WEB PHISHING DETECTION

PROJECT REPORT

Submitted by

SANTHOSH J	**LEADER**
SARVESHKUMAR S	**MEMBER 1**
SARATHKUMAR R	**MEMBER 2**
SARAVANAN B	**MEMBER 3**

in partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

BHARATHIYAR INSTITUTE OF ENGINEERING FOR WOMEN

DEVIYAKURICHI-636 112

ANNA UNIVERSITY : CHENNAI – 600 025

NOV- 2022

ABSTRACT

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

Common threats of web phishing:

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

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

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

1.INTRODUCTION

1.1 Project Overview:

Phishing emails are a routine occurrence for anyone with email in the Internet age. Masking themselves as reputable companies such as using devious means such as the use of company logos and standards, malicious actors, again and again, attempt to lure in individuals from a wide range of technical shades. Phishing ranges extensively in sophistication: from mass-produced misspelt requests for overly specific details to sophisticated spear phishing attacks focused on the details of the individual. The ability of phishing attacks to innocuously harvest your private credentials can leave you mercilessly exposed in our data-intensive world. All it takes is for a user to make the critical mistake of clicking on a single malicious link. Through a simple mistake, a user exposes themselves and their data from anything from drive-by downloads, cross-site scripting attacks to the harvesting of their details in an innocuous web form. Phishing has two main delivery vehicles: emails and websites. Emails being the foremost of these, are most classically associated with phishing. These often include malicious URLs to direct users towards maliciously crafted content. By obfuscating the real destination of a URL through a few simple manipulations, users can quickly find themselves on unknown and insecure ground. Therefore it is vital to tackle this massive worldwide problem. In the United Kingdom (UK) alone, phishing is expected to cost the UK economy as much as £280 million per year. This is encouraging companies such as Google to look into the future of Uniform Resource Locators (URLs) themselves. To tackle the problem of phishing, my project has been focused on tackling the malicious URLs included in them as ―more than 75% of phishing mails include malicious URLs to phishing sites‖. Existing techniques to handle URLs involve automated phishing detecting (mainly employing machine learning techniques), user training (the best results of which are gained from embedded training) and automated security indicators (providing information to help the users decide). I aim to create a system which incorporates aspects of these techniques, to inform and protect users from malicious.

1.2 Purpose

To solve this problem, I have been building a user-focused tool which seeks to catch problematic URLs before they reach their full malicious potential. The tool that I have developed is a Phishing Learning and Detection tool, built for the Chrome platform in the form of an extension called Catch-Phish. It classifies URLs into one of three safety states using a combination of natural language processing and knowledge acquisition, before providing users with the necessary information to understand how this classification was derived. 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. This is important due to the significant average availability time of phishing links, which according to Canova et al. [11], was 32 hours and 32 minutes in the first half of 2014. Machine learning techniques arevery 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 they train end users [60]. For these reasons, it is important to train users to detected phishing themselves. As the first year of my Mini project, the focus this year was working on both designing the tool and implementing and evaluating the means for how the user would interact with the tool. The project has involved a welcome and generous amount of advice from a PhD student who is an expert in phishing, computer security and URLs and is working on a similar project. This student has produced alot of the research referenced in this report. However, I make a clear distinction between the work doneby myself and this student throughout the report. 1.2.1 Results and Accomplishment

URLs of benign websites were collected from www.alexa.com and The URLs of phishing websites were collected from www.phishtank.com. The data set consists of total 36,711 URLs which include 17058 benign URLs and 19653 phishing URLs. Benign URLs are labeled as ―0‖ and phishing
URLs are labeled.

2. LITERATURE SURVEY

2.1. Existing Problem:

According to this paper we people are highly dependent on the internet. For performing online shopping and online activities like banking, mobile recharge and more activities are done only through internet. Here phishing is nothing but a type of website threat which illegally collects the original websiteinformation such as login id, password and credit card information. Here we will use an efficient machine learning based web phishing detection technique

Problem Identification

There are many users who purchase products through online platform and the payment is done through ebanking.

There are some fake banking websites in which they collect the more sensitive information like username, password, credit card details etc , for illegal purpose.

This type of websites are called phishing website.

Here web phishing is one of the security threat to webservices on the internet.

Problem Solution

To overcome the problem of phishing website whenever we are clicking on one website it must show an alert box like it is a secure website or it is not a secure website.

Then another way is that we can scan the website in order to prevent our system or mobile from the phishing attack.

Even though technologies are there we as the user have to be aware of the websites whether it is secure or not. We should not click any unwanted websites.

2.2. Reference:

[1] Higashino, M., et al. An Anti-phishing Training System for Security Awareness and Education Considering Prevention of Information Leakage. in 2019 5th International Conference on Information Management (ICIM). 2019.

[2] H. Bleau, Global Fraud and Cybercrime Forecast,. 2017.

[3] Michel Lange, V., et al., Planning and production of grammatical and lexical verbs in multi-word messages. PloS one, 2017. 12(11): p. e0186685-e018668

2.3. Problem Statement Definition:

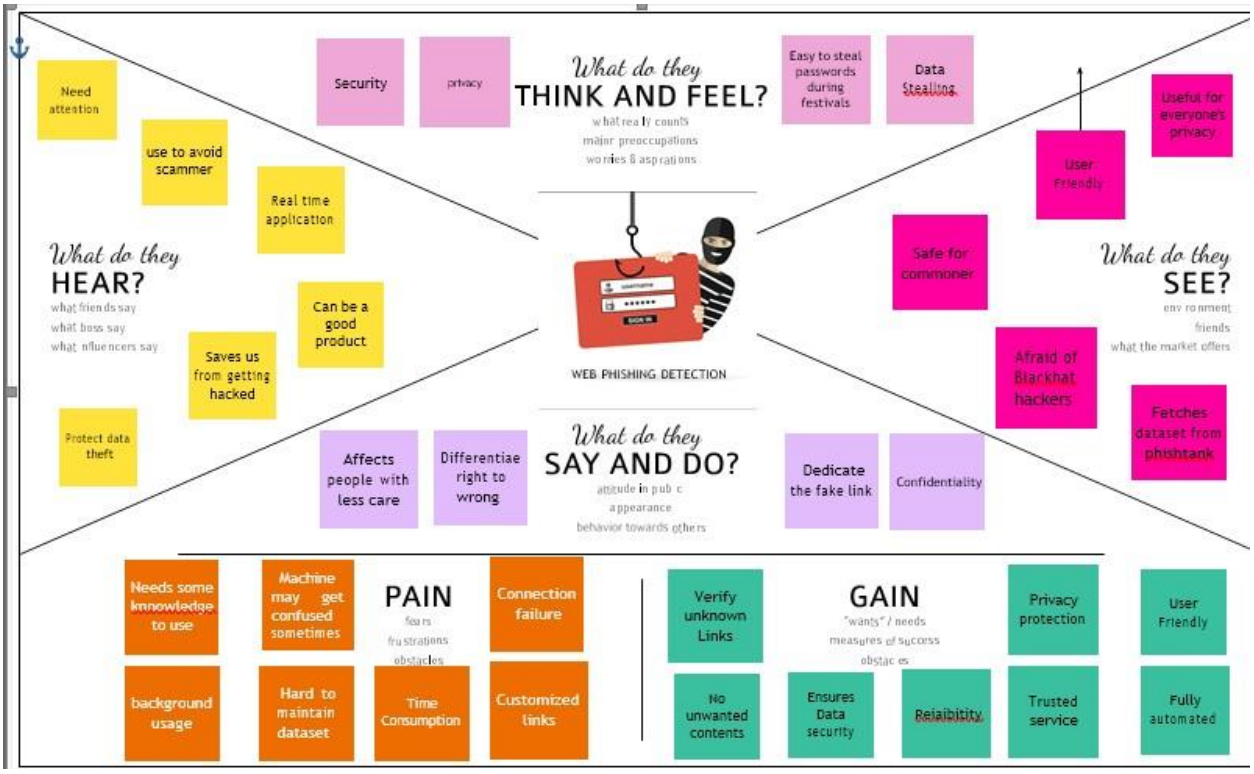
Phishing is a fraudulent technique that uses social and technological tricks to steal customer identification and financial credentials . Social media systems use spoofed e-mails from legitimate companies and agencies to enable users to use fake websites to divulge financial details like usernames and passwords. Hackers install malicious software on computers to steal credentials, often using systems to intercept username and passwords of consumers' online accounts. Phishers use multiple methods, including email, Uniform Resource Locators (URL), instant messages, forum postings, telephone calls, and text messages to steal user information. The structure of phishing content is similar to the original content and trick users to access the content in order to obtain their sensitive data. The primary objective of phishing is to gain certain personal information for financial gain or use of identity theft. Phishing attacks are causing

severe economic damage around the world. Moreover, Most phishing attacks target financial/payment institutions and webmail, according to the Anti-Phishing Working Group (APWG) latest Phishing pattern studies.

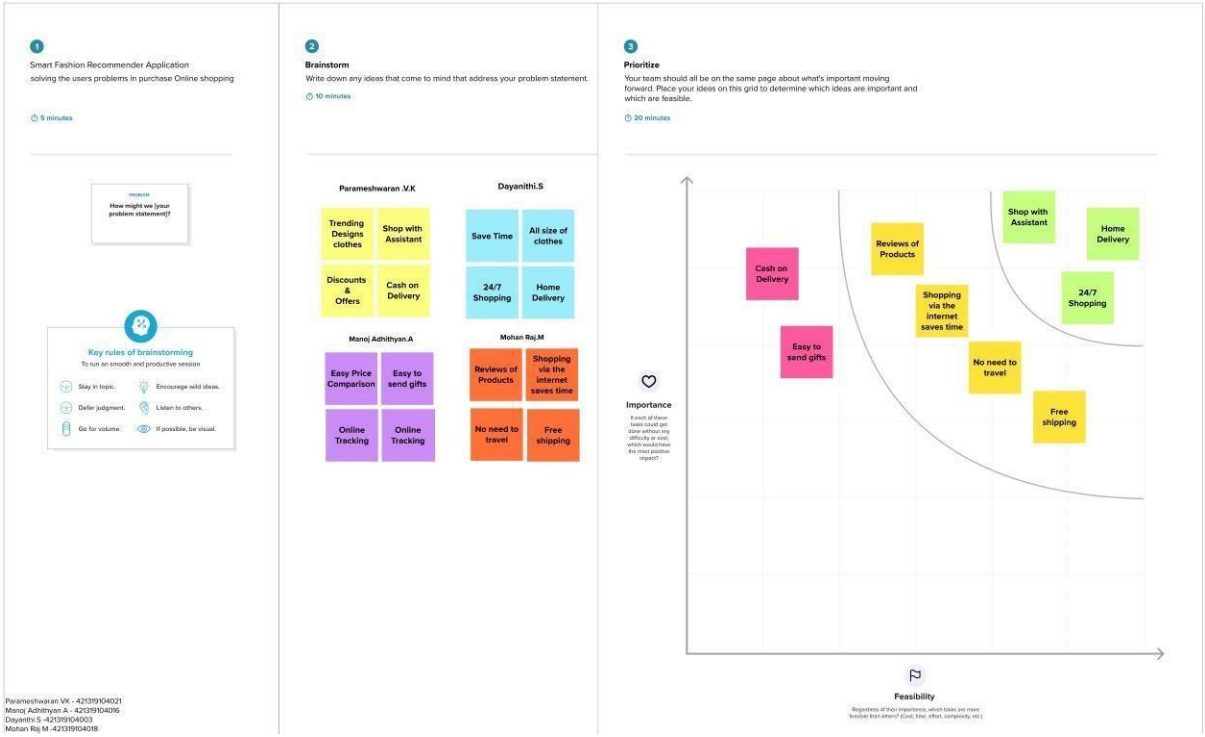
Problem Statement(PS)	I am	I'm trying to	But	Because	Which makes me feel
PS – 1	Internet user	Browse the internet	I identify a scam	An attacker masquerades as a reputable entity	Unsafe about my information that is shared over the network
PS - 2	Enterprise user	Open emails in the cloud server	I detect malicious protocols	They are not cryptographically signed	Emails are unverified and third party intrusion

3. IDEATION AND PROPOSED SOLUTION

3.1. Empathy Map Canvas



3.2. Ideation & Brainstorming:



3.3. Proposed Solution:

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

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Web phishing tends to steal a lots of information from the user during online transaction like username, password, important documents that has been attached to that websites. There are Multiple Types of Attacks happens here every day, but there is no auto detection Process through Machine Learning is achieved
2.	Idea / Solution description	Through ML and data mining techniques like classification algorithm user can able to attain a warning signal to notify these phishing websites which helps the user to safeguard their identities and their login credentials etc. python is the language that helps to enable these techniques for the online users

3.	Novelty / Uniqueness	This project not only able to identify the malicious websites it also has the ability to automatically block these kind of websites completely in the future when it has been identified and also blocks some various mails /ads from these malicious websites
4.	Social Impact / Customer Satisfaction	This web phishing detection project attains the customer satisfaction by discarding variouskinds of malicious websites to protect their privacy. This project is not only capable of using by an single individual ,a large social community and a organisation can use this web phishing detection to protect their privacy. This project helps to block various malicious websites simultaneously.
5.	Business Model (Revenue Model)	This developed model can be used as anenterprise applications by organisations which handles sensitive information and also can be sold to government agencies to prevent the loss of potential important data.
6.	Scalability of the Solution	This project's performance rate will be high and it also provide many capabilities to the user without reducing its efficieny to detect the malicious websites. thus scalability of this project will be high .

3.4. Problem Solution Fit:

Identify strong TR & EM	**4. EMOTIONS: BEFORE / AFTER** When the customers do not use phishing detection websites they will be in the fear of the details getting leaked, scare of the money in bank account getting looted. Once they start using phishing detection websites they will be confident in providing the details.	EM	There will be no problem when the customer is offline as they can't use any website when they go offline.	Identify strong TR & EM

4. REQUIRMENT ANALYSIS

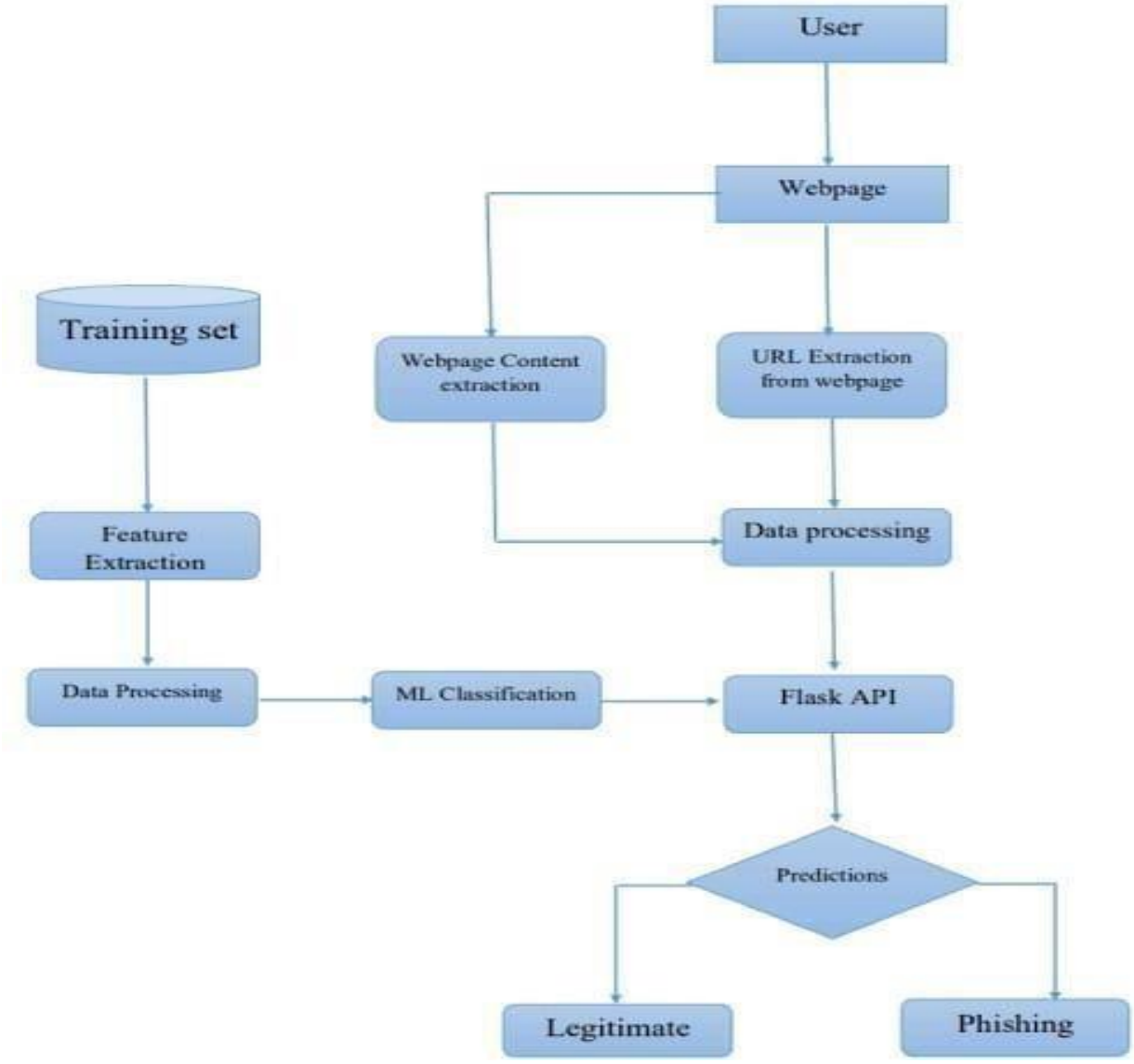
4.1. Functional Requirement

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 and Whitelist 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 produces final result.
FR-6	Announcement	Model then displays whether website is a legal site or a phishing site.
FR-7	Events	This model needs the capability of retrieving and displaying accurate result for a website

5. PROJECT DESIGN

5.1 Data Flow Diagram:

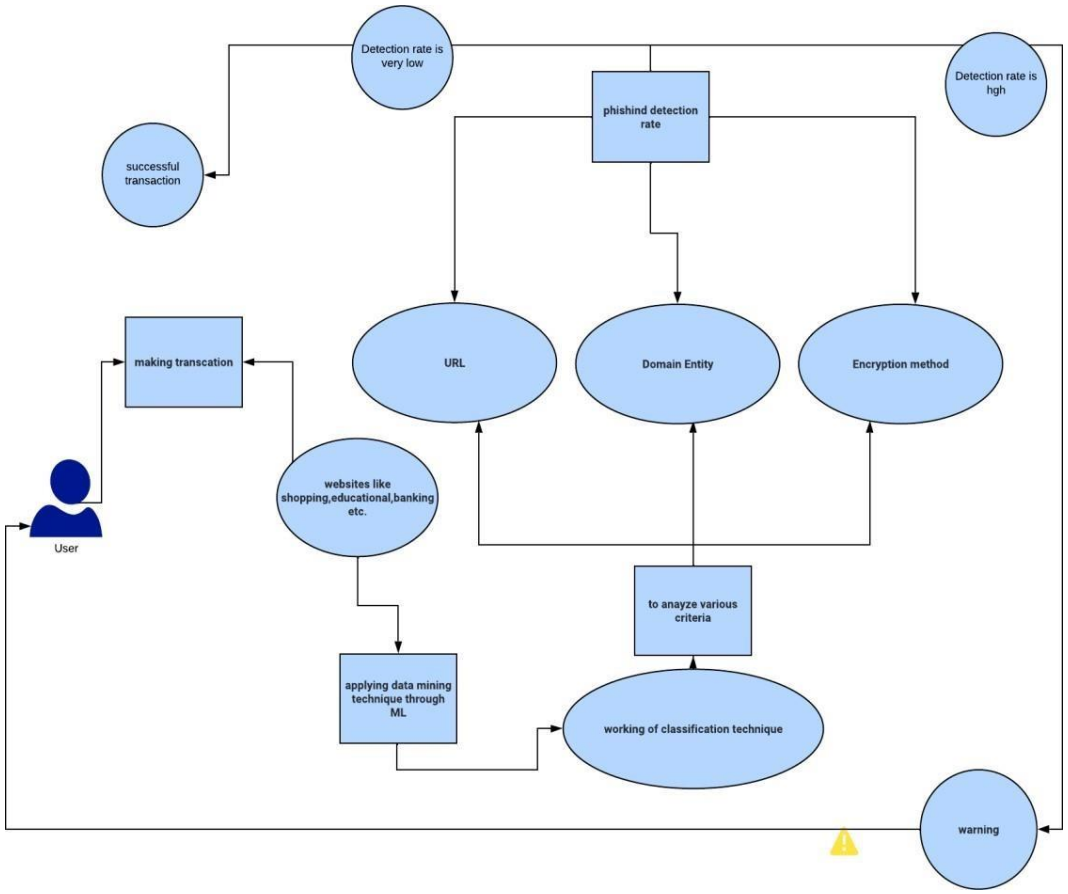
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It showshow data enters and leaves the system, what changes the information, and where data is stored.



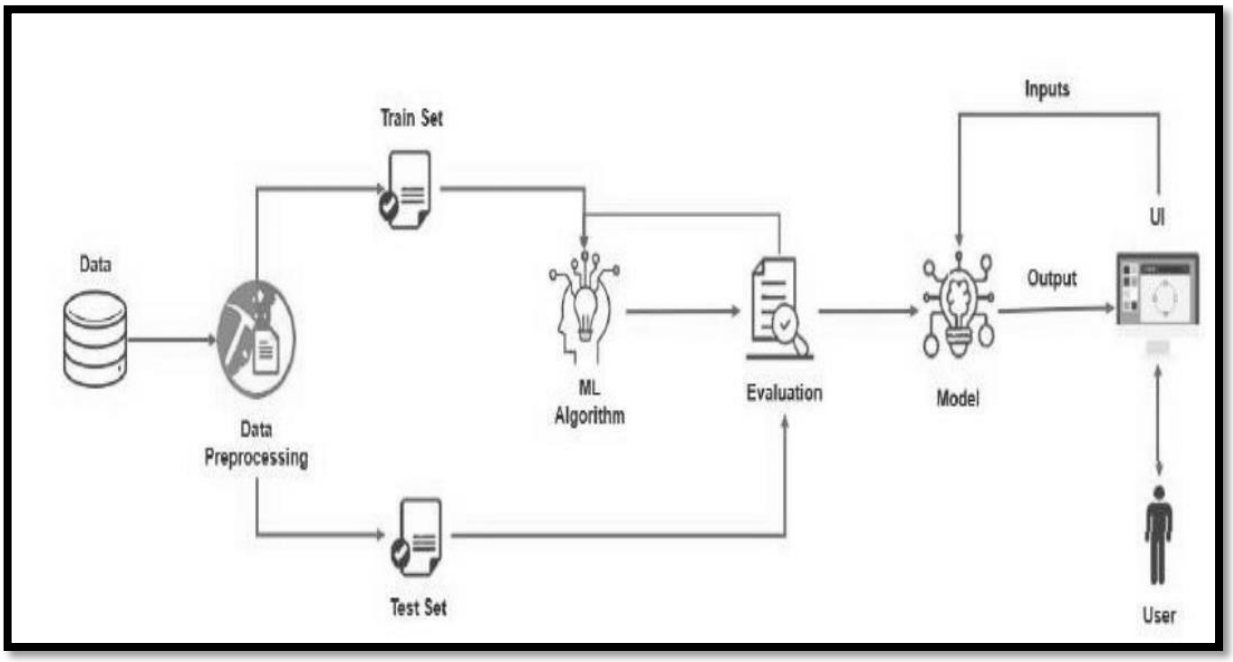
5.2 Solution Architecture

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.



Technical Architecture:



5.3 User Stories:

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through		Medium	Sprint-1

			Gmail			
	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 Logistic Regression, KNN	In this i can have correct prediction on the particular algorithms	High	Sprint-1
	Classifier	USN-2	Here i will send all the model output to classifier in order to produce final result.	I this i will find the correct classifier for producing the result	Medium	Sprint-2

6. PROJECT PLANNING & SCHEDULING

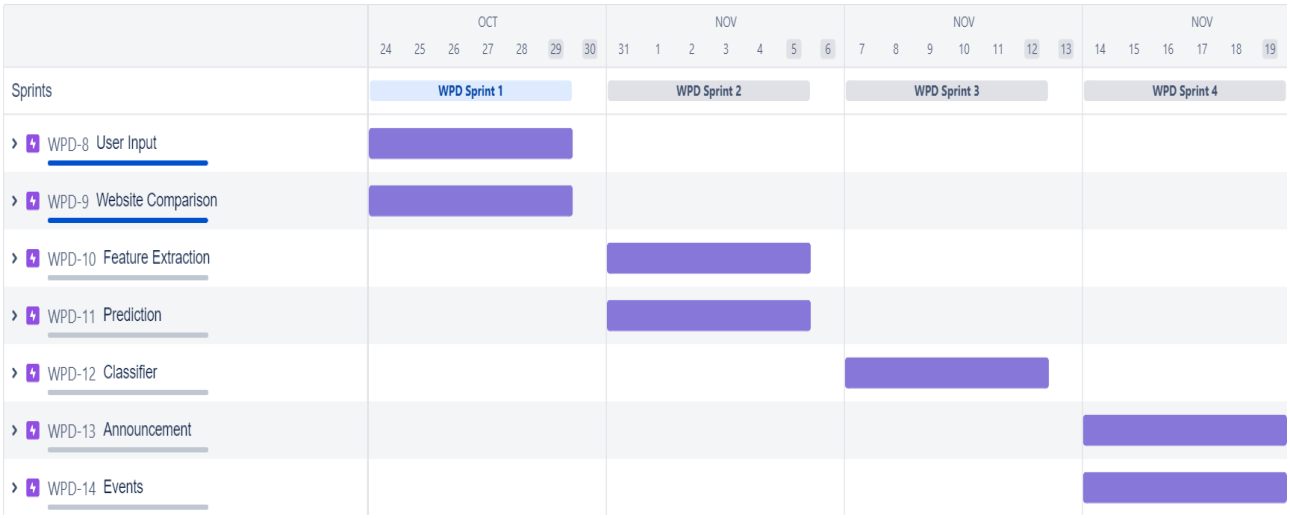
6.1 Sprint Planning & Estimation:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	User Input	USN-1	User inputs an URL in the required field to check its validation	1	Medium	Nivethini.R
Sprint-1	Website Comparison	USN-2	Model compares the websites using Blacklist and Whitelist approach	1	High	Megala.V
Sprint-2	Feature Extraction	USN-3	After comparison, if none foundon comparison then it extracts feature using heuristic and visual similarity	2	High	Ruthika.P
Sprint-2	Prediction	USN-4	Model predicts the URL using Machine learning algorithms such as logistic Regression, KNN.	1	Medium	Nivethini.R
Sprint-3	Classifier	USN-5	Model then displays whether the website is legal site or a phishing site.	1	Medium	Megala.V
Sprint-4	Announcement	USN-6	Model then displays whether the website is legal site or a phishing site	1	High	Ruthika.P
Sprint-4	Events	USN-7	This model needs the capability of retrieving and displaying accurate result for a website.	1	High	Nivethini.R

6.2 Sprint Delivery Schedule:

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

6.3 Reports from JIRA:



7.CODING & SOLUTIONING

7.1 Feature 1

· 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.

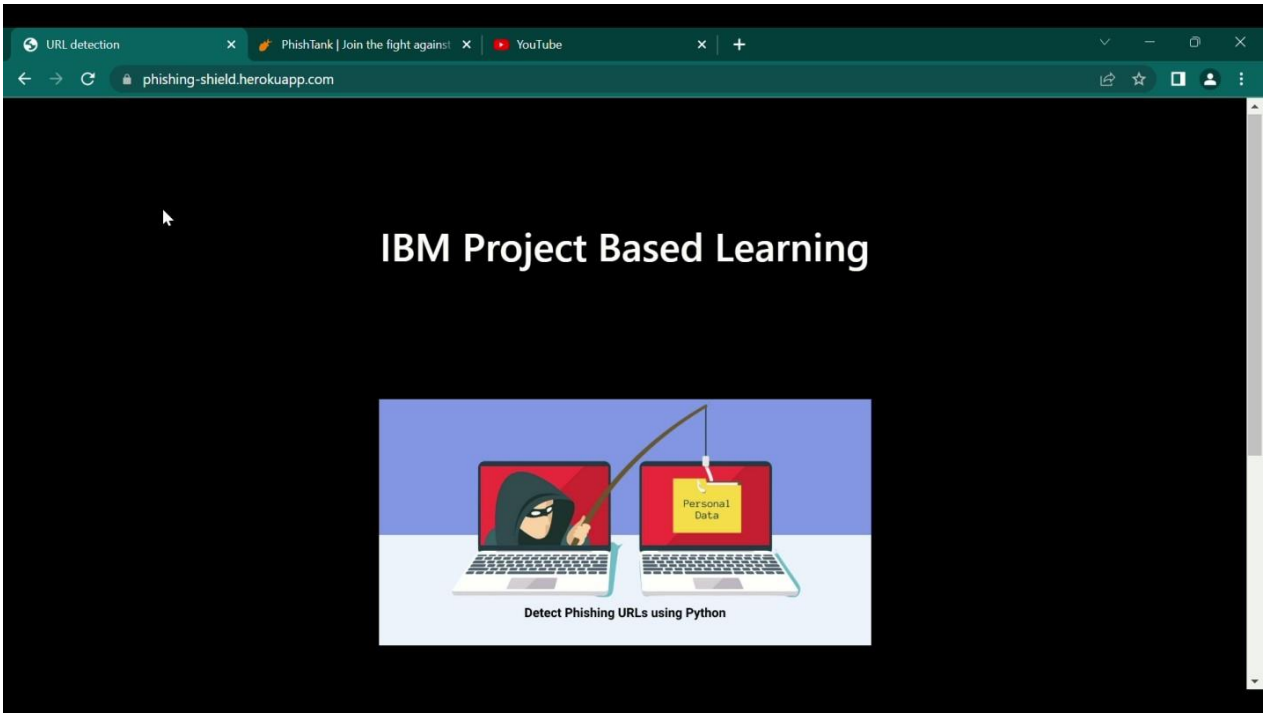
REGISTERATION:

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body {
font-family: Arial, Helvetica, sans-serif;
background-color: black;
}
* {
box-sizing: border-box;
}
/* Add padding to containers */
.container {
padding: 16px;
background-color: white;
}
/* Full-width input fields */
input[type=text], input[type=password] {
width: 100%;
padding: 15px;
margin: 5px 0 22px 0;
display: inline-block;
border: none;
background: #f1f1f1;
}
```

```
input[type=text]:focus, input[type=password]:focus {
background-color: #ddd
outline: none;
}
/* Overwrite default styles of hr */
hr {
border: 1px solid #f1f1f1;
margin-bottom: 25px;
}
/* Set a style for the submit button */
.registerbtn {
background-color: #04AA6D;
color: white;
padding: 16px 20px;
margin: 8px 0;
border:  none;
cursor: pointer;
width: 100%;
opacity: 0.9;
}
.registerbtn:hover {
opacity: 1;
}
/* Add a blue text color to links */
a {
color: dodgerblue;
}
/* Set a grey background color and center the text of the "sign in" section */
.signin {
background-color: #f1f1f1;
text-align: center;
}
</style>
</head>
<body>
<form action="success.html" method="POST">
<div class="container">
<h1>Register</h1>
```

```
<p>Please fill in this form to create an account.</p>
<hr>
<label for="email"><b>Email</b></label>
<input type="text" placeholder="Enter Email" name="email" id="email" required>
<label for="psw"><b>Password</b></label>
<input type="password" placeholder="Enter Password" name="psw" id="psw"
required>
<button type="submit" class="registerbtn">Register</button>
</div>
</form>
</body>
</html
```

SOLUTIONING:



Feature 2:

· Accuracy of an URL

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

Coding:

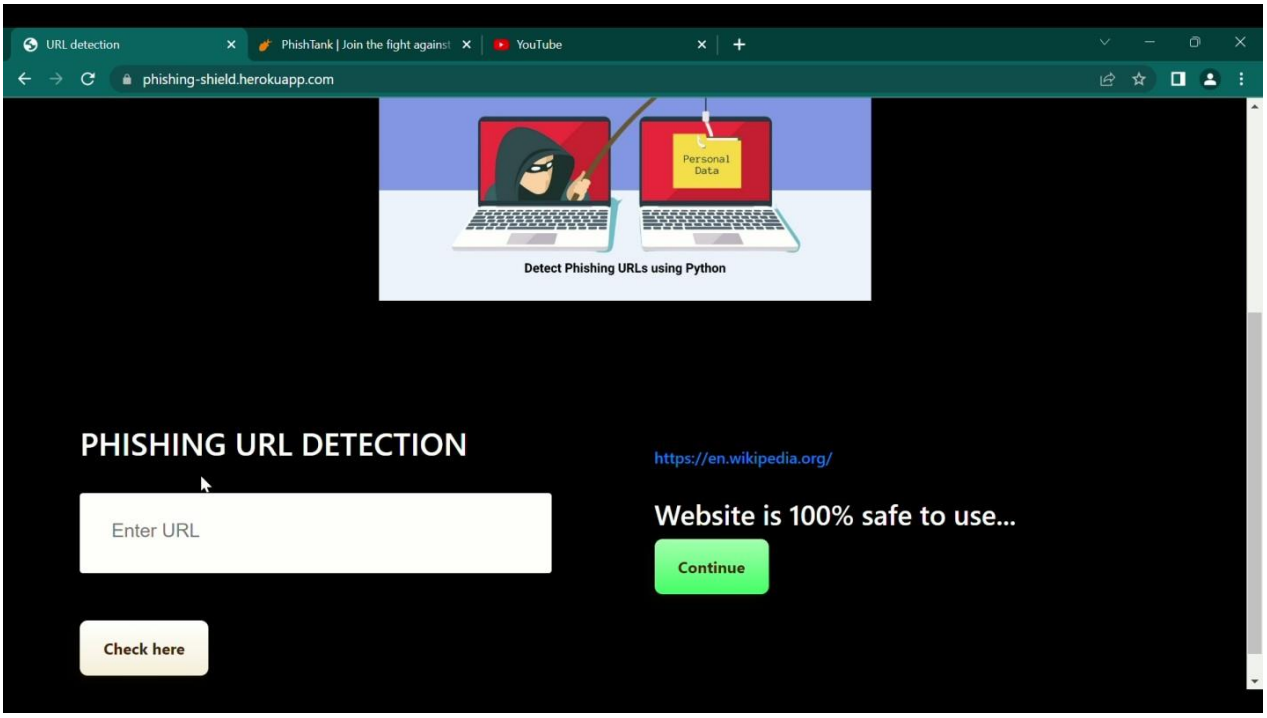
```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body {
font-family: Arial, Helvetica, sans-serif;
background-color: black;
}
* {
box-sizing: border-box;
}
/* Add padding to containers */
.container {
padding: 16px;
background-color: white;
}
/* Full-width input fields */
input[type=text], input[type=password] {
width: 100%;
padding: 15px;
margin: 5px 0 22px 0;
display: inline-block;
border: none;
background: #f1f1f1;
}
input[type=text]:focus, input[type=password]:focus {
background-color: #ddd;
outline: none;
```

```
}
/* Overwrite default styles of hr */
hr {
border: 1px solid #f1f1f1;
margin-bottom: 25px;
}
/* Set a style for the submit button */
.registerbtn {
background-color: blue;
color: white;
padding: 16px 20px;
margin: 8px 0;
border:  none;
cursor: pointer;
width: 100%;
opacity: 0.9;
}
.registerbtn:hover {
opacity: 1;
}
/* Add a blue text color to links */
a {
color: dodgerblue;
}
/* Set a grey background color and center the text of the "sign in" section */
.signin {
background-color: #f1f1f1;
text-align: center;

}
</style>
</head>
<body>
<form name=form action='/form_login' method="POST">
<div class="container">
<h1>Login</h1>
<p>Welcome back!!!</p>
<hr>
```

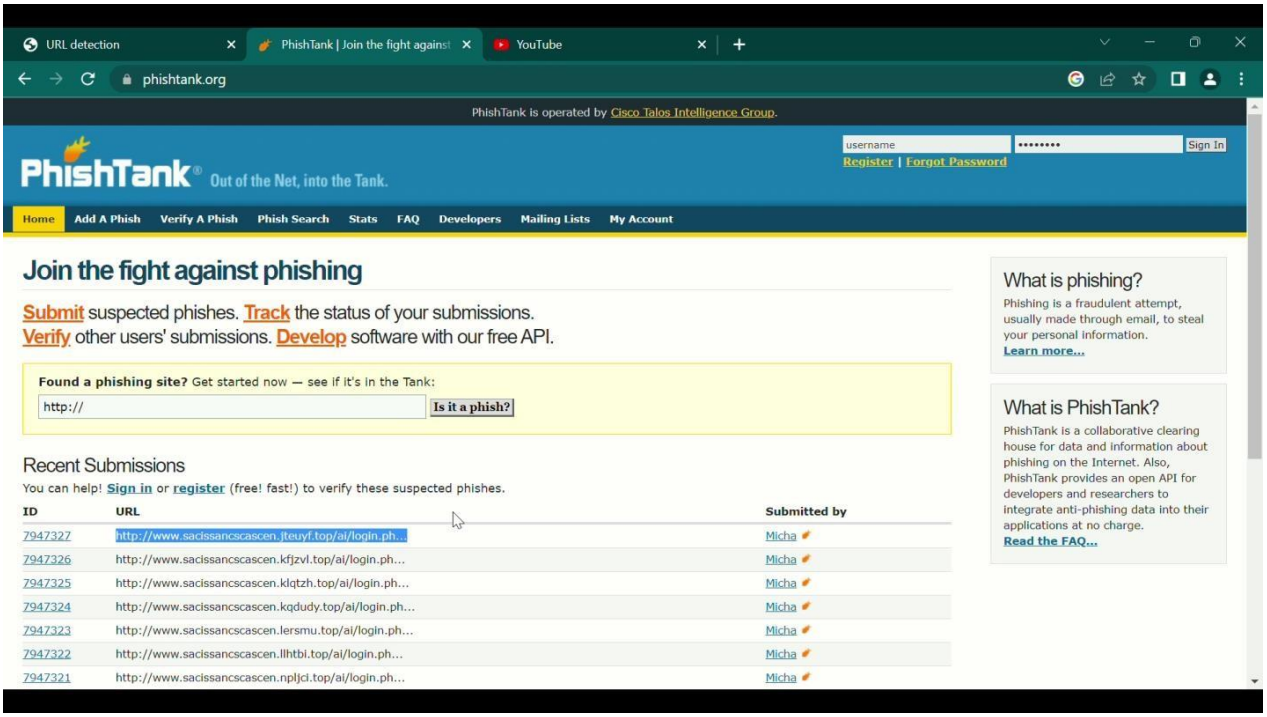
```
<label for="email"><b>Email</b></label>
<input type="text" placeholder="Enter Email" name="email" id="email" required>
<label for="psw"><b>Password</b></label>
<input type="password" placeholder="Enter Password" name="psw" id="psw"
required>
<button type="submit" class="registerbtn">LOGIN</button>
</div>
</form>
<h2><center>{{info}}</center></h2>
</body>
</html>
```

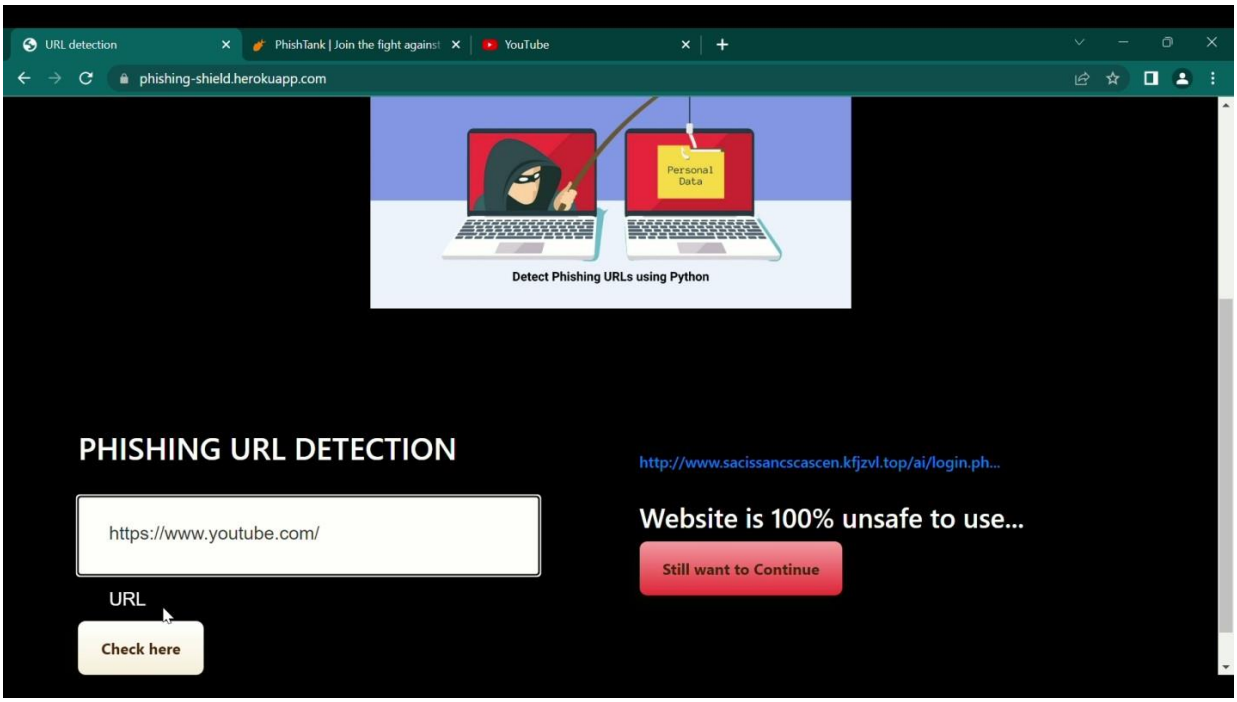
Solutioning:



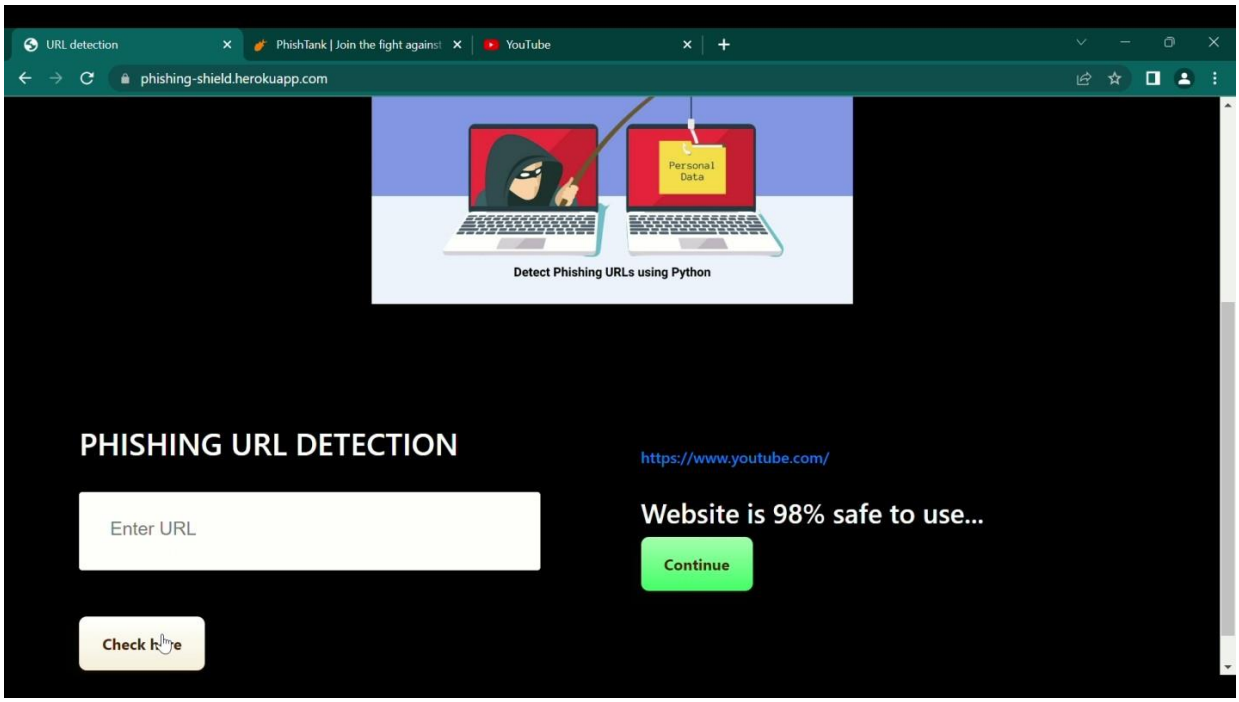
Coding:

```
<!DOCTYPE html>
<html>
<head>
<style>
#rcorners1 {
border-radius: 25px;
background: lavender;
padding: 200px;
width: 200px;
height: 150px;
}
@keyframes example {
0% {background-color:#6699ff; left:0px; top:0px;}
25% {background-color: #0066ff; left:200px; top:0px;}
50% {background-color:#0000ff; left:200px; top:200px;}
75% {background-color:#000099; left:0px; top:200px;}
100% {background-color:#0000cc; left:0px; top:0px;}
}
</style>
</head>
<body>
<center>
<div id="rcorners1">
<h2><b>WELCOME TO WEB PHISHING DETECTION !!!You have
successfully logged
in.</b></h2>
</div>
</center>
</body>
</html>
```

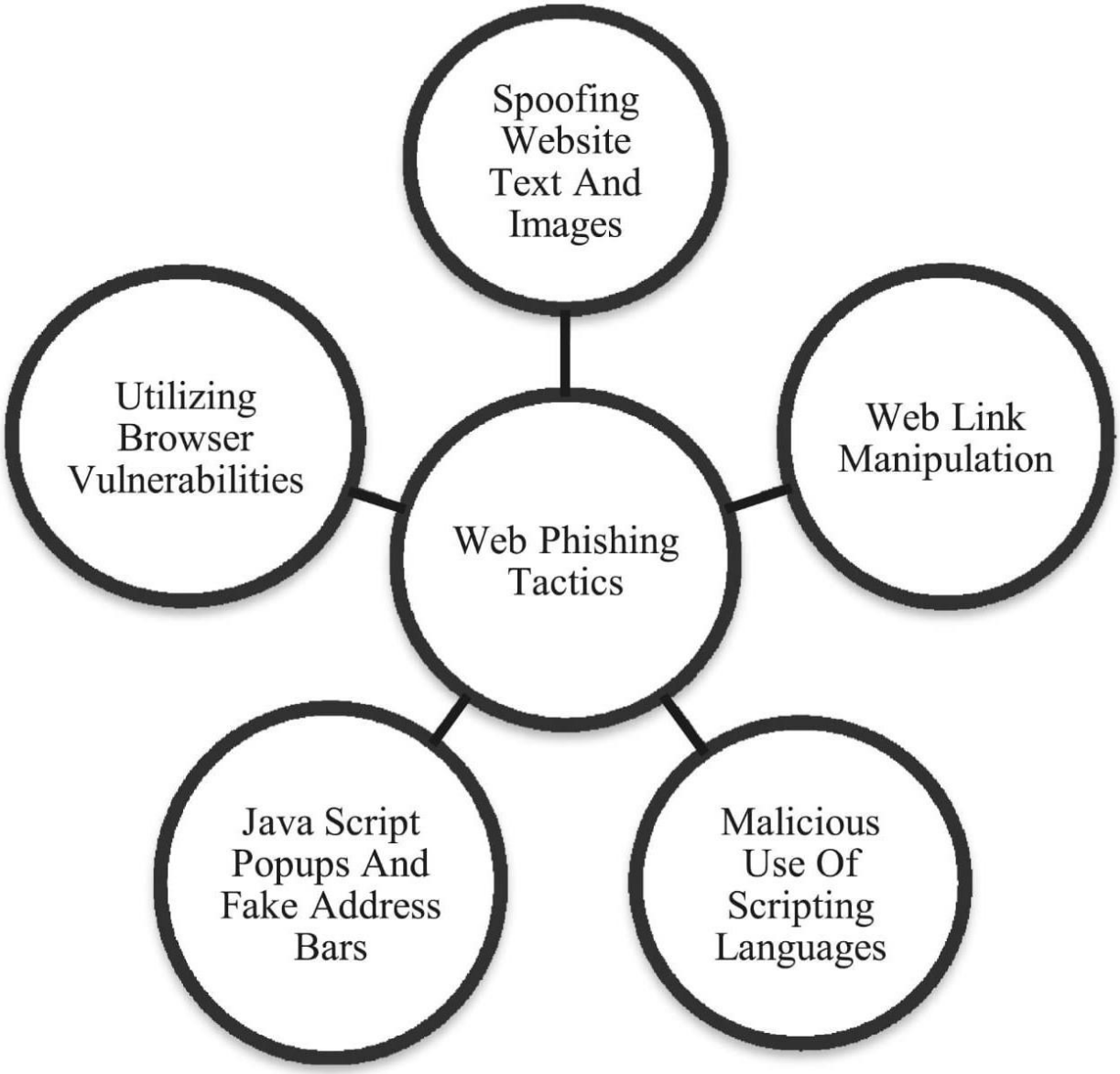

URL detection
PhishTank | Join the fight against
YouTube
phishtank.org
PhishTank is operated by Cisco Talos Intelligence Group.
PhishTank® Out of the Net, into the Tank.
username
Sign In
Register | Forgot Password
Home
Add A Phish
Verify A Phish
Phish Search
Stats
FAQ
Developers
Mailing Lists
My Account
Join the fight against phishing
Submit suspected phishes. Track the status of your submissions.
Verify other users' submissions. Develop software with our free API.
Found a phishing site? Get started now — see if it's in the Tank:
http://
Is it a phish?
Recent Submissions
You can help! Sign in or register (free! fast!) to verify these suspected phishes.
ID
URL
Submitted by
7947327 http://www.sacissancscascen.jteuyf.top/ai/login.ph... Micha
7947326 http://www.sacissancscascen.kfjzvl.top/ai/login.ph... Micha
7947325 http://www.sacissancscascen.klqtzh.top/ai/login.ph... Micha
7947324 http://www.sacissancscascen.kqdudy.top/ai/login.ph... Micha
7947323 http://www.sacissancscascen.lersmu.top/ai/login.ph... Micha
7947322 http://www.sacissancscascen.llhtbi.top/ai/login.ph... Micha
7947321 http://www.sacissancscascen.npljci.top/ai/login.ph... Micha
What is phishing?
Phishing is a fraudulent attempt, usually made through email, to steal your personal information.
Learn more...
What is PhishTank?
PhishTank is a collaborative clearing house for data and information about phishing on the Internet. Also, PhishTank provides an open API for developers and researchers to integrate anti-phishing data into their applications at no charge.
Read the FAQ...


URL detection
PhishTank | Join the fight against
YouTube
phishing-shield.herokuapp.com
Personal Data
Detect Phishing URLs using Python
PHISHING URL DETECTION
https://www.youtube.com/
URL
Check here
http://www.sacissancscascen.kfjzvl.top/ai/login.ph...
Website is 100% unsafe to use...
Still want to Continue

Result:


URL detection
PhishTank | Join the fight against
YouTube
phishing-shield.herokuapp.com
Personal Data
Detect Phishing URLs using Python
PHISHING URL DETECTION
Enter URL
Check here
https://www.youtube.com/
Website is 98% safe to use...
Continue

7.3 Database Schema:

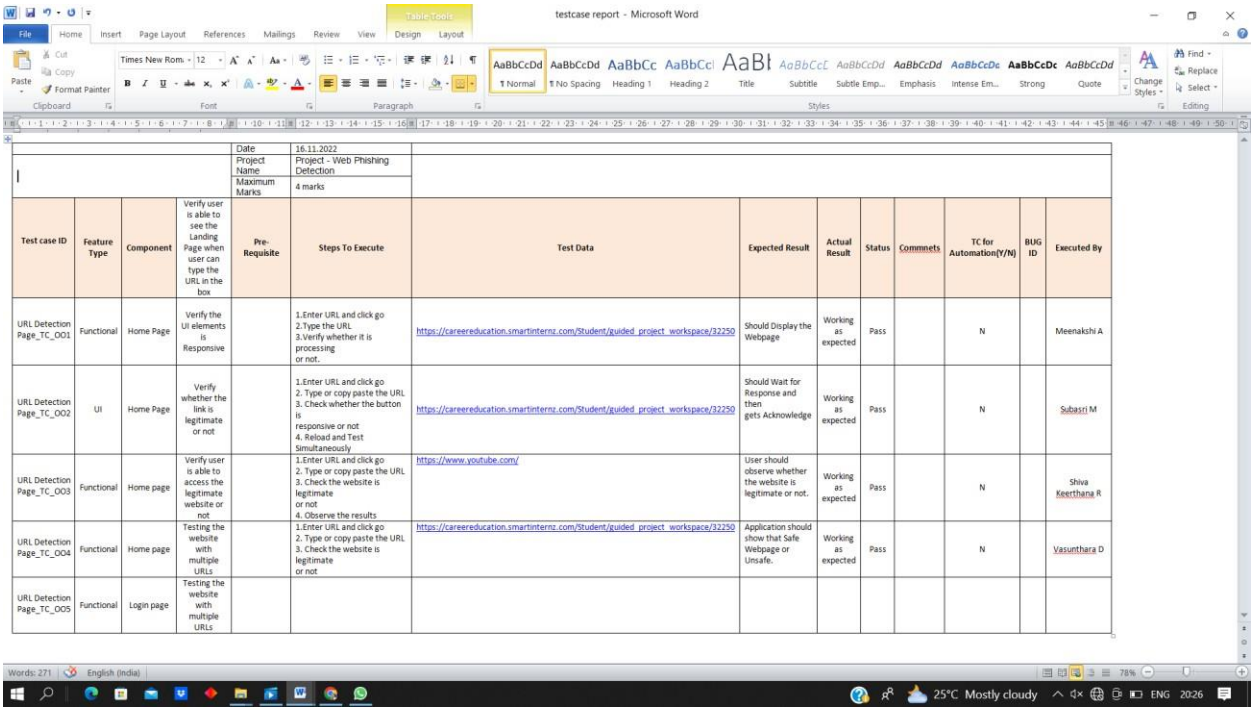


7.3 Work flow

We propose a phishing detection approach, which extracts efficient features from the URL 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

8. TESTING

8.1 Test Cases:



testcase report - Microsoft Word

				Date	16.11.2022								
				Project Name	Project - Web Phishing Detection								
				Maximum Marks	4 marks								
Test case ID	Feature Type	Component	Verify user is able to see the Landing Page when user can type the URL in the box	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
URL Detection Page_TC_OO1	Functional	Home Page	Verify the UI elements is Responsive		1.Enter URL and click go 2.Type the URL 3.Verify whether it is processing or not.	https://careereducation.smartinternz.com/Student/guided_project_workspace/32250	Should Display the Webpage	Working as expected	Pass		N		Meenakshi A
URL Detection Page_TC_OO2	UI	Home Page	Verify whether the link is legitimate or not		1.Enter URL and click go 2. Type or copy paste the URL 3. Check whether the button is responsive or not 4. Reload and Test Simultaneously	https://careereducation.smartinternz.com/Student/guided_project_workspace/32250	Should Wait for Response and then gets Acknowledge	Working as expected	Pass		N		Subasri M
URL Detection Page_TC_OO3	Functional	Home page	Verify user is able to access the legitimate website or not		1.Enter URL and click go 2. Type or copy paste the URL 3. Check the website is legitimate or not 4. Observe the results	https://www.youtube.com/	User should observe whether the website is legitimate or not.	Working as expected	Pass		N		Shiva Keerthana R
URL Detection Page_TC_OO4	Functional	Home page	Testing the website with multiple URLs		1.Enter URL and click go 2. Type or copy paste the URL 3. Check the website is legitimate or not	https://careereducation.smartinternz.com/Student/guided_project_workspace/32250	Application should show that Safe Webpage or Unsafe.	Working as expected	Pass		N		Vasunthara D
URL Detection Page_TC_OO5	Functional	Login page	Testing the website with multiple URLs										

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, 15 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. The class with five sub-classes checks the 5 different heuristic rules, length of the URL, number of dots and slashes in the URL, presence of @ symbols in the URL, IP address mentioned in the URL, and the presence of special character such as ',', '_', ';' in the URL. Initially, only a single test case was created for the class with five sub-classes, but it was failing as this class has five methods as shown in Figure 4.11. After applying the getter setter method, all the test cases passed without any issues. The test results are shown in Figure 4.12. assert Not Null() is used to check if the input URL is not empty, and assert Array Equals() is used to compare the result from the detection method with the expected result.

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.

UAT is performed by –

· Client

· End users

UAT PROCESS

Need of User Acceptance Testing arises once software has undergone Unit, Integration and System testingbecause developers might have built software based on requirements document.. by their own understanding further required changes during development may not be effectively communicated to them, so for testing whether the final product is accepted by client/end-user, user acceptance testing is needed.

9 RESULT

9.1 Performance Metrics

In a classification problem, the category or classes of data is identified based on training da ta. 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 performance of a classification model, different metrics are used, and some of them are follows:

· Accuracy

· Confusion Matrix

· Precision

· Recall

· F-Score

· AUC(Area Under the Curve)-ROC

10 ADVANTAGES & DISADVANTAGES

Advantages

Blacklists

· 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

Blacklists

· 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

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 attacksthrough 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 utilitieswere 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 wereused to unit test one of the modules of the ISOT phishing detection system. In this report, thedata 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 97.14% 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 97.14% detection accuracy using random forest algorithm with lowest false positive rate. Also result shows that classifiers give better performance whenwe 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 use

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 determiningthe 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

13 APPENDIX

SOURCE CODE

App.py

```
#importing required libraries
From flask import Flask, request, render _template
import requests
import numpy as np
import warnings
import pickle
warnings.filterwarnings('ignore')
from feature import Feature Extraction
file = open("model.pkl","rb")
gbc = pickle.load(file)
file.close()
API_KEY = "UWEsUaH1i-FABXxbCpQ9lcPk5E0jIaivG8i-veVF9zJj" 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
```

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

app=Flask(_name_)

@app.route("/", methods=["GET", "POST"])

def index():

if request.method == "POST":

url =  request.form["url"]

obj = FeatureExtraction(url)

x = np.array(obj.getFeaturesList()).reshape(1,30)

#1 is safe

#-1 is unsafe

y_pro_phishing = gbc.predict_proba(x)[0,0]

y_pro_non_phishing = gbc.predict_proba(x)[0:,1]

print( y_pro_phishing,y_pro_non_phishing)

# if(y_pred ==1 ):

pred = "It is {0:.2f} % safe to go ".format(y_pro_phishing*100)

payload_scoring = {"input_data": [{"field":

[["UsingIP","LongURL","ShortURL","Symbol@","Redirecting//","PrefixSuffix-
```

","SubDomains","HTTPS","DomainRegLen","Favicon","NonStdPort","HTTPSDomain URL","RequestURL","AnchorURL","LinksInScriptTags","ServerFormHandler","Info Email","AbnormalURL","WebsiteForwarding","StatusBarCust","DisableRightClick"," UsingPopupWindow","IframeRedirection","AgeofDomain","DNSRecording","WebsiteT raffic","PageRank","GoogleIndex","LinksPointingToPage","StatsReport"]], "values":obj}]}

```
response_scoring = requests.post('https://us
south.ml.cloud.ibm.com/ml/v4/deployments/phishing_1/predictions?version=2022-11-11',
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("The Website is secure.. Continue")
else:
print("The Website is not Legitimate... BEWARE!!")
return render_template('index.html',xx =round(y_pro_non_phishing,2),url=url ) return
render_template("index.html", xx = -1)
if _name_ == "_main_":
app.run(debug=True)
```

PROJECT DEMO LINK

URL detection - Google Chrome 2022-11-17 18-30-55.mp4

GITHUB

https://github.com/IBM-EPBL/IBM-Project-22929-1659861252