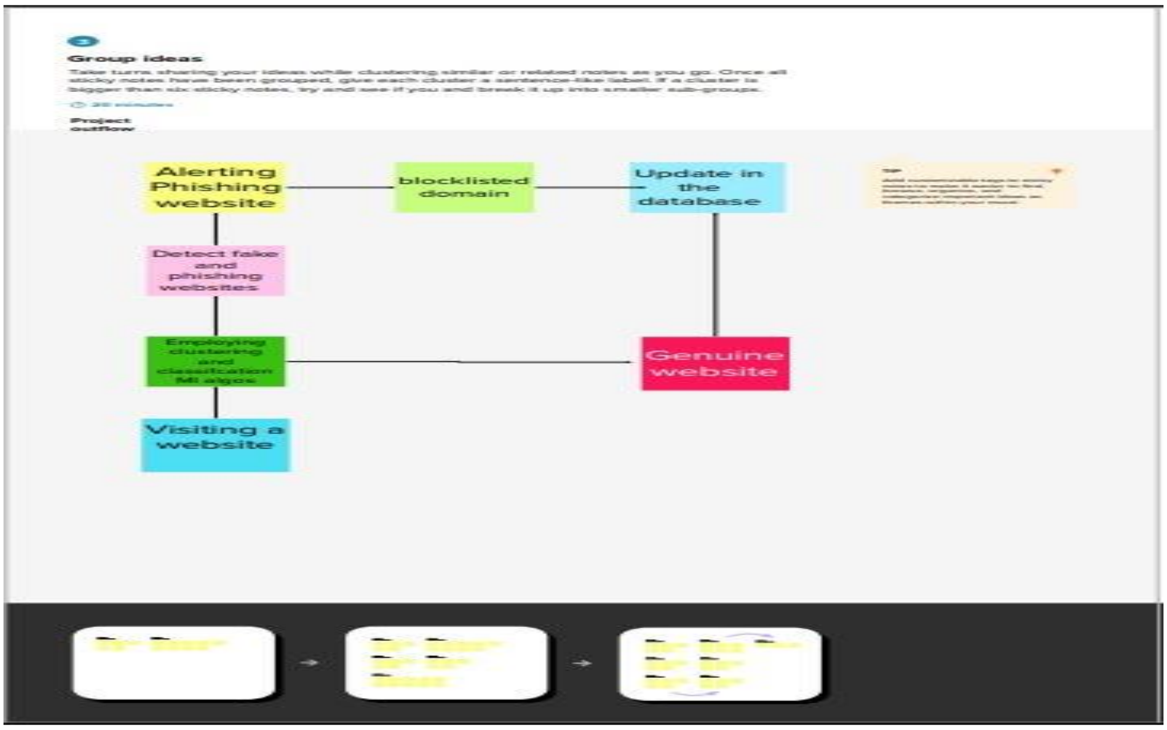
How might we [your problem statement]?

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.

The lines between ideation and brainstorming have become a bit more blurred with the development of several brainstorming software programs, such as Bright idea and Idea wake. These software programs are designed to encourage employees of companies to generate new ideas for improving the companies' operations and, ultimately, bottom-line profitability. The programs often combine the processes of ideation and brainstorming in that individual employee can use them, but companies may simulate brainstorming sessions by having several employees all utilize the software to generate new ideas intended to address a specific purpose.


2
Brainstorm
Write down any ideas that come to mind that address your problem statement.
10 minutes
TIP
Sharon Roseline S
Sowmika D
Sruthi S
Arumesh R


3

Group ideas

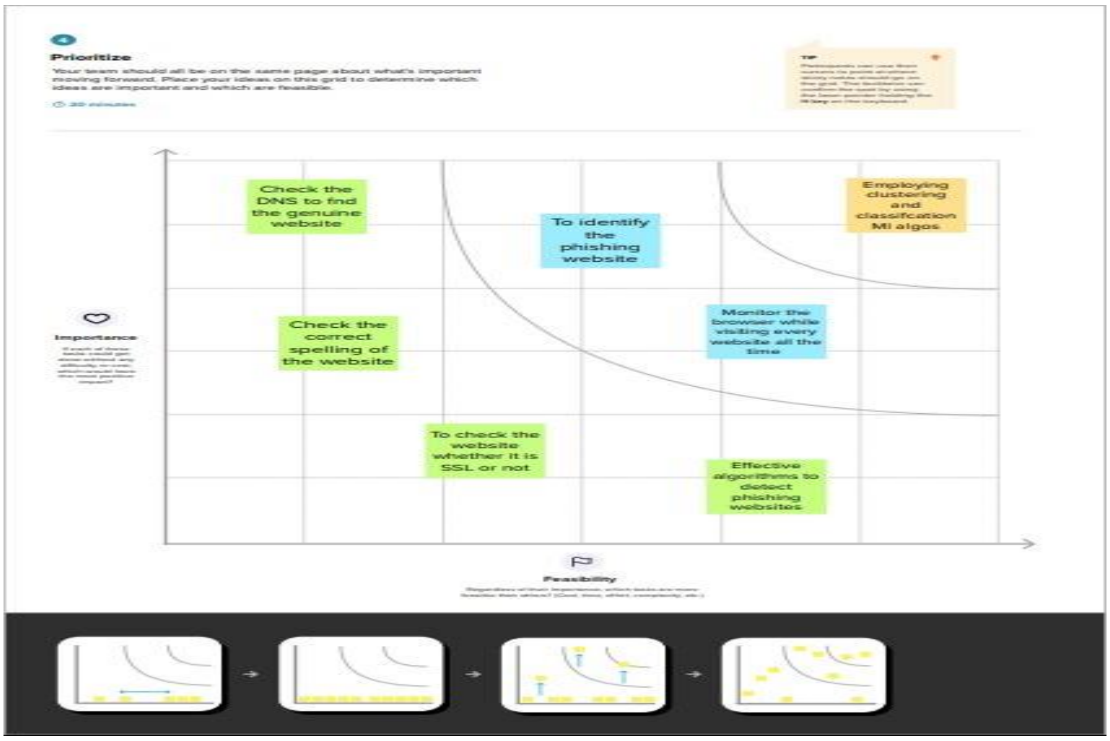
Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

Project outflow

Alerting Phishing website
blocklisted domain
Update in the database
Detect fake and phishing websites
Employing clustering and classifcation Ml algos
Genuine website
Visiting a website

Participants in a brainstorming session are encouraged to freely toss out whatever ideas may occur to them. The thinking is that by generating a large number of ideas, the brainstorming group is likely to come up with a suitable solution for whatever issue they are addressing


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

Check the DNS to find the genuine website
To identify the phishing website
Employing clustering and classifcation Ml algos
Importance
Check the correct spelling of the website
Monitor the browser while visiting every website all the time
To check the website whether it is SSL or not
Effective algorithms to detect phishing websites
Feasibility

3.3 Proposed Solution

Having hooked your audience into the problem, now you want to paint a picture of what the world will be like when you solve the problem. Your proposed solution should relate the current situation to a desired result and describe the benefits that will accrue when the desired result is achieved. So, begin your proposed solution by briefly describing this desired result.

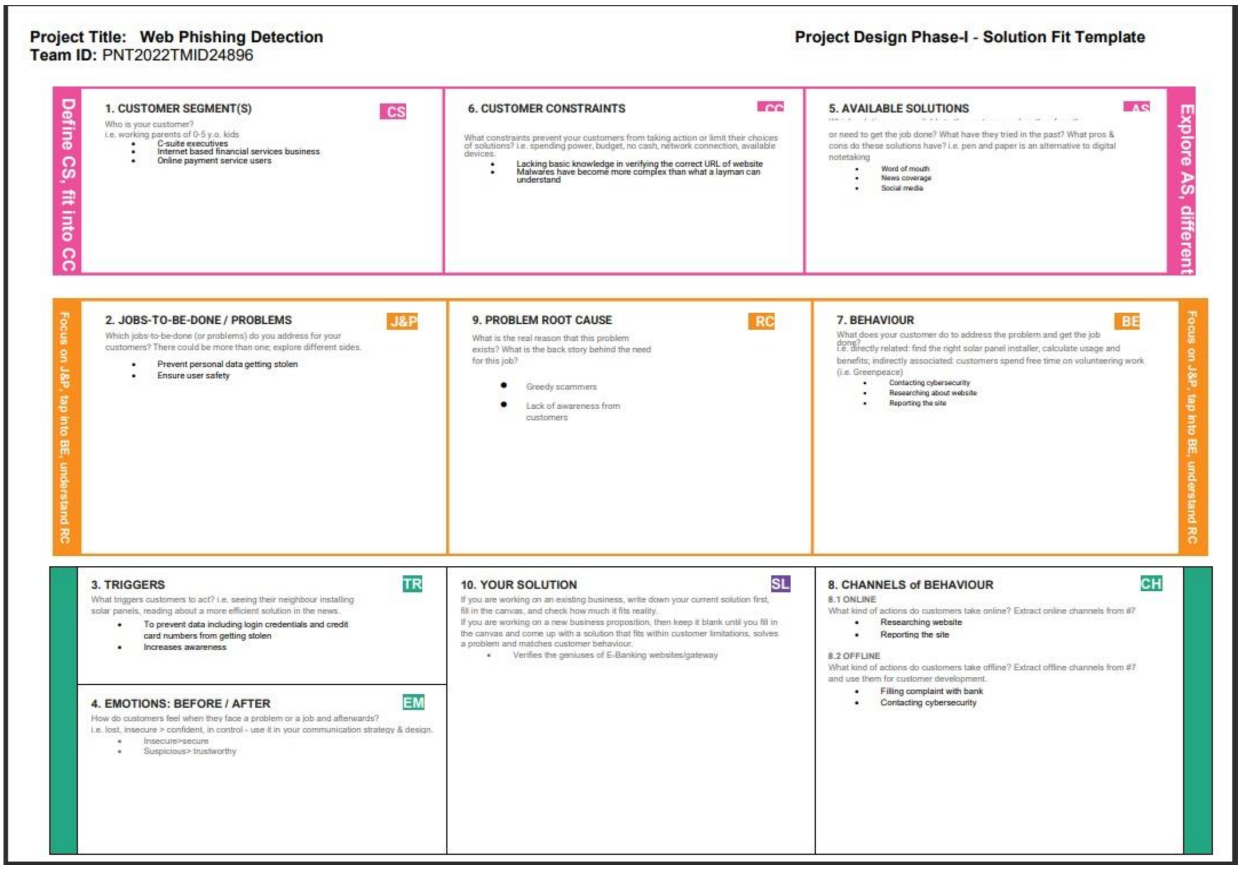
	PARAMETER	DESCRIPTION
1	Problem Statement	• Web phishing is one of the major problems which handles with sensitive information. • Malicious link will often steal users' credentials without their consent which must be solved • The main objective is to identify phishing e-payment website and safeguard user information from phishing to protect users' privacy
2	Idea / solution description	• Our proposed model is capable of detecting phishing websites by use of Machine learning & classification algo's • As we came to know while doing literature survey, efficient models like decision tree, random forest could also be tried out. • use of pre-defined blacklisted website dataset
3	Novelty / Uniqueness	• As we are providing a service (SAAS) which doesn't need any kind of computational resources. • The specialized feature is that we provide users to enable our project as a

		Chrome extension with user-friendly UI/UX which gives them a higher level of confidence while doing transactions or web surfing. • Our model is designed in such a way which gives alerts while entering into phishing websites.
4	Social Impact	• Our project will have a definite impact on society by making users free from data theft. • secure users from proxies and scams • Using our product people can feel safer and secure from the cyber-attack like web phishing
5	Business Model	• We are providing the product as Software as a service model, so that users can easily use the product without any difficulties. • Even though our product is not designed to generate revenue directly, it helps indirectly for e-commerce & banking institutions by providing trustiness over the products which further increase the intakes of services.
6	Scalability of the solution	• Apart from E-banking and e-commerce sector the idea proposed can be developed into platform independent model also. • Machine Learning models and effective feature engineering techniques helps identify phishing websites

		and come up with key features that are common in most phishing websites.

3.4 Problem Solution fit

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem.



Project Title: Web Phishing Detection
Team ID: PNT2022TMID24896

Project Design Phase-I - Solution Fit Template

Define CS, fit into CC
Explore AS, different
Focus on J&P, tap into BE, understand RC
Focus on J&P, tap into BE, understand RC
CS
CC
AS
J&P
RC
BE
TR
SL
CH
EM

1. CUSTOMER SEGMENT(S)
Who is your customer?
i.e. working parents of 0-5 y.o. kids
- C-suite executives
- Internet based financial services business
- Online payment service users

6. CUSTOMER CONSTRAINTS
What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.
- Lacking basic knowledge in verifying the correct URL of website
- Malwares have become more complex than what a layman can understand

5. AVAILABLE SOLUTIONS
[illegible] or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking
- Word of mouth
- News coverage
- Social media

2. JOBS-TO-BE-DONE / PROBLEMS
Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.
- Prevent personal data getting stolen
- Ensure user safety

9. PROBLEM ROOT CAUSE
What is the real reason that this problem exists? What is the back story behind the need for this job?
- Greedy scammers
- Lack of awareness from customers

7. BEHAVIOUR
What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)
- Contacting cybersecurity
- Researching about website
- Reporting the site

3. TRIGGERS
What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.
- To prevent data including login credentials and credit card numbers from getting stolen
- Increases awareness

4. EMOTIONS: BEFORE / AFTER
How do customers feel when they face a problem or a job and afterwards?
i.e. lost, insecure > confident, in control - use it in your communication strategy & design.
- Insecure>secure
- Suspicious> trustworthy

10. YOUR SOLUTION
If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.
- Verifies the geniuses of E-Banking websites/gateway

8. CHANNELS of BEHAVIOUR
8.1 ONLINE
What kind of actions do customers take online? Extract online channels from #7
- Researching website
- Reporting the site

8.2 OFFLINE
What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.
- Filling complaint with bank
- Contacting cybersecurity

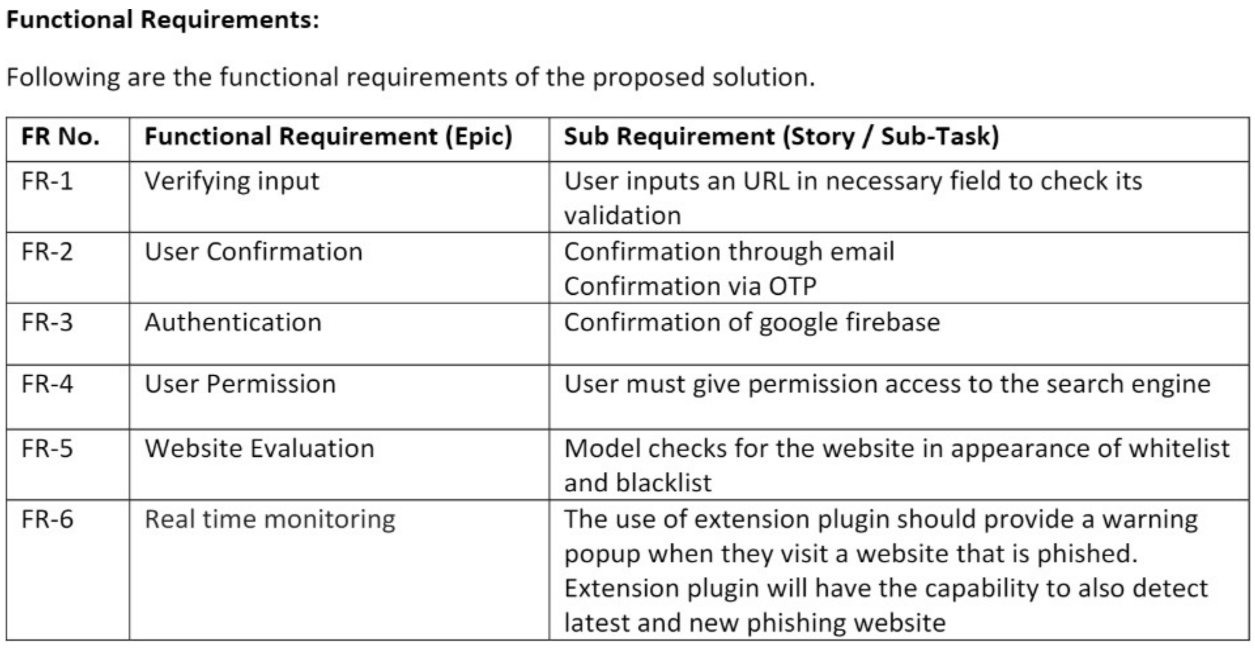
CHAPTER 4

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

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

- Address Bar based Features.
- Abnormal Based Features
- HTML and JavaScript Based Features.
- Domain Based Features.



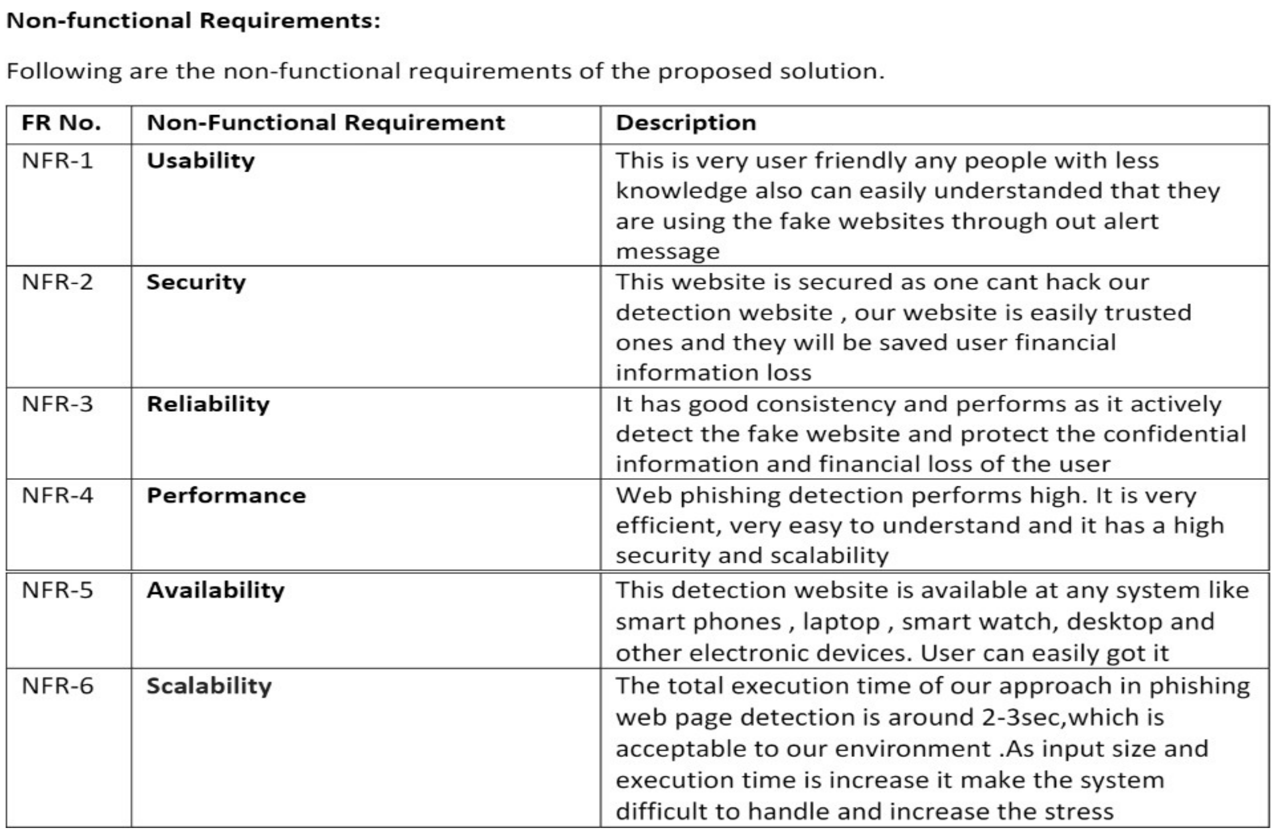
Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Verifying input	User inputs an URL in necessary field to check its validation
FR-2	User Confirmation	Confirmation through email Confirmation via OTP
FR-3	Authentication	Confirmation of google firebase
FR-4	User Permission	User must give permission access to the search engine
FR-5	Website Evaluation	Model checks for the website in appearance of whitelist and blacklist
FR-6	Real time monitoring	The use of extension plugin should provide a warning popup when they visit a website that is phished. Extension plugin will have the capability to also detect latest and new phishing website

4.2 Non-Functional requirements

Non-functional Requirements (NFRs) define system attributes such as security, reliability, performance, maintainability, scalability, and usability. They serve as constraints or restrictions on the design of the system across the different backlogs.



Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

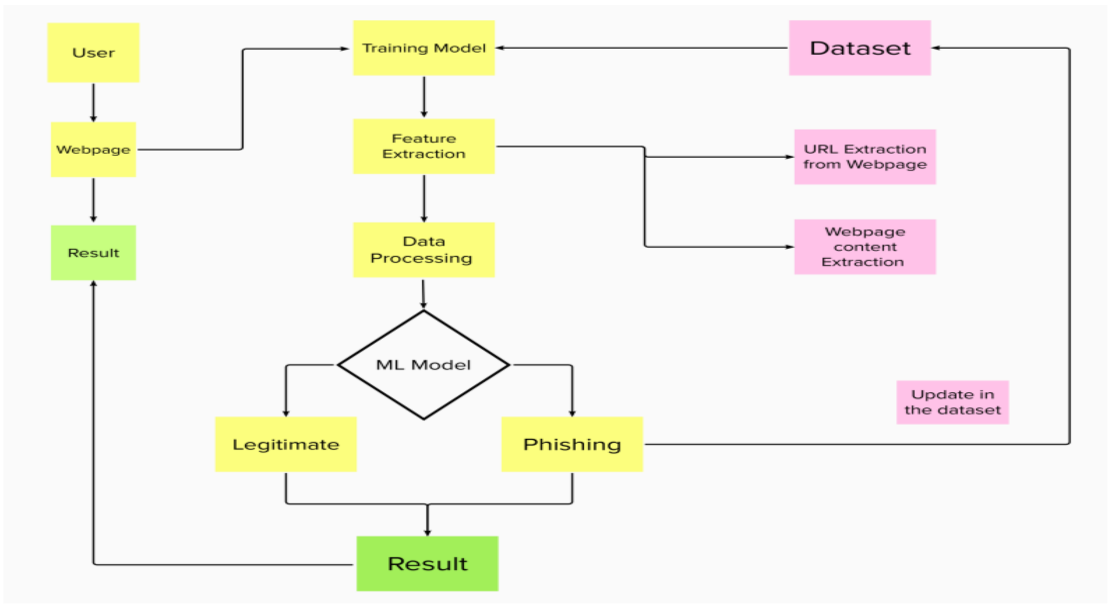
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This is very user friendly any people with less knowledge also can easily understanded that they are using the fake websites through out alert message
NFR-2	**Security**	This website is secured as one cant hack our detection website , our website is easily trusted ones and they will be saved user financial information loss
NFR-3	**Reliability**	It has good consistency and performs as it actively detect the fake website and protect the confidential information and financial loss of the user
NFR-4	**Performance**	Web phishing detection performs high. It is very efficient, very easy to understand and it has a high security and scalability
NFR-5	**Availability**	This detection website is available at any system like smart phones , laptop , smart watch, desktop and other electronic devices. User can easily got it
NFR-6	**Scalability**	The total execution time of our approach in phishing web page detection is around 2-3sec,which is acceptable to our environment .As input size and execution time is increase it make the system difficult to handle and increase the stress

CHAPTER 5

5. PROJECT DESIGN

5.1 Data Flow Diagrams

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually —sayǁ things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That's why DFDs remain so popular after all these years.

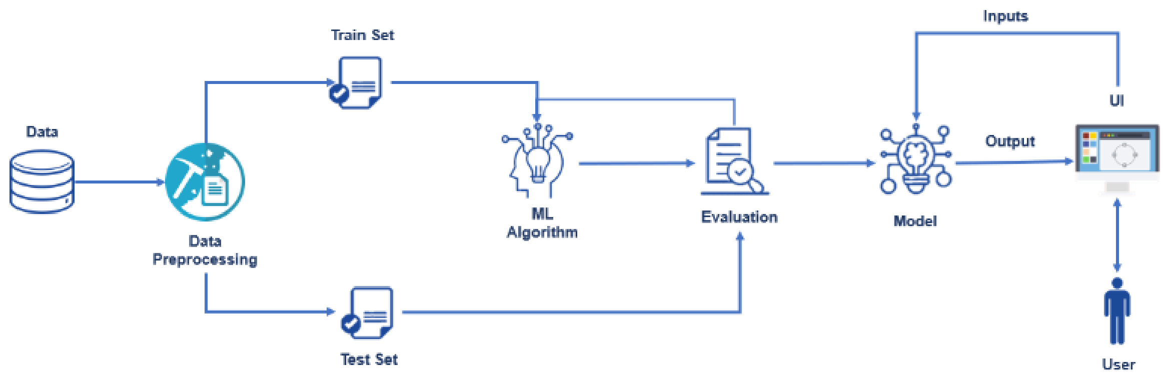


5.2 Solution & Technical Architecture

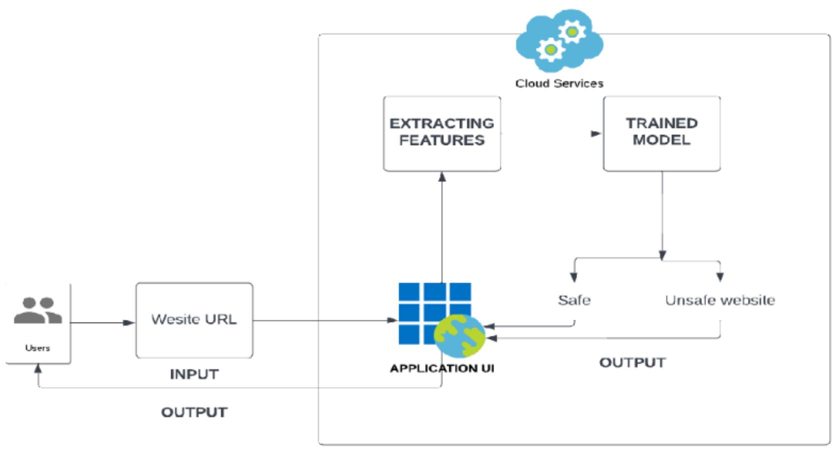
This project will be approached based on decision tree, random forest and SVM algorithms.

- The data is collected from the dataset and it is pre-processed.
- Then it is applied to the machine learning algorithmic model and analysed.
- The model is trained and tested according to the problem specified.
- Python Flask and IBM cloud facility is used in developing this project.

SOLUTION ARCHITECTURE:



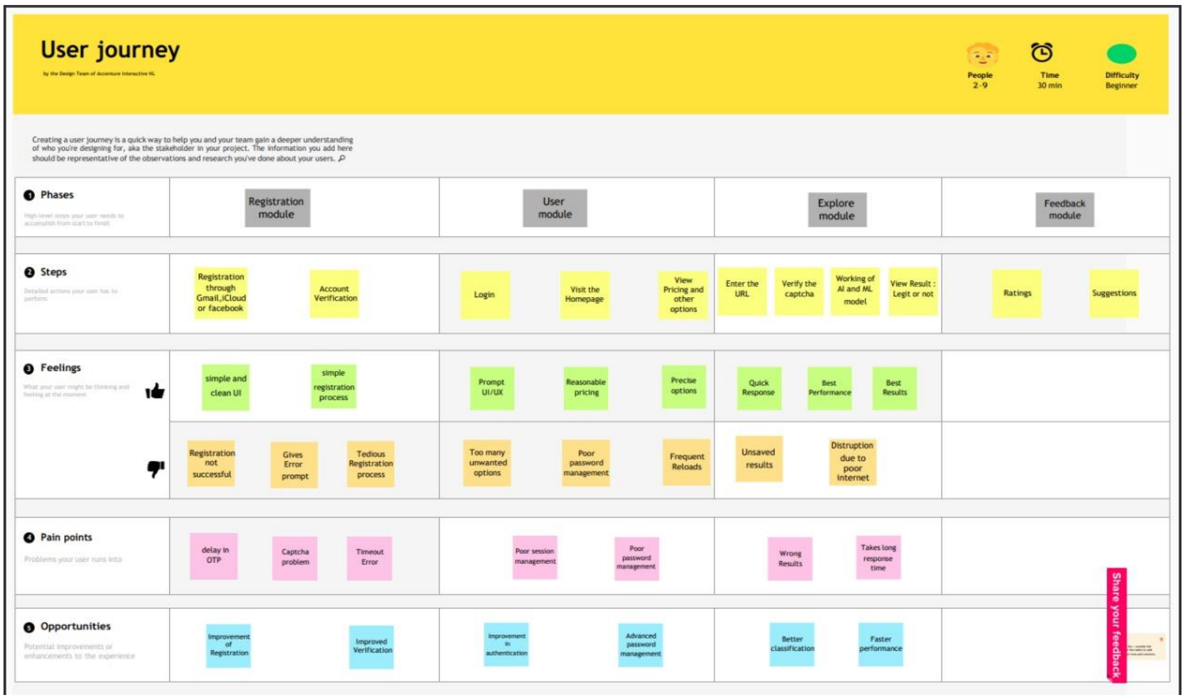
TECHNICAL ARCHITECTURE:



S.No	Component	Description	Technology
1.	User Interface	User interacts with application e.g.Web UI, App, Chatbot etc.	HTML, CSS, Python - Flask
2.	URL - Input	The website URL which the user wants to check whether it is legit or not	HTML, CSS, Python - Flask
3.	URL - Output	A prediction on whether the site is legit or not	HTML, CSS, Python - Flask
4.	APP UI	User interacts with application. Logic for a process in the application.	HTML, CSS, Python - Flask
5.	Feature Extraction	Extract the URL attributes, data type, Configurations, etc.	Python
6.	Trained Model	Best metrics, accuracy, classify whether legit or not.	Python, Jupyter Notebook
7.	Model Output	Classification whether it is legit or not	Python, Jupyter Notebook
8.	Infrastructure (Server / Cloud) - IBM	Application Deployment on Local System / Cloud Local Server Configuration Cloud Server Configuration	IBM Cloud

5.3 User Stories

A user story is the smallest unit of work in an agile framework. It’s an end goal, not a feature, expressed from the software user’s perspective. A user story is an informal, general explanation of a software feature written from the perspective of the end user or customer. The purpose of a user story is to articulate how a piece of work will deliver a particular value back to the customer. Note that "customers" don't have to be external end users in the traditional sense, they can also be internal customers or colleagues within your organization who depend on your team.



User journey

by the Design Team of Accenture Interactive NL

People 2-9 | Time 30 min | Difficulty Beginner

Creating a user journey is a quick way to help you and your team gain a deeper understanding of who you're designing for, aka the stakeholder in your project. The information you add here should be representative of the observations and research you've done about your users.

1 Phases High-level steps your user needs to accomplish from start to finish	Registration module	User module	Explore module	Feedback module
2 Steps Detailed actions your user has to perform	Registration through Gmail,iCloud or facebook; Account Verification	Login; Visit the Homepage; View Pricing and other options	Enter the URL; Verify the captcha; Working of AI and ML model; View Result : Legit or not	Ratings; Suggestions
3 Feelings What your user might be thinking and feeling at the moment 👍	simple and clean UI; simple registration process	Prompt UI/UX; Reasonable pricing; Precise options	Quick Response; Best Performance; Best Results	
👎	Registration not successful; Gives Error prompt; Tedious Registration process	Too many unwanted options; Poor password management; Frequent Reloads	Unsaved results; Distruption due to poor internet	
4 Pain points Problems your user runs into	delay in OTP; Captcha problem; Timeout Error	Poor session management; Poor password management	Wrong Results; Takes long response time	
5 Opportunities Potential improvements or enhancements to the experience	Improvement of Registration; Improved Verification	Improvement in authentication; Advanced password management	Better classification; Faster performance	

Share your feedback

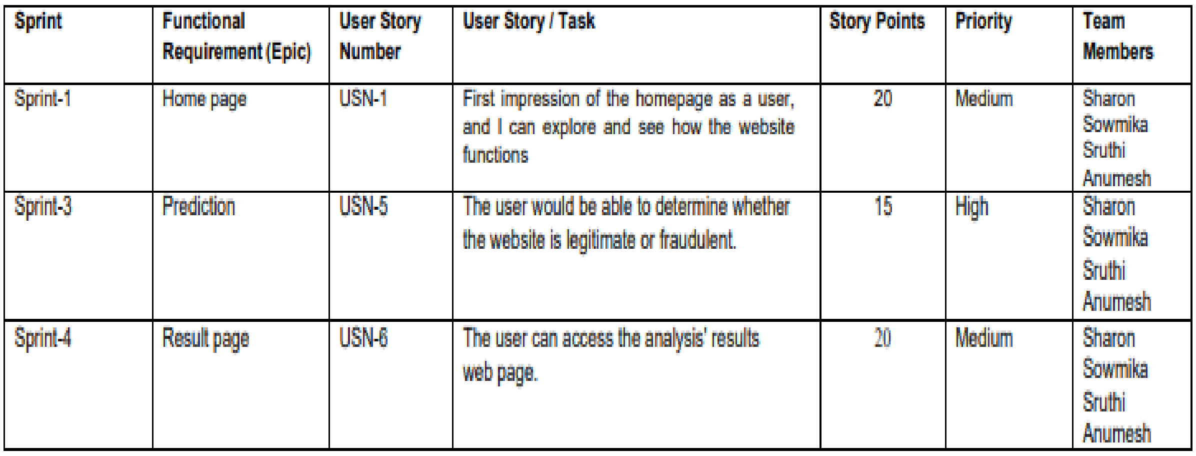
List-based phishing detection methods use either whitelist or blacklist-based technique. A blacklist contains a list of suspicious domains, URLs, and IP addresses, which are used to validate if a’ URL is fraudulent.

CHAPTER 6

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

During sprint planning, we break the stories down into tasks, estimate those tasks, and compare the task estimates against our capacity. It's that, not points, that keep us from overcommitting in this sprint. No need to change the estimate.
Estimation is an essential management operation used to determine an approximate time frame required to start and finish any process in a controlled environment. It's crucial to any project planning to not go past the time limits, set budgets, and available resources.

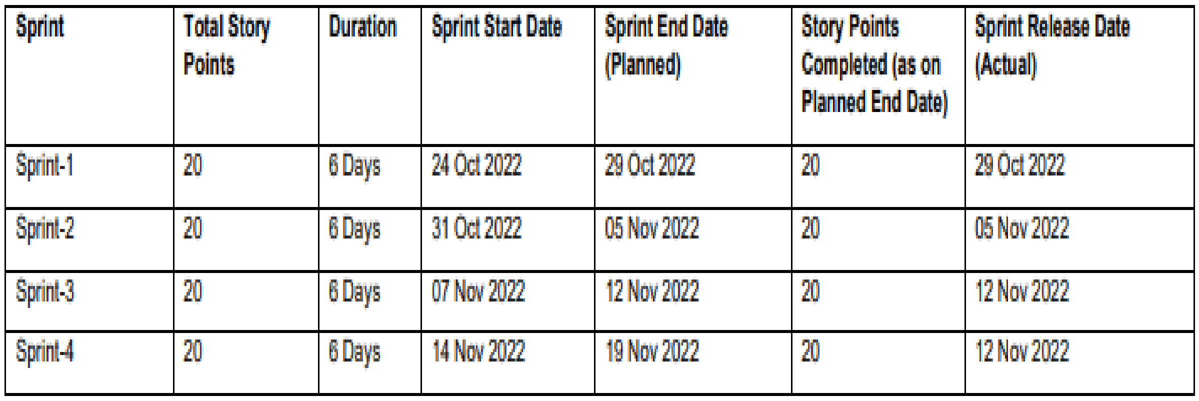


Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Home page	USN-1	First impression of the homepage as a user, and I can explore and see how the website functions	20	Medium	Sharon Sowmika Sruthi Anumesh
Sprint-3	Prediction	USN-5	The user would be able to determine whether the website is legitimate or fraudulent.	15	High	Sharon Sowmika Sruthi Anumesh
Sprint-4	Result page	USN-6	The user can access the analysis' results web page.	20	Medium	Sharon Sowmika Sruthi Anumesh

For super-fast Agile estimation, the items to be estimated are simply placed by the group in one of three categories: big, uncertain and small. The group starts by discussing a few together, and then, like the Bucket System, uses divide-and- conquer to go through the rest of the items.

6.2 Sprint Delivery Schedule

sprint is a set period of time during which specific work has to be completed and made ready for review. Each sprint begins with a planning meeting. During the meeting, the product owner (the person requesting the work) and the development team agree upon exactly what work will be accomplished during the sprint. The development team has the final say when it comes to determining how much work can realistically be accomplished during the sprint, and the product owner has the final say on what criteria need to be met for the work to be approved and accepted.



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

The deliverables of a sprint aren't as predictable as they are for other projects. Sprint participants have produced sketches and drawings, writing, photographs, comic strips, videos and fully coded working prototypes. The answer is whatever's right to answer the problem. The Scrum Developer is the professional responsible for creating the project deliverables, together with the rest of the Scrum team. As described in the Scrum Guide, there are three core roles in Scrum, responsible for meeting the project objectives: the Product Owner, the Scrum Master and the Development Team. Grooming the backlog and deciding which work to complete in the upcoming sprint. Agile delivery is an iterative approach to software delivery in which teams build software incrementally at the beginning of a project rather than ship it at once upon completion. Agile development means taking iterative, incremental, and lean approaches to streamline and accelerate the delivery of projects. the Scrum Master is not responsible for delivery. Not directly. The Scrum Master is part of the Scrum Team. The Scrum Team is responsible for delivery of working software.

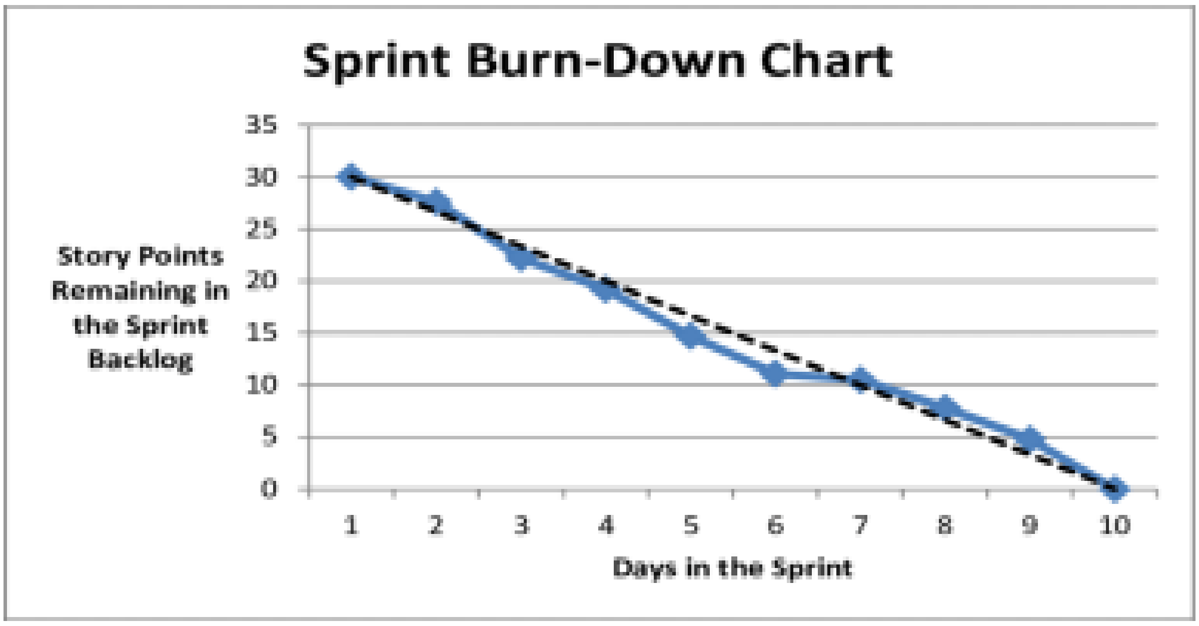
Velocity:

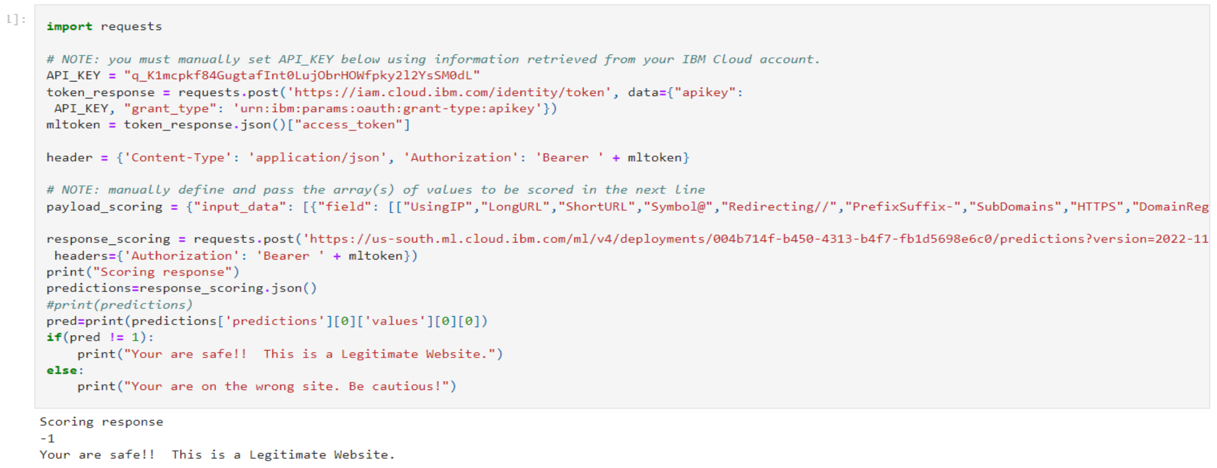
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)

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

Burndown Chart:

A burn down chart shows the amount of work that has been completed in an epic or sprint, and the total work remaining. Burn down charts are used to predict your team's likelihood of completing their work in the time available. They're also great for keeping the team aware of any scope creep that occurs.





```
1]:
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "q_K1mcpkf84GugtafInt0LujObrHOWfpky2l2YsSM0dL"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

# NOTE: manually define and pass the array(s) of values to be scored in the next line
payload_scoring = {"input_data": [{"field": [["UsingIP","LongURL","ShortURL","Symbol@","Redirecting//","PrefixSuffix-","SubDomains","HTTPS","DomainReg

response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/004b714f-b450-4313-b4f7-fb1d5698e6c0/predictions?version=2022-11
 headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
predictions=response_scoring.json()
#print(predictions)
pred=print(predictions['predictions'][0]['values'][0][0])
if(pred != 1):
    print("Your are safe!!  This is a Legitimate Website.")
else:
    print("Your are on the wrong site. Be cautious!")
```

```
Scoring response
-1
Your are safe!!  This is a Legitimate Website.
```

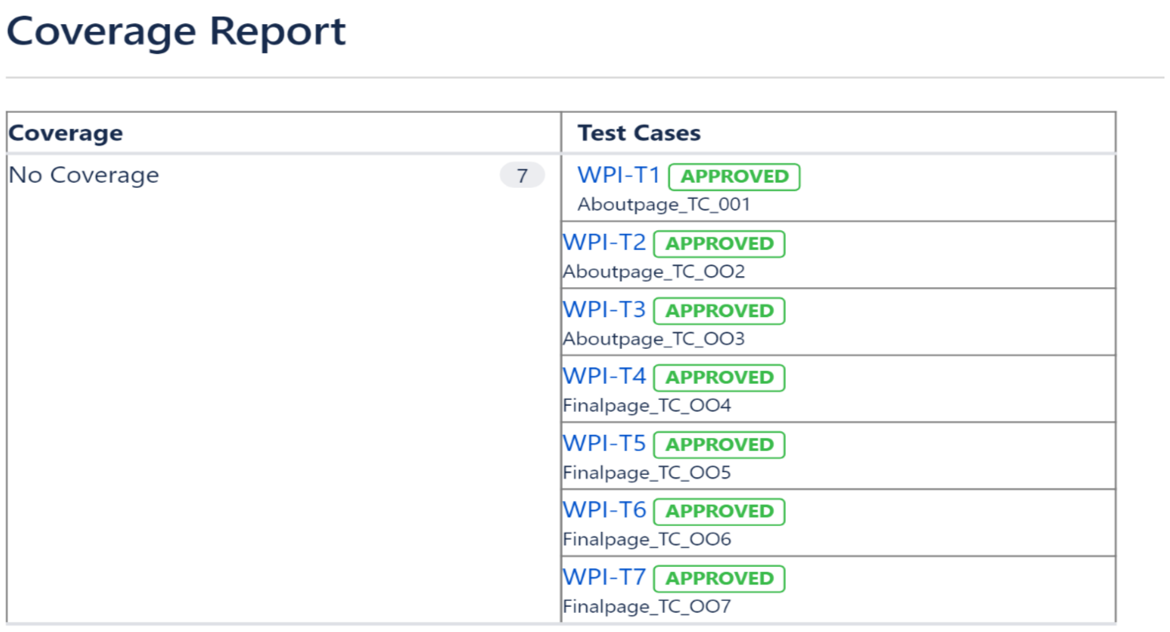
6.3 Report from JIRA

When JIRA sends either standard notifications or user invitations to a mail server, they are listed as phishing attempts rather than legitimate emails

The mail server is receiving a constant stream of concurrent multiple email notifications from the same sender which, in turn, triggers security measures on the server which handle these messages as phishing attempts.

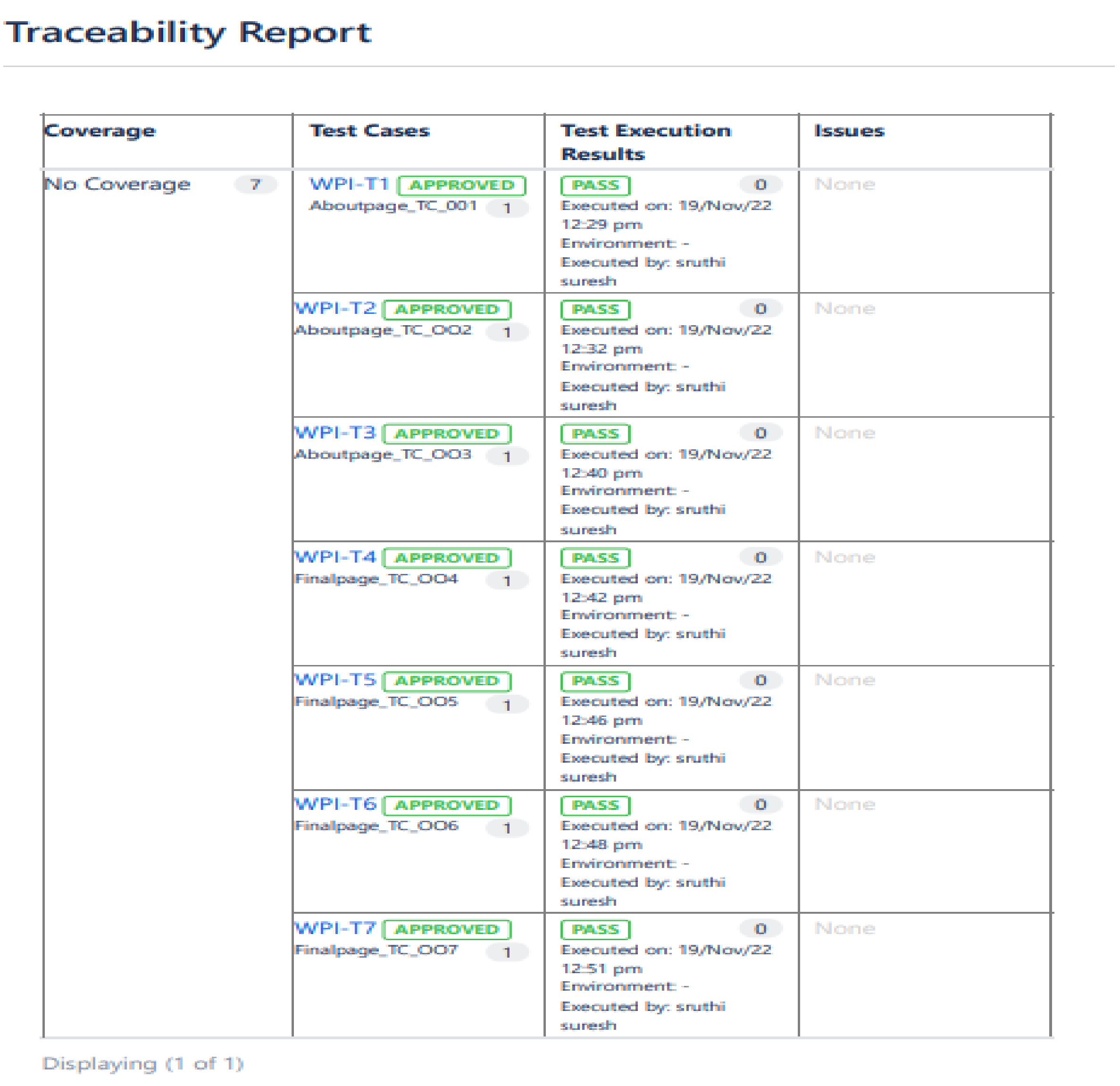
Resolution

- Check the base url of JIRA to see if it is set as a direct ip address with portnumber.
- Example: **http://10.10.10.10:8080**
- Some email servers (such as Microsoft Outlook) will consider messages from non-DNS urls as phishing attempts. You can correct this behaviour bysetting JIRA's base url to a url address such as **http://my-jira.com.**
- Sometimes when certain mail servers receive multiple emails from the same sender security measures are triggered that will then list those emails as phishing messages. For this, it is best to check with the local mail server administrator for further assistance and confirmation.



Coverage Report

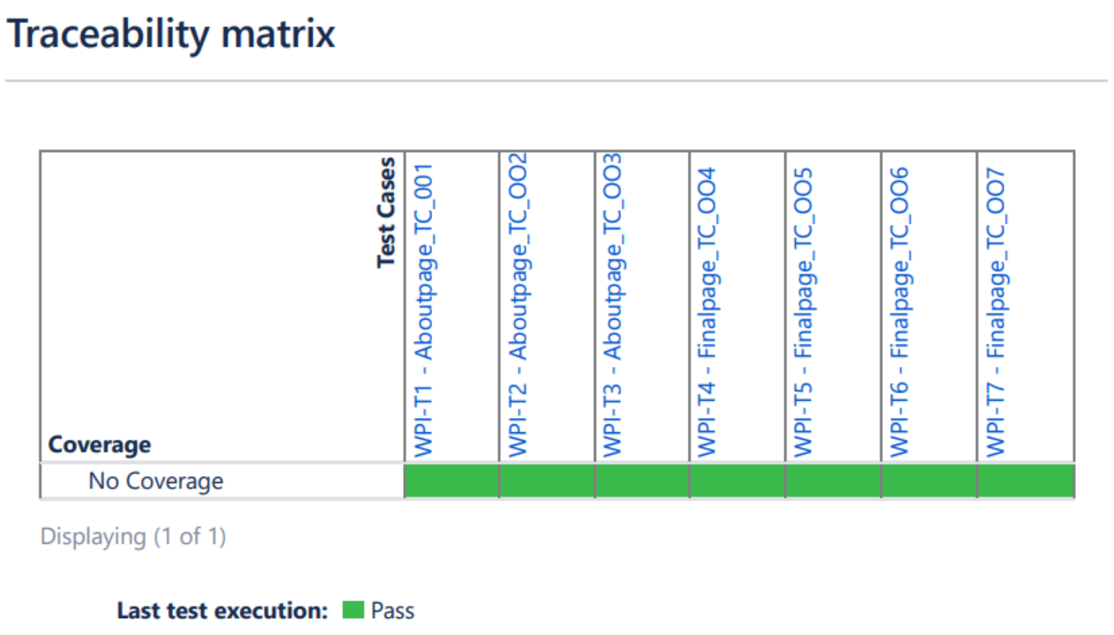
Coverage	Test Cases
No Coverage 7	WPI-T1 APPROVED Aboutpage_TC_001
	WPI-T2 APPROVED Aboutpage_TC_OO2
	WPI-T3 APPROVED Aboutpage_TC_OO3
	WPI-T4 APPROVED Finalpage_TC_OO4
	WPI-T5 APPROVED Finalpage_TC_OO5
	WPI-T6 APPROVED Finalpage_TC_OO6
	WPI-T7 APPROVED Finalpage_TC_OO7



Traceability Report

Coverage	Test Cases	Test Execution Results	Issues
No Coverage 7	WPI-T1 APPROVED Aboutpage_TC_001 1	PASS 0 Executed on: 19/Nov/22 12:29 pm Environment: - Executed by: sruthi suresh	None
	WPI-T2 APPROVED Aboutpage_TC_OO2 1	PASS 0 Executed on: 19/Nov/22 12:32 pm Environment: - Executed by: sruthi suresh	None
	WPI-T3 APPROVED Aboutpage_TC_OO3 1	PASS 0 Executed on: 19/Nov/22 12:40 pm Environment: - Executed by: sruthi suresh	None
	WPI-T4 APPROVED Finalpage_TC_OO4 1	PASS 0 Executed on: 19/Nov/22 12:42 pm Environment: - Executed by: sruthi suresh	None
	WPI-T5 APPROVED Finalpage_TC_OO5 1	PASS 0 Executed on: 19/Nov/22 12:46 pm Environment: - Executed by: sruthi suresh	None
	WPI-T6 APPROVED Finalpage_TC_OO6 1	PASS 0 Executed on: 19/Nov/22 12:48 pm Environment: - Executed by: sruthi suresh	None
	WPI-T7 APPROVED Finalpage_TC_OO7 1	PASS 0 Executed on: 19/Nov/22 12:51 pm Environment: - Executed by: sruthi suresh	None

Displaying (1 of 1)

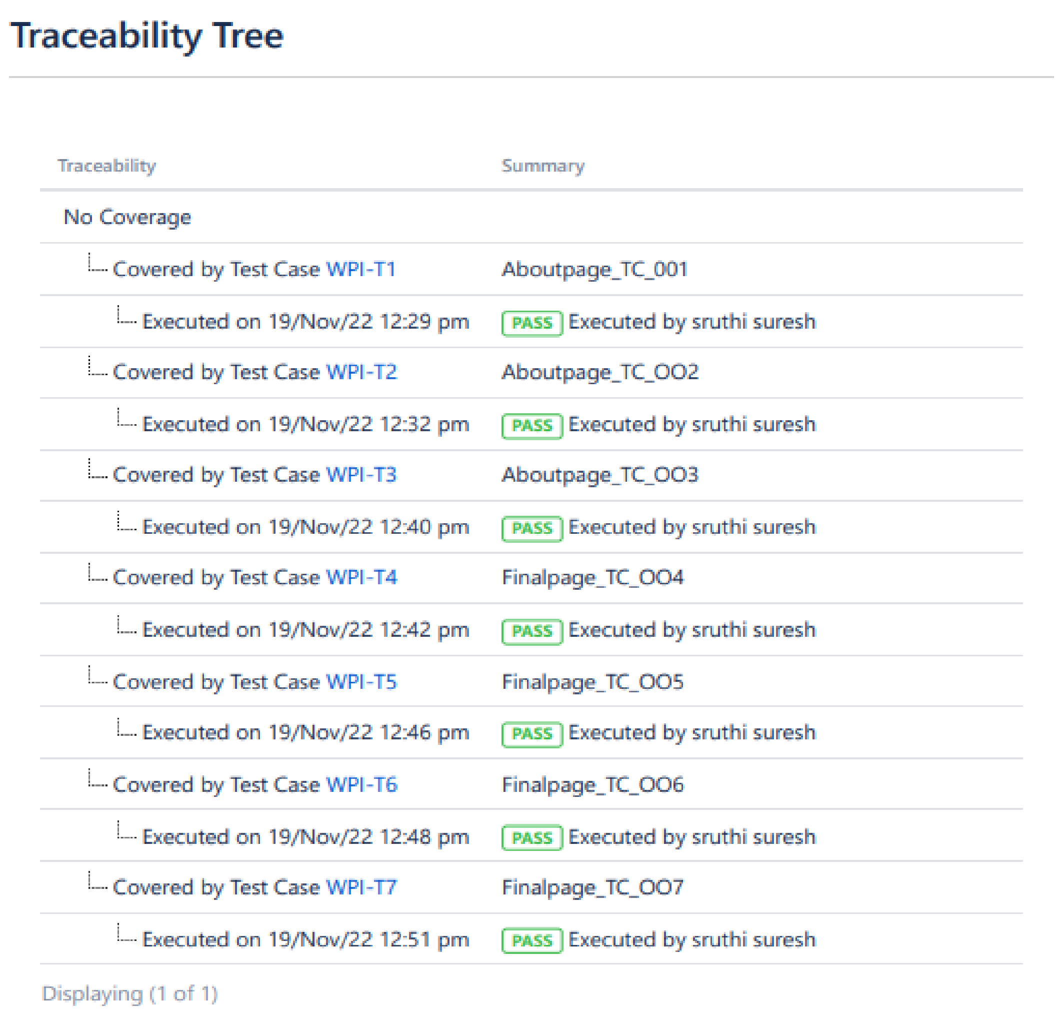


Traceability matrix

Coverage / Test Cases	WPI-T1 - Aboutpage_TC_001	WPI-T2 - Aboutpage_TC_OO2	WPI-T3 - Aboutpage_TC_OO3	WPI-T4 - Finalpage_TC_OO4	WPI-T5 - Finalpage_TC_OO5	WPI-T6 - Finalpage_TC_OO6	WPI-T7 - Finalpage_TC_OO7
No Coverage	🟩	🟩	🟩	🟩	🟩	🟩	🟩

Displaying (1 of 1)

Last test execution: 🟩 Pass



Traceability Tree

Traceability	Summary
No Coverage	
└ Covered by Test Case WPI-T1	Aboutpage_TC_001
└ Executed on 19/Nov/22 12:29 pm	PASS Executed by sruthi suresh
└ Covered by Test Case WPI-T2	Aboutpage_TC_OO2
└ Executed on 19/Nov/22 12:32 pm	PASS Executed by sruthi suresh
└ Covered by Test Case WPI-T3	Aboutpage_TC_OO3
└ Executed on 19/Nov/22 12:40 pm	PASS Executed by sruthi suresh
└ Covered by Test Case WPI-T4	Finalpage_TC_OO4
└ Executed on 19/Nov/22 12:42 pm	PASS Executed by sruthi suresh
└ Covered by Test Case WPI-T5	Finalpage_TC_OO5
└ Executed on 19/Nov/22 12:46 pm	PASS Executed by sruthi suresh
└ Covered by Test Case WPI-T6	Finalpage_TC_OO6
└ Executed on 19/Nov/22 12:48 pm	PASS Executed by sruthi suresh
└ Covered by Test Case WPI-T7	Finalpage_TC_OO7
└ Executed on 19/Nov/22 12:51 pm	PASS Executed by sruthi suresh

Displaying (1 of 1)

CHAPTER 7

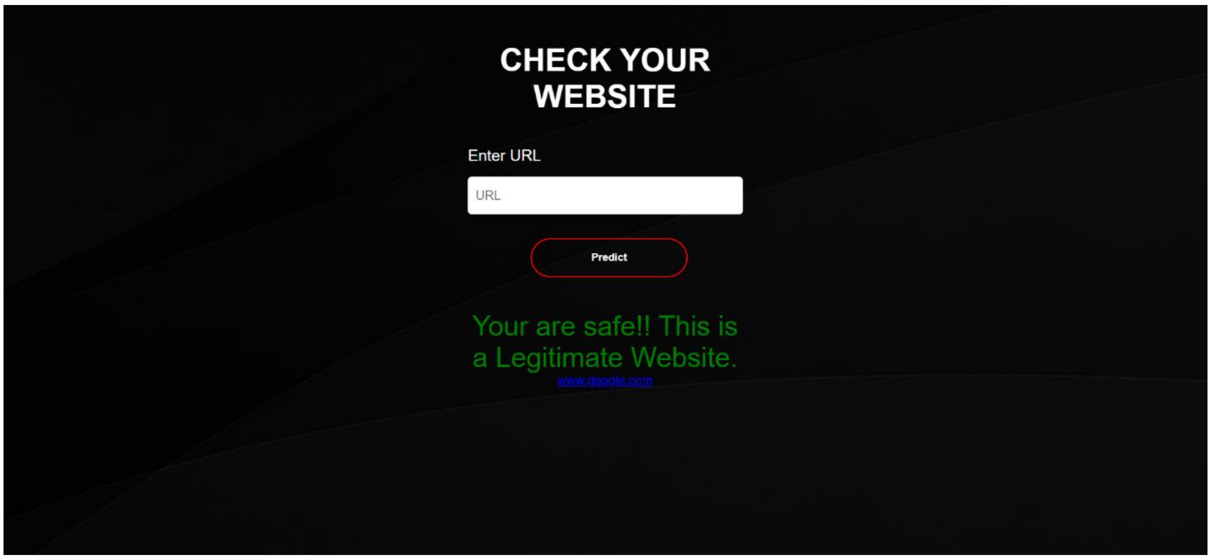
7. CODING & SOLUTIONING

7.1 Libraries to be installed

```
import pandas as pd
import numpy as np
import keras
import pickle
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from xgboost import XGBClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix,accuracy_score
```

7.2 Phishing URL Detection

Phishing is a form of fraud in which the attacker tries to learn sensitive information such as login credentials of person in email or other communication channels.



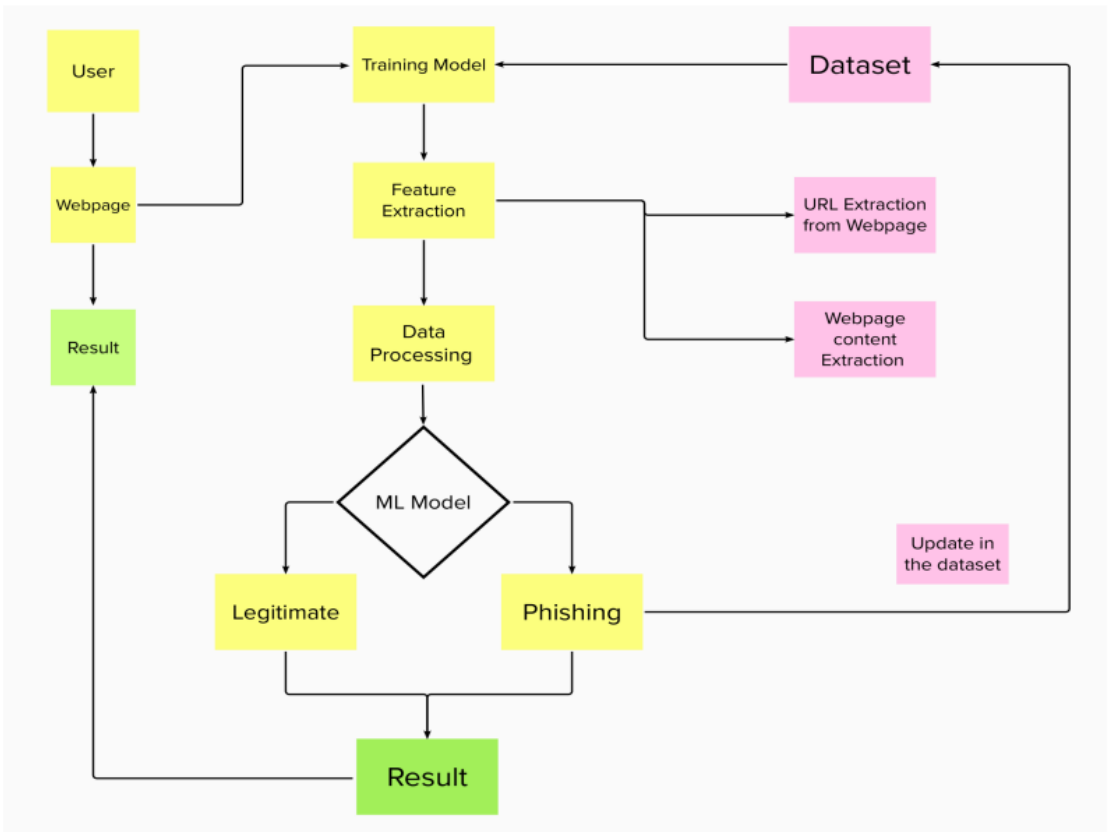
Your are safe!! This is a Legitimate Website.
www.google.com

- **Accuracy of an URL**

we present a way to detect such malicious URL addresses with almost 100% accuracy using convolutional neural network.

- **Work flow**

We propose a phishing detection approach, which extracts efficient features from theURL and HTML of the given webpage without relying on third-party services. Thus,it can be adaptable at the client side and specify better privacy. An efficient solution for phishing detection that extracts the features from website's URL and HTML source code proposed.



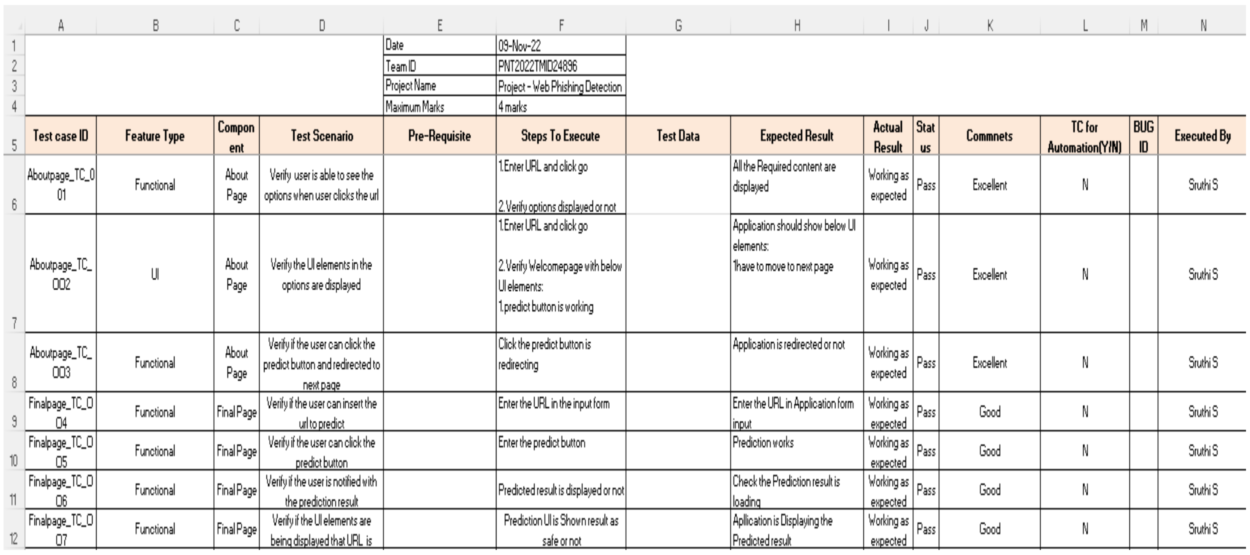
CHAPTER 8

8. TESTING

For the URL verifier module in the ISOT phishing detection system, phishing detection is done using 16 different heuristic rules. In the system, 11 main classes were defined, and 1 class was defined with 5 sub-classes. This covers all 16 heuristic rules. To test the system, 7 test cases were designed using assertion methods. Ten test cases were designed to test the 10 main classes and 5 test cases were designed to test the class with five sub-classes. The getter-setter method was used to test the class with five sub-classes. The getter method is used to obtain or retrieve a variable value from the class, and the setter method is used to store the variables.

8.1 Test scenarios

- Verify if user can see the options when user clicks the url .
- Verify if the UI elements are getting displayed properly.
- Verify if the user can click the predict button and redirected to next page.
- Verify if the user can insert the url to predict .
- Verify if the user can click the predict button.
- Verify if the user is notified with the prediction result.
- Verify if the UI elements are being displayed .
- Verify if the application can predict the url safe or not.



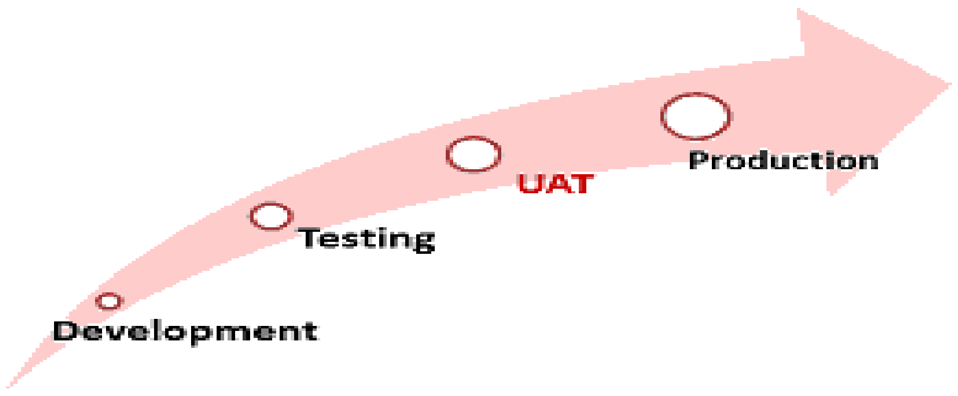
Date	09-Nov-22
Team ID	PNT2022TMID24896
Project Name	Project - Web Phishing Detection
Maximum Marks	4 marks

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
Aboutpage_TC_001	Functional	About Page	Verify user is able to see the options when user clicks the url		1.Enter URL and click go 2.Verify options displayed or not		All the Required content are displayed	Working as expected	Pass	Excellent	N		Sruthi S
Aboutpage_TC_OO2	UI	About Page	Verify the UI elements in the options are displayed		1.Enter URL and click go 2.Verify Welcomepage with below UI elements: 1.predict button is working		Application should show below UI elements: 1have to move to next page	Working as expected	Pass	Excellent	N		Sruthi S
Aboutpage_TC_OO3	Functional	About Page	Verify if the user can click the predict button and redirected to next page		Click the predict button is redirecting		Application is redirected or not	Working as expected	Pass	Excellent	N		Sruthi S
Finalpage_TC_O04	Functional	Final Page	Verify if the user can insert the url to predict		Enter the URL in the input form		Enter the URL in Application form input	Working as expected	Pass	Good	N		Sruthi S
Finalpage_TC_O05	Functional	Final Page	Verify if the user can click the predict button		Enter the predict button		Prediction works	Working as expected	Pass	Good	N		Sruthi S
Finalpage_TC_O06	Functional	Final Page	Verify if the user is notified with the prediction result		Predicted result is displayed or not		Check the Prediction result is loading	Working as expected	Pass	Good	N		Sruthi S
Finalpage_TC_O07	Functional	Final Page	Verify if the UI elements are being displayed that URL is		Prediction UI is Shown result as safe or not		Apllication is Displaying the Predicted result	Working as expected	Pass	Good	N		Sruthi S

Figure 8.1: All test cases are passed.

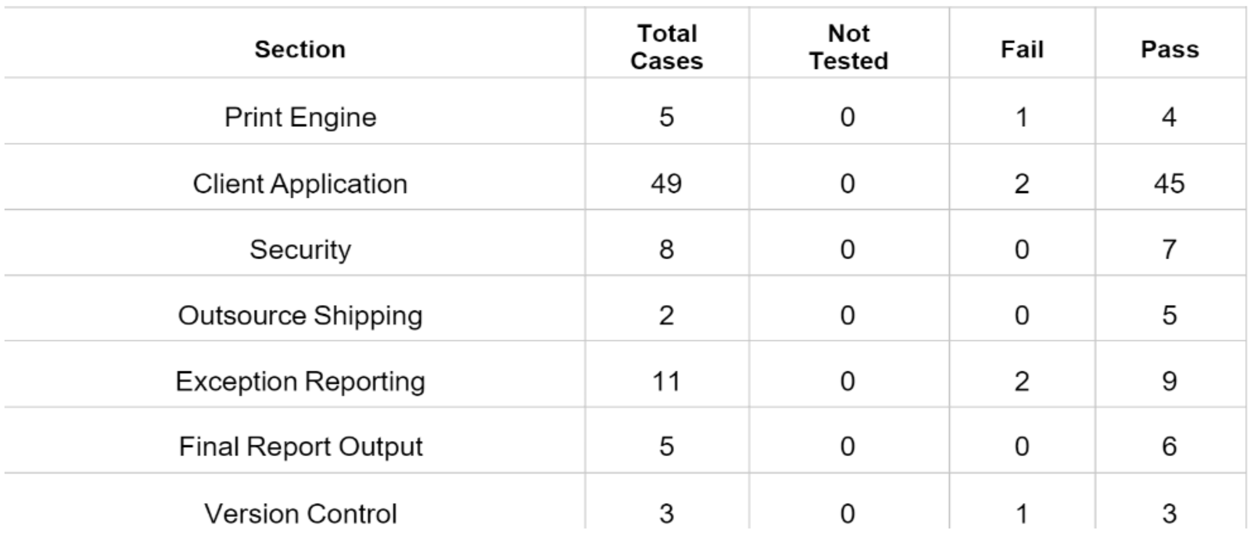
8.2 User Acceptance Testing

User Acceptance Testing (UAT) is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done.



The main **Purpose of UAT** is to validate end to end business flow. It does not focus on cosmetic errors, spelling mistakes or system testing. User Acceptance Testing is carried out in a separate testing environment with production-like data setup. It is kind of black box testing where two or more end-users will be involved.

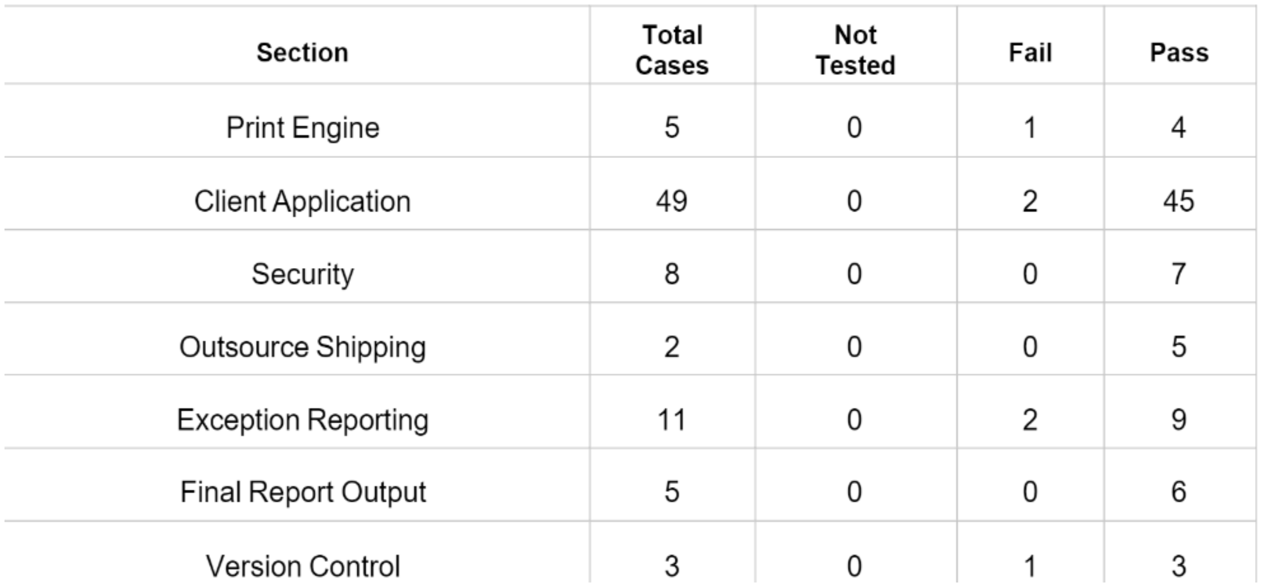
8.2.1 Testcase Analysis:



Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	1	4
Client Application	49	0	2	45
Security	8	0	0	7
Outsource Shipping	2	0	0	5
Exception Reporting	11	0	2	9
Final Report Output	5	0	0	6
Version Control	3	0	1	3

Figure:8.2.1 Testcase analysis

8.2.2 Defect Analysis:



Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	1	4
Client Application	49	0	2	45
Security	8	0	0	7
Outsource Shipping	2	0	0	5
Exception Reporting	11	0	2	9
Final Report Output	5	0	0	6
Version Control	3	0	1	3

Figure: 8.2.2 Defect analysis

CHAPTER 9

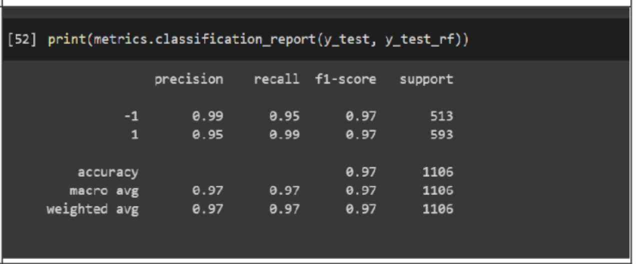
9.RESULT

9.1 Performance Testing

In a classification problem, the category or classes of data is identified based on training data. The model learns from the given dataset and then classifies the new data into classes or groups based on the training. It predicts class labels as the output, such as Yes or No, 0 or 1, Spam or Not Spam, etc. To evaluate the performance of a classification model, different metrics are used, and some ofthem are as follows:

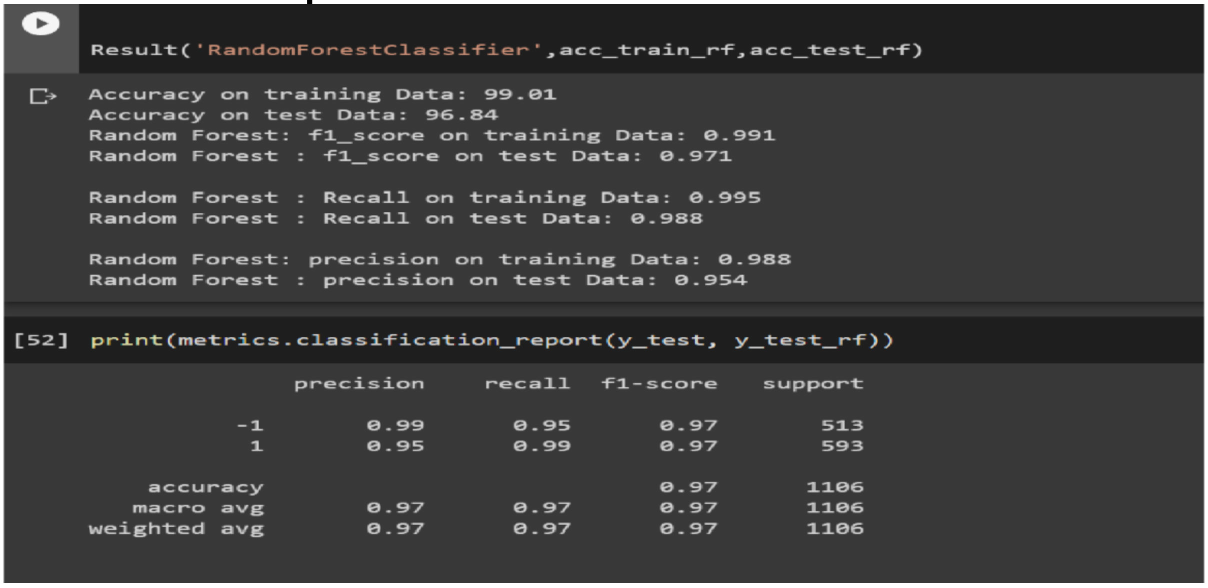
- Accuracy
- Confusion Matrix
- Precision
- Recall
- F-Score

Model performance testing:

Parameter	Values	Screenshot
Metrics	**Classification Model:** **Random Forest Classification** Train Accuracy Score- 99.01% Test Accuracy Score-96.47%	

```
[52] print(metrics.classification_report(y_test, y_test_rf))

              precision    recall  f1-score   support

          -1       0.99      0.95      0.97       513
           1       0.95      0.99      0.97       593

    accuracy                           0.97      1106
   macro avg       0.97      0.97      0.97      1106
weighted avg       0.97      0.97      0.97      1106
```

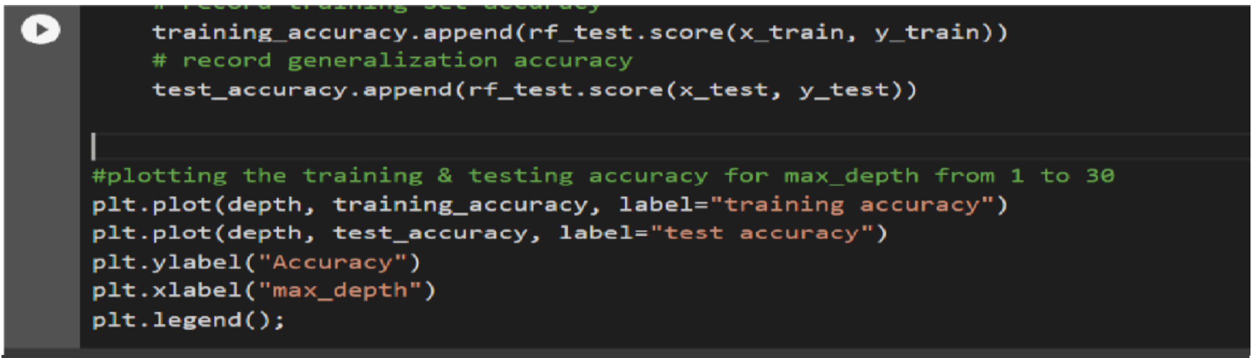
Classification Report:



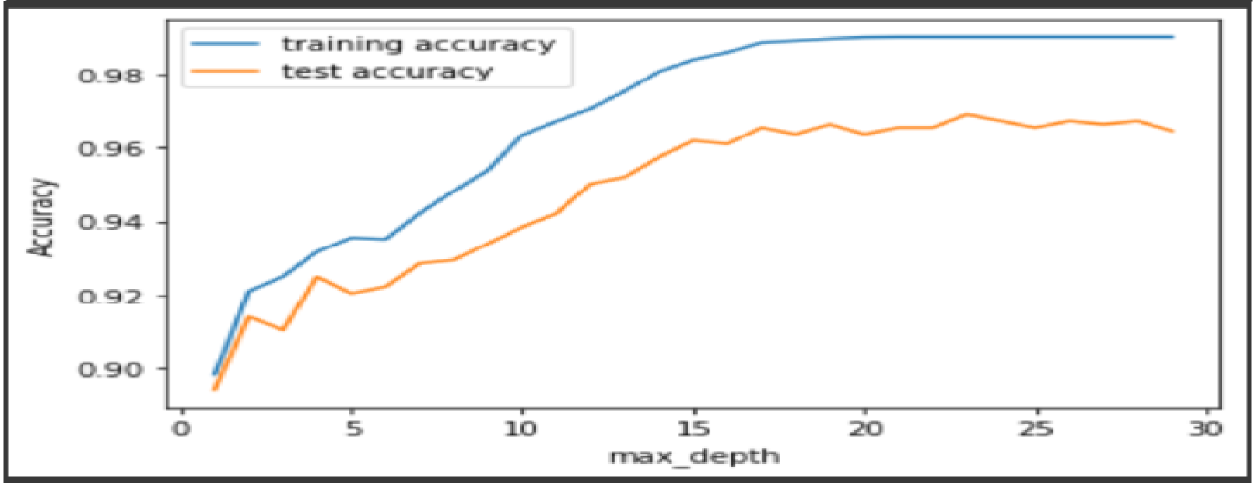
```
Result('RandomForestClassifier',acc_train_rf,acc_test_rf)

Accuracy on training Data: 99.01
Accuracy on test Data: 96.84
Random Forest: f1_score on training Data: 0.991
Random Forest : f1_score on test Data: 0.971

Random Forest : Recall on training Data: 0.995
Random Forest : Recall on test Data: 0.988

Random Forest: precision on training Data: 0.988
Random Forest : precision on test Data: 0.954
```

```
[52] print(metrics.classification_report(y_test, y_test_rf))

              precision    recall  f1-score   support

          -1       0.99      0.95      0.97       513
           1       0.95      0.99      0.97       593

    accuracy                           0.97      1106
   macro avg       0.97      0.97      0.97      1106
weighted avg       0.97      0.97      0.97      1106
```

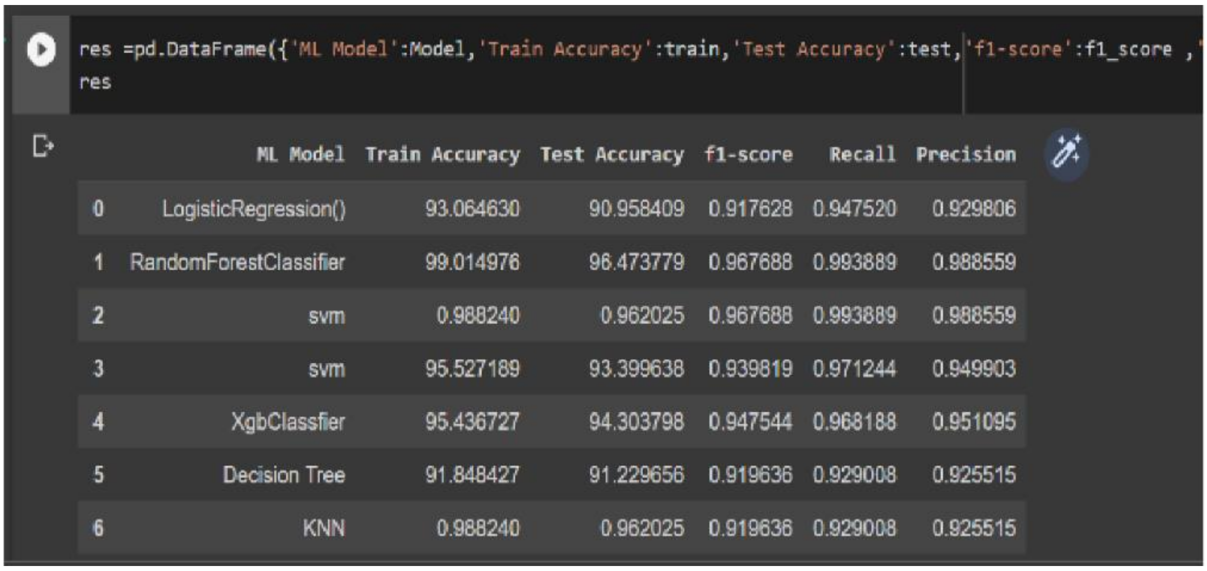
Performance:



```
    training_accuracy.append(rf_test.score(x_train, y_train))
    # record generalization accuracy
    test_accuracy.append(rf_test.score(x_test, y_test))

#plotting the training & testing accuracy for max_depth from 1 to 30
plt.plot(depth, training_accuracy, label="training accuracy")
plt.plot(depth, test_accuracy, label="test accuracy")
plt.ylabel("Accuracy")
plt.xlabel("max_depth")
plt.legend();
```



COMPARISON OF ACCURACY



```
res =pd.DataFrame({'ML Model':Model,'Train Accuracy':train,'Test Accuracy':test,'f1-score':f1_score ,'
res
```

	ML Model	Train Accuracy	Test Accuracy	f1-score	Recall	Precision
0	LogisticRegression()	93.064630	90.958409	0.917628	0.947520	0.929806
1	RandomForestClassifier	99.014976	96.473779	0.967688	0.993889	0.988559
2	svm	0.988240	0.962025	0.967688	0.993889	0.988559
3	svm	95.527189	93.399638	0.939819	0.971244	0.949903
4	XgbClassfier	95.436727	94.303798	0.947544	0.968188	0.951095
5	Decision Tree	91.848427	91.229656	0.919636	0.929008	0.925515
6	KNN	0.988240	0.962025	0.919636	0.929008	0.925515

CHAPTER 10

10. ADVANTAGES & DISADVANTAGES

Advantages:

- Requiring low resources on host machine
- Effective when minimal FP rates are required Heuristics and visual similarity
- Mitigate zero hour attacks. Machine Learning
- Mitigate zero-hour attacks.
- Construct own classification models.

Disadvantages:

- Mitigation of zero-hour phishing attacks.
- Can result in excessive queries with heavily loaded server. Heuristics and visual similarity
- Higher FP rate than blacklists.
- High computational cost. Machine Learning
- Time consuming
- Costly
- Huge number of rules
- loss of money
- loss of intellectual property
- damage to reputation
- disruption of operational activities
- Phishing scams involve sending out emails or texts disguised as legitimate sources.

CHAPTER 11

11. CONCLUSION:

Phishing websites are being designed to trick people into submitting credentials to access private information and assets by making the sites look like legitimate websites. Phishing attacks through URLs embedded in emails or SMS messages are one of the major issues faced by Internet users because of the huge number of online transactions performed daily. Phishing attacks cause losses to organizations, customers and Internet users. In this project, testing utilities were developed to support test automation for different aspects of the ISOT phishing detection system. In particular, two kind of test utilities were developed. 1. A mechanism to populate live test email accounts from different datasets, including spam, phishing, and legitimate emails, allowing online testing of the ISOT phishing detection system. 2. Automated test cases were used to unit test one of the modules of the ISOT phishing detection system. In this report, the data collection process for phishing, spam and legitimate emails was discussed. Further, the ISOT message security system was explained, and the design and implementation of the service to read the eml files from the collected datasets and send them to the test accounts was explained. The testing was done using the Junit framework to check the functionality of the different heuristic rules of the system. The results of the unit tests were verified using assertion methods in the Junit framework.

We achieved 99.01% detection accuracy using random forest algorithm with lowest false positive rate. Also result shows that classifiers give better performance when we used more data as training data. This paper aims to enhance detection method to detect phishing websites using machine learning technology. We achieved 99.01% detection accuracy using random forest algorithm with lowest false positive rate. Also result shows that classifiers give better performance when we used more data as training data. In future hybrid technology will be implemented to detect phishing websites more accurately, for which random forest algorithm of machine learning technology and blacklist method will be used.

CHAPTER 12

12. FUTURE SCOPE:

In the future, optimization can be done in the test units and these units can be made fully automated using Robot-Framework. This is important if more heuristics rules are included in the detection system. If the URL length is very long i.e. more than a million characters, then the system may crash. To prevent this situation, a timeout feature can be added when determining the URL length.

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

CHAPTER 13

13. APPENDIX

13.1 SOURCE CODE:

Model Building code:

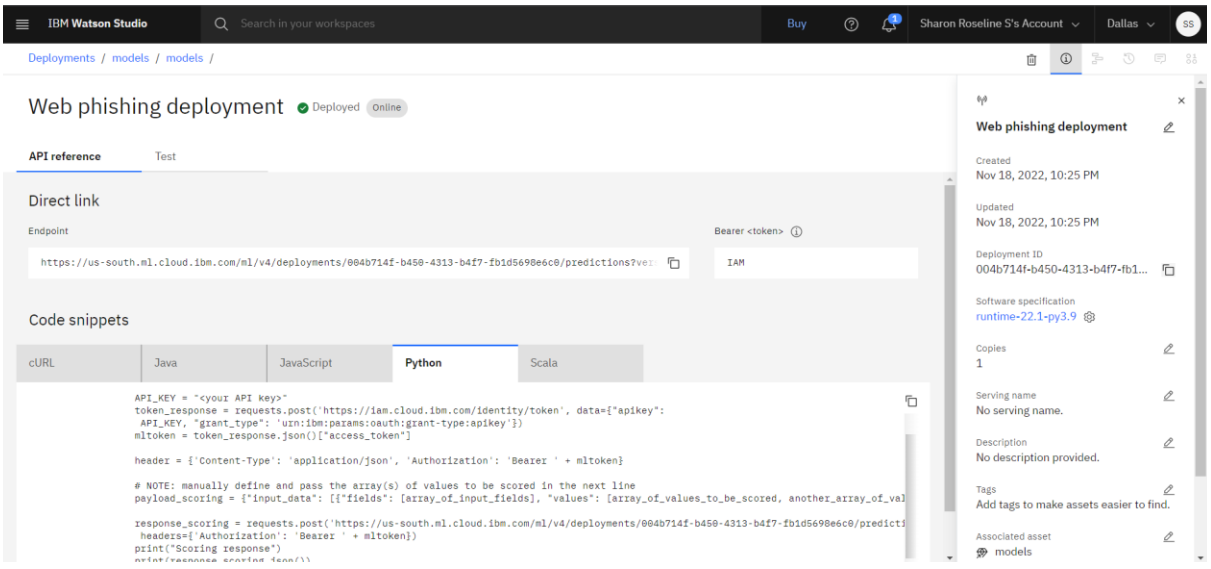
```
import pandas as pd
import numpy as np
import keras
import pickle
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import confusion_matrix,accuracy_score
ds =pd.read_csv("dataset_website.csv")
ds.head()
ds.shape
ds.columns
ds.info()
ds.isnull().any()
ds.describe()
y = ds['Result']
X = ds.drop('Result',axis=1)
X.shape, y.shape
x=ds.iloc[:,1:31].values
y=ds.iloc[:,-1].values
print(x,y)

**5.ML Model Training**
Model=[]
train=[]
test=[]
def Result(model,a,b):
  Model.append(model)
  train.append(round(a,5))
  test.append(round(b,5))
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.1,random_state=0)
x_train

**Logistic Regression**
from sklearn.linear_model import LogisticRegression
```

```
lr=LogisticRegression()
lr.fit(x_train,y_train)

y_test_lr = lr.predict(x_test)
y_train_lr = lr.predict(x_train)

acc_train_lr = accuracy_score(y_train,y_train_lr) *100
acc_test_lr = accuracy_score(y_test,y_test_lr)*100

print("Accuracy on training Data: {:.2f}".format(acc_train_lr))
print("Accuracy on test Data: {:.2f}".format(acc_test_lr))
Result(lr,acc_train_lr,acc_test_lr)

**Random Forest**

from sklearn.ensemble import RandomForestClassifier
rf= RandomForestClassifier()
rf.fit(x_train,y_train)

y_test_rf = rf.predict(x_test)
y_train_rf = rf.predict(x_train)

acc_train_rf = accuracy_score(y_train,y_train_rf)*100
acc_test_rf = accuracy_score(y_test,y_test_rf)*100

print("Accuracy on training Data: {:.2f}".format(acc_train_rf))
print("Accuracy on test Data: {:.2f}".format(acc_test_rf))
Result('RandomForestClassifier',acc_train_rf,acc_test_rf)

**SVM**
from sklearn.svm import SVC
svm = SVC()
svm.fit(x_train, y_train)

y_test_svm = svm.predict(x_test)
y_train_svm = svm.predict(x_train)

acc_train_svm = accuracy_score(y_train,y_train_svm)*100
acc_test_svm = accuracy_score(y_test,y_test_svm)*100

print("Accuracy on training Data: {:.3f}".format(acc_train_svm))
print("Accuracy on test Data: {:.3f}".format(acc_test_svm))
Result('svm',acc_train_svm,acc_test_svm)
```

```
**XGBoost**
from xgboost import XGBClassifier
xg = XGBClassifier()
xg.fit(x_train,y_train)

y_test_xg = xg.predict(x_test)
y_train_xg = xg.predict(x_train)

acc_train_xg = accuracy_score(y_train,y_train_xg) *100
acc_test_xg = accuracy_score(y_test,y_test_xg)*100

print("Accuracy on training Data: {:.3f}".format(acc_train_xg))
print("Accuracy on test Data: {:.3f}".format(acc_test_xg))
Result('XgbClassfier',acc_train_xg,acc_test_xg)

** Decision Tree**
from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier(criterion='gini', max_depth=4, random_state=0)
dt.fit(x_train, y_train)

y_test_dt = dt.predict(x_test)
y_train_dt = dt.predict(x_train)

acc_train_dt = accuracy_score(y_train,y_train_dt) *100
acc_test_dt = accuracy_score(y_test,y_test_dt)*100

print("Accuracy on training Data: {:.3f}".format(acc_train_dt))
print("Accuracy on test Data: {:.3f}".format(acc_test_dt))
Result('Decision Tree',acc_train_dt,acc_test_dt)

**6.Comparison of models**
res =pd.DataFrame({'ML Model':Model,'Train Accuracy':train,'Test Accuracy':test})
res.sort_values(by=['Test Accuracy', 'Train Accuracy'], ascending=False)

pickle.dump(lr,open('Phishing_Website.pkl','wb'))
pickle.dump(dt,open('Phishing_Website_with_dt.pkl','wb'))
```

Deploying model on IBM cloud:

```
import requests
API_KEY = "q_K1mcpkf84GugtafInt0LujObrHOWfpky2l2YsSM0dL"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]
header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

payload_scoring= {"input_data":
 [{"field": [["UsingIP","LongURL","ShortURL","Symbol@","Redirecting//","PrefixSuffix-","SubDomains","HTTPS","DomainRegLen","Favicon","NonStdPort","HTTPSDomainURL","RequestURL","AnchorURL","LinksInScriptTags","ServerFormHandler","InfoEmail","AbnormalURL","WebsiteForwarding","StatusBarCust","DisableRightClick","UsingPopupWindow","IframeRedirection","AgeofDomain","DNSRecording","WebsiteTraffic","PageRank","GoogleIndex","LinksPointingToPage","StatsReport"]], "values":
[[1,1,1,1,1,-1,-1,-1,-1,1,1,1,1,1,-1,-1,1,1,1,1,0,1,1,1,1,-1,-1,-1,-1,1,1,0,1]]}]}
response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/004b714f-b450-4313-b4f7-fb1d5698e6c0/predictions?version=2022-11-18', json=payload_scoring,
 headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
predictions=response_scoring.json()
#print(predictions)
pred=print(predictions['predictions'][0]['values'][0][0])
if(pred != 1):
    print("Your are safe!!  This is a Legitimate Website.")
else:
    print("Your are on the wrong site. Be cautious!")
```


IBM Watson Studio
Search in your workspaces
Buy
Sharon Roseline S's Account
Dallas
SS
Deployments / models / models /

Web phishing deployment ● Deployed Online

API reference | Test

Direct link

Endpoint: https://us-south.ml.cloud.ibm.com/ml/v4/deployments/004b714f-b450-4313-b4f7-fb1d5698e6c0/predictions?ver

Bearer <token>: IAM

Code snippets

cURL
Java
JavaScript
Python
Scala

```
API_KEY = "<your API key>"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

# NOTE: manually define and pass the array(s) of values to be scored in the next line
payload_scoring = {"input_data": [{"fields": [array_of_input_fields], "values": [array_of_values_to_be_scored, another_array_of_val

response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/004b714f-b450-4313-b4f7-fb1d5698e6c0/predicti
 headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
```

Web phishing deployment

Created: Nov 18, 2022, 10:25 PM

Updated: Nov 18, 2022, 10:25 PM

Deployment ID: 004b714f-b450-4313-b4f7-fb1...

Software specification: runtime-22.1-py3.9

Copies: 1

Serving name: No serving name.

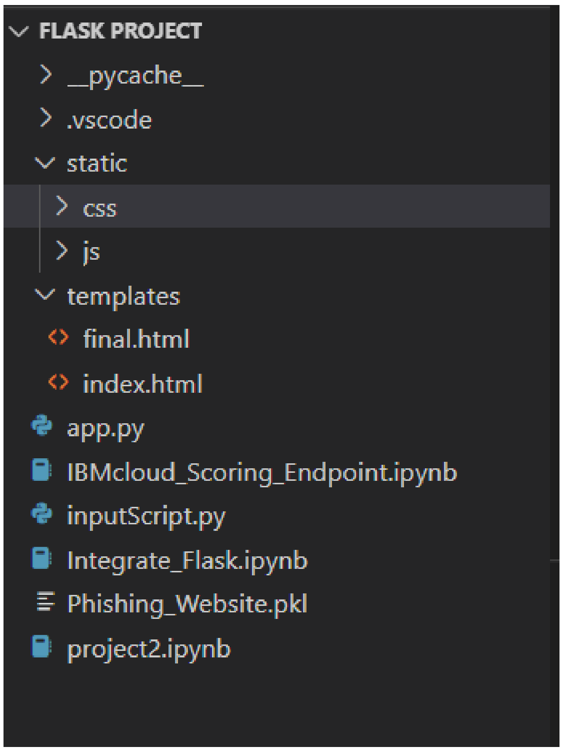
Description: No description provided.

Tags: Add tags to make assets easier to find.

Associated asset: models

APPLICATION BUILDING:

Folder Structure:



```
FLASK PROJECT
  > __pycache__
  > .vscode
  v static
    > css
    > js
  v templates
    final.html
    index.html
  app.py
  IBMcloud_Scoring_Endpoint.ipynb
  inputScript.py
  Integrate_Flask.ipynb
  Phishing_Website.pkl
  project2.ipynb
```

App.py

```
from flask import Flask, render_template,request,jsonify
import pickle
import inputScript
import numpy as np
app = Flask(__name__)
loaded_model = pickle.load(open("Phishing_Website.pkl", 'rb'))
@app.route('/')
def home():
   return render_template('index.html')

@app.route('/final')
def  final():

    return render_template("final.html")

@app.route('/y_predict',methods=['POST'])
def y_predict():
    url = request.form['URL']
    checkprediction = inputScript.main(url)
    print(checkprediction)
    prediction =loaded_model.predict(checkprediction)
    print(prediction)
    output=prediction[0]
    if(output==1):
        pred="Your are safe!!  This is a Legitimate Website."
    else:
        pred="Your are on the wrong site. Be cautious!"
    return render_template('final.html', prediction_text='{}'.format(pred),url=url)

@app.route('/predict_api',methods=['POST'])
def predict_api():
    '''
    For direct API calls trought request
    '''
    data = request.get_json(force=True)
    prediction = loaded_model.y_predict([np.array(list(data.values()))])

    output = prediction[0]
    return jsonify(output)
if __name__ == '__main__':
        app.run()
```

InputScript.py

```
import regex
from tldextract import extract
import ssl
import socket
from bs4 import BeautifulSoup
import urllib.request
import whois
import datetime
from googlesearch import search
import datetime
import requests
import favicon
import re
import parser
def url_having_ip(url):
```

```
#using regular function
    match=regex.search(
   '(([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-
4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\/)|'  #IPv4
                    '((0x[0-9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\.(0x[0-
9a-fA-F]{1,2})\\/)'  #IPv4 in hexadecimal
                    '(?:[a-fA-F0-9]{1,4}:){7}[a-fA-F0-9]{1,4}',url)     #Ipv6
    if match:
        #print match.group()
        return -1
    else:
        #print 'No matching pattern found'
        return 1


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


def url_short(url):

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

'yfrog\.com|migre\.me|ff\.im|tiny\.cc|url4\.eu|twit\.ac|su\.pr|twurl\.nl|snipurl\.com|'

'short\.to|BudURL\.com|ping\.fm|post\.ly|Just\.as|bkite\.com|snipr\.com|fic\.kr|loopt\.u
s|'

'doiop\.com|short\.ie|kl\.am|wp\.me|rubyurl\.com|om\.ly|to\.ly|bit\.do|t\.co|lnkd\.in|'

'db\.tt|qr\.ae|adf\.ly|goo\.gl|bitly\.com|cur\.lv|tinyurl\.com|ow\.ly|bit\.ly|ity\.im|'

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

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

```
    if match:
        return -1
    else:
        return 1

def having_at_symbol(url):
    symbol=regex.findall(r'@',url)
    if(len(symbol)==0):
        return -1
    else:
        return 1

def doubleSlash(url):
    for i in range(8,len(url)):
        if(url[i]=='/'):

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

def prefix_suffix(url):
    for i in range(8,len(url)):
        if(url[i]=='/'):

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

def sub_domain(url):
    subDomain, domain, suffix = extract(url)
    if(subDomain.count('.')==0):
        return -1
    elif(subDomain.count('.')==1):
        return 0
    else:
        return 1

def SSLfinal_State(url):
    try:
#check wheather contains https
        if(regex.search('^https',url)):
            usehttps = 1
        else:
            usehttps = 0
```

```
#getting the certificate issuer to later compare with trusted issuer
        #getting host name
        subDomain, domain, suffix = extract(url)
        host_name = domain + "." + suffix
        context = ssl.create_default_context()
        sct = context.wrap_socket(socket.socket(), server_hostname = host_name)
        sct.connect((host_name, 443))
        certificate = sct.getpeercert()
        issuer = dict(x[0] for x in certificate['issuer'])
        certificate_Auth = str(issuer['commonName'])
        certificate_Auth = certificate_Auth.split()
        if(certificate_Auth[0] == "Network" or certificate_Auth == "Deutsche"):
            certificate_Auth = certificate_Auth[0] + " " + certificate_Auth[1]
        else:
            certificate_Auth = certificate_Auth[0]
        trusted_Auth =
['Comodo','Symantec','GoDaddy','GlobalSign','DigiCert','StartCom','Entrust','Verizon','
Trustwave','Unizeto','Buypass','QuoVadis','Deutsche Telekom','Network
Solutions','SwissSign','IdenTrust','Secom','TWCA','GeoTrust','Thawte','Doster','VeriSi
gn']
#getting age of certificate
        startingDate = str(certificate['notBefore'])
        endingDate = str(certificate['notAfter'])
        startingYear = int(startingDate.split()[3])
        endingYear = int(endingDate.split()[3])
        Age_of_certificate = endingYear-startingYear

#checking final conditions
        if((usehttps==1) and (certificate_Auth in trusted_Auth) and
(Age_of_certificate>=1) ):
            return -1 #legitimate
        elif((usehttps==1) and (certificate_Auth not in trusted_Auth)):
            return 0 #suspicious
        else:
            return 1 #phishing

    except Exception as e:

        return 1

def domain_registration(url):
    try:
        w = whois.whois(url)
        updated = w.updated_date
        exp = w.expiration_date
        length = (exp[0]-updated[0]).days
        if(length<=365):
            return 1
        else:
            return -1
    except:
        return 0

def favicon(url):
    subDomain, domain, suffix = extract(url)
    b=domain
    try:
        icons = favicon.get(url)
        icon = icons[0]
        subDomain, domain, suffix =extract(icon.url)
        a=domain
        if(a==b):
            return 1
        else:
            return -1
    except:
        return -1

def port(url):
    #ongoing
    return 0

def https_token(url):
    subDomain, domain, suffix = extract(url)
    host =subDomain +'.' + domain + '.' + suffix
    if(host.count('https')): #attacker can trick by putting https in domain part
        return 1
    else:
        return -1

def request_url(url):
    try:
        subDomain, domain, suffix = extract(url)
        websiteDomain = domain

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

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

        for video in vids:
            subDomain, domain, suffix = extract(video['src'])
            vidDomain = domain
            if(websiteDomain==vidDomain or vidDomain==''):
                linked_to_same = linked_to_same + 1
        linked_outside = total-linked_to_same
        if(total!=0):
            avg = linked_outside/total

        if(avg<0.22):
            return -1
        elif(0.22<=avg<=0.61):
            return 0
        else:
            return 1
    except:
        return 0


def url_of_anchor(url):
    try:
        subDomain, domain, suffix = extract(url)
        websiteDomain = domain

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

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

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

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

        if(avg<0.25):
            return -1
        elif(0.25<=avg<=0.81):
            return 0
        else:
            return 1
    except:
        return 0

def sfh(url):
    #ongoing
    return -1

def email_submit(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        if(soup.find('mailto:')):
            return 1
        else:
            return -1
    except:
        return 0

def abnormal_url(url):
    subDomain, domain, suffix = extract(url)
    try:
        domain = whois.whois(url)
        hostname=domain.domain_name[0].lower()
        match=re.search(hostname,url)
        if match:
            return 1
        else:
            return -1
    except:
        return -1

def redirect(url):
    try:
        request = requests.get(url)
        a=request.history
        if(len(a)<=1):
            return 1
        else:
            return 0

    except:
        return 0
```

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

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

def rightClick(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        if(soup.find_all('script',mousedown=True)):
            return -1
        else:
            return 1
    except:
        return -1

def popup(url):
    #ongoing
    return 1

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

```
def age_of_domain(url):
    try:
        w = whois.whois(url)
        start_date = w.creation_date
        current_date = datetime.datetime.now()
        age =(current_date-start_date[0]).days
        if(age>=180):
            return -1
        else:
            return 1
    except Exception as e:
        print(e)
        return 0

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

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

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

def page_rank(url):
    #ongoing
    return 1
```

```
def google_index(url):
    try:
        subDomain, domain, suffix = extract(url)
        a=domain + '.' + suffix
        query = url
        for j in search(query, tld="co.in", num=5, stop=5, pause=2):
            subDomain, domain, suffix = extract(j)
            b=domain + '.' + suffix
        if(a==b):
            return 1
        else:
            return -1
    except:
        return -1


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

def statistical(url):
    hostname = url
    h = [(x.start(0), x.end(0)) for x in
regex.finditer('https://|http://|www.|https://www.|http://www.', hostname)]
    z = int(len(h))
    if z != 0:
        y = h[0][1]
        hostname = hostname[y:]
        h = [(x.start(0), x.end(0)) for x in regex.finditer('/', hostname)]
        z = int(len(h))
        if z != 0:
            hostname = hostname[:h[0][0]]
```

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

ip_match=regex.search('146\.112\.61\.108|213\.174\.157\.151|121\.50\.168\.88|192\.
185\.217\.116|78\.46\.211\.158|181\.174\.165\.13|46\.242\.145\.103|121\.50\.168\.40|
83\.125\.22\.219|46\.242\.145\.98|107\.151\.148\.44|107\.151\.148\.107|64\.70\.19\.2
03|199\.184\.144\.27|107\.151\.148\.108|107\.151\.148\.109|119\.28\.52\.61|54\.83\.4
3\.69|52\.69\.166\.231|216\.58\.192\.225|118\.184\.25\.86|67\.208\.74\.71|23\.253\.12
6\.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|1
03\.243\.24\.98|199\.59\.243\.120|31\.170\.160\.61|213\.19\.128\.77|62\.113\.226\.13
1|208\.100\.26\.234|195\.16\.127\.102|195\.16\.127\.157|34\.196\.13\.28|103\.224\.21
2\.222|172\.217\.4\.225|54\.72\.9\.51|192\.64\.147\.141|198\.200\.56\.183|23\.253\.16
4\.103|52\.48\.191\.26|52\.214\.197\.72|87\.98\.255\.18|209\.99\.17\.27|216\.38\.62\.1
8|104\.130\.124\.96|47\.89\.58\.141|78\.46\.211\.158|54\.86\.225\.156|54\.82\.156\.19
|37\.157\.192\.102|204\.11\.56\.48|110\.34\.231\.42',ip_address)
    except:
        return -1

    if url_match:
        return -1
    else:
        return 1

def main(url):

    check = [[url_having_ip(url),url_length(url),url_short(url),having_at_symbol(url),
            doubleSlash(url),prefix_suffix(url),sub_domain(url),SSLfinal_State(url),

domain_registration(url),favicon(url),port(url),https_token(url),request_url(url),

url_of_anchor(url),Links_in_tags(url),sfh(url),email_submit(url),abnormal_url(url),
             redirect(url),on_mouseover(url),rightClick(url),popup(url),iframe(url),

age_of_domain(url),dns(url),web_traffic(url),page_rank(url),google_index(url),
             links_pointing(url),statistical(url)]]


    print(check)
    return check
```

HTML Files:

Index.html

```
<html>
    <head>
        <title>Web Phishing</title>
        <link rel="stylesheet" href="{{url_for('static', filename='css/style.css')}}">
        <script>
            function myFunction()
            {
                window.location.href="final";
            }
        </script>
    </head>
    <body>
        <div class="banner">

            <div class="navbar">
                <h3 class="logo">Web Phishing</h3>
                <ul>
                    <li><a href="#">Home</a></li>
                    <li><a href="#">About</a></li>
                    <li><a href="#">Contact</a></li>
                </ul>
            </div>
            <div class="content">
                <h1 style="font-size:50px;">WEB PHISHING PREDICTION</h1>
                <p>Welcome! Our Website uses ML model to mimic how a person would
                    look at, understand, and draw a verdict on a suspicious website.
                    <br>Our engine learns from high quality, proprietary datasets containing
                     millions of  samples for high accuracy detection.
                     <br>We detect whether the website is legit or not in seconds, not days
                     or weeks like other blocklist-based phishing protection softwares</p>
                <h1 style="font-size:50px;">ABOUT</h1>
                <P>Web service is one of the key communications software services for the
internet.
                Web phishing is one of many security threats to we services on the
internet.<br>
                Web phishing aims to steal private information such os usernames,
passwords, and credit card details,
                by way of impersonating a legutumate entity.<br> The recipient is then
tricked information clicking
```

```
                malicious link, which can lead to the installation of malware,<br> the freezing
of the system as part of
                a ransonware attack or the revealing of the sensitive information. It will lead
to information disclosure and property damage.
                </P><br>
                <h3> CHECK YOUR WEBSITE</h3>
                <P>Understanding if the website is a valid one or not is important and plays
a vital role in securing the data.<br>
                    To know if the URL is a valid one or your information is at risk check your
website.
                </P>
                <div>
                    <button   type="button"   onclick="myFunction()"><span></span>Predict
Your Website</button>
                </div>
            </div>
        </div>
    </body>
</html>
```

Final.html

```
<html>
    <head>
        <title>Check Your Website</title>
        <link rel="stylesheet" href="{{url_for('static', filename='css/style.css')}}">
    </head>
    <body>

        <div class="predict-page">
            <h1>Check Your Website</h1>
            <form action="{{ url_for('y_predict') }}" method="post">
                <p>Enter URL</p>
                <input type="text" name="URL" placeholder="URL">
                <a href= {{ url }} >
                <button  style="right:  -20%;  top:  10px;"  type="submit"><span></span>
Predict </button>
            </a>
            </form>
            <div style="text-align: center ;">
                <div  id='result',  style="color:  green;padding-top:  2rem;font-size:  2.2rem;"
font-size:30px;>{{ prediction_text }}</div>
```

```html
                <a href=" {{ url }} "> {{ url }} </a>
              </div>
        </div>
    </body>
</html>
```

CSS File:

Style.css

```css
*{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
}
body{
    width: 100%;
    height: 100vh;
    background-image: linear-gradient(rgba(0,0,0,0.75),rgba(0,0,0,0.75)),url(bg.jpg);
    background-size: cover;
    background-position: center;
}

.banner{
    width: 100%;
    height: 100vh;
    background-image: linear-gradient(rgba(0,0,0,0.75),rgba(0,0,0,0.75)),url(bg.jpg);
    background-size: cover;
    background-position: center;
}

.navbar{
    width: 85%;
    margin: auto;
    padding: 35px 0;
    display: flex;
    align-items: center;
    justify-content: space-between;
}
```

```
.logo{
   width: 120px;
   cursor: pointer;
}

.navbar ul li{
   list-style: none;
   display: inline-block;
   margin: 0 20px;
   position: relative;
}

.navbar ul li a{
   text-decoration: none;
   color: white;
   text-transform: uppercase;
}

.navbar ul li::after{
   content: "";
   height: 3px;
   width: 0;
   background: red;
   position: absolute;
   left: 0;
   bottom: -10px;
   transition: 0.3s;
}

.navbar ul li:hover::after{
   width: 100%;
}

.content{
   width: 100%;
   position: absolute;
   top: 55%;
   transform: translateY(-50%);
   text-align: center;
   color: white;
}

.content p{
   margin: 20px auto;
   font-weight: 100;
   line-height: 25px;
}
button{
   width: 200px;
   padding: 15px 0;
   text-align: center;
   margin: 20px 10px;
   border-radius: 25px;
   font-weight: bold;
   border: 2px solid red;
   background: transparent;
   color: white;
   cursor: pointer;
   position: relative;
   overflow: hidden;
}

span{
   background: red;
   height: 100%;
   width: 0;
   border-radius: 25px;
   position: absolute;
   left: 0;
   bottom: 0;
   z-index: -1;
   transition: 0.3s;
}

button:hover span{
   width: 100%;
}

button:hover{
   border: none;
}

/*******************************************/

.predict-page{
   width: 350px;
   top: 35%;
   left: 50%;
   transform: translate(-50%,-50%);
   position: absolute;
   color: white;
}

.predict-page h1{
   font-size: 40px;
   text-align: center;
   text-transform: uppercase;
   margin: 40px 0;
}
.predict-page p{
   font-size:  20px;
   margin: 15px 0;
}
.predict-page input{
   font-size: 16px;
   width: 100%;
   padding: 15px 10px;
   border: 0;
   outline: none;
   border-radius: 5px;
}
```

13.2 SOURCE CODE & PROJECT DEMO LINK

DEMO LINK:

https://drive.google.com/drive/folders/1Iec2LuwAXqv

09r6uj-7HfGmu4-SJoIG7?usp=sharing

GITHUB:

https://github.com/IBM-EPBL/IBM-Project-50363-1660904796