WEB PHISHING DETECTION USING MACHINE LEARNING

1. INTRODUCTION

1.1 Project Overview

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. 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.

1.2 Purpose

- 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.

● So we develop this detection method to predict whether the website is phishing website or not.

● By this, we can save or rescue the missing data.

2. LITERATURE SURVEY

2.1 Existing problem

TOPIC: Phishing Website Detection Using Machine Learning.

AUTHORS: AdarshMandadi;SaikiranBoppana;Vishnu Ravella;R Kavitha.

Phishing is an Internet scam in which an attacker sends fake messages thatappear to come from a trusted source. The goal was to get as many people as possible to click on a link or open an infected file. There are different approaches to detect this type of attack. One approach is machine learning. The URLs received from the user are fed into the machine learning model, then the algorithm processes the input and displays the output, whether it is phishing or legitimate. The proposed approach deals with Random Forest, Decision Tree classifiers. The proposed approach effectively classified phishing and legitimate URLs with an accuracy of 87.0% and 82.4%, respectively, for the Random Forest and Decision Tree classifiers.

TOPIC: An Ensemble Method for PhishingWebsites Detection Based on XGBoost.

AUTHORS: Jiaqi Gu;HuiXu.

Today, the Internet is spreading all over the world. Apart from the benefits that the Internet brings to its users, there are many potential harms such as phishing scams.In this paper, I present an ensemble model for detecting phishing websites using URL functions. I used the dataset called &quot;Phishing

Website Detector - Phishing Website Dataset&quot; from Kaggle. Then, several models were built using all sorts of common ensemble algorithms (like stacking, boosting, and bagging). Finally, various methods of measuring the performance of the models have been used. The reason why only ensemble models are chosen as methods to implement is mainly because of their good overall performance. In terms of results, the proposed XGBoost model combining Random Forest and K-Nearest Neighborsoutperforms all other models (Random Forest, AdaBoost, Parameterized XGBoost, Stacking and Voting), with an accuracy of 99.74% on data from training and 96.44% on test data.

TOPIC: Phishing Detection and Prevention using Chrome Extension.
AUTHORS:M. Amir SyafiqRohmat Rose ; NurlidaBasir ; NurFatin Nabila RafieHeng ; Nurzi Juana MohdZaizi ; MadihahMohd Saudi.
During the COVID-19 pandemic outbreaks, the number of cyber attacks, including phishing activities, skyrocketed. Many technical solutions for phishing detection have been developed today, but these approaches have failed or failed to identify phishing pages and efficiently detect malicious code. One of the drawbacks is poor detection accuracy and poor adaptability to new phishing connections.In this paper, an intelligent phishing detection and prevention model is designed. The proposed model uses a self-destruct detection algorithm in which machine learning, especially supervised learning algorithm, has been used. All rules used in the algorithm focus on URL-based web properties that attackers rely on to redirect victims to the simulated websites. Accordingly, based on the proposed model, Phishing Detection for Chrome Extensions has been developed to prevent phishing attacks with appropriate countermeasure and alert users of phishing when

visiting illegal websites. This smart phishing detection and prevention model is believed to be able to prevent scam and spam websites and reduce cybercrime and cybercrisis happening year by year.

2.2 References

1. Phishing Website Detection Using Machine Learning

Adarsh Mandadi;Saikiran Boppana;Vishnu Ravella;R Kavitha

2022 IEEE 7th International conference for Convergence in Technology (I2CT)

https://ieeexplore.ieee.org/document/9824801/

2.An Ensemble Method for Phishing Websites Detection Based on XGBoost

Jiaqi Gu;Hui Xu

2022 14th International Conference on Computer Research and Development (ICCRD)

Year: 2022 | Conference Paper | Publisher: IEEE

https://ieeexplore.ieee.org/document/9730579/

3.Phishing Detection and Prevention using Chrome Extension

M. Amir Syafiq Rohmat Rose;Nurlida Basir;Nur Fatin Nabila Rafie Heng;Nurzi Juana Mohd

Zaizi;Madihah Mohd Saudi

2022 10th International Symposium on Digital Forensics and Security (ISDFS)

https://ieeexplore.ieee.org/document/9800826/

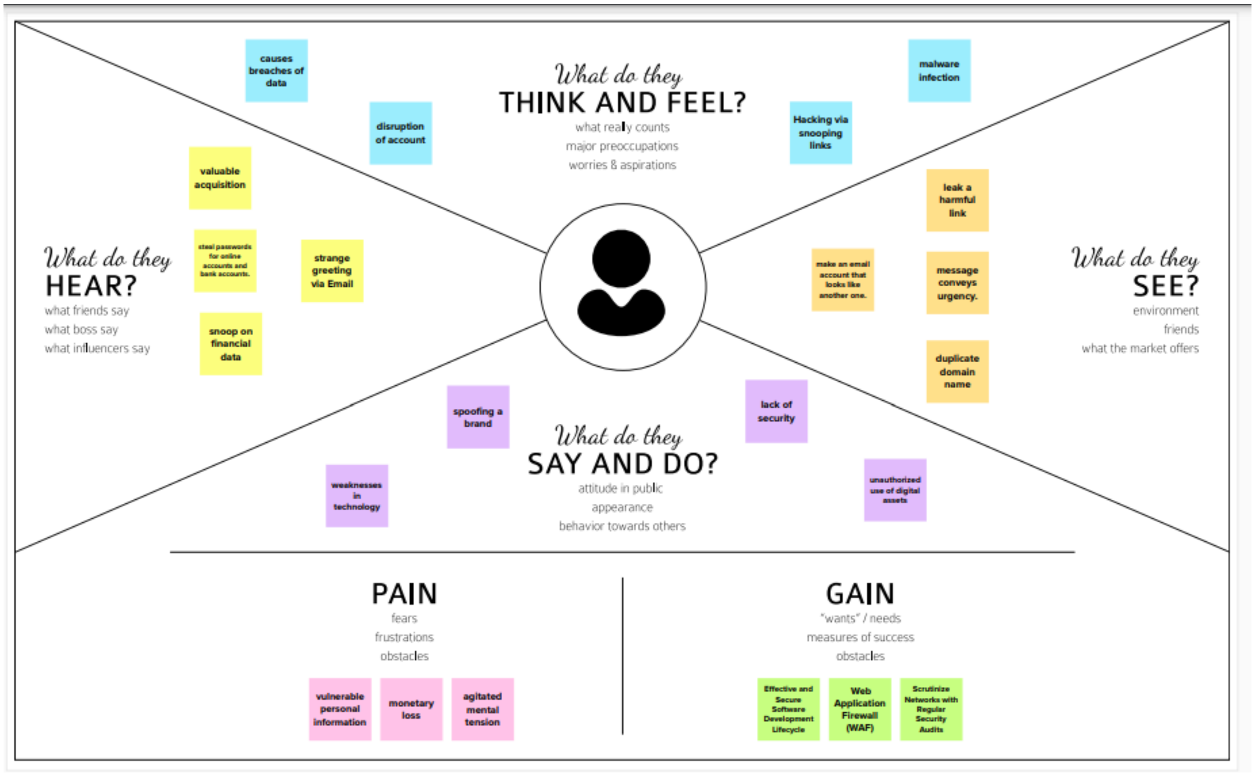
2.3 Problem Statement Definition

Phishing is a type of fraud in which the perpetrator sends emails or other communications claiming to be from a respectable company or individual in an attempt to get sensitive information like login passwords or account information from respective users. The intention is to get the users to reveal their fnancial information, system credentials or other sensitive data.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas:

Teams can utilise an empathy map as a collaborative tool to learn more about their clients. An empathy map can depict a group of users, such as a consumer segment, in a manner similar to user personas. Teams can better understand a principal user's motivations, issues, and user experience by using the empathy map, which depicts that person. Empathy mapping is a straightforward yet powerful workshop that can be used with a wide range of users, including stakeholders, specific use cases, or entire teams.



3.2 Ideation & Brainstorming

Brainstorm, Idea Listing and Grouping:

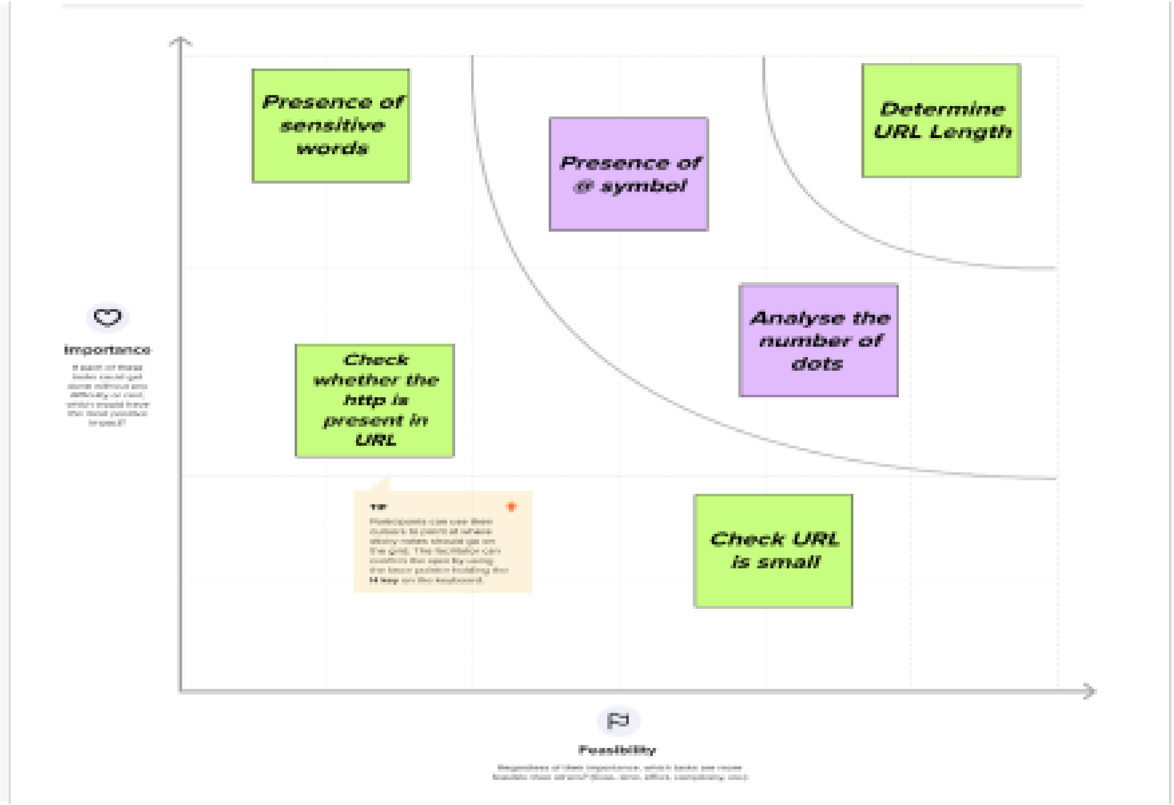
Brainstorm is nothing but to suggest ideas for the project before starting the project. The process of brainstorming can assist the group focus its ideas and find solutions.

Project Ideas are where you begin documenting proposals for future research grant applications. During this stage, you are recording important project-related details as well as locating collaborators, potential funders, budget details, and project-related metadata. You can also make tasks and assign them to project-related people.

An administrative grouping of projects is known as a project group. Project groups make it possible for administrative operations to have an impact on several projects and users at once.

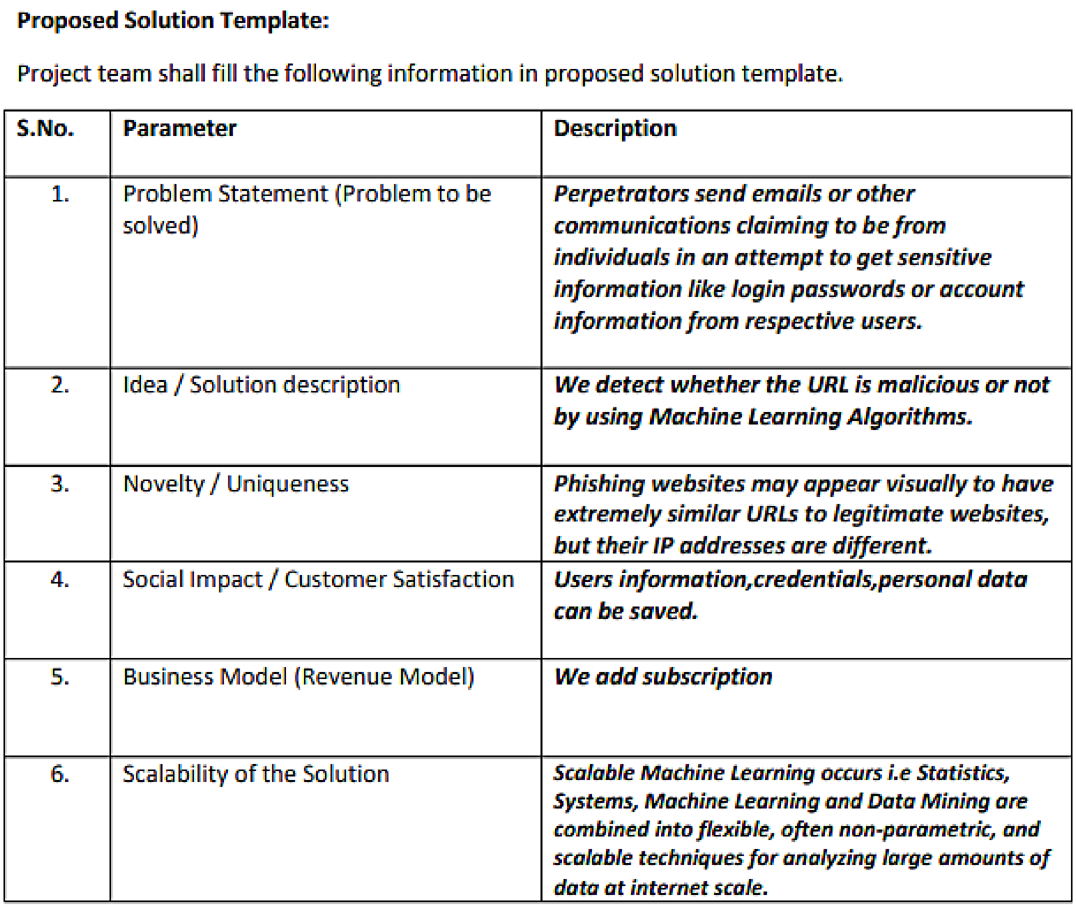
Idea prioritization:

Only a small portion of the idea management process involves idea prioritisation. It takes time to develop an organised idea management strategy and a methodical approach to gathering, analysing, and prioritising new ideas.


Presence of sensitive words
Presence of @ symbol
Determine URL Length
Importance
Analyse the number of dots
Check whether the http is present in URL
TIP
Check URL is small
Feasibility

3.3 Proposed Solution

Proposed Solution refers to the technical response that the Implementation agency will offer in response to the Project's requirements and objectives.The proposed solution of our project is given below:



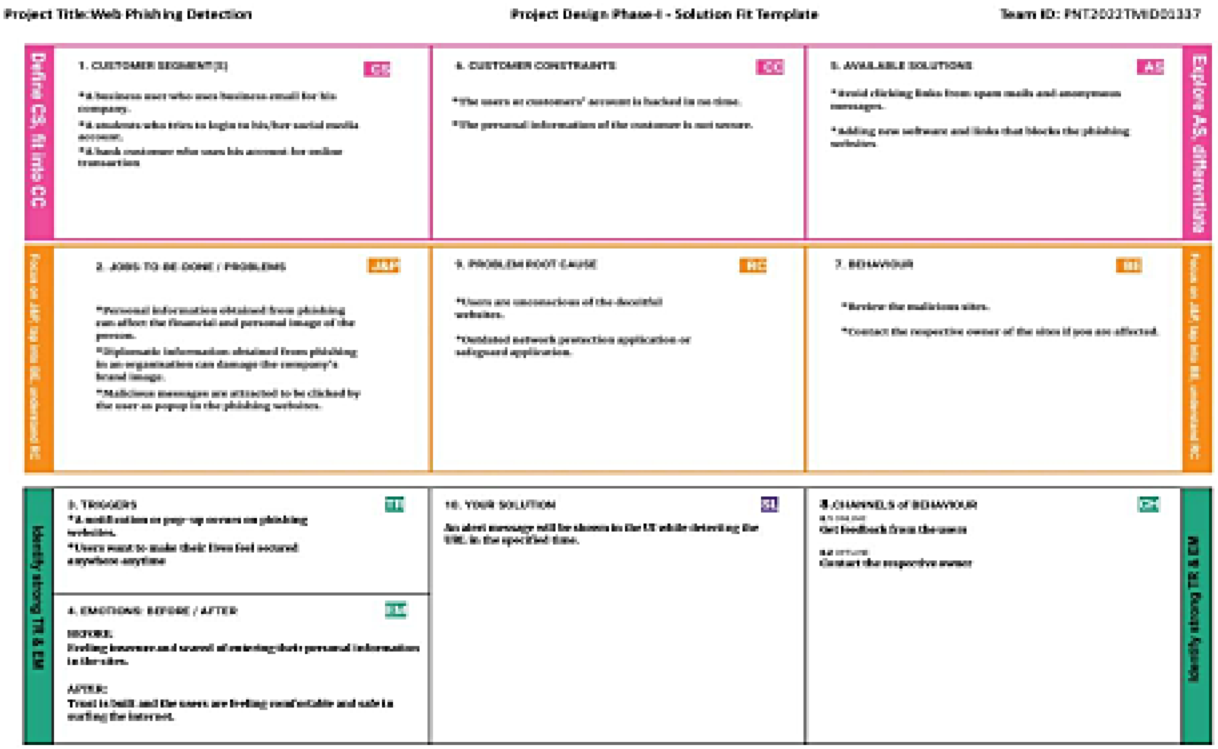
Proposed Solution Template:

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

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	***Perpetrators send emails or other communications claiming to be from individuals in an attempt to get sensitive information like login passwords or account information from respective users.***
2.	Idea / Solution description	***We detect whether the URL is malicious or not by using Machine Learning Algorithms.***
3.	Novelty / Uniqueness	***Phishing websites may appear visually to have extremely similar URLs to legitimate websites, but their IP addresses are different.***
4.	Social Impact / Customer Satisfaction	***Users information,credentials,personal data can be saved.***
5.	Business Model (Revenue Model)	***We add subscription***
6.	Scalability of the Solution	***Scalable Machine Learning occurs i.e Statistics, Systems, Machine Learning and Data Mining are combined into flexible, often non-parametric, and scalable techniques for analyzing large amounts of data at internet scale.***

3.4 Problem Solution fit

Proposed solution fit is nothing but identifying an existing problem and to solve it with a solution that customers find useful and satisfying.



Project Title:Web Phishing Detection | Project Design Phase-I - Solution Fit Template | Team ID: PNT2022TMID01337

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) CS *A business user who uses business email for his company. *A students who tries to login to his/her social media account. *A bank customer who uses his account for online transaction	6. CUSTOMER CONSTRAINTS CC *The users or customers' account is hacked in no time. *The personal information of the customer is not secure.	5. AVAILABLE SOLUTIONS AS *Avoid clicking links from spam mails and anonymous messages. *Adding new software and links that blocks the phishing websites.	Explore AS, differentiate
Focus on J&P, tap into BE, understand RC	2. JOBS-TO-BE-DONE / PROBLEMS J&P *Personal information obtained from phishing can affect the financial and personal image of the person. *Diplomatic information obtained from phishing in an organization can damage the company's brand image. *Malicious messages are attracted to be clicked by the user as popup in the phishing websites.	9. PROBLEM ROOT CAUSE RC *Users are unconscious of the deceitful websites. *Outdated network protection application or safeguard application.	7. BEHAVIOUR BE *Review the malicious sites. *Contact the respective owner of the sites if you are affected.	Focus on J&P, tap into BE, understand RC
Identify strong TR & EM	3. TRIGGERS TR *A notification or pop-up occurs on phishing websites. *Users want to make their lives feel secured anywhere anytime 4. EMOTIONS: BEFORE / AFTER EM BEFORE: Feeling insecure and scared of entering their personal information in the sites. AFTER: Trust is built and the users are feeling comfortable and safe in surfing the internet.	10. YOUR SOLUTION SL An alert message will be shown in the UI while detecting the URL in the specified time.	8.CHANNELS of BEHAVIOUR CH 8.1 ONLINE Get feedback from the users 8.2 OFFLINE Contact the respective owner	Identify strong TR & EM

REQUIREMENT ANALYSIS

4.1 Functional requirement

The desired operations of a programme or system are referred to as functional requirements in software development and systems engineering. Product features or functions must be implemented by developers in order for users to complete their duties. For the development team as well as the stakeholders, it is crucial to make them apparent. Functional requirements often explain how a system will behave under particular circumstances.

FunctionalRequirements:

Followingarethefunctionalrequirementsof theproposedsolution.

FRNo.	**Functional requirement(Epic)**	**Sub Requirement(Story/Sub-Task)**
FR-1	Usercommunication	User Must Know the SignLanguage
FR-2	Usercommunication	The user Has tocommunicate in Front of theCamera

4.2 Non-Functional requirements

Non-functional requirements list a system's fundamental characteristics. They are sometimes referred to as qualities. It defines characteristics of the system such as usability, scalability, maintainability, and performance. They act as limitations or restrictions on how the system is designed for the various backlogs.

Non-functionalRequirements:

Followingarethenon-functionalrequirementsoftheproposedsolution.

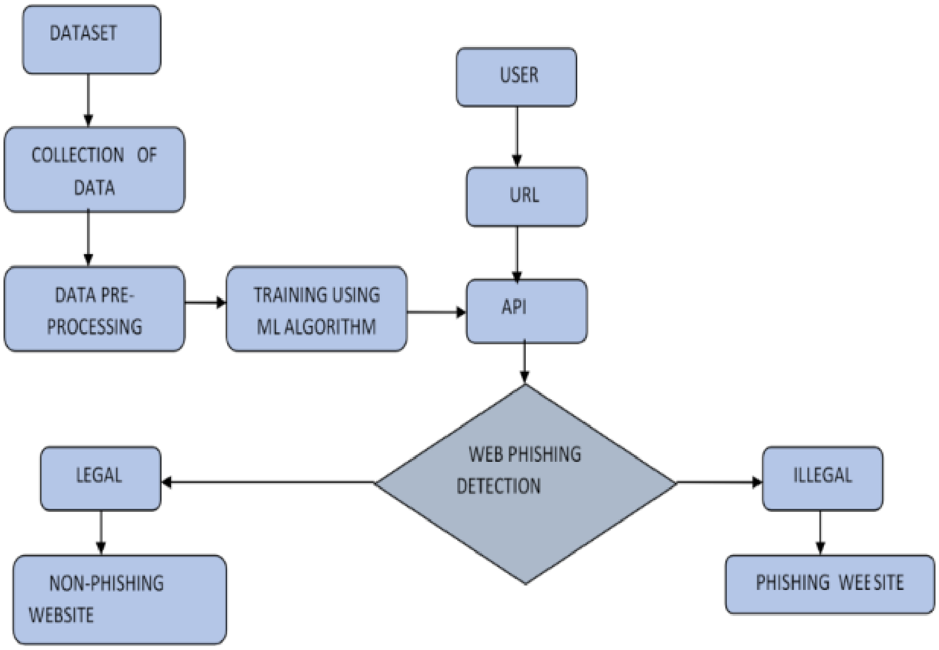
FRNo	Non-FunctionalRequirement	Description
NFR-1	Usability	The camera captures allexpressions including facialexpressions and hand gestureswhichcanbeeasilyusedbyall age groups.
NFR-2	Reliability	The system is very liable, itcanlastforlongamountsof timeifwellmaintained.
NFR-3	Performance	Thecost-effectivenatureofthe system makes it extremelyliableandthus,efficient.
NFR-4	Availability	The solution fits all the signlanguages when we train themodel for all the signlanguages. So, it is used by allthecountrieswithdifferent languages.
NFR-5	Scalability	The system gives outputrapidly. It also predicts quicklywhen it gets so many inputs atatime.Itpredictsdifferent types of sign language at atime.

5. PROJECT DESIGN

5.1 Data Flow Diagrams

Data Flow Diagrams It demonstrates the many types of data that will be input into and exported from the system, as well as where the data will be stored. A DFD is frequently an expansion of a context diagram to reveal more of the system's finer details that were initially depicted by the context diagram.

Data Flow Diagrams:

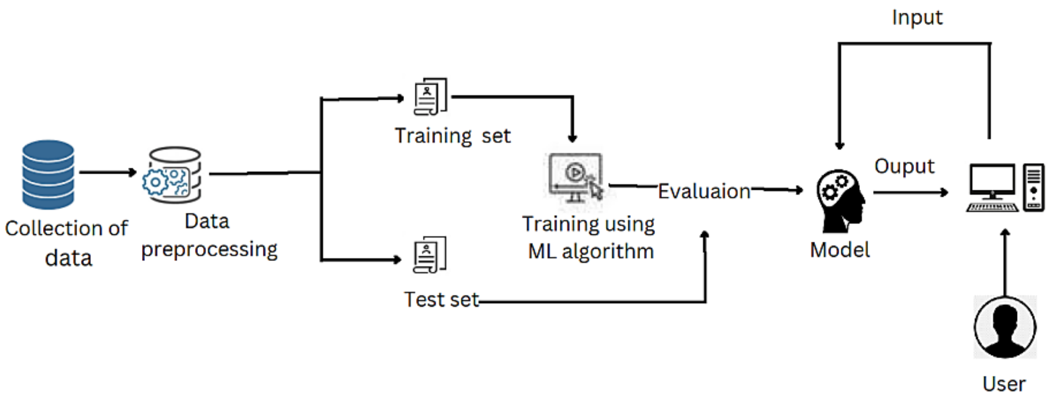


5.2 Solution & Technical Architecture

Technical Architecture:

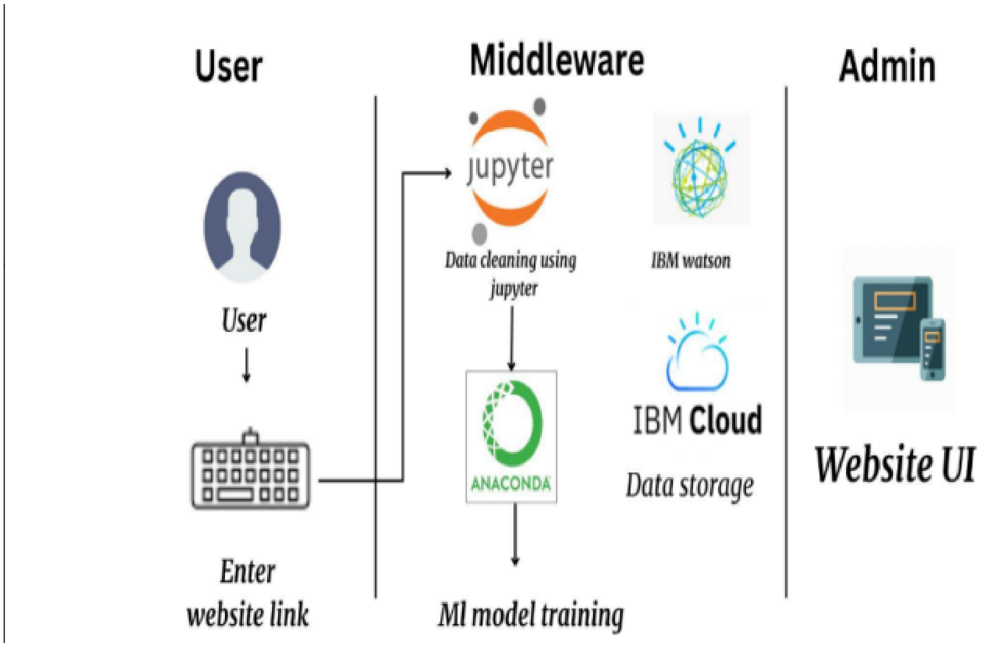
The main system components, their connections, and the agreements that specify how the components interact are all included in the technical architecture. The objective of technical architects is to fulfil all business requirements with an application that is both performance- and security-optimized. creating the framework for technological systems. controlling the execution of programmes. collaborating with the software development group to make sure the system functions properly.

Solution Architecture:



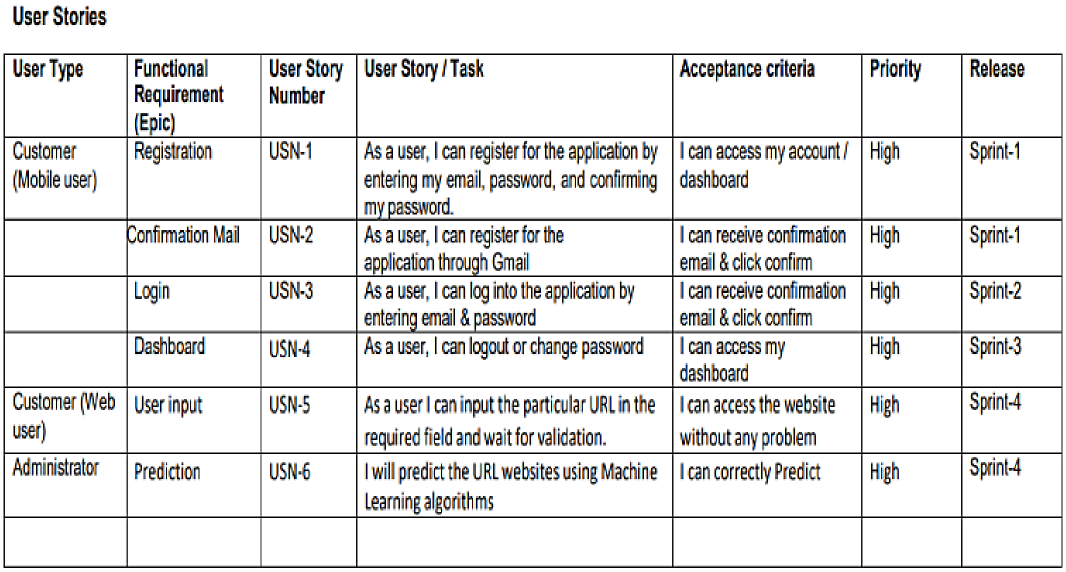
Solution Architecture:

By outlining the functional requirements and implementation stages of IT solutions and customising them to particular business objectives, solution architecture serves as the foundation for software development initiatives. It is divided into numerous subprocesses, each of which is guided by a different perspective on corporate architecture. The solution architecture makes it possible to guarantee that a new system will work in the current business context. A solution architect must comprehend how the processes, operating systems, and application architectures all interact in order to carry out this role. The Solution Architecture of the web phishing detection is given below:



5.3 User Stories

A user story is a casual, all-inclusive description of a software feature written from the viewpoint of the client or end user. A user story's objective is to describe how a piece of work will provide the customer with a specific value. The fact that user stories, unlike requirements or use cases, are not intended to stand alone may be the most significant advantage of using user stories in agile product development. Every user story is instead a standing placeholder for a future discussion with the development team.



User Stories

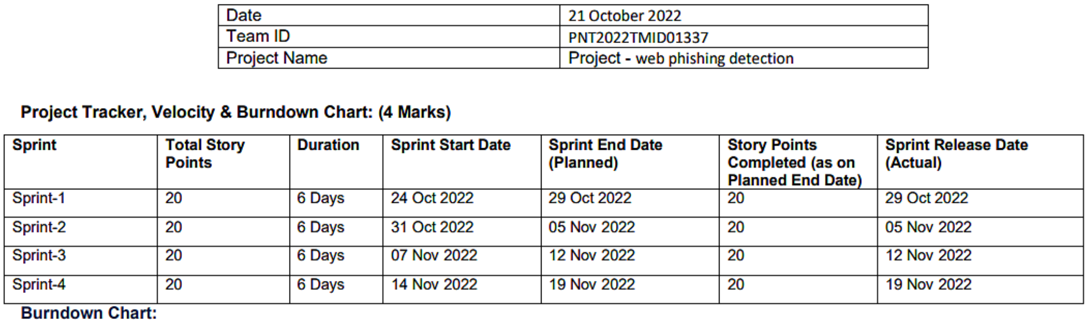
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
	Confirmation Mail	USN-2	As a user, I can register for the application through Gmail	I can receive confirmation email & click confirm	High	Sprint-1
	Login	USN-3	As a user, I can log into the application by entering email & password	I can receive confirmation email & click confirm	High	Sprint-2
	Dashboard	USN-4	As a user, I can logout or change password	I can access my dashboard	High	Sprint-3
Customer (Web user)	User input	USN-5	As a user I can input the particular URL in the required field and wait for validation.	I can access the website without any problem	High	Sprint-4
Administrator	Prediction	USN-6	I will predict the URL websites using Machine Learning algorithms	I can correctly Predict	High	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

The definition of a sprint is a dedicated period of time in which a set amount of work will be completed on a project. It's part of the agile methodology, and an Agile project will be broken down into a number of sprints, each sprint taking the project closer to completion.

In the scrum process, sprint planning marks the beginning of the sprint. Sprint planning's goal is to specify what can be completed in a sprint and how it will be done. The entire scrum team collaborates on sprint planning.

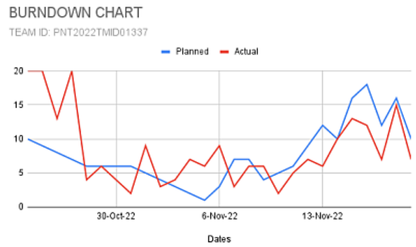


Date	21 October 2022
Team ID	PNT2022TMID01337
Project Name	Project - web phishing detection

Project Tracker, Velocity & Burndown Chart: (4 Marks)

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	19 Nov 2022

Burndown Chart:



6.2 Sprint Delivery Schedule

A sprint schedule is a written description of the entire sprint planning process. It's one of the initial steps in the agile sprint planning process, and it calls for sufficient investigation, preparation, and coordination. It centres on a product backlog, which is a list of open requests for development and iteration.

A burndown chart, which displays how rapidly a team is progressing through a customer's user stories, is a project management chart. This agile tool records the description of a feature from the viewpoint of the end user and compares the overall effort to the quantity of work for each agile sprint.

Date	21 October 2022
Team ID	PNT2022TMID01337
Project Name	Project - web phishing detection
Maximum Marks	8 Marks

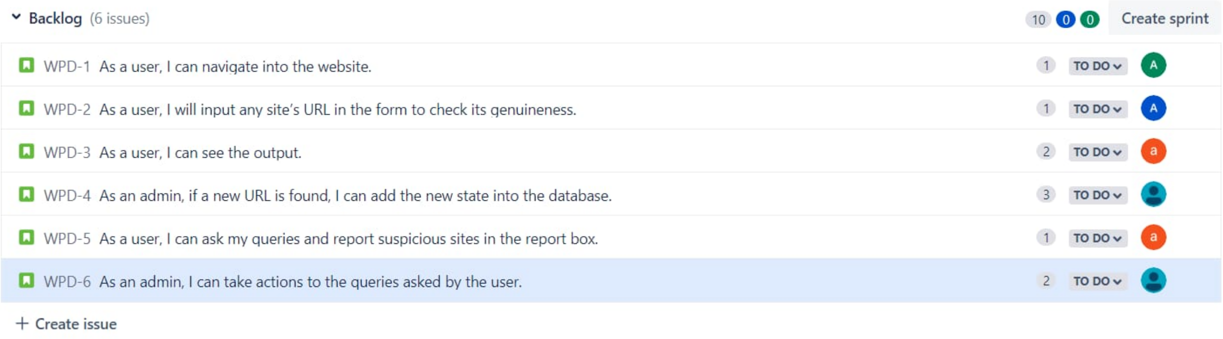
Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	lydia
Sprint-1	conformation Mail	USN-2	As a user, I can register for the application through Gmail	2	Medium	praveena
Sprint-2	Login	USN-3	As a user, I can log into the application by entering email & password	1	High	janaki
Sprint-3	Dashboard	USN-4	As a user, I can logout or change password	1	Medium	muthulakshmi
Sprint-4	User input	USN-5	As a user I can input the particular URL in the required field and wait for validation.	2	High	lydia
Sprint-4	Prediction	USN-6	I will predict the URL websites using Machine Learning algorithms	2	High	praveena

6.3 Reports from JIRA

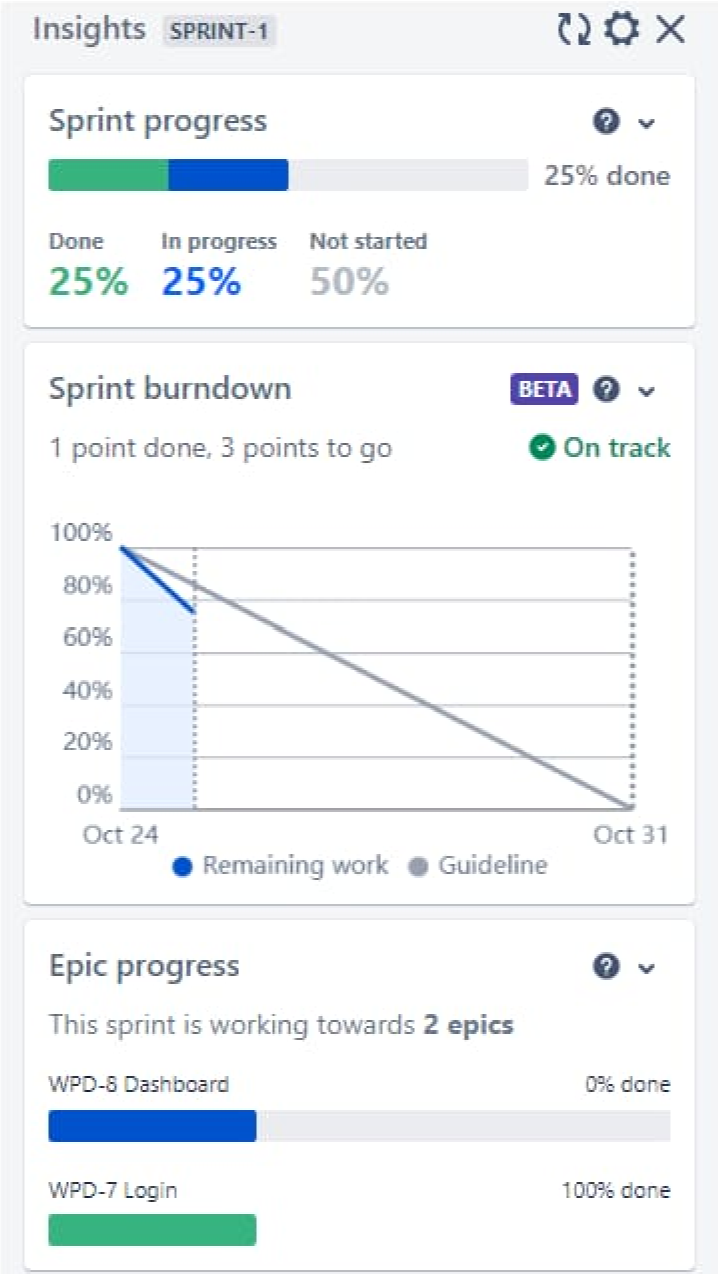
JIRA brings teams together for everything from agile software development and customer service to start-ups and companies. Jira assists teams in planning, assigning, tracking, reporting, and managing work. Jira Software, the tool for agile teams, helps software teams produce better work.



Backlog (6 issues) 10 0 0 Create sprint

WPD-1	As a user, I can navigate into the website.	1	TO DO	A
WPD-2	As a user, I will input any site's URL in the form to check its genuineness.	1	TO DO	A
WPD-3	As a user, I can see the output.	2	TO DO	a
WPD-4	As an admin, if a new URL is found, I can add the new state into the database.	3	TO DO	
WPD-5	As a user, I can ask my queries and report suspicious sites in the report box.	1	TO DO	a
WPD-6	As an admin, I can take actions to the queries asked by the user.	2	TO DO	

+ Create issue



Insights SPRINT-1

Sprint progress

25% done

Done 25%

In progress 25%

Not started 50%

Sprint burndown BETA

1 point done, 3 points to go

✓ On track

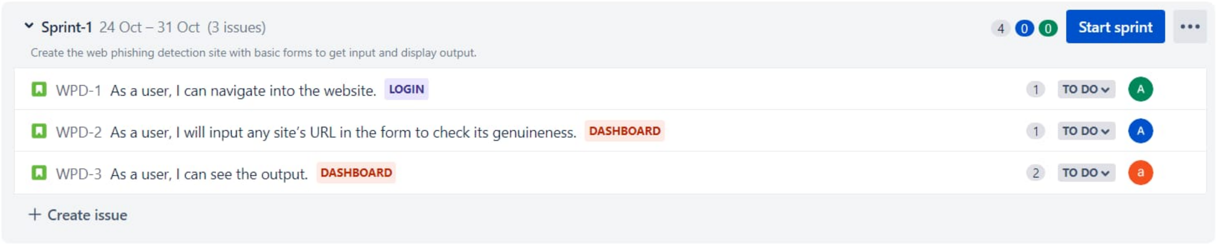
Epic progress

This sprint is working towards **2 epics**

WPD-8 Dashboard 0% done

WPD-7 Login 100% done

BACKLOG:

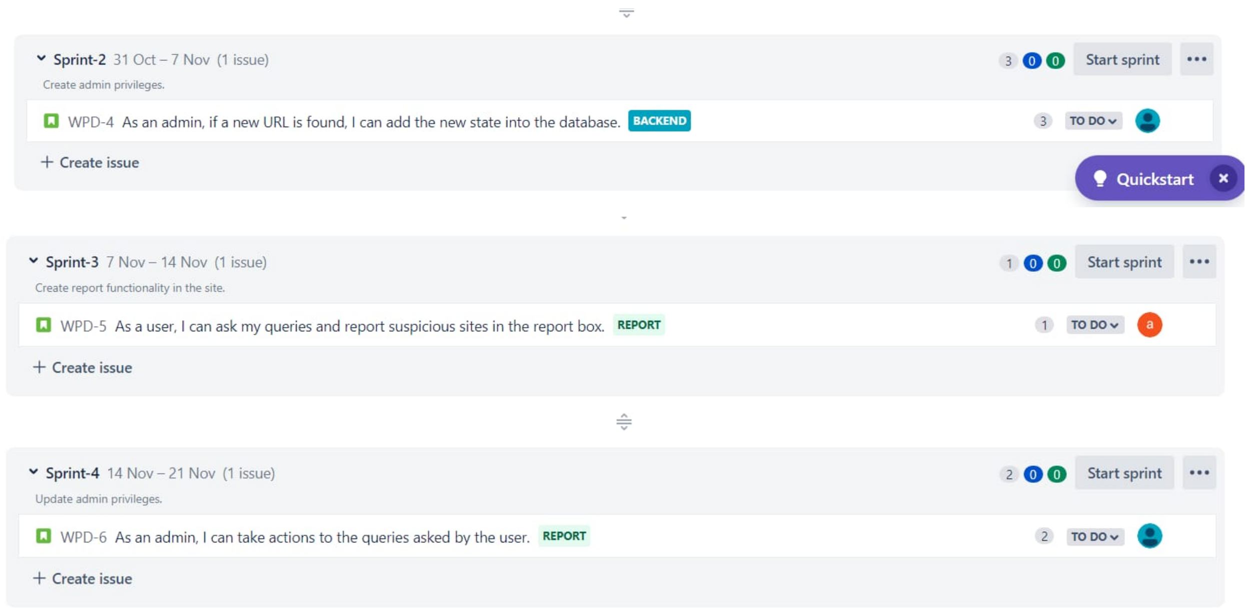


Sprint-1 24 Oct – 31 Oct (3 issues) 4 0 0 Start sprint •••

Create the web phishing detection site with basic forms to get input and display output.

WPD-1	As a user, I can navigate into the website.	LOGIN	1	TO DO	A
WPD-2	As a user, I will input any site's URL in the form to check its genuineness.	DASHBOARD	1	TO DO	A
WPD-3	As a user, I can see the output.	DASHBOARD	2	TO DO	a

+ Create issue



Sprint-2 31 Oct – 7 Nov (1 issue) 3 0 0 Start sprint •••

Create admin privileges.

WPD-4 As an admin, if a new URL is found, I can add the new state into the database. BACKEND 3 TO DO

+ Create issue

Quickstart

Sprint-3 7 Nov – 14 Nov (1 issue) 1 0 0 Start sprint •••

Create report functionality in the site.

WPD-5 As a user, I can ask my queries and report suspicious sites in the report box. REPORT 1 TO DO

+ Create issue

Sprint-4 14 Nov – 21 Nov (1 issue) 2 0 0 Start sprint •••

Update admin privileges.

WPD-6 As an admin, I can take actions to the queries asked by the user. REPORT 2 TO DO

+ Create issue

7. CODING & SOLUTIONING

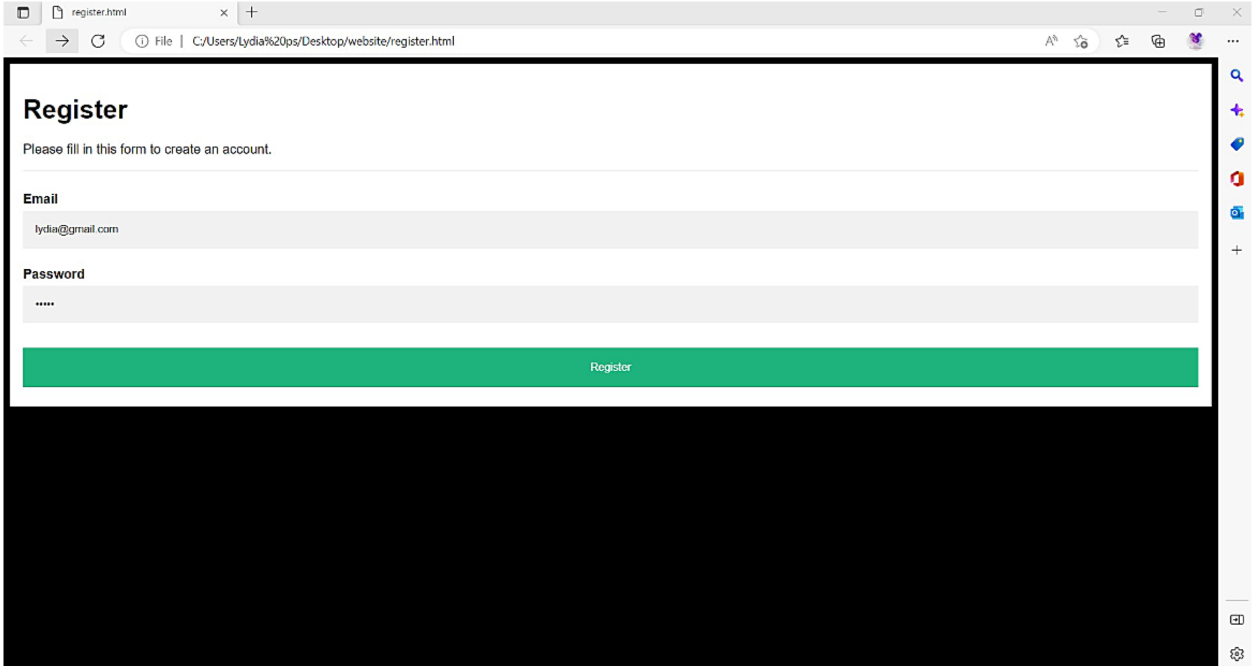
7.1 Feature 1

Registration Page:

```
<!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>
```



register.html

File | C:/Users/Lydia%20ps/Desktop/website/register.html

Register

Please fill in this form to create an account.

Email

lydia@gmail.com

Password

•••••

Register

7.2 Feature 2

Login Page (python)

```
#!/usr/bin/env
python
# coding: utf-8

# In[ ]:


from flask import Flask,request,render_template
import pickle

app = Flask(__name__)


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

database={'lydia@gmail.com':'lydia','janaki@gmail.com':'janaki',

'praveena@gmail.com':'praveena','muthulakshmi@gmail.com':'muthulakshmi'}
```

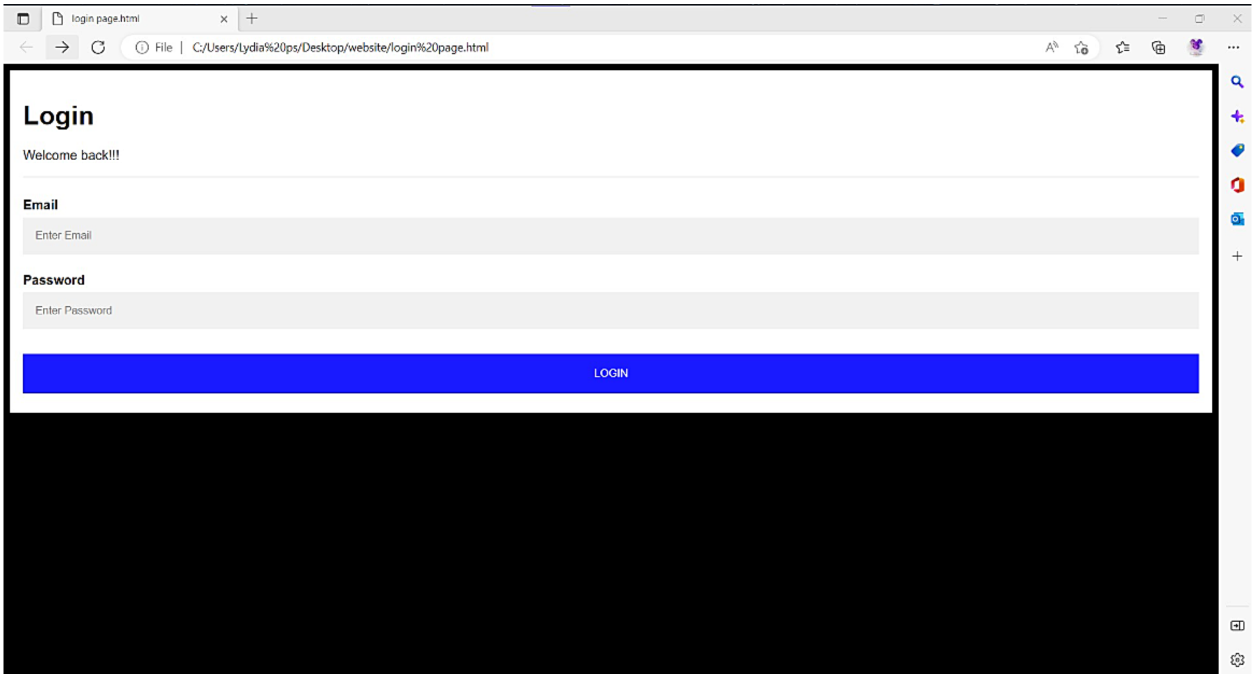
```
@app.route('/form_login',methods=['POST','GET'])
def login():
    name1=request.form['email']
    pwd=request.form['psw']
    if name1 not in database:
           return render_template('login_page.html',info='Invalid E-mail')
    else:
        if database[name1]!=pwd:
            return render_template('login_page.html',info='Invalid
Password')
        else:
                return render_template('login_success.html',email=name1)

if __name__ == '__main__':
    app.run()


# In[ ]:




# In[ ]:
```

Login Page (html)

```
<!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>
```


login page.html
File | C:/Users/Lydia%20ps/Desktop/website/login%20page.html

Login

Welcome back!!!

Email

Enter Email

Password

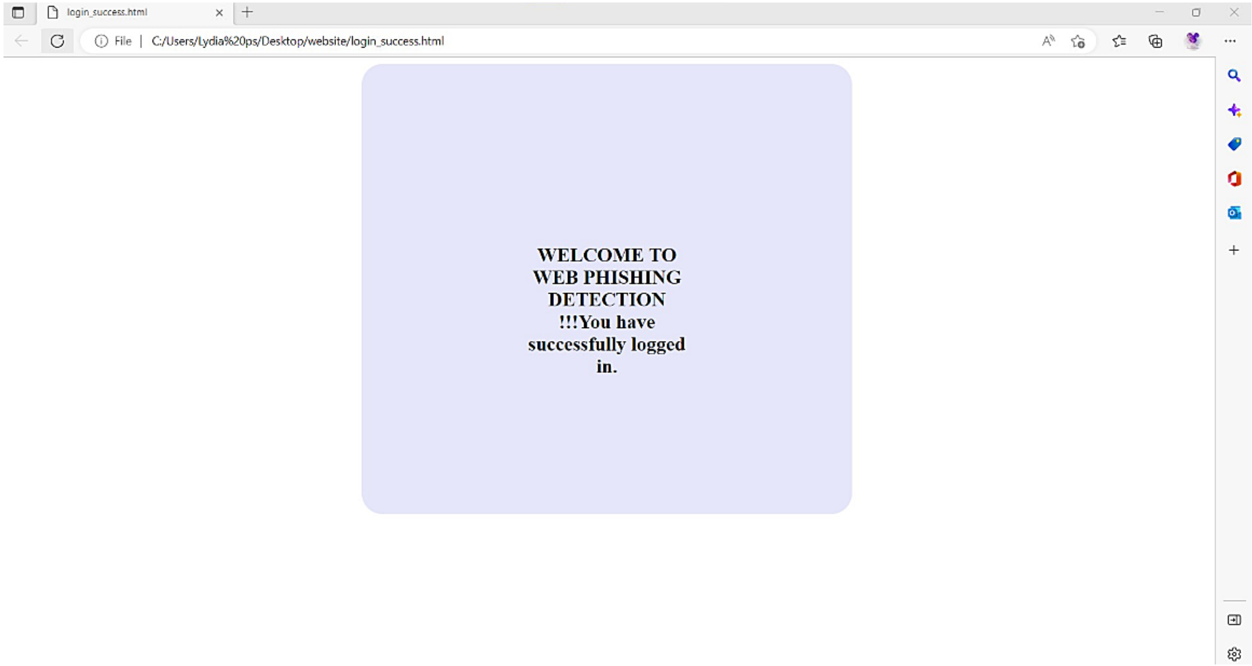
Enter Password

LOGIN

Login Page Registration success:

```
<!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;}
```

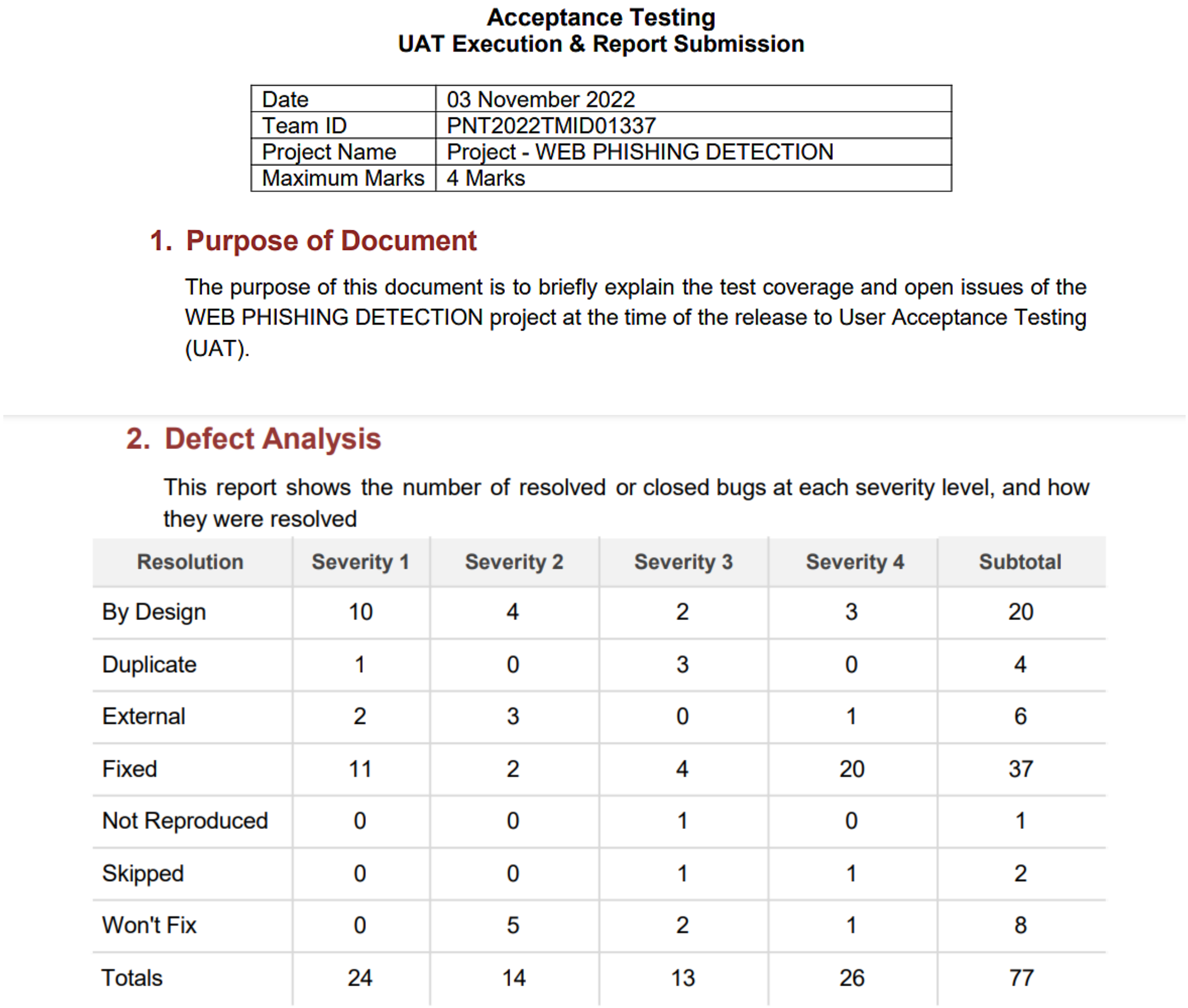
```
  75%  {background-color:#000099; left:0px; top:200px;}
  100% {background-color:#0000cc; left:0px; top:0px;}
}
</style>
</head>
<body>
<center>
<div id="rcorners1">
<h2><b>WELCOME TO     WEB PHISHING DETECTION !!!You have successfully logged
in.</b></h2>
</div>
</center>

</body>
</html>
```



WELCOME TO WEB PHISHING DETECTION !!!You have successfully logged in.

8. TESTING

8.1 Test cases



Acceptance Testing
UAT Execution & Report Submission

Date	03 November 2022
Team ID	PNT2022TMID01337
Project Name	Project - WEB PHISHING DETECTION
Maximum Marks	4 Marks

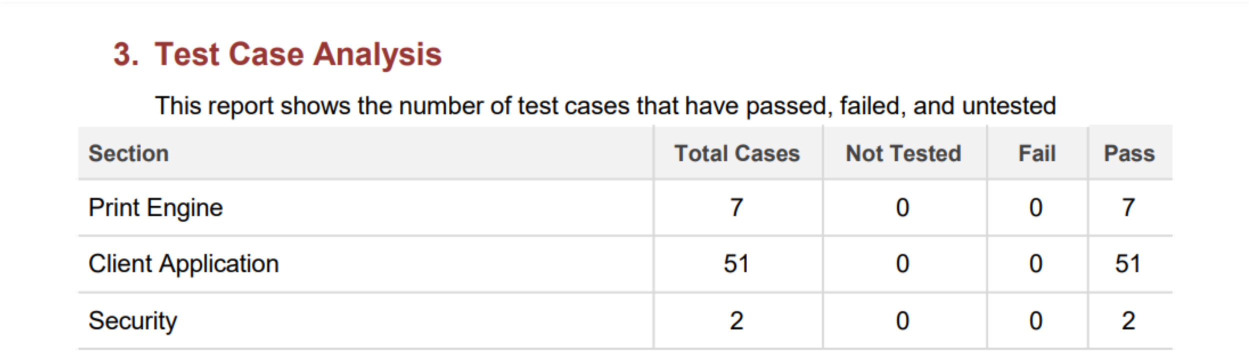
1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the WEB PHISHING DETECTION project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

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

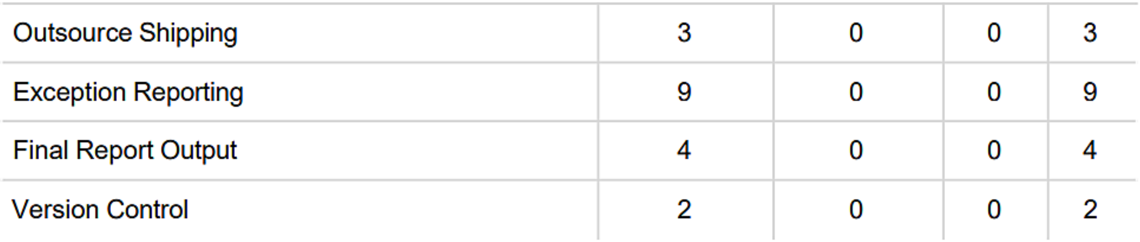
Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77



3. Test Case Analysis

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

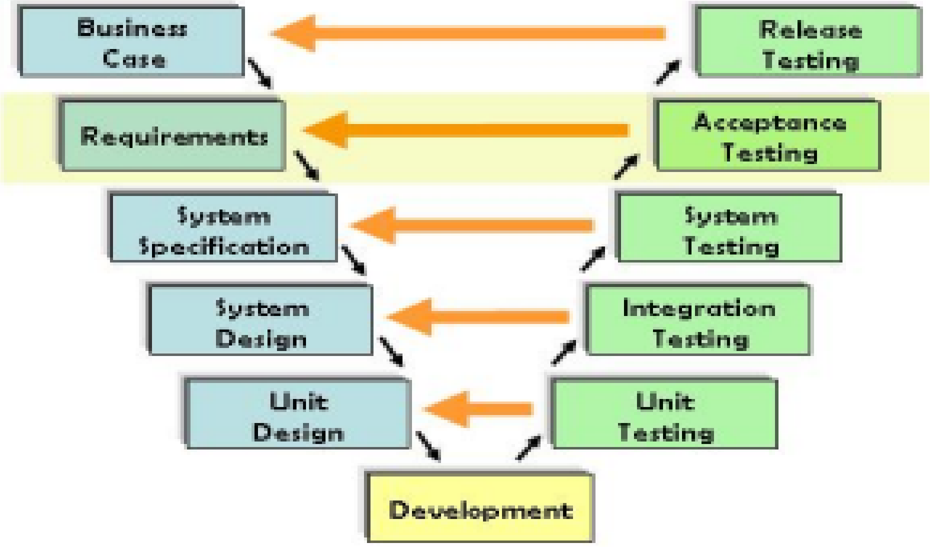
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	51	0	0	51
Security	2	0	0	2



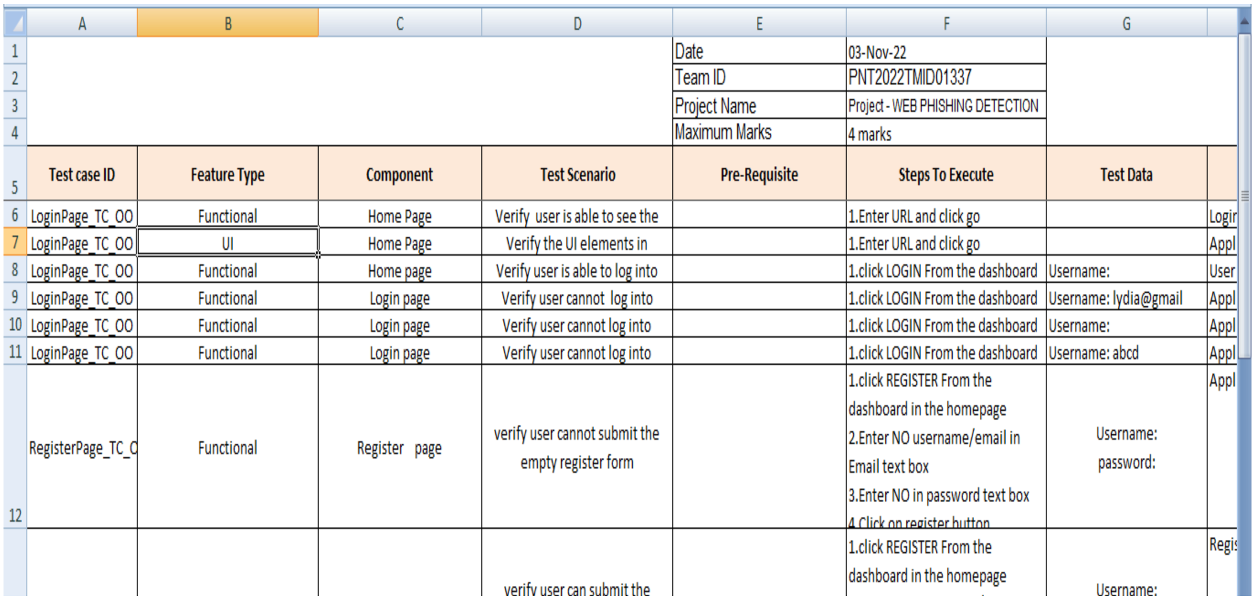
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

8.2 User acceptance testing

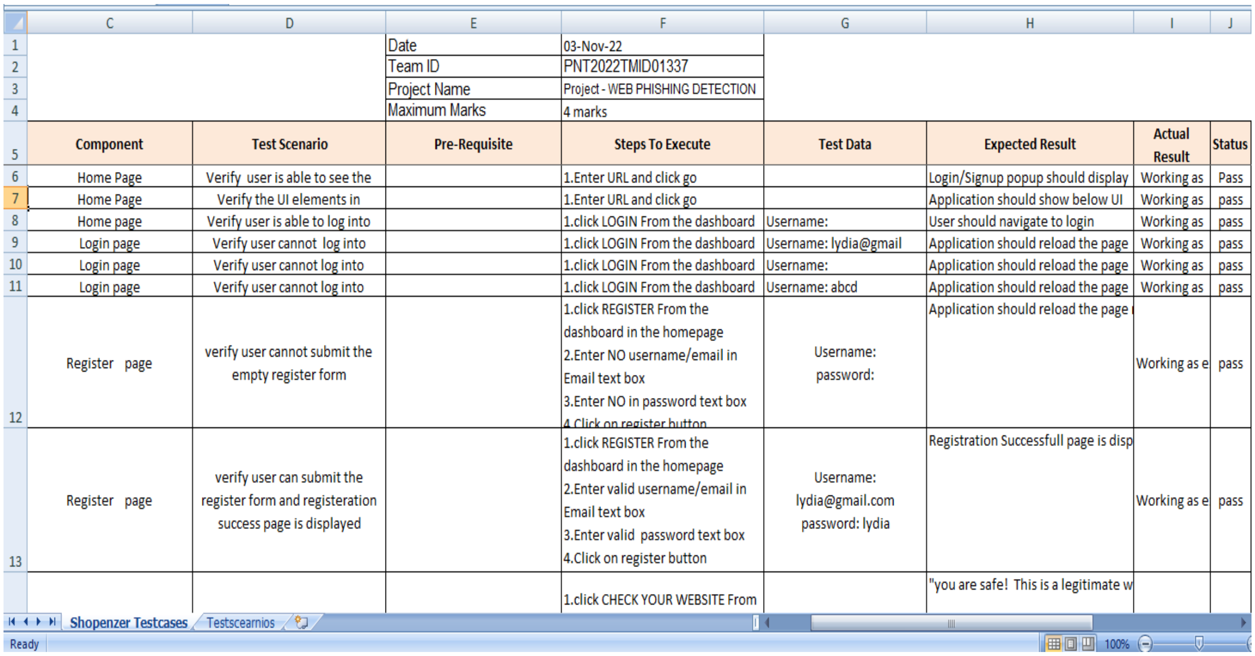
User Acceptance Testing (UAT) is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done.The 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 a kind of black box testing where two or more end-users will be involved.



- Business Requirements must be available.
- Application Code should be fully developed
- Unit Testing, Integration Testing & System Testing should be completed
- No Showstoppers, High, Medium defects in System Integration Test Phase
- Only Cosmetic error is acceptable before UAT
- Regression Testing should be completed with no major defects
- All the reported defects should be fixed and tested before UAT
- Traceability matrix for all testing should be completed
- UAT Environment must be ready
- Sign off mail or communication from System Testing Team that the system is ready for UAT execution



	A	B	C	D	E	F	G	
1					Date	03-Nov-22		
2					Team ID	PNT2022TMID01337		
3					Project Name	Project - WEB PHISHING DETECTION		
4					Maximum Marks	4 marks		
5	Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	
6	LoginPage_TC_OO	Functional	Home Page	Verify user is able to see the		1.Enter URL and click go		Logir
7	LoginPage_TC_OO	UI	Home Page	Verify the UI elements in		1.Enter URL and click go		Appl
8	LoginPage_TC_OO	Functional	Home page	Verify user is able to log into		1.click LOGIN From the dashboard	Username:	User
9	LoginPage_TC_OO	Functional	Login page	Verify user cannot log into		1.click LOGIN From the dashboard	Username: lydia@gmail	Appl
10	LoginPage_TC_OO	Functional	Login page	Verify user cannot log into		1.click LOGIN From the dashboard	Username:	Appl
11	LoginPage_TC_OO	Functional	Login page	Verify user cannot log into		1.click LOGIN From the dashboard	Username: abcd	Appl
12	RegisterPage_TC_C	Functional	Register page	verify user cannot submit the empty register form		1.click REGISTER From the dashboard in the homepage 2.Enter NO username/email in Email text box 3.Enter NO in password text box 4.Click on register button	Username: password:	Appl
				verify user can submit the		1.click REGISTER From the dashboard in the homepage	Username:	Regis

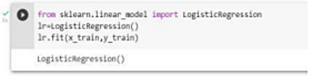
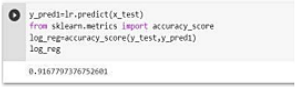
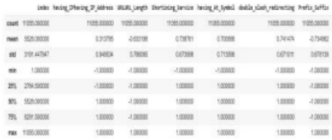
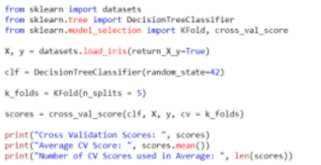
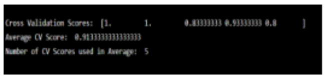


	C	D	E	F	G	H	I	J
1			Date	03-Nov-22				
2			Team ID	PNT2022TMID01337				
3			Project Name	Project - WEB PHISHING DETECTION				
4			Maximum Marks	4 marks				
5	**Component**	**Test Scenario**	**Pre-Requisite**	**Steps To Execute**	**Test Data**	**Expected Result**	**Actual Result**	**Status**
6	Home Page	Verify user is able to see the		1.Enter URL and click go		Login/Signup popup should display	Working as	Pass
7	Home Page	Verify the UI elements in		1.Enter URL and click go		Application should show below UI	Working as	pass
8	Home page	Verify user is able to log into		1.click LOGIN From the dashboard	Username:	User should navigate to login	Working as	pass
9	Login page	Verify user cannot log into		1.click LOGIN From the dashboard	Username: lydia@gmail	Application should reload the page	Working as	pass
10	Login page	Verify user cannot log into		1.click LOGIN From the dashboard	Username:	Application should reload the page	Working as	pass
11	Login page	Verify user cannot log into		1.click LOGIN From the dashboard	Username: abcd	Application should reload the page	Working as	pass
12	Register page	verify user cannot submit the empty register form		1.click REGISTER From the dashboard in the homepage 2.Enter NO username/email in Email text box 3.Enter NO in password text box 4.Click on register button	Username: password:	Application should reload the page	Working as e	pass
13	Register page	verify user can submit the register form and registeration success page is displayed		1.click REGISTER From the dashboard in the homepage 2.Enter valid username/email in Email text box 3.Enter valid password text box 4.Click on register button	Username: lydia@gmail.com password: lydia	Registration Successfull page is disp	Working as e	pass
				1.click CHECK YOUR WEBSITE From		"you are safe! This is a legitimate w		

9. RESULTS

9.1 Performance Metrics

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

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** **supervised learning classification** MAE - , MSE - , RMSE - , R2 score - **Classification Model:** **Logistic Regression** Confusion Matrix - , Accuray Score- & Classification Report -	 
2.	Tune the Model	Hyperparameter Tuning -	
		Validation Method -	 

```
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

```
LogisticRegression()
```

```
y_pred1=lr.predict(x_test)
from sklearn.metrics import accuracy_score
log_reg=accuracy_score(y_test,y_pred1)
log_reg
```

```
0.9167797376752601
```

	index	having_IPhaving_IP_Address	URLURL_Length	Shortining_Service	having_At_Symbol	double_slash_redirecting	Prefix_Suffix
count	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000
mean	5528.000000	0.313795	-0.633198	0.738761	0.700588	0.741474	-0.734962
std	3191.447947	0.949534	0.766095	0.673998	0.713598	0.671011	0.678139
min	1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
25%	2764.500000	-1.000000	-1.000000	1.000000	1.000000	1.000000	-1.000000
50%	5528.000000	1.000000	-1.000000	1.000000	1.000000	1.000000	-1.000000
75%	8291.500000	1.000000	-1.000000	1.000000	1.000000	1.000000	-1.000000
max	11055.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

```
from sklearn import datasets
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import KFold, cross_val_score

X, y = datasets.load_iris(return_X_y=True)

clf = DecisionTreeClassifier(random_state=42)

k_folds = KFold(n_splits = 5)

scores = cross_val_score(clf, X, y, cv = k_folds)

print("Cross Validation Scores: ", scores)
print("Average CV Score: ", scores.mean())
print("Number of CV Scores used in Average: ", len(scores))
```

```
Cross Validation Scores:  [1.         1.         0.83333333 0.93333333 0.8        ]
Average CV Score:  0.9133333333333333
Number of CV Scores used in Average:  5
```

10. ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

- Enhance SEG and phishing awareness training inefficiencies

SEGs and phishing awareness training continue to be essential tools in the fight against malware and phishing. Even even qualified security experts, increasingly complex phishing assaults like BEC are becoming harder to spot. Therefore, there is a pressing need for the channel to offer clients technology that not only aims to prevent intrusion but also has the ability to assist users once an attack has gone through the secure email gateway. By examining account data and learning about users' communication preferences, a mailbox-level anti-phishing solution adds an extra degree of security. This provides a higher level of phishing security to recognise assaults more rapidly, notify users, and eliminate risks as soon as possible.

- It Lightens the Security Team's Load

Customers can now choose from a wide variety of technologies to improve the security of their email. The finest of these use machine learning and artificial intelligence to more accurately identify some of the potential risks. In addition to enhancing security, this can considerably lighten the responsibilities of the IT and security teams. Less than one in five firms, according to a survey by Fidelis Cybersecurity, have a dedicated threat hunting team, and only half of those teams are capable of managing more than eight investigations each day. Security teams need all the assistance they can get, so they must go beyond human intelligence to other tools that can help safeguard the integrity of their company's data.

- It Offers a Solution, Not a Tool

Security aims to provide solutions rather than just bring tools to the table. In the end, resellers must make sure that solutions are effective for the company, which necessitates listening to clients, comprehending the channels, concerns, and how they are affecting the company. Basic tools only provide information and straightforward applications, but automated and sophisticated phishing threat protection solutions can assist in addressing the problems that consumers experience. As a result, the channel is better able to discuss solutions with their clients and provide a broad overview and architectural framework for addressing the problems that are now intrinsic to email security. Automated advanced phishing threat defence uses a system that constantly learns to protect against threats from today and future while increasing security awareness throughout the entire business.

- Differentiate You from Your Rivals

The channel can be hesitant to adopt some of the more cutting-edge technologies, so those who adopt them quickly will have an advantage over their rivals. Simply engaging in these discussions will set you apart from many of your rivals. It displays that you are on top of the most recent dangers and have an understanding of how machine learning and artificial intelligence can strengthen security postures without burdening them more.Given the surge in business email compromise assaults, this is particularly crucial. According to the FBI, these assaults reached record levels in 2017, and they can cost their victims' organisations anywhere between $25,000 and $75,000 on average.

DISADVANTAGES:

- The procedure of detecting phishing can occasionally take a while.
- The majority of web phishing techniques consume a lot of memory.
- There are numerous categorization algorithms, which takes time.
- Huge mail server and lots of memory needed.

11. CONCLUSION:

Use machine learning technologies to improve the detection process for phishing websites. With the least amount of false positives, we used the logistic regression method to reach a detection accuracy of 90%. Additionally, the results demonstrate that classifiers perform better when more data is used as training data. In the future, hybrid technology that combines the blacklist approach with the random forest algorithm of machine learning technology will be utilized to more reliably detect phishing websites.

12. FUTURE SCOPE:

Attacks that use phishing are still among the most important ones that need to be properly addressed. Effective phishing detection models utilising deep learning algorithms have recently been established thanks to advances in these techniques. Even though many distinct models have been created thus far, several problems still have no clear-cut solutions. In order to address nine research issues, we conducted a Systematic Literature Review (SLR) study, looked into 43 high-quality articles, and assessed when and how deep learning algorithms were applied. Additionally, we discussed the difficulties and available fixes for deep learning-based phishing detection models.

Because there was a dearth of a comprehensive overview of deep learning-based phishing detection models, our SLR study addressed nine research issues to fill this gap.43 excellent deep learning-based phishing detection publications were carefully examined, and the necessary information was then taken and combined to address the study questions.Algorithms for deep learning were briefly introduced.The most popular deep learning algorithms, the most popular datasets, the different machine learning architectures, the platforms used for development, the evaluation measures, the methodologies used for validation, the data sources, and the feature selection algorithms were all listed in depth.The difficulties and knowledge gaps were listed.

13. APPENDIX

Source code

```
from flask import Flask, request, jsonify, render_template
import pickle
import numpy as np
import pandas
import inputscript

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

@app.route('/')
def home():
    return render_template('welcomepage.html')

@app.route('/website')
def predict():
    return render_template('website.html')

ans = ""
bns = ""

@app.route('/result_processing_function', methods=['POST','GET'])
def y_predict():
    url = request.form['url']
    checkprediction = inputscript.main(url)
    prediction = model.predict(checkprediction)
    print(prediction)
    output=prediction[0]
    if(output==1):
        pred="You are safe!!  This is a legitimate Website."
        return render_template('website.html',bns=pred)

    else:
        pred="You are on the wrong site. Be cautious!"
        return render_template('website.html',ans=pred)
```

```
@app.route('/predict_api', methods=['POST'])
def predict_api():

    data = request.get_json(force=True)
    prediction = model.y_predict([np.array(list(data.values()))])

    output=prediction[0]
    return jsonify(output)





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

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



@app.route('/login')
def login1():
    return render_template("login_page.html")

database={'lydia@gmail.com':'lydia','janaki@gmail.com':'janaki',

'praveena@gmail.com':'praveena','muthulakshmi@gmail.com':'muthulakshmi'}
```

```
@app.route('/form_login',methods=['POST','GET'])
def login2():
    name1=request.form['email']
    pwd=request.form['psw']
    if name1 not in database:
              return render_template('login_page.html',info='Invalid E-mail')
    else:
        if database[name1]!=pwd:
            return render_template('login_page.html',info='Invalid Password')
        else:
                   return render_template('login_success.html',email=name1)



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 datetime
import requests
import re
```

```
def having_IPhaving_IP_Address(url):
    match=regex.search(
                '(([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-
5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\.([01]?\\d\\d?|2[0-4]\\d|25[0-5])\\/)|'
                '((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})\\/)'
                '(?:[a-fA-F0-9]{1,4}:){7}[a-fA-F0-9]{1,4}',url)

    if match:
      #print match.group()
      return -1
    else:
      #print 'No matching pattern found'
      return 1

"""
Check for the URL length. Return 1 (Legitimate) if the URL length is less than 54
characters
Return 0 if the length is between 54 and 75
Else return -1
"""

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

```
"""
Check with the shortened URLs.
Return -1 if any shortened URLs used.
Else return 1
"""


def Shortining_Service (url):

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

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

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

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

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

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

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

#Checking for @ symbol. Returns 1 if no @ symbol found. Else returns 0.
```

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

#Checking for Double Slash redirections. Returns -1 if // found. Else returns 1

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

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

#Checking for - in Domain. Returns -1 if '-' is found else returns 1.

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

"""
Check the Subdomain. Return 1 if the subDomain contains less than 1 '.'
Return 0 if the subDomain contains less than 2 '.'
Return -1 if the subDomain contains more than 2 '.'
"""

def having_Sub_Domain(url):
    subDomain, domain, suffix = extract(url)
    if(subDomain.count('.')<=2):
        if(subDomain.count('.')<=1):
            return 1
        else:
            return 0
    else:
        return -1

#Checking the SSL. Returns 1 if it returns the respomse code and -1 if exceptions are thrown.

def SSLfinal_State(url):
    try:
        response = requests.get(url)
        return 1
    except Exception as e:
        return -1

#domains expires on ≤ 1 year returns -1, otherwise returns 1

def Domain_registeration_length(url):
    try:
        domain = whois.whois(url)
        exp=domain.expiration_date[0]
        up=domain.updated_date[0]
        domainlen=(exp-up).days
        if(domainlen<=365):
            return -1
        else:
            return 1
    except:
        return -1

#Checking the Favicon. Returns 1 if the domain of the favicon image and the URL domain match else returns -1.

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

#Checking the Port of the URL. Returns 1 if the port is available else returns -1.

def port(url):
    try:
        a_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        location=(url[7:],80)
        result_of_check = a_socket.connect_ex(location)
        if result_of_check == 0:
            return 1
        else:
            return -1
        a_socket.close
    except:
        return -1

# HTTPS token in part of domain of URL returns -1, otherwise returns 1

def HTTPS_token(url):
    match=re.search('https://|http://',url)
    if (match and match.start(0)==0):
        url=url[match.end(0):]
    match=re.search('http|https',url)
    if match:
        return -1
    else:
        return 1



#% of request URL<22% returns 1, otherwise returns -1

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

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

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

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

        if(avg<0.22):
            return 1
        else:
            return -1
    except:
        return -1

#:% of URL of anchor<31% returns 1, % of URL of anchor ≥ 31% and ≤ 67%
returns 0, otherwise returns -1

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

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

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

"""
% of links in <meta>, <script>and<link>tags < 25% returns 1, % of links in
<meta>,
<script> and <link> tags ≥ 25% and ≤ 81% returns 0, otherwise returns -1
"""

def Links_in_tags(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        no_of_meta =0
        no_of_link =0
        no_of_script =0
        anchors=0
        avg =0
        for meta in soup.find_all('meta'):
            no_of_meta = no_of_meta+1
        for link in soup.find_all('link'):
            no_of_link = no_of_link +1
        for script in soup.find_all('script'):
            no_of_script = no_of_script+1
        for anchor in soup.find_all('a'):
            anchors = anchors+1
        total = no_of_meta + no_of_link + no_of_script+anchors
        tags = no_of_meta + no_of_link + no_of_script
        if(total!=0):
            avg = tags/total

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

#Server Form Handling
#SFH is "about: blank" or empty → phishing, SFH refers to a different domain →
suspicious, otherwise → legitimate
def SFH(url):
    #ongoing
    return -1

#:using "mail()" or "mailto:" returning -1, otherwise returns 1
def Submitting_to_email(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        if(soup.find('mailto:','mail():')):
            return -1
        else:
            return 1
    except:
        return -1

#Host name is not in URL returns -1, otherwise returns 1
def Abnormal_URL(url):
    subDomain, domain, suffix = extract(url)
    try:
        domain = whois.whois(url)
        hostname=domain.domain_name[0].lower()
        match=re.search(hostname,url)
        if match:
            return 1
        else:
            return -1
    except:
        return -1

#number of redirect page ≤ 1 returns 1, otherwise returns 0
def Redirect(url):
    try:
        request = requests.get(url)
        a=request.history
        if(len(a)<=1):
            return 1
        else:
            return 0

    except:
        return 0


#onMouseOver changes status bar returns -1, otherwise returns 1
def on_mouseover(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')

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

#right click disabled returns -1, otherwise returns 1
def RightClick(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        if(soup.find_all('script',mousedown=True)):
            return -1
        else:
            return 1
    except:
        return -1

#popup window contains text field → phishing, otherwise → legitimate
def popUpWidnow(url):
    #ongoing
    return 1

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


#:age of domain ≥ 6 months returns 1, otherwise returns -1
def age_of_domain(url):
    try:
        w = whois.whois(url).creation_date[0].year
        if(w<=2018):
            return 1
        else:
            return -1
    except Exception as e:
        return -1

#no DNS record for domain returns -1, otherwise returns 1
def DNSRecord(url):

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

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

#website rank < 100.000 returns 1, website rank > 100.000 returns 0, otherwise returns -1
def web_traffic(url):
    try:
        rank = BeautifulSoup(urllib.request.urlopen("http://data.alexa.com/data?cli=10&dat=s&url=" + url).read(), "lxml").find("REACH")['RANK']
    except TypeError:
        return -1
    rank= int(rank)
    if (rank<100000):
        return 1
    else:
        return 0

#:PageRank < 0,2 → phishing, otherwise → legitimate
def Page_Rank(url):
    #ongoing
    return 1

#webpage indexed by Google returns 1, otherwise returns -1
def Google_Index(url):
    try:
        subDomain, domain, suffix = extract(url)
        a=domain + '.' + suffix
        query = url
        for j in search(query, tld="co.in", num=5, stop=5, pause=2):
            subDomain, domain, suffix = extract(j)
            b=domain + '.' + suffix
        if(a==b):
            return 1
        else:
            return -1
    except:
        return -1


#:number of links pointing to webpage = 0 returns 1, number of links pointing to
webpage> 0
#and ≤ 2 returns 0, otherwise returns -1

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

#:host in top 10 phishing IPs or domains returns -1, otherwise returns 1
def Statistical_report (url):
    hostname = url
    h = [(x.start(0), x.end(0)) for x in
regex.finditer('https://|http://|www.|https://www.|http://www.', hostname)]
    z = int(len(h))
    if z != 0:
        y = h[0][1]
        hostname = hostname[y:]
        h = [(x.start(0), x.end(0)) for x in regex.finditer('/', hostname)]
        z = int(len(h))
        if z != 0:
            hostname = hostname[:h[0][0]]

url_match=regex.search('at\.ua|usa\.cc|baltazarpresentes\.com\.br|pe\.hu|esy\.es|hol\
.es|sweddy\.com|myjino\.ru|96\.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\.4
0|83\.125\.22\.219|46\.242\.145\.98|107\.151\.148\.44|107\.151\.148\.107|64\.70\.19
```

```
\.203|199\.184\.144\.27|107\.151\.148\.108|107\.151\.148\.109|119\.28\.52\.61|54\.8
3\.43\.69|52\.69\.166\.231|216\.58\.192\.225|118\.184\.25\.86|67\.208\.74\.71|23\.25
3\.126\.58|104\.239\.157\.210|175\.126\.123\.219|141\.8\.224\.221|10\.10\.10\.10|43
\.229\.108\.32|103\.232\.215\.140|69\.172\.201\.153|216\.218\.185\.162|54\.225\.10
4\.146|103\.243\.24\.98|199\.59\.243\.120|31\.170\.160\.61|213\.19\.128\.77|62\.113
\.226\.131|208\.100\.26\.234|195\.16\.127\.102|195\.16\.127\.157|34\.196\.13\.28|10
3\.224\.212\.222|172\.217\.4\.225|54\.72\.9\.51|192\.64\.147\.141|198\.200\.56\.183|
23\.253\.164\.103|52\.48\.191\.26|52\.214\.197\.72|87\.98\.255\.18|209\.99\.17\.27|2
16\.38\.62\.18|104\.130\.124\.96|47\.89\.58\.141|78\.46\.211\.158|54\.86\.225\.156|5
4\.82\.156\.19|37\.157\.192\.102|204\.11\.56\.48|110\.34\.231\.42',ip_address)
    except:
        return -1

    if url_match:
        return -1
    else:
        return 1

#returning scrapped data to calling function in app.py
def main(url):



    check = [[having_IPhaving_IP_Address
(url),URLURL_Length(url),Shortining_Service(url),having_At_Symbol(url),

double_slash_redirecting(url),Prefix_Suffix(url),having_Sub_Domain(url),SSLfina
l_State(url),

Domain_registeration_length(url),Favicon(url),port(url),HTTPS_token(url),Reques
t_URL(url),
URL_of_Anchor(url),Links_in_tags(url),SFH(url),Submitting_to_email(url),Abnormal_URL(url),

Redirect(url),on_mouseover(url),RightClick(url),popUpWidnow(url),Iframe(url),

age_of_domain(url),DNSRecord(url),web_traffic(url),Page_Rank(url),Google_Index(url),
            Links_pointing_to_page(url),Statistical_report(url)]]


    print(check)
    return check
```

Links:

github - https://github.com/IBM-EPBL/IBM-Project-4635-1658736526

Demo - https://drive.google.com/file/d/1GzgIs03NtqqpEaFuxgCuSZd43r7WhHyx/view?usp=drivesdk

(We have also uploaded the same video on gitHub too, incase of any technical issues while accessing the demo link.)