RMK ENGINEERING COLLEGE

(An Autonomous Institution)

R.S.M. Nagar, Kavaraipettai-601 206, Gummidipoondi Taluk, Thiruvallur District.

PROJECT

WEB PHISHING DETECTION

DONE BY

TEAM ID: PNT2022TMID15890

VASANTHAZHAGAN R A (111719104170)

SHRINIWAAZ K G (111719104144)

SATHISH S (111719104138)

THARUN M V (111719104163)

INDEX

1. INTRODUCTION

For an average individual, it is very difficult to differentiate a phishing website from a legitimate website. It is not feasible to identify a fake website just by looking at them because they look similar to the original ones. This makes it hard to trust any websites. Users also share sensitive information like passwords, card details, etc. in websites in order to utilize the websites. Presence of these fake makes us feel less secure. These phishing sites also affects the reputation of the original website and its organization. Users feel frustrated, unsecured, confused, stressed, worried, annoyed, etc. which makes them give up.

1.1. Project Overview

These days most sites ask users for their personal data to process further. There are times where these data can be stolen. Those stolen data are often used to threaten the user mainly for ransom. One such way of stealing data is Web Phishing. The harassers create a fraudulent website that resembles the original websites and make the users to type in the data by scamming them. Thereby, stealing the user's data.

1.2. Purpose

Web Phishing Detection is a technology where different machine learning algorithm can be used to differentiate a fraudulent website from an authentic one. This really helps users getting robbed of their personal data and prevent them from cyber harassers.

2. LITERATURE SURVEY

2.1. Existing problem

Detecting Phishing Websites Using Machine Learning [1] by Amani Alswailem, Bashayr Alabdullah, Norah Alrumayh, Dr. Aram Alsedrani

The system is based on a machine learning method, particularly supervised learning. We have selected the Random Forest technique due to its good performance in classification. Accuracy of 98.8% and combination of 26 features. There are 36 features that can features that can be extracted from URL, page content and page rank. Using the combinations of these features irrelevant features are removed.

Result - Random combination of features and found it took the shape of normal distribution curve.

Review: Phishing Detection Approaches [2] by AlMaha Abu Zuraiq, Mouhammd Alkasassbeh

Uses different phishing detection approaches which include: Content-Based, Heuristic-Based, and Fuzzy rule-based approaches. Content-based approach does a deep analysis on pages' content. Heuristic Based Approach s discriminative features extracted by understanding and analyzing the structure of phishing web pages.

Result - Each approach has its advantages and disadvantages and improving these approaches is always required.

Real Time Detection of Phishing Websites [3] by Abdulghani Ali Ahmed, Nurul Amirah Abdullah

Proposes a detection technique of phishing websites based on checking Uniform Resources Locators (URLs) of web pages by checking the Uniform Resources Locators (URLs) of suspected web pages. There are few features that used to identify fake site from a legitimate one. Some are URLs, domain identity, security & encryption, source code, page style & contents, web address bar and social human factor. Features of URL and domain names are checked using several criteria such as IP Address, long URL address, adding a prefix or suffix, redirecting using the symbol "//", and URLs having the symbol "@".

Result - The paper checks the authenticity of the Universal Resource Locator (URLs) based only on few characteristics for detecting phishing attacks.

Detection of Phishing Websites by using Machine Learning-Based URL Analysis [4] by Mehmet Korkmaz, Ozgur Koray Sahingoz, Banu Diri

Aimed to implement a phishing detection system by analyzing the URL of the webpage. detected 58 different features on the web URL which included words, digits, "=", "?", IP address, etc. They implemented the system by using 8 different algorithms Logistic Regression (LR), K-Nearest Neighborhood (KNN), Support Vector Machine (SVM), Decision Tree (DT), Naive Bayes (NB), XGBoost, Random Forest (RF) and Artificial Neural Network (ANN).

Result - They used 3 datasets and obtained results in 8 different algorithms. Random Forest (RF) is the one seems to produce highest accuracy rate with 94.59%, 90.5%, 91.26% in the 3 datasets respectively.

Phishing Website Detection using Machine Learning Algorithms [5] Rishikesh Mahajan, Irfan Siddavatam

Deals with ML technologies for detection of phishing URLs by extracting and analyzing different features of legitimate and phishing URL. Python programming language is used to extract the features from URL. This paper talks about 3 machine learning algorithm Decision Tree, Random Forest and Support Vector Machine.

Result - Achieved 97.14% detection accuracy using Random Forest Algorithm with lowest false positive rate.

Phishing website detection using novel machine learning fusion approach [6] by A. Lakshmanarao,P.Surya Prabhakara Rao,M M Bala Krishna

Various machine learning algorithms like logistic regression, decision tree classifier, random forest classifier, AdaBoost, gradient boosting classifier for the phishing detection. A dataset from the UCI machine learning repository for the experiments. Two priority algorithms PA1, PA2. Based on the results of priority-based algorithms final fusion model was decided.

Result - A fusion classifier and achieved an accuracy of 97%. The proposed model was tested on one dataset only.

2.2.References

1. Amani Alswailem, Bashayr Alabdullah, Norah Alrumayh, Dr. Aram Alsedrani ‘s ***“Detecting Phishing Websites Using Machine Learning”*** published during 2019 2nd International Conference on Computer Applications & Information Security (ICCAIS), 01-03 May 2019.

2. AlMaha Abu Zuraiq, Mouhammd Alkasassbeh ‘s ***“Review: Phishing Detection*** **Approaches”** published during 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS), 09-11 October 2019.

3. Abdulghani Ali Ahmed, Nurul Amirah Abdullah ‘s ***“Real Time Detection of Phishing Websites”*** published during 2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), 13-15 October 2016.

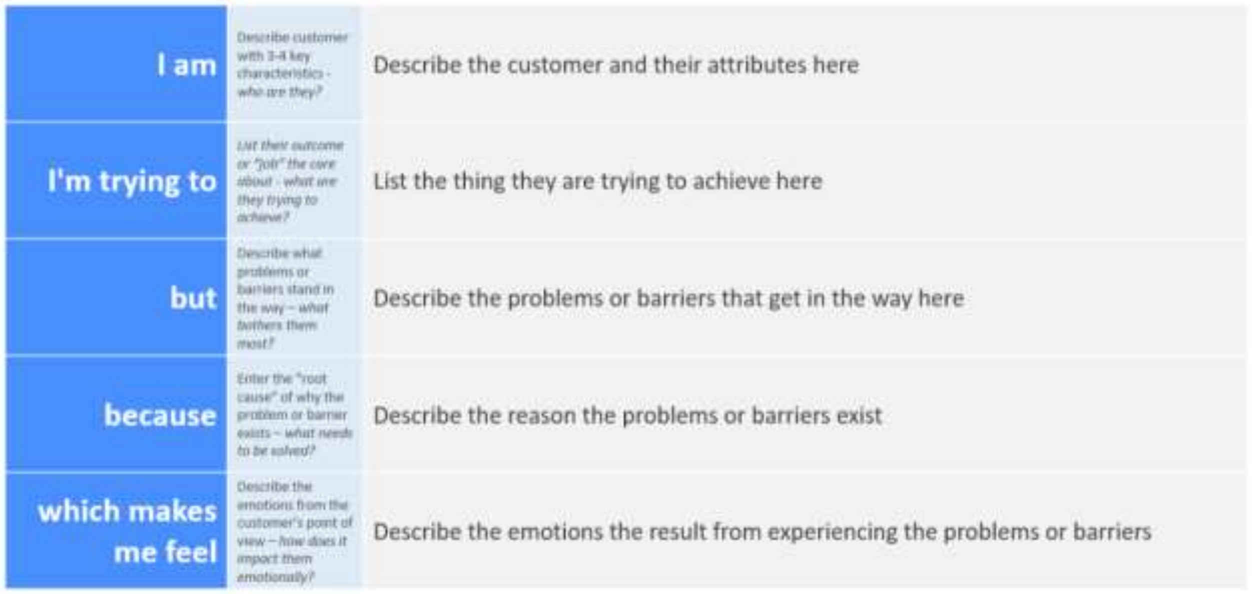
4. Mehmet Korkmaz, Ozgur Koray Sahingoz, Banu Diri ‘s ***“Detection of Phishing Websites by using Machine Learning-Based URL Analysis”*** published during 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 01-03 July 2020.

5. Rishikesh Mahajan, Irfan Siddavatam ‘s ***“Phishing Website Detection using Machine Learning Algorithms”*** published during International Journal of Computer Applications, October 2018.

6. A. Lakshmanarao,P.Surya Prabhakara Rao,M M Bala Krishna ‘s ***“Phishing website detection using novel machine learning fusion approach”*** published during 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), 25-27 March 2021.

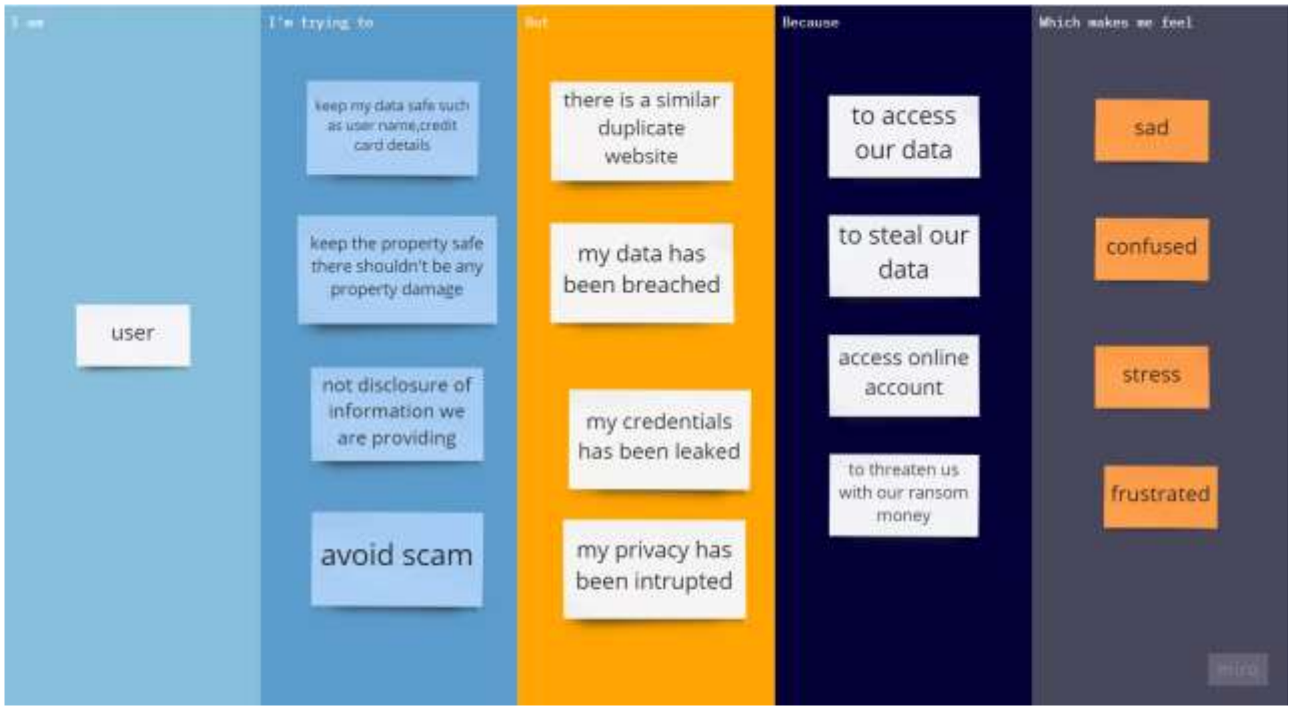
2.3.Problem Statement Definition

Problem Statement defines the problem in hand. It helps the developers to understand the problem from the customer point of view and assist them in determining the ideal solution.



I am	*Describe customer with 3-4 key characteristics - who are they?*	Describe the customer and their attributes here
I'm trying to	*List their outcome or "job" the care about - what are they trying to achieve?*	List the thing they are trying to achieve here
but	*Describe what problems or barriers stand in the way – what bothers them most?*	Describe the problems or barriers that get in the way here
because	*Enter the "root cause" of why the problem or barrier exists – what needs to be solved?*	Describe the reason the problems or barriers exist
which makes me feel	*Describe the emotions from the customer's point of view – how does it impact them emotionally?*	Describe the emotions the result from experiencing the problems or barriers

2.2.1 Problem Statement

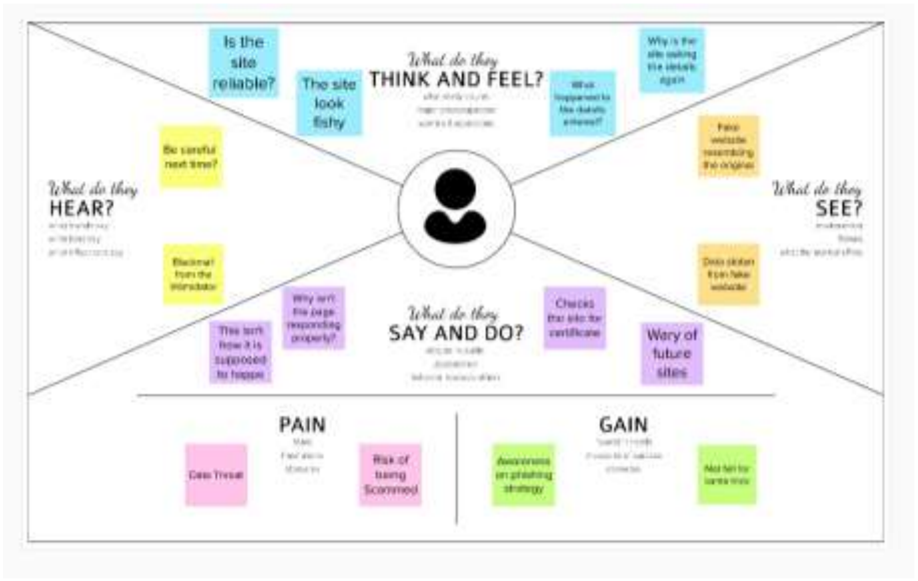

I am
I'm trying to
But
Because
Which makes me feel
user
keep my data safe such as user name,credit card details
keep the property safe there shouldn't be any property damage
not disclosure of information we are providing
avoid scam
there is a similar duplicate website
my data has been breached
my credentials has been leaked
my privacy has been intrupted
to access our data
to steal our data
access online account
to threaten us with our ransom money
sad
confused
stress
frustrated
miro

2.2.2 User Problem Statement

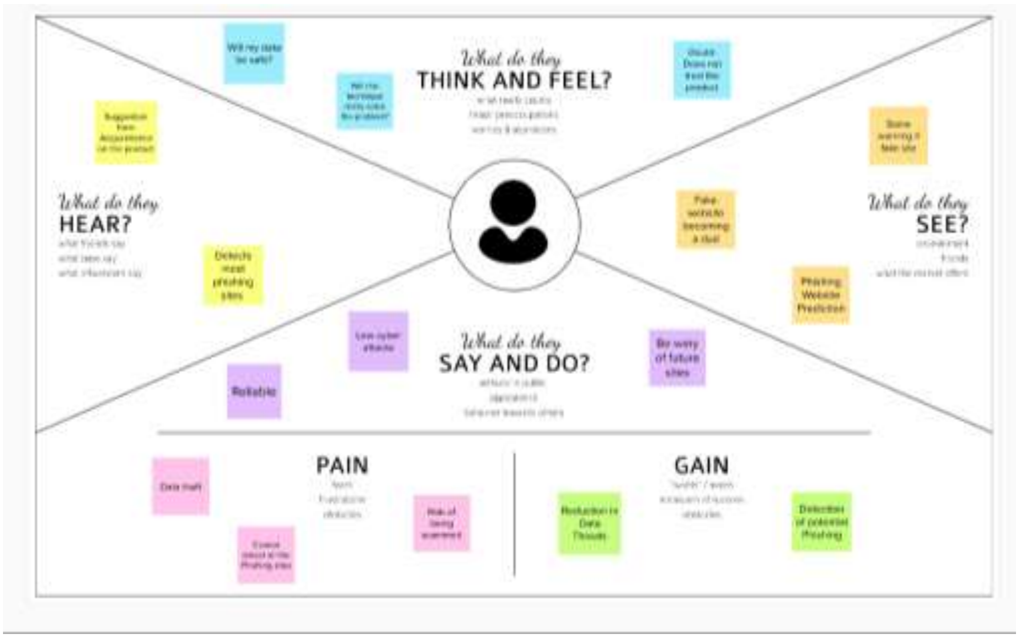
3. IDEATION & PROPOSED SOLUTION

3.1. Empathy Map Canvas

Empathy map is a way of determining the user's behavior and attitude. Web phishing concerns users and their data. So, it is required that the developers of the web phishing detection application understand user's perspective on web phishing and the application.



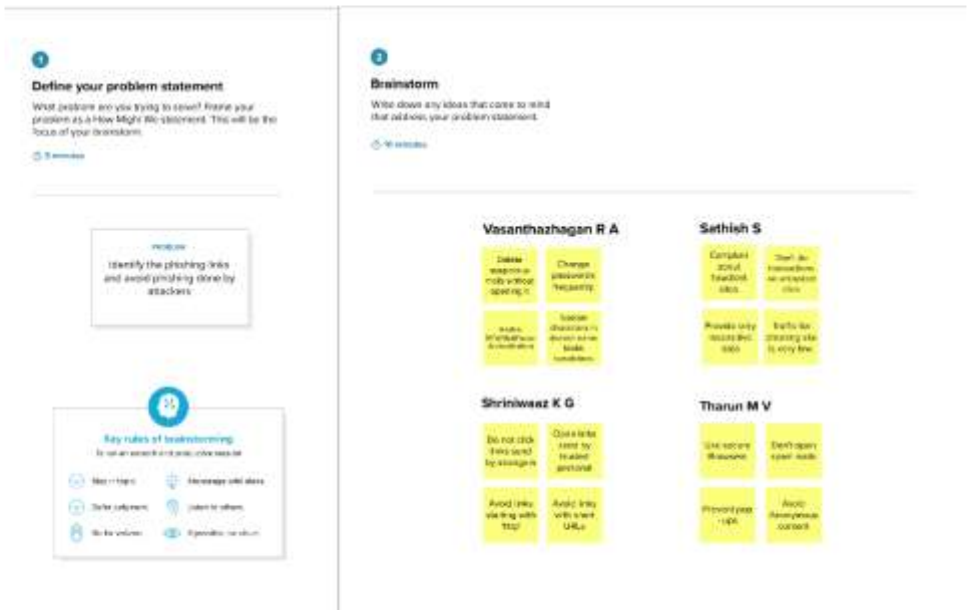
3.1.1 Empathy Map on Web Phishing



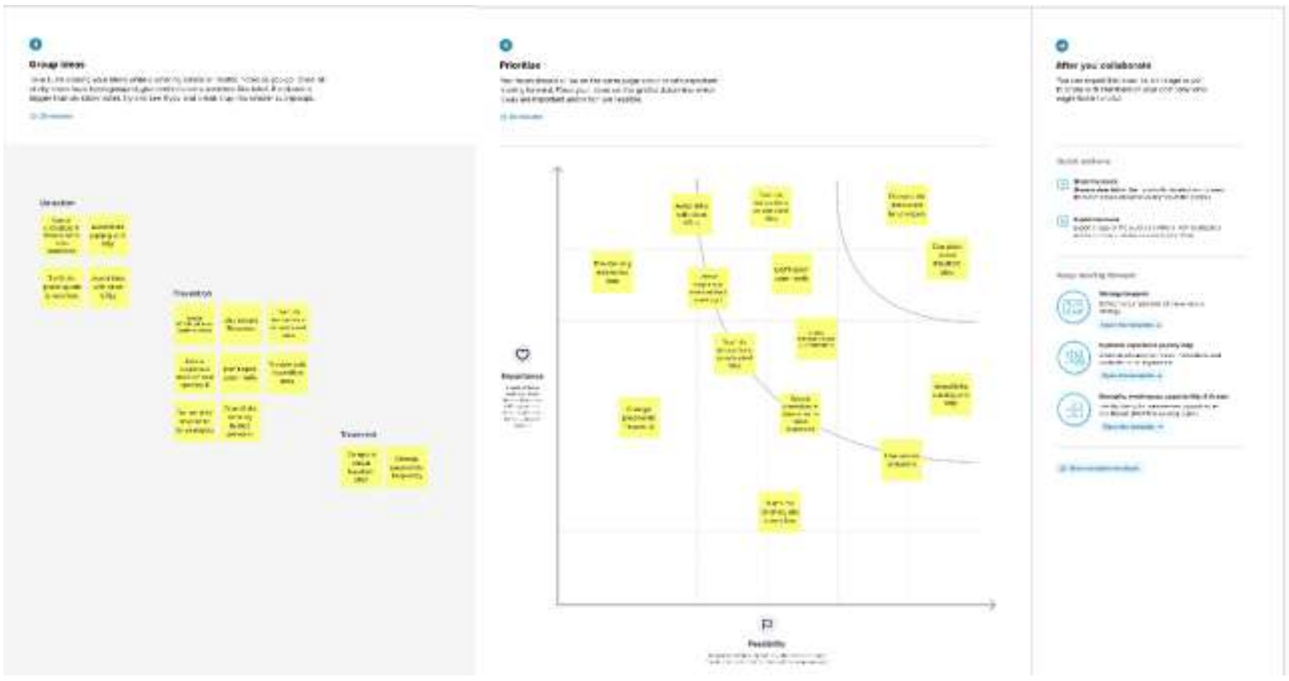
3.1.2 Empathy Map on Web Phishing Detection System

3.2. Ideation & Brainstorming

Finding the solution is not necessarily a single step process. It requires all the minds in the team to come together to find a solution. Ideation & Brainstorming is the phase where each member comes up with their idea and suggestion and the effective one is chosen.


Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
PROBLEM
Identify the phishing links and avoid phishing done by attackers
Key rules of brainstorming
Brainstorm
Write down any ideas that come to mind that address your problem statement.
Vasanthazhagan R A
Change passwords frequently
Sathish S
Provide only secured web sites
Shriniwaaz K G
Do not click links send by strangers
Click links send by trusted personal
Avoid links starting with http
Avoid links with short URLs
Tharun M V
Use secure Browser
Don't open spam mails
Prevent pop-ups
Avoid Anonymous content

3.2.1 Ideation a)



3.2.2 Ideation b)

3.3. Proposed Solution

Proposed solution is the solution that the developers comes up with from the ideas from the team members. It includes several areas like solution description, uniqueness, customer satisfaction and scalability.

The proposed solution describes an application that analyses the URL of the given site and produces a report showing if the website is a phishing website or an original one. The solution will detect 30 different features related to the given URL for the website like length, IP address, HTTPS token, page rank, google index etc. Then uses these features to detect if it is a phishing website or not. We have tested with three algorithms - Logistic Regression, Decision tree and Random Forest Tree. Among them, Random Forest Tree has given best result with the highest accuracy. It is an application to secure users from phishing websites.

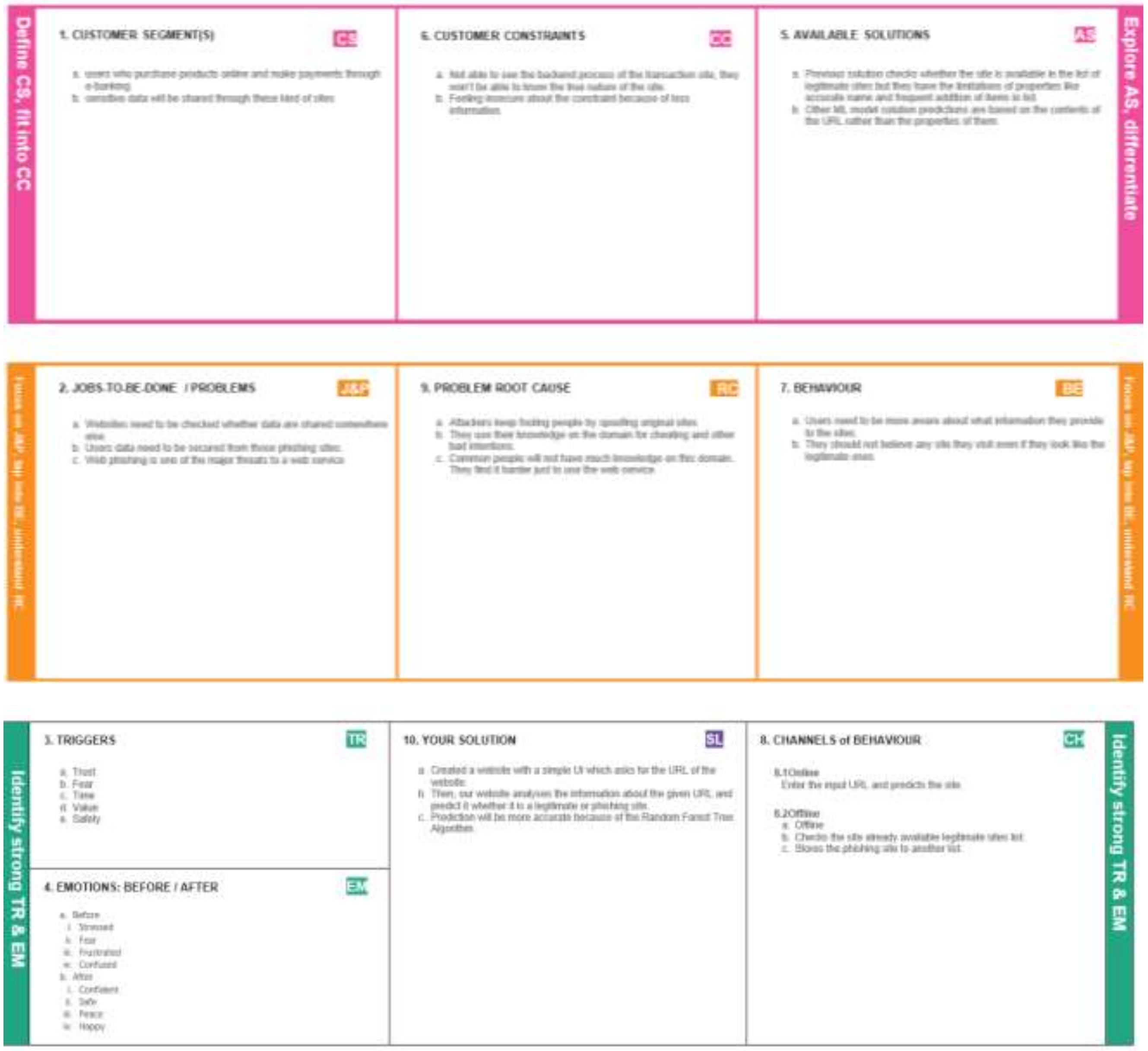
Sl. No.	**Parameter**	**Description**
1.	Problem Statement (Problem to be solved)	For an average individual, it is very difficult to differentiate a phishing website from a legitimate website. It is not feasible to identify a fake website just by looking at them because they look similar to the original ones. This makes it hard to trust any websites. Users also share sensitive information like passwords, card details, etc in websites in order to utilize the websites. Presence of these fake makes us feel less secure. These phishing sites also affects the reputation of the original website and its organisation. Users feel frustrated, unsecured, confused, stressed, worried, annoyed, etc. which makes them give up.
2.	Idea / Solution description	The solution to the problems described is a Phishing Website Detection application which analyses the URL of the site and give report about the site. It will detect 30 different features related to the URL of the website like length, IP address, HTTPS token etc. We had tested with three algorithms like Logistic Regression, Decision tree, Random Forest Tree. Among them, Random Forest Tree has given best result with more accuracy. It is an application to secure users from phishing websites.

3.	Novelty / Uniqueness	It uses Random Forest Tree algorithm, which provided higher accuracy rate among others with concept of binary classification on the website's properties like IP address, HTTPS token, etc to identify the originality of the websites.
4.	Social Impact / Customer Satisfaction	Phishing sites created fear among users' mind about losing their sensitive data. It has major impact on user's mental health and their privacy. By identifying the fake certificates, the user can access and provide information without any fear. It makes them feel that their data was not leaked. It provides immense customer satisfaction as it is very user-friendly and is easily accessible. It is a place where user can find security of the website.
5.	Business Model (Revenue Model)	The revenue is generated usage of the application. More the user using the application the more the revenue generated. If successful, it can also be developed as a browser extension.
6.	Scalability of the Solution	This application is very scalable as it can identify fake websites and original websites using binary classification ML models.

3.3.1 Table: Proposed Solution

3.4.Problem Solution fit

Problem solution fit is the one solution amongst many that actually solves the defined user problem



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

a. users who purchase products online and make payments through e-banking.
b. sensitive data will be shared through these kind of sites

6. CUSTOMER CONSTRAINTS CC

a. Not able to see the backend process of the transaction site, they won't be able to know the true nature of the site.
b. Feeling insecure about the constraint because of less information.

5. AVAILABLE SOLUTIONS AS

a. Previous solution checks whether the site is available in the list of legitimate sites but they have the limitations of properties like accurate name and frequent addition of items in list.
b. Other ML model solution predictions are based on the contents of the URL rather than the properties of them.

Explore AS, differentiate

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

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

a. Websites need to be checked whether data are shared somewhere else.
b. Users data need to be secured from those phishing sites.
c. Web phishing is one of the major threats to a web service.

9. PROBLEM ROOT CAUSE RC

a. Attackers keep fooling people by spoofing original sites.
b. They use their knowledge on the domain for cheating and other bad intentions.
c. Common people will not have much knowledge on this domain. They find it harder just to use the web service.

7. BEHAVIOUR BE

a. Users need to be more aware about what information they provide to the sites.
b. They should not believe any site they visit even if they look like the legitimate ones.

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

Identify strong TR & EM

3. TRIGGERS TR

a. Trust
b. Fear
c. Time
d. Value
e. Safety

4. EMOTIONS: BEFORE / AFTER EM

a. Before
 i. Stressed
 ii. Fear
 iii. Frustrated
 iv. Confused
b. After
 i. Confident
 ii. Safe
 iii. Peace
 iv. Happy

10. YOUR SOLUTION SL

a. Created a website with a simple UI which asks for the URL of the website.
b. Then, our website analyses the information about the given URL and predict it whether it is a legitimate or phishing site.
c. Prediction will be more accurate because of the Random Forest Tree Algorithm.

8. CHANNELS of BEHAVIOUR CH

8.1Online
Enter the input URL and predicts the site.

8.2Offline
a. Offline
b. Checks the site already available legitimate sites list.
c. Stores the phishing site to another list.

Identify strong TR & EM

3.4.1 Problem Solution Fit

4. REQUIREMENT ANALYSIS

4.1. Functional requirement

Functional requirements include the functionality and the functional features that are required for the proposed solution.

FR No.	Functional Requirement	Sub Requirement (Story / Sub-Task)
FR-1	**User Registration**	Registration through Gmail
FR-2	**Registered User Login**	Login through Gmail and Password
FR-3	**Data Fetch**	Get the URL of the website need to be verified
FR-4	**Result Prediction**	Predict the category of the website using the ML model trained under a certain dataset
FR-5	**Storing Result**	Store the outcome in the appropriate place
FR-6	**Displaying Result**	Notify the user about the nature of the given website

4.1.1 Table: Functional Requirement

4.2. Non-Functional requirements

Non-Functional requirements include the non-functionality features that are required for the proposed solution.

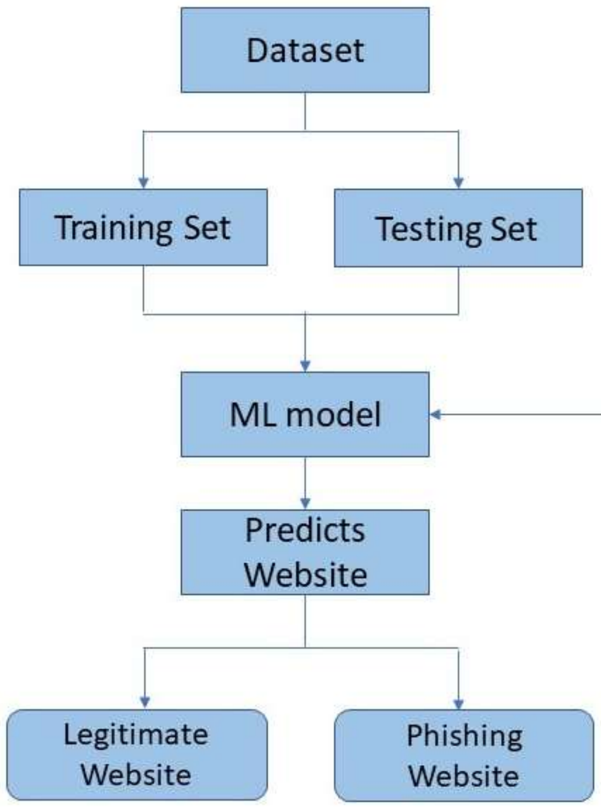
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Users will be able to register and access the website for detecting phishing sites easily. The User Interface should be simple and usable at ease.
NFR-2	**Security**	Data provided by the user should be secure and highest priority should be given to protect those data. Also, admin and authorized personnel can only be allowed to access those data.
NFR-3	**Reliability**	The data will be updated after every prediction. So, the database will be refreshed more often by adding more new websites.
NFR-4	**Performance**	The website shall be simple, dynamic and responsive. While the authentication of the website and other internal process will be done on background.
NFR-5	**Availability**	The website should be developed to identify phishing websites mainly among the payment and banking related domain. These are to be made available to user easily.
NFR-6	**Scalability**	The website shall be accessed by a greater number of users at the same time.

4.2.1 Table: Non - Functional Requirement

5. PROJECT DESIGN

5.1.Data Flow Diagrams

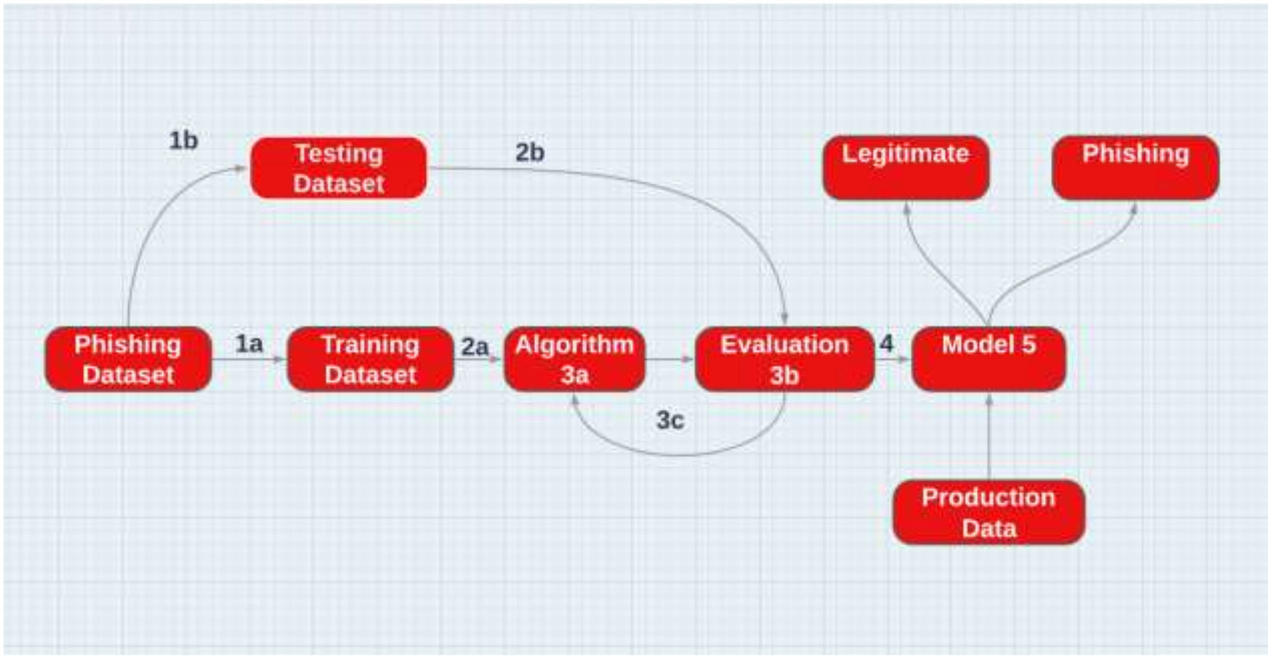
Data flow diagrams are the way of representing the flow of data and information in the system. It depicts the way in which the data travels through the system, how the data enters the system, what happens to the data in - between, where the data are stored.



5.1.1 Data Flow Diagram

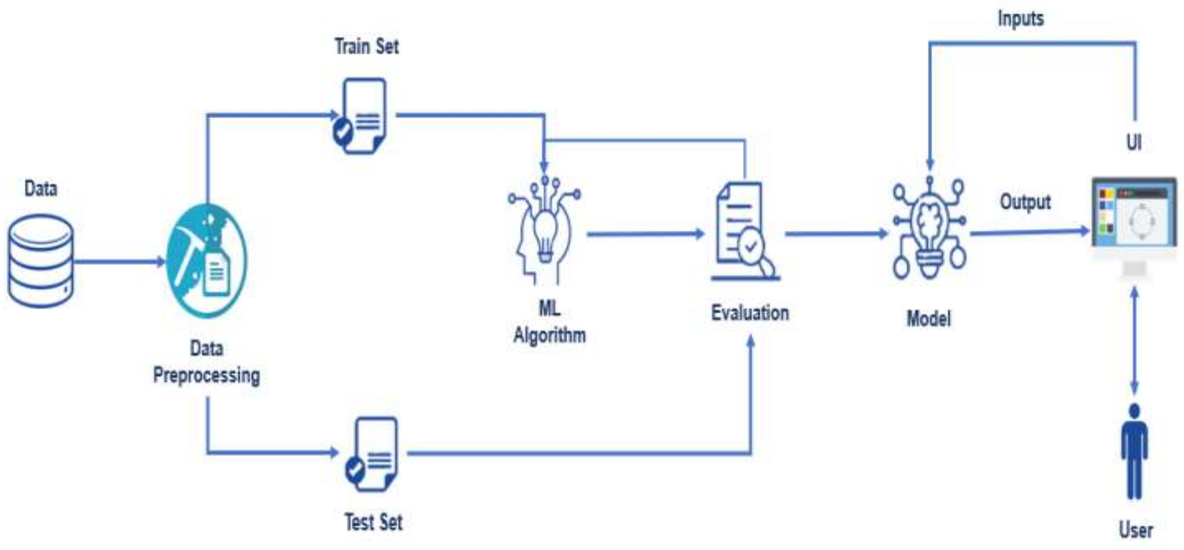
5.2. Solution & Technical Architecture

Solution architecture is a way of dividing the processes in the application into smaller sub processes and provides the specification for understanding and defining the solution.



5.2.1 Solution Architecture

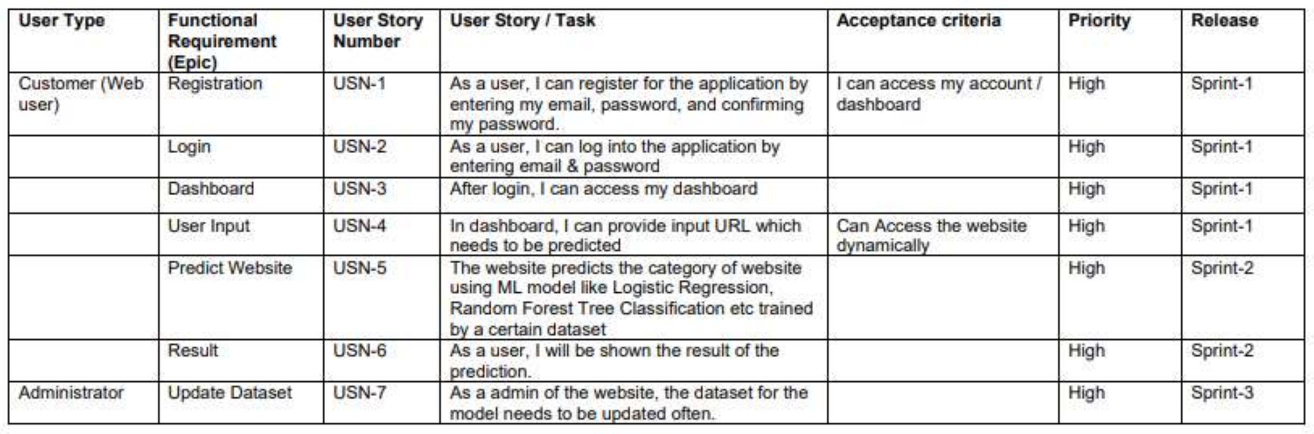
Technical Stack or Technical Architecture defines about the components and technologies used along with the characteristics of the application. Technical Architecture depicts the neat flow of the technologies being used.



5.2.2 Technical Architecture/Stack

5.3. User Stories

User stories represent the stories and actions that are actually performed by the user while using the application. It helps developers to gain knowledge on the user's perspective and determine how they use them.



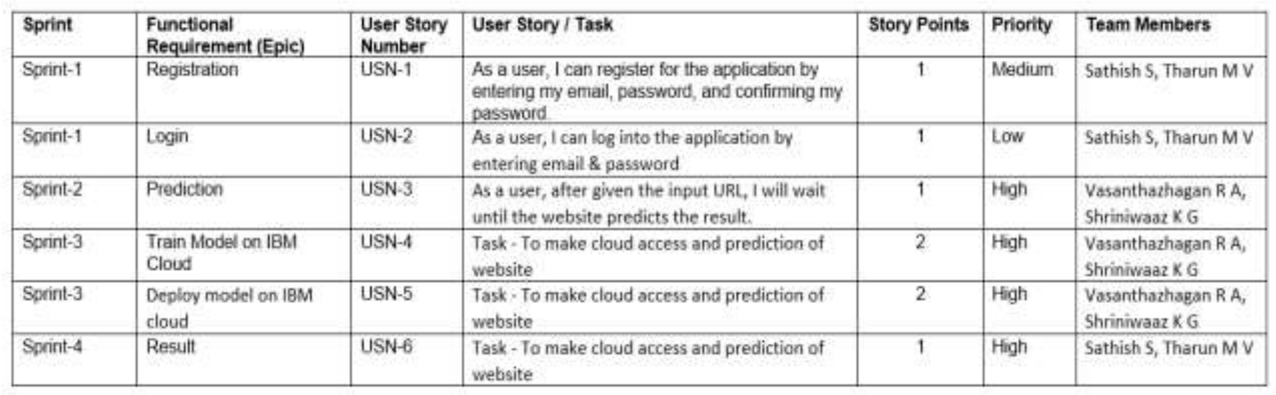
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
	Login	USN-2	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard	USN-3	After login, I can access my dashboard		High	Sprint-1
	User Input	USN-4	In dashboard, I can provide input URL which needs to be predicted	Can Access the website dynamically	High	Sprint-1
	Predict Website	USN-5	The website predicts the category of website using ML model like Logistic Regression, Random Forest Tree Classification etc trained by a certain dataset		High	Sprint-2
	Result	USN-6	As a user, I will be shown the result of the prediction.		High	Sprint-2
Administrator	Update Dataset	USN-7	As a admin of the website, the dataset for the model needs to be updated often.		High	Sprint-3

5.3.1 Table: User Stories

6. PROJECT PLANNING & SCHEDULING

6.1. Sprint Planning & Estimation

Sprint planning is used to create a plan or schedule with an estimated time for completion that can be completed within the time period. The plan created shall segregate task for each team members.

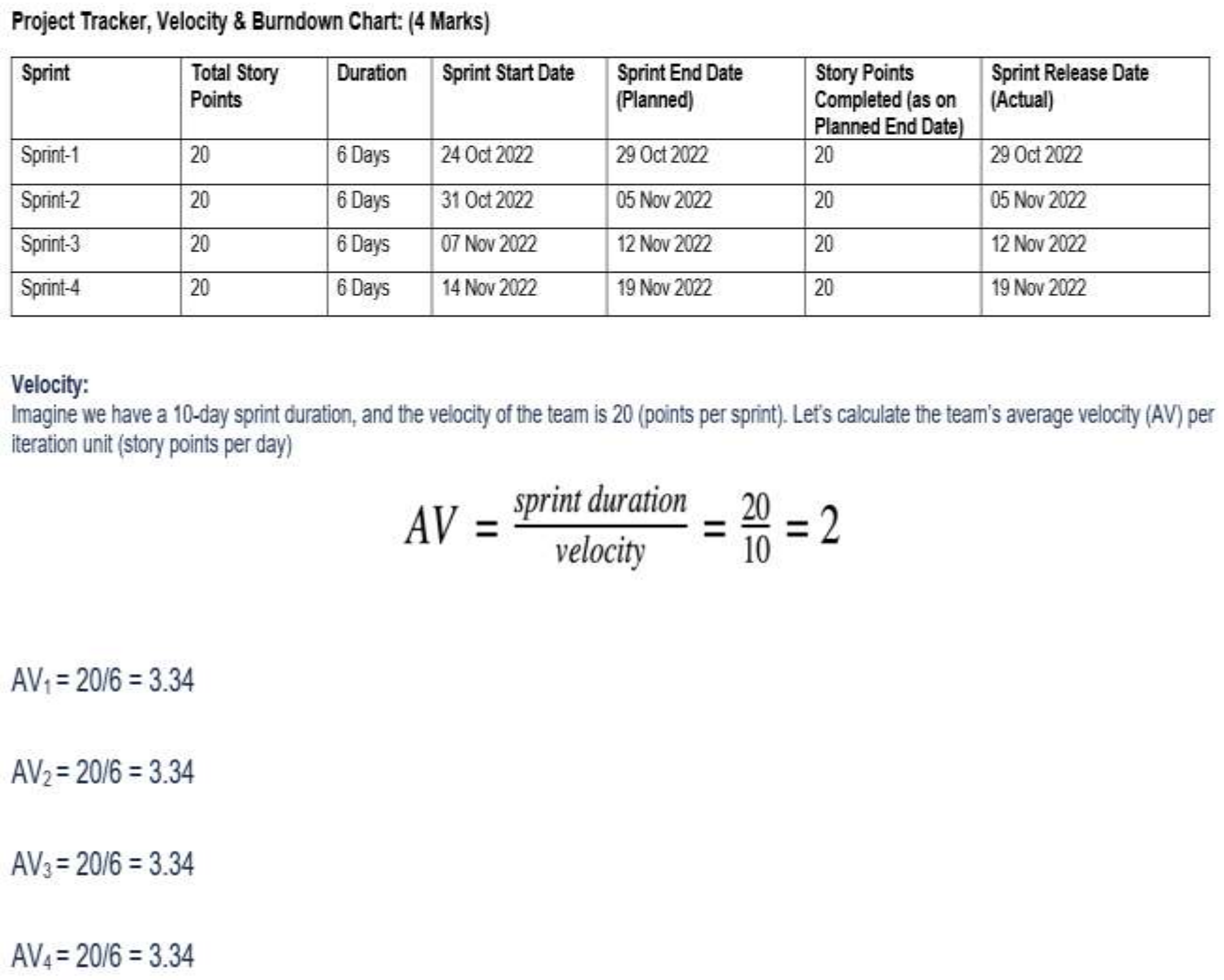


Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	1	Medium	Sathish S, Tharun M V
Sprint-1	Login	USN-2	As a user, I can log into the application by entering email & password	1	Low	Sathish S, Tharun M V
Sprint-2	Prediction	USN-3	As a user, after given the input URL, I will wait until the website predicts the result.	1	High	Vasanthazhagan R A, Shriniwaaz K G
Sprint-3	Train Model on IBM Cloud	USN-4	Task - To make cloud access and prediction of website	2	High	Vasanthazhagan R A, Shriniwaaz K G
Sprint-3	Deploy model on IBM cloud	USN-5	Task - To make cloud access and prediction of website	2	High	Vasanthazhagan R A, Shriniwaaz K G
Sprint-4	Result	USN-6	Task - To make cloud access and prediction of website	1	High	Sathish S, Tharun M V

6.1.1 Table: Sprint Planning & Estimation

6.2. Sprint Delivery Schedule

Sprint Delivery Schedule gives the report for the Sprint Planning & Estimation along with a burn down chart.



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

Velocity:

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

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

$AV_1 = 20/6 = 3.34$

$AV_2 = 20/6 = 3.34$

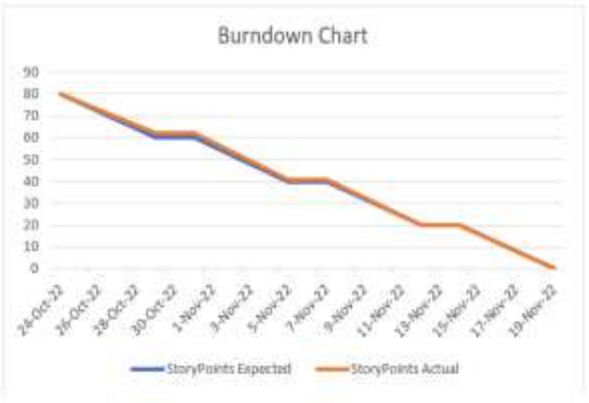
$AV_3 = 20/6 = 3.34$

$AV_4 = 20/6 = 3.34$

6.2.1 Sprint Delivery Schedule

Burndown Chart:

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

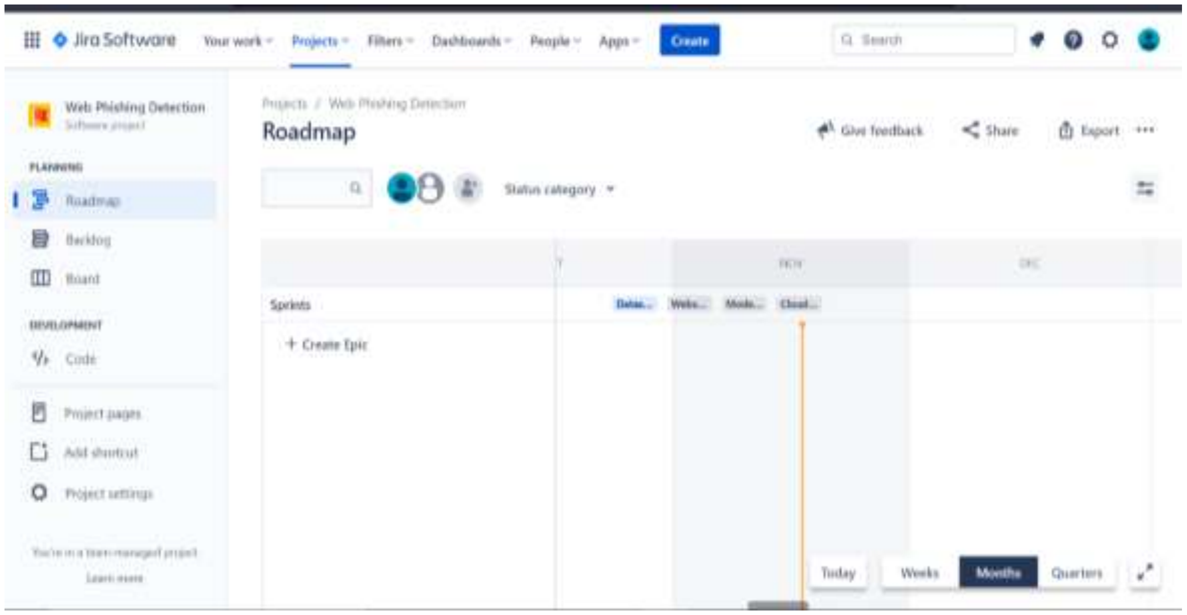


6.2.2 Burndown Chart

6.3.Reports from JIRA

Roadmap:

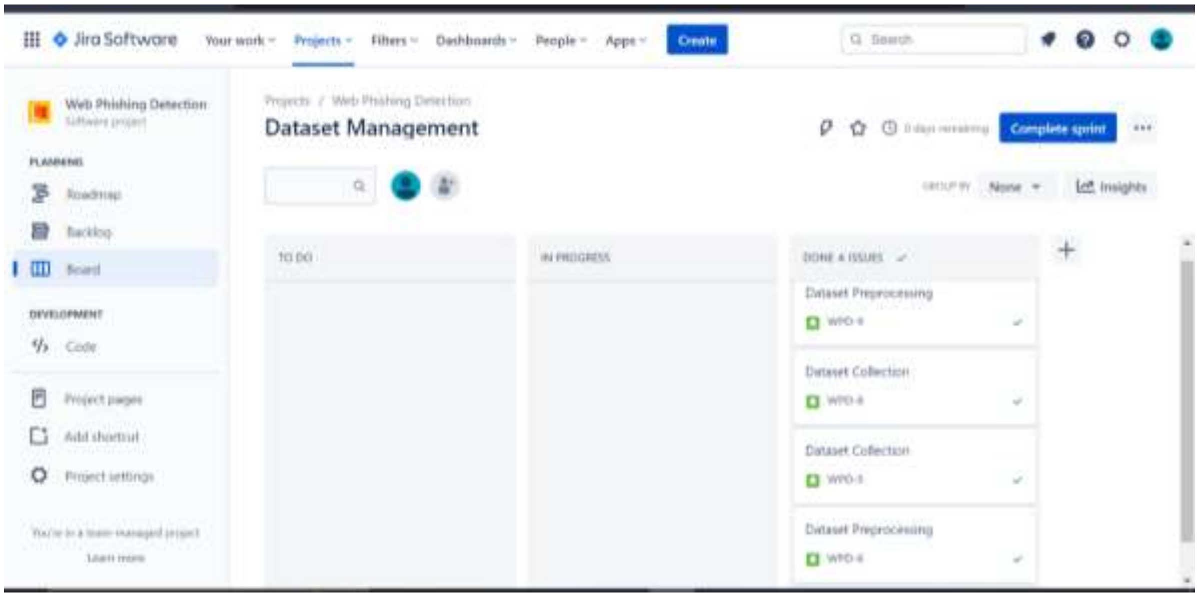
Roadmap depicts the flow of the project as per the duration scheduled.



6.3.1 Roadmap

Board:

Board contains the detailed view on each sprint of the backlog.



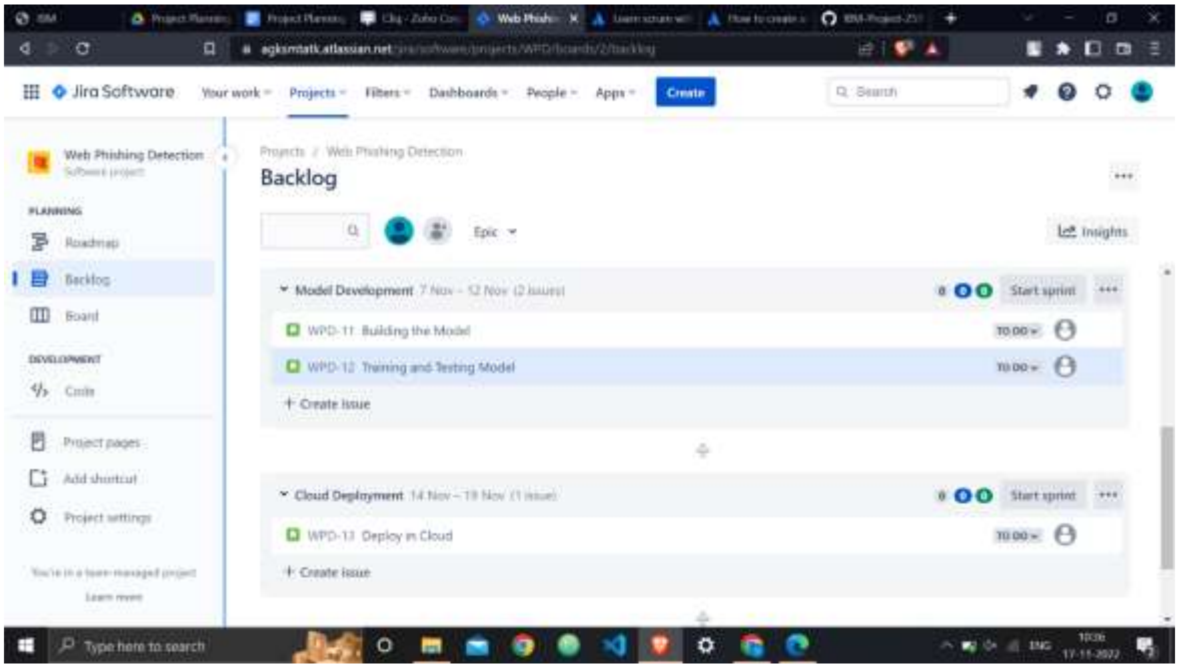
Dataset Management

TO DO	IN PROGRESS	DONE 4 ISSUES
		Dataset Preprocessing WPD-9
		Dataset Collection WPD-8
		Dataset Collection WPD-3
		Dataset Preprocessing WPD-6

6.3.2 Board

Backlog:

Backlog contains the sprints and issues for each sprint.



Backlog

Model Development 7 Nov – 12 Nov (2 issues) — Start sprint

- WPD-11 Building the Model — TO DO
- WPD-12 Training and Testing Model — TO DO
- + Create issue

Cloud Deployment 14 Nov – 19 Nov (1 issue) — Start sprint

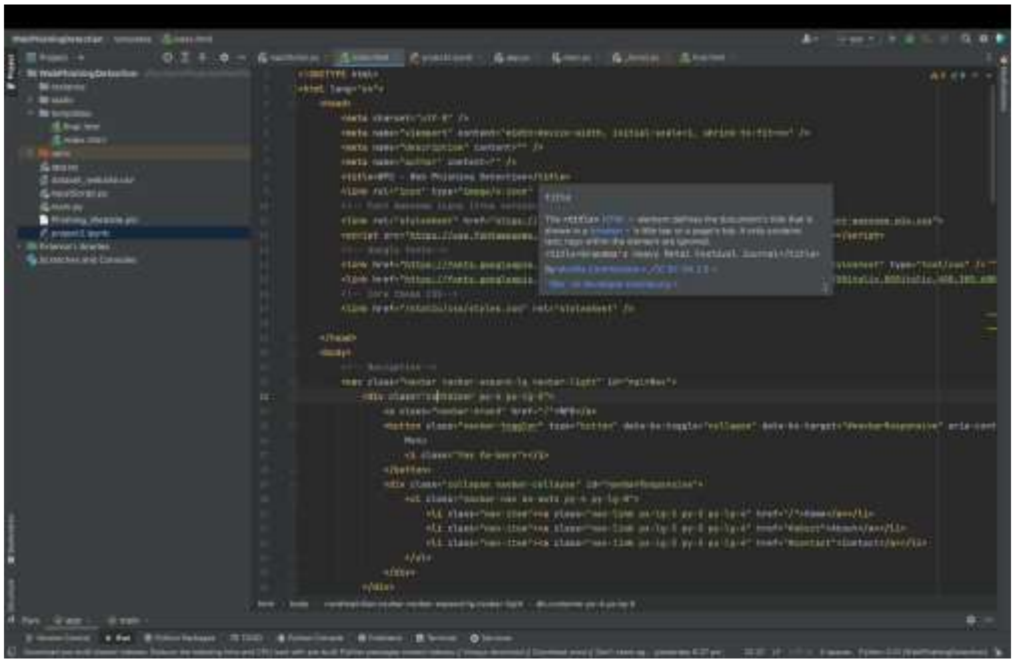
- WPD-13 Deploy in Cloud — TO DO
- + Create issue

6.3.3 Backlog

7. CODING & SOLUTIONING

7.1.Home Page

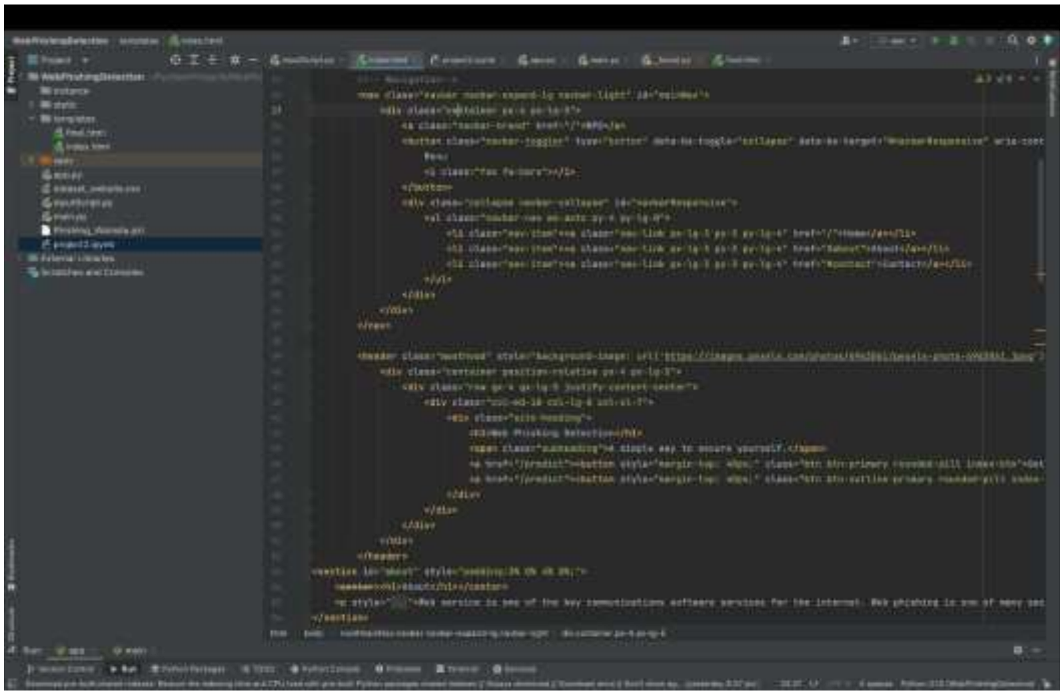
Home page is the starting page where the application takes the user to. The ‘Get Started’ Button takes the user to the next page, i.e. the ‘Prediction Page’.



```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />
        <meta name="description" content="" />
        <meta name="author" content="" />
        <title>WPD - Web Phishing Detection</title>
        <link rel="icon" type="image/x-icon" [illegible]
        <!-- Font Awesome icons (free version) [illegible]
        <link rel="stylesheet" href="https://[illegible] [illegible]-awesome.min.css">
        <script src="https://use.fontawesome.[illegible] [illegible]</script>
        <!-- Google fonts-->
        <link href="https://fonts.googleapis.[illegible] [illegible]stylesheet" type="text/css" />
        <link href="https://fonts.googleapis.[illegible] [illegible]
        <!-- Core theme CSS-->
        <link href="/static/css/styles.css" rel="stylesheet" />

    </head>
    <body>
        <!-- Navigation-->
        <nav class="navbar navbar-expand-lg navbar-light" id="mainNav">
            <div class="container px-4 px-lg-5">
                <a class="navbar-brand" href="/">WPD</a>
                <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarResponsive" aria-cont
                    Menu
                    <i class="fas fa-bars"></i>
                </button>
                <div class="collapse navbar-collapse" id="navbarResponsive">
                    <ul class="navbar-nav ms-auto py-4 py-lg-0">
                        <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="/">Home</a></li>
                        <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="#about">About</a></li>
                        <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="#contact">Contact</a></li>
                    </ul>
                </div>
            </div>
```

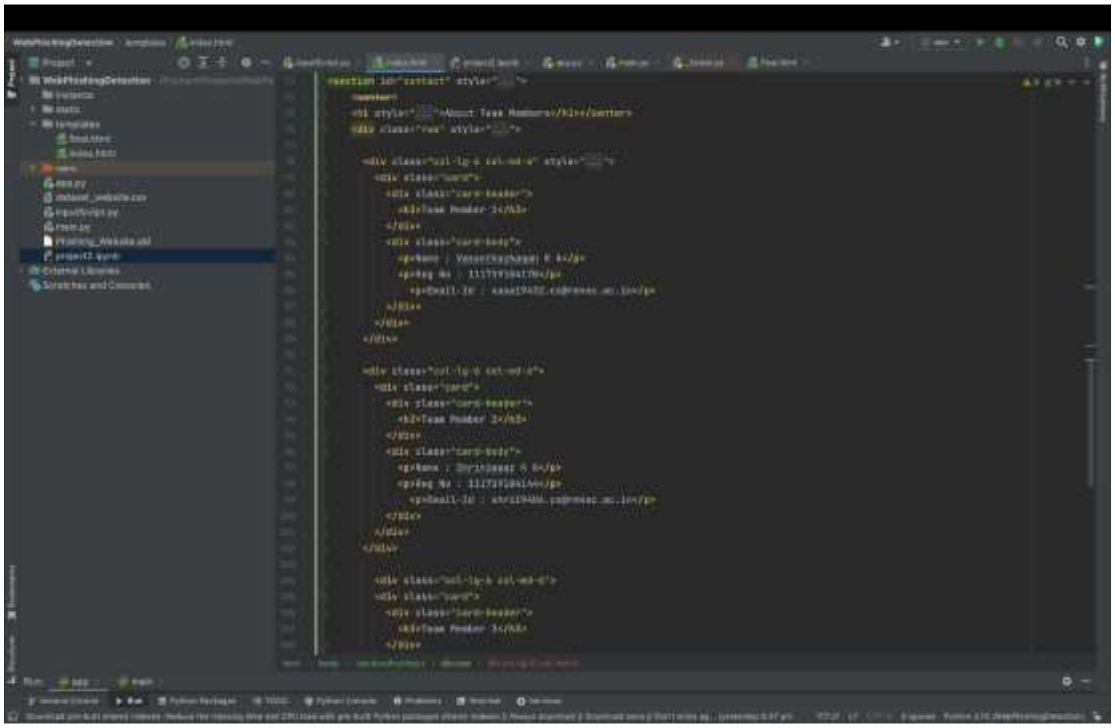
```
title
The <title> HTML element defines the document's title that is shown in a browser's title bar or a page's tab. It only contains text; tags within the element are ignored.
<title>Grandma's Heavy Metal Festival Journal</title>
By Mozilla Contributors, CC BY-SA 2.5
Title on developer.mozilla.org
```

7.1.1 Home Page Code a)



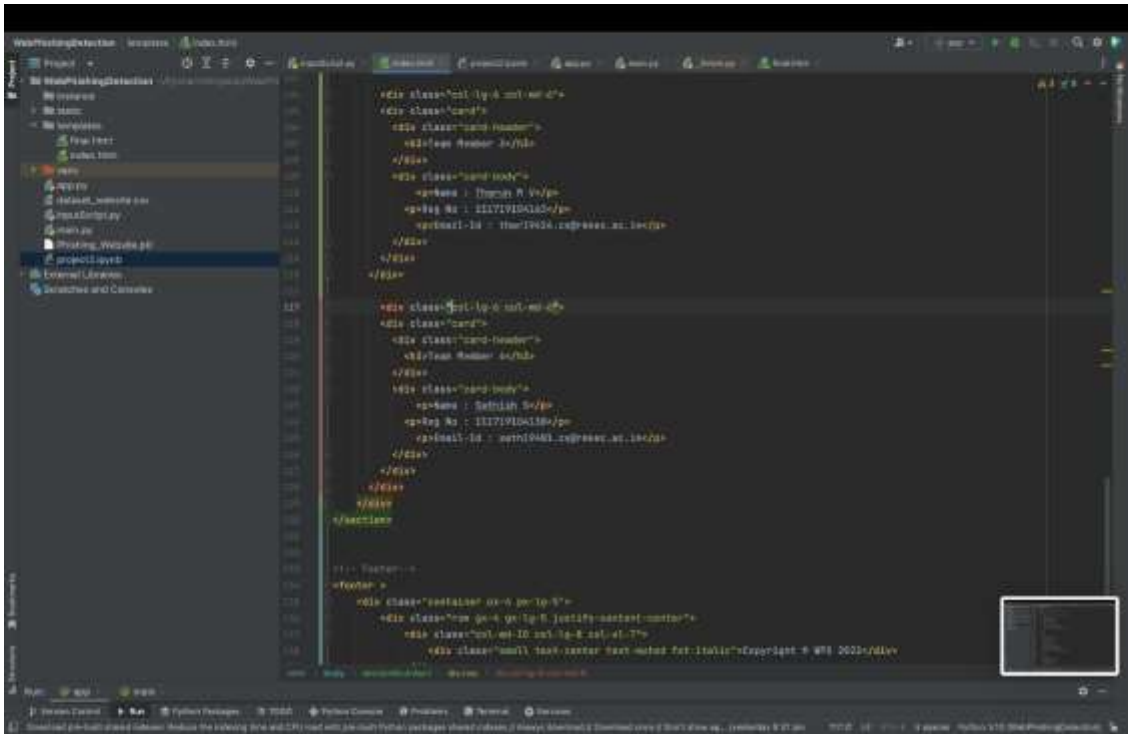
```
        <!-- Navigation-->
        <nav class="navbar navbar-expand-lg navbar-light" id="mainNav">
            <div class="container px-4 px-lg-5">
                <a class="navbar-brand" href="/">WPD</a>
                <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarResponsive" aria-cont
                    Menu
                    <i class="fas fa-bars"></i>
                </button>
                <div class="collapse navbar-collapse" id="navbarResponsive">
                    <ul class="navbar-nav ms-auto py-4 py-lg-0">
                        <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="/">Home</a></li>
                        <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="#about">About</a></li>
                        <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="#contact">Contact</a></li>
                    </ul>
                </div>
            </div>
        </nav>

        <header class="masthead" style="background-image: url('https://images.pexels.com/photos/[illegible]
            <div class="container position-relative px-4 px-lg-5">
                <div class="row gx-4 gx-lg-5 justify-content-center">
                    <div class="col-md-10 col-lg-8 col-xl-7">
                        <div class="site-heading">
                            <h1>Web Phishing Detection</h1>
                            <span class="subheading">A simple way to ensure yourself.</span>
                            <a href="/predict"><button style="margin-top: 40px;" class="btn btn-primary rounded-pill index-btn">Get
                            <a href="/predict"><button style="margin-top: 40px;" class="btn btn-outline-primary rounded-pill index-
                        </div>
                    </div>
                </div>
            </div>
        </header>
    <section id="about" style="padding:3% 0% 4% 3%;">
        <center><h1>About</h1></center>
        <p style="[illegible]">Web service is one of the key communications software services for the internet. Web phishing is one of many sec
    </section>
```

7.1.2 Home Page Code b)



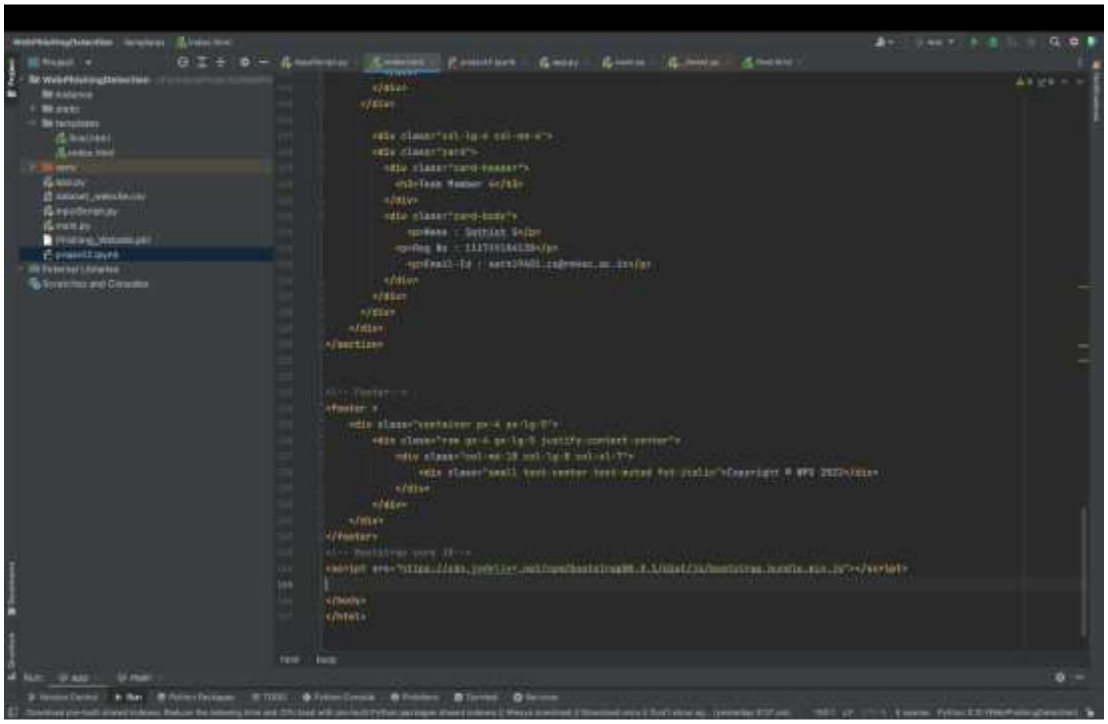
```
<section id="contact" style="...">
    <center>
    <h1 style="...">About Team Members</h1></center>
    <div class="row" style="...">

      <div class="col-lg-6 col-md-6" style="...">
        <div class="card">
          <div class="card-header">
            <h3>Team Member 1</h3>
          </div>
          <div class="card-body">
            <p>Name : Vasanthazhagan K A</p>
            <p>Reg No : [illegible]</p>
              <p>Email-Id : [illegible].cs@rmkec.ac.in</p>
          </div>
        </div>
      </div>

      <div class="col-lg-6 col-md-6">
        <div class="card">
          <div class="card-header">
            <h3>Team Member 2</h3>
          </div>
          <div class="card-body">
            <p>Name : Shrinivasan K B</p>
            <p>Reg No : [illegible]</p>
              <p>Email-Id : [illegible].cs@rmkec.ac.in</p>
          </div>
        </div>
      </div>

        <div class="col-lg-6 col-md-6">
        <div class="card">
          <div class="card-header">
            <h3>Team Member 3</h3>
          </div>
```

7.1.3 Home Page Code c)



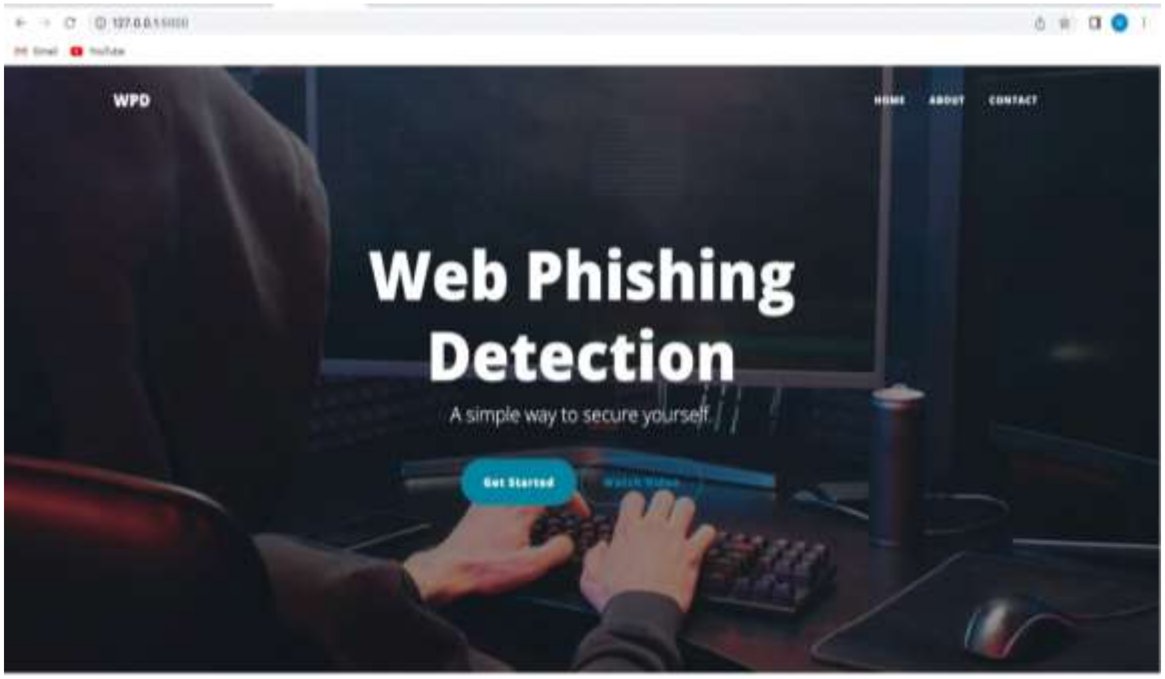
```
        <div class="col-lg-6 col-md-6">
        <div class="card">
          <div class="card-header">
            <h3>Team Member 3</h3>
          </div>
          <div class="card-body">
              <p>Name : Tharun M V</p>
            <p>Reg No : 111719104[illegible]</p>
              <p>Email-Id : thar19426.cs@rmkec.ac.in</p>
          </div>
        </div>
      </div>

        <div class="col-lg-6 col-md-6">
        <div class="card">
          <div class="card-header">
            <h3>Team Member 4</h3>
          </div>
          <div class="card-body">
              <p>Name : Sathiah S</p>
            <p>Reg No : 111719104138</p>
              <p>Email-Id : sath19481.cs@rmkec.ac.in</p>
          </div>
        </div>
      </div>
    </div>
</section>


<!-- Footer-->
<footer >
    <div class="container px-4 px-lg-5">
        <div class="row gx-4 gx-lg-5 justify-content-center">
            <div class="col-md-10 col-lg-8 col-xl-7">
                <div class="small text-center text-muted fst-italic">Copyright © WPS 2022</div>
```

7.1.4 Home Page Code d)



```
            </div>
        </div>

            <div class="col-lg-4 col-md-6">
            <div class="card">
              <div class="card-header">
                <h5>Team Member 4</h5>
              </div>
              <div class="card-body">
                  <p>Name : Sothish S</p>
                <p>Reg No : [illegible]</p>
                  <p>Email-Id : [illegible]</p>
              </div>
            </div>
        </div>
    </div>
</section>


<!-- Footer -->
<footer >
    <div class="container px-4 px-lg-5">
        <div class="row gx-4 gx-lg-5 justify-content-center">
            <div class="col-md-10 col-lg-8 col-xl-7">
                <div class="small text-center text-muted fst-italic">Copyright © WPD 2022</div>
            </div>
        </div>
    </div>
</footer>
<!-- Bootstrap core JS-->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.[illegible]/dist/js/bootstrap.bundle.min.js"></script>

</body>
</html>
```

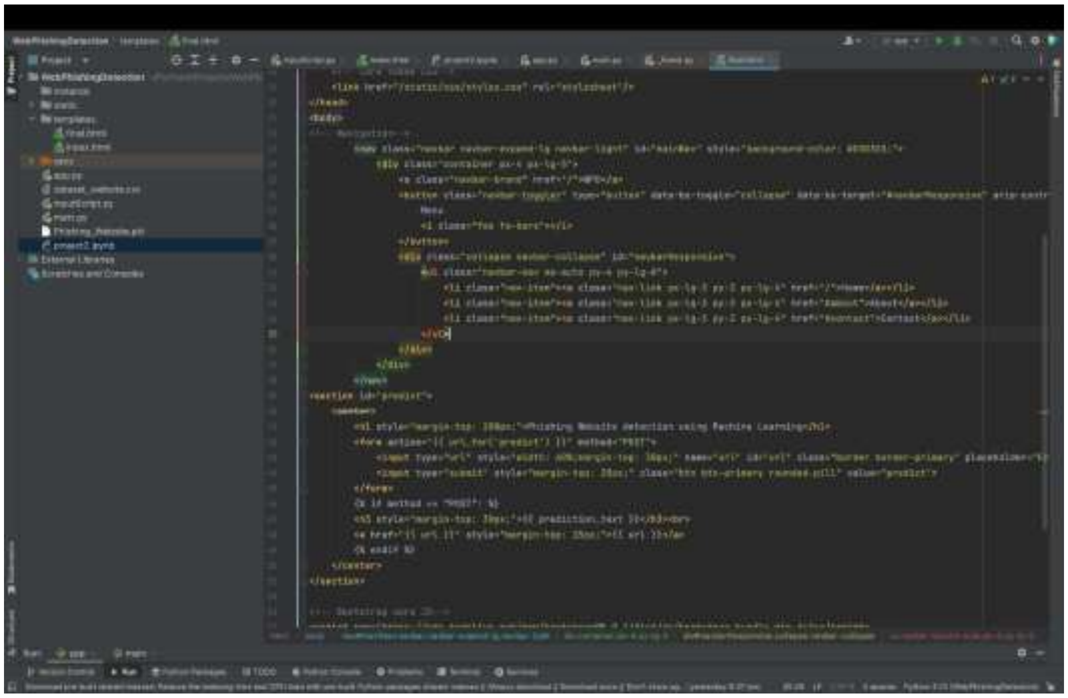
7.1.5 Home Page Code e)



7.1.6 Home Page Output

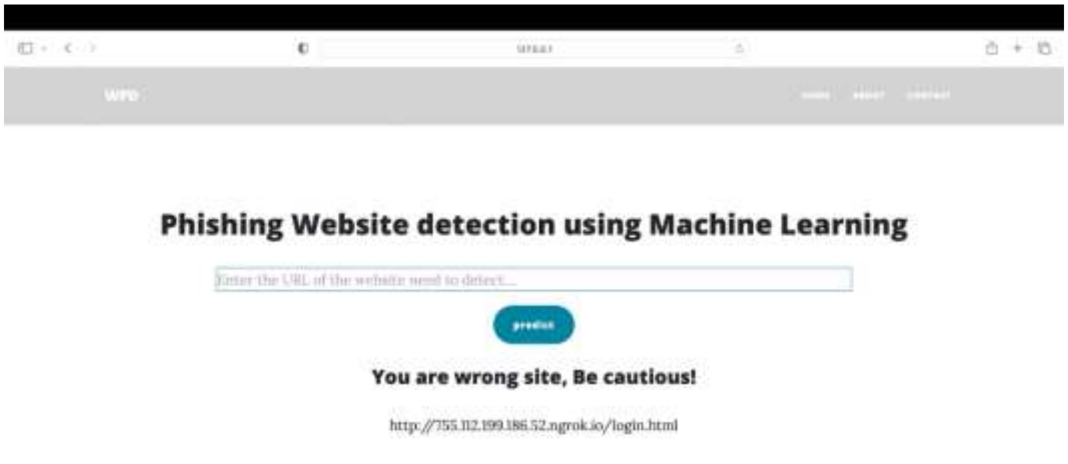
7.2.Prediction Page

Prediction Page allows the user to input the website's URL and clicking the 'Predict' button tells the user if the page is Legitimate or suspicious one.



```
    <link href="/static/css/styles.css" rel="stylesheet"/>
</head>
<body>
<!-- Navigation-->
        <nav class="navbar navbar-expand-lg navbar-light" id="mainNav" style="background-color: [illegible];">
            <div class="container px-4 px-lg-5">
                <a class="navbar-brand" href="/">WPD</a>
                <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarResponsive" aria-contr
                    Menu
                    <i class="fas fa-bars"></i>
                </button>
                <div class="collapse navbar-collapse" id="navbarResponsive">
                    <ul class="navbar-nav ms-auto py-4 py-lg-0">
                        <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="/">Home</a></li>
                        <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="#about">About</a></li>
                        <li class="nav-item"><a class="nav-link px-lg-3 py-3 py-lg-4" href="#contact">Contact</a></li>
                    </ul>
                </div>
            </div>
        </nav>
<section id="predict">
    <center>
        <h1 style="margin-top: 150px;">Phishing Website detection using Machine Learning</h1>
        <form action="{{ url_for('predict') }}" method="POST">
            <input type="url" style="width: 40%;margin-top: 30px;" name="url" id="url" class="border border-primary" placeholder="E
            <input type="submit" style="margin-top: 20px;" class="btn btn-primary rounded-pill" value="predict">
        </form>
        {% if method == "POST": %}
        <h5 style="margin-top: 30px;">{{ prediction_text }}</h5><br>
        <a href="{{ url }}" style="margin-top: 20px;">{{ url }}</a>
        {% endif %}
    </center>
</section>

<!-- Bootstrap core JS-->
```

7.2.1 Prediction Page Code



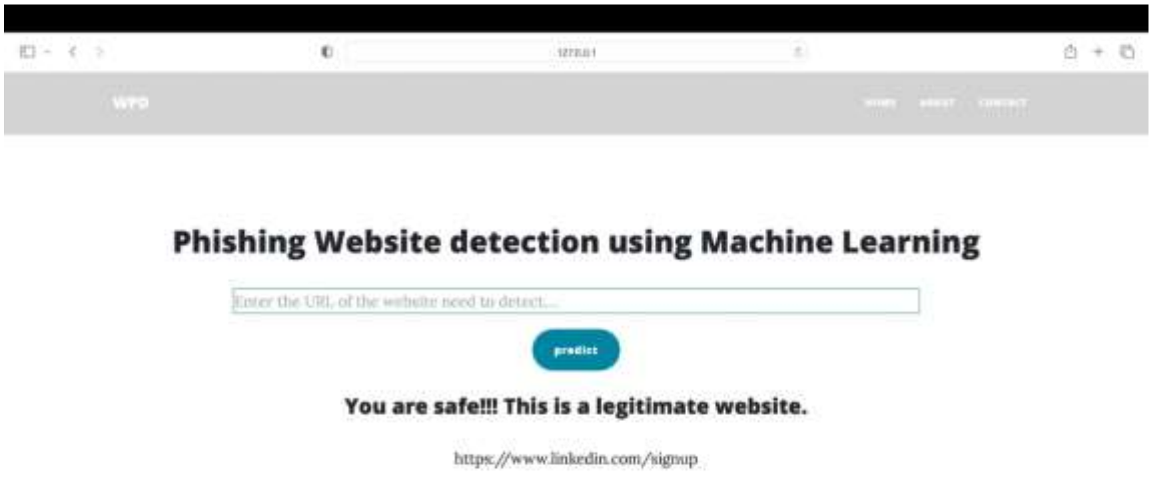
Phishing Website detection using Machine Learning

Enter the URL of the website need to detect...

You are wrong site, Be cautious!

http://755.112.199.186.52.ngrok.io/login.html

7.2.2 Prediction Page Output a)



Phishing Website detection using Machine Learning

Enter the URL of the website need to detect...

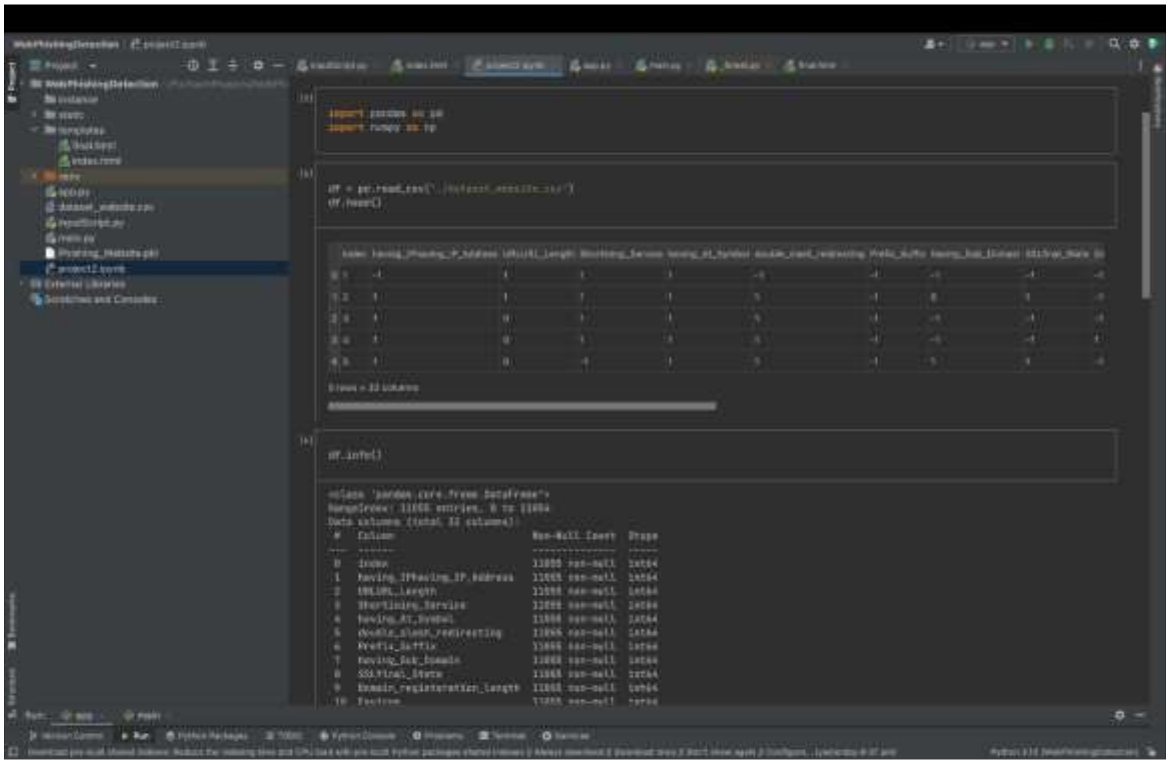
You are safe!!! This is a legitimate website.

https://www.linkedin.com/signup

7.2.3 Prediction Page Output b)

7.3.Model Building using Random Forest Classifier

The model is build using Random Forest Classifier.



```
import pandas as pd
import numpy as np
```

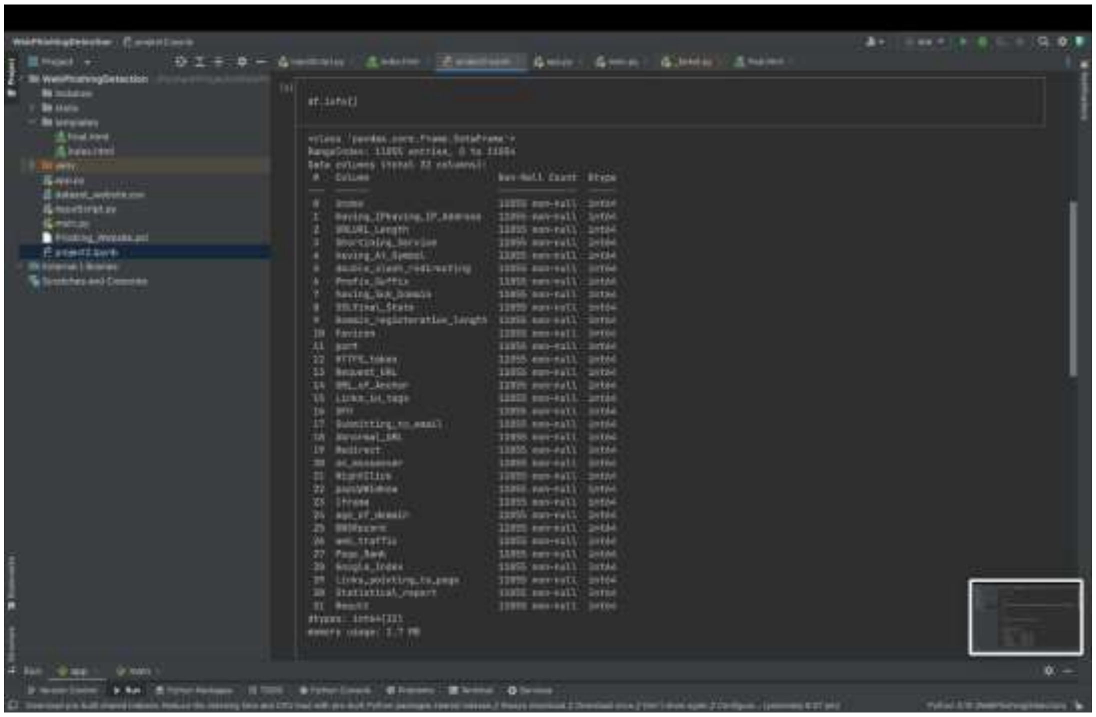
```
df = pd.read_csv('./dataset_website.csv')
df.head()
```

```
  index having_IPhaving_IP_Address URLURL_Length Shortining_Service having_At_Symbol double_slash_redirecting Prefix_Suffix having_Sub_Domain SSLfinal_State Do
0 1     -1                         1             1                  1                -1                       -1            -1                -1             -1
1 2     1                          1             1                  1                1                        -1            0                 1              -1
2 3     1                          0             1                  1                1                        -1            -1                -1             -1
3 4     1                          0             1                  1                1                        -1            -1                -1             1
4 5     1                          0             -1                 1                1                        -1            1                 1              -1

5 rows × 32 columns
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11055 entries, 0 to 11054
Data columns (total 32 columns):
 #   Column                       Non-Null Count   Dtype
---  ------                       --------------   -----
 0   index                        11055 non-null   int64
 1   having_IPhaving_IP_Address   11055 non-null   int64
 2   URLURL_Length                11055 non-null   int64
 3   Shortining_Service           11055 non-null   int64
 4   having_At_Symbol             11055 non-null   int64
 5   double_slash_redirecting     11055 non-null   int64
 6   Prefix_Suffix                11055 non-null   int64
 7   having_Sub_Domain            11055 non-null   int64
 8   SSLfinal_State               11055 non-null   int64
 9   Domain_registeration_length  11055 non-null   int64
 10  Favicon                      11055 non-null   int64
```

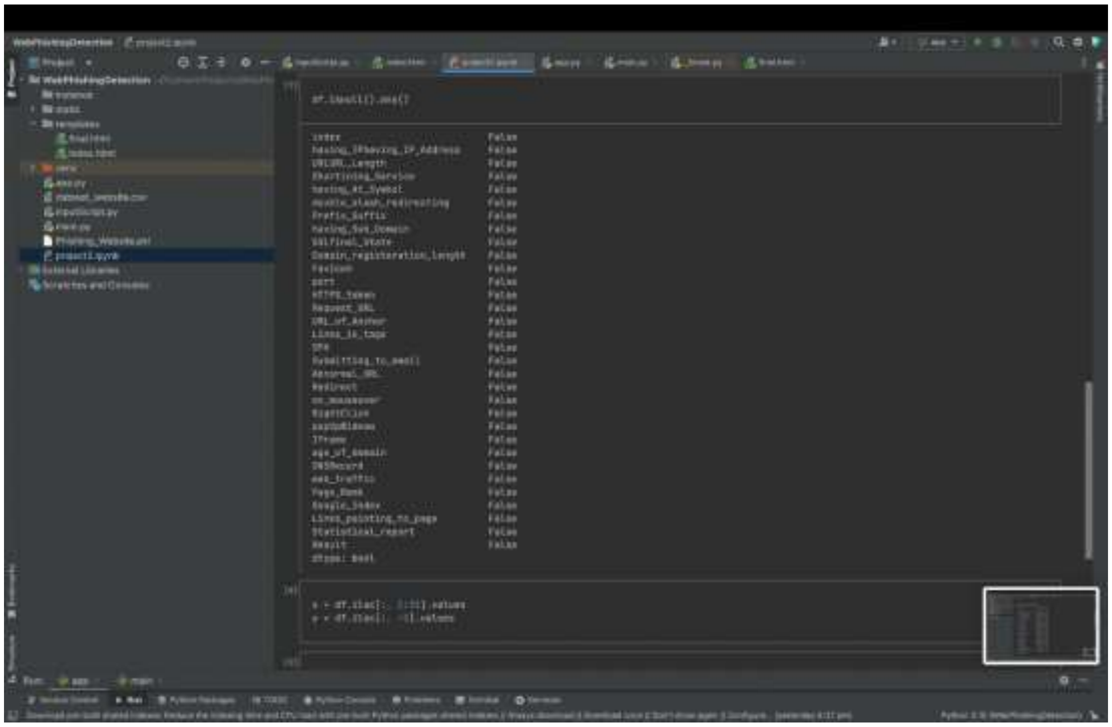
7.3.1 Model Building Code a)



```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11055 entries, 0 to 11054
Data columns (total 32 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   index                        11055 non-null  int64
 1   having_IPhaving_IP_Address   11055 non-null  int64
 2   URLURL_Length                11055 non-null  int64
 3   Shortining_Service           11055 non-null  int64
 4   having_At_Symbol             11055 non-null  int64
 5   double_slash_redirecting     11055 non-null  int64
 6   Prefix_Suffix                11055 non-null  int64
 7   having_Sub_Domain            11055 non-null  int64
 8   SSLfinal_State               11055 non-null  int64
 9   Domain_registeration_length  11055 non-null  int64
 10  Favicon                      11055 non-null  int64
 11  port                         11055 non-null  int64
 12  HTTPS_token                  11055 non-null  int64
 13  Request_URL                  11055 non-null  int64
 14  URL_of_Anchor                11055 non-null  int64
 15  Links_in_tags                11055 non-null  int64
 16  SFH                          11055 non-null  int64
 17  Submitting_to_email          11055 non-null  int64
 18  Abnormal_URL                 11055 non-null  int64
 19  Redirect                     11055 non-null  int64
 20  on_mouseover                 11055 non-null  int64
 21  RightClick                   11055 non-null  int64
 22  popUpWidnow                  11055 non-null  int64
 23  Iframe                       11055 non-null  int64
 24  age_of_domain                11055 non-null  int64
 25  DNSRecord                    11055 non-null  int64
 26  web_traffic                  11055 non-null  int64
 27  Page_Rank                    11055 non-null  int64
 28  Google_Index                 11055 non-null  int64
 29  Links_pointing_to_page       11055 non-null  int64
 30  Statistical_report           11055 non-null  int64
 31  Result                       11055 non-null  int64
dtypes: int64(32)
memory usage: 2.7 MB
```

7.3.2 Model Building Code b)

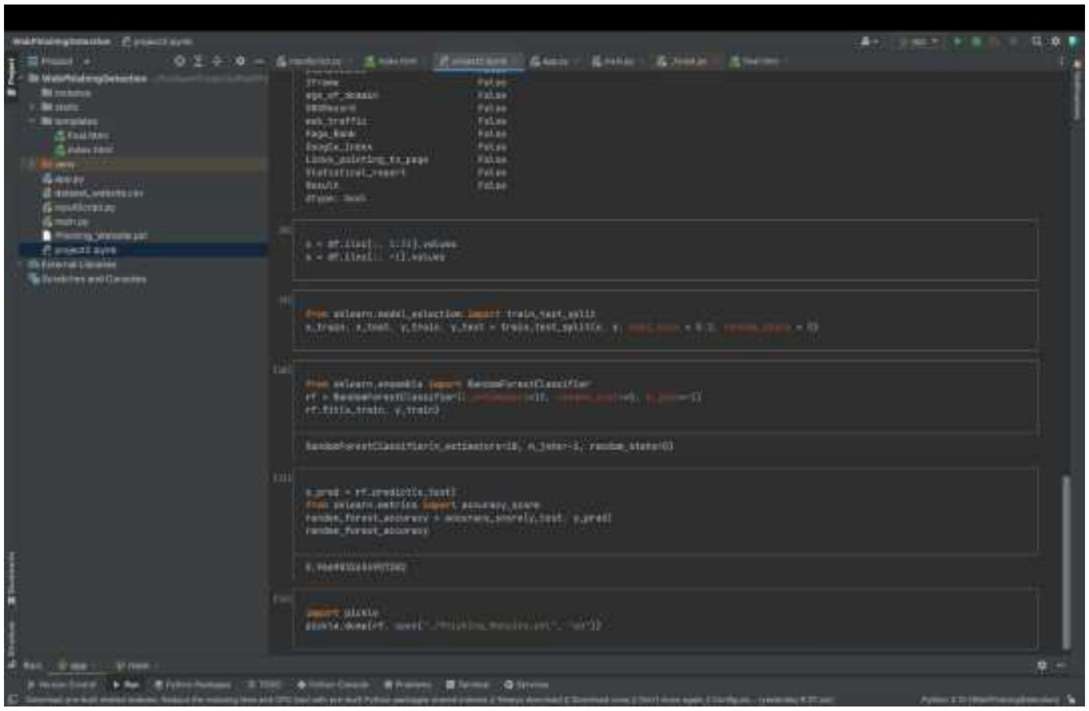


```
df.isnull().any()
```

```
index                          False
having_IPhaving_IP_Address     False
URLURL_Length                  False
Shortining_Service             False
having_At_Symbol               False
double_slash_redirecting       False
Prefix_Suffix                  False
having_Sub_Domain              False
SSLfinal_State                 False
Domain_registeration_length    False
Favicon                        False
port                           False
HTTPS_token                    False
Request_URL                    False
URL_of_Anchor                  False
Links_in_tags                  False
SFH                            False
Submitting_to_email            False
Abnormal_URL                   False
Redirect                       False
on_mouseover                   False
RightClick                     False
popUpWidnow                    False
Iframe                         False
age_of_domain                  False
DNSRecord                      False
web_traffic                    False
Page_Rank                      False
Google_Index                   False
Links_pointing_to_page         False
Statistical_report             False
Result                         False
dtype: bool
```

```
x = df.iloc[:, 1:31].values
y = df.iloc[:, -1].values
```

7.3.3 Model Building Code c)



```
iFrame                    False
age_of_domain             False
DNSRecord                 False
web_traffic               False
Page_Rank                 False
Google_Index              False
Links_pointing_to_page    False
Statistical_report        False
Result                    False
dtype: bool
```

```
x = df.iloc[:, 1:31].values
y = df.iloc[:, -1].values
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)
```

```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=10, random_state=0, n_jobs=-1)
rf.fit(x_train, y_train)
```

```
RandomForestClassifier(n_estimators=10, n_jobs=-1, random_state=0)
```

```
y_pred = rf.predict(x_test)
from sklearn.metrics import accuracy_score
random_forest_accuracy = accuracy_score(y_test, y_pred)
random_forest_accuracy
```

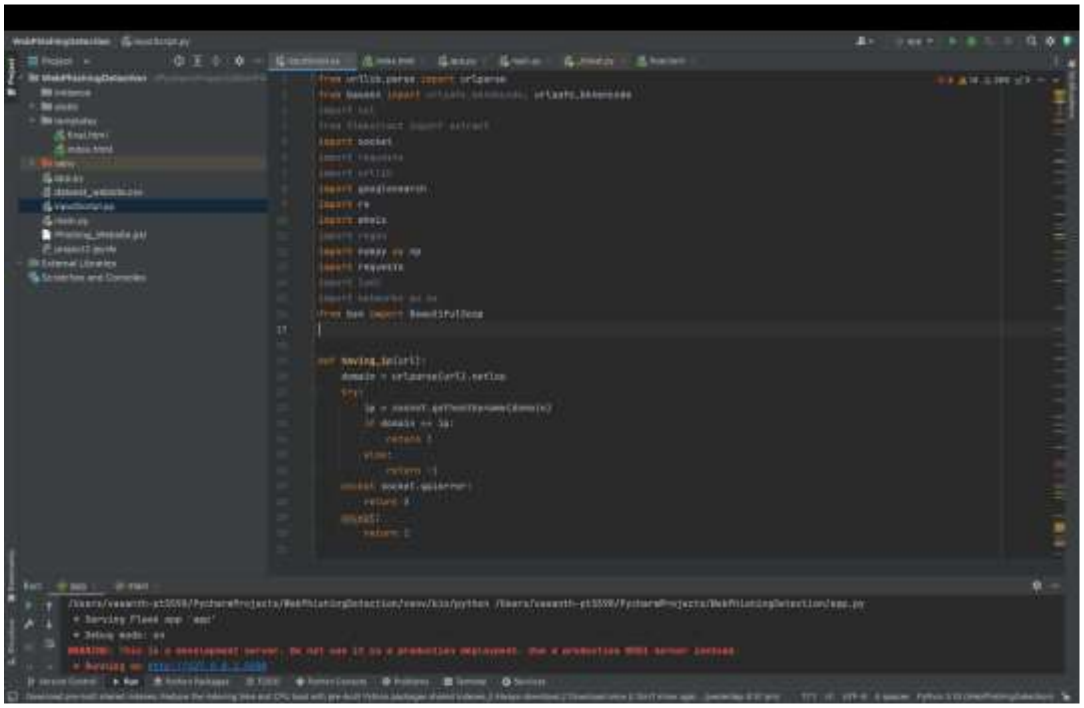
```
0.9[illegible]
```

```
import pickle
pickle.dump(rf, open("./Phishing_Website.pkl", "wb"))
```

7.3.4 Model Building Code d)

7.4. URL Features checking

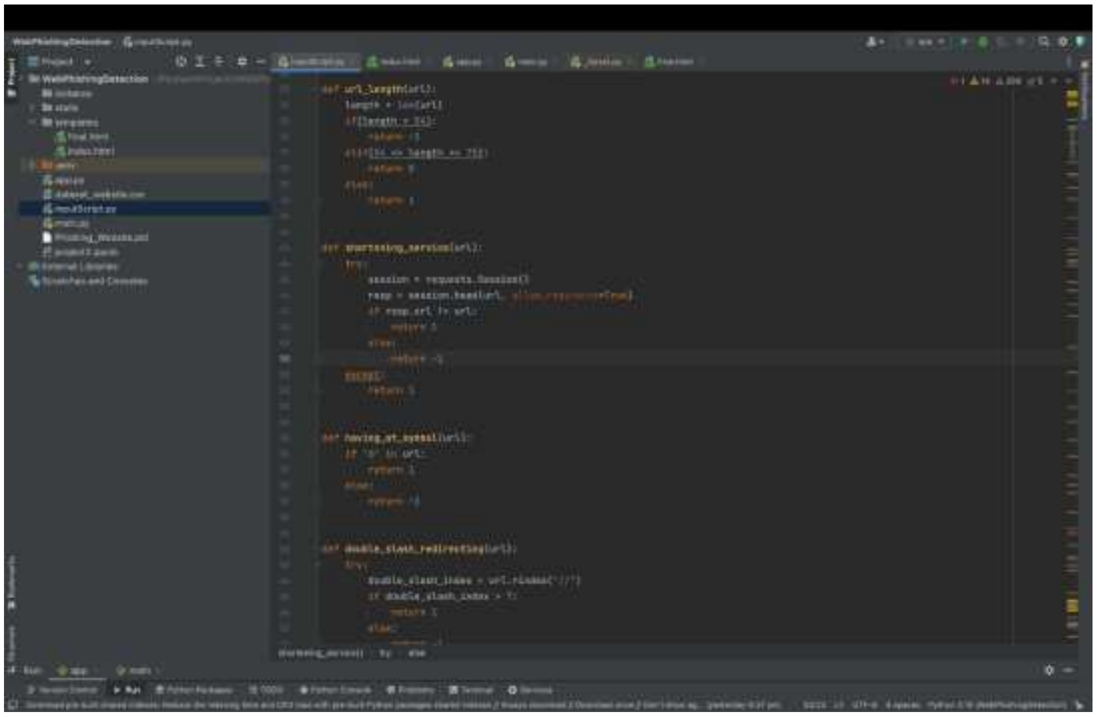
Takes 30 features from the URL and to predict the fraudulence.



```
from urllib.parse import urlparse
from base64 import urlsafe_b64encode, urlsafe_b64decode
import ssl
from tldextract import extract
import socket
import requests
import urllib
import googlesearch
import re
import whois
import regex
import numpy as np
import requests
import json
import networkx as nx
from bs4 import BeautifulSoup


def having_ip(url):
    domain = urlparse(url).netloc
    try:
        ip = socket.gethostbyname(domain)
        if domain == ip:
            return 1
        else:
            return -1
    except socket.gaierror:
        return 0
    except:
        return 1
```

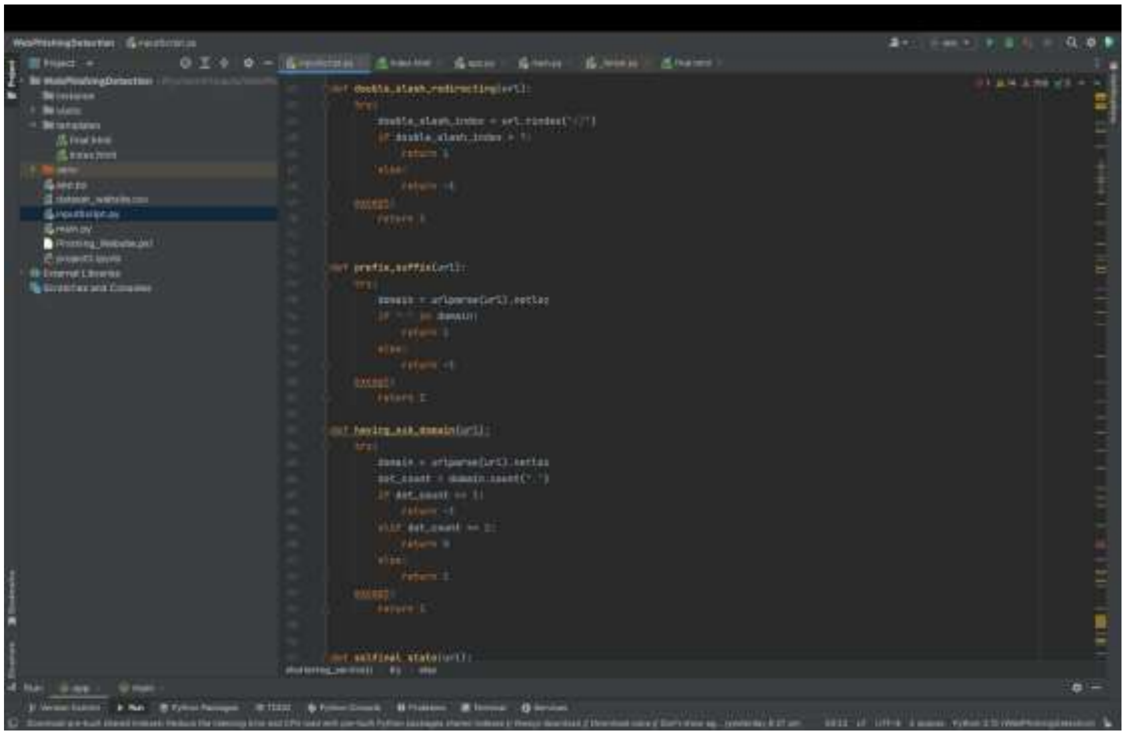
```
/Users/vasanth-pt5558/PycharmProjects/WebPhishingDetection/venv/bin/python /Users/vasanth-pt5558/PycharmProjects/WebPhishingDetection/app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
```

7.4.1 URL Feature Checking Code a)



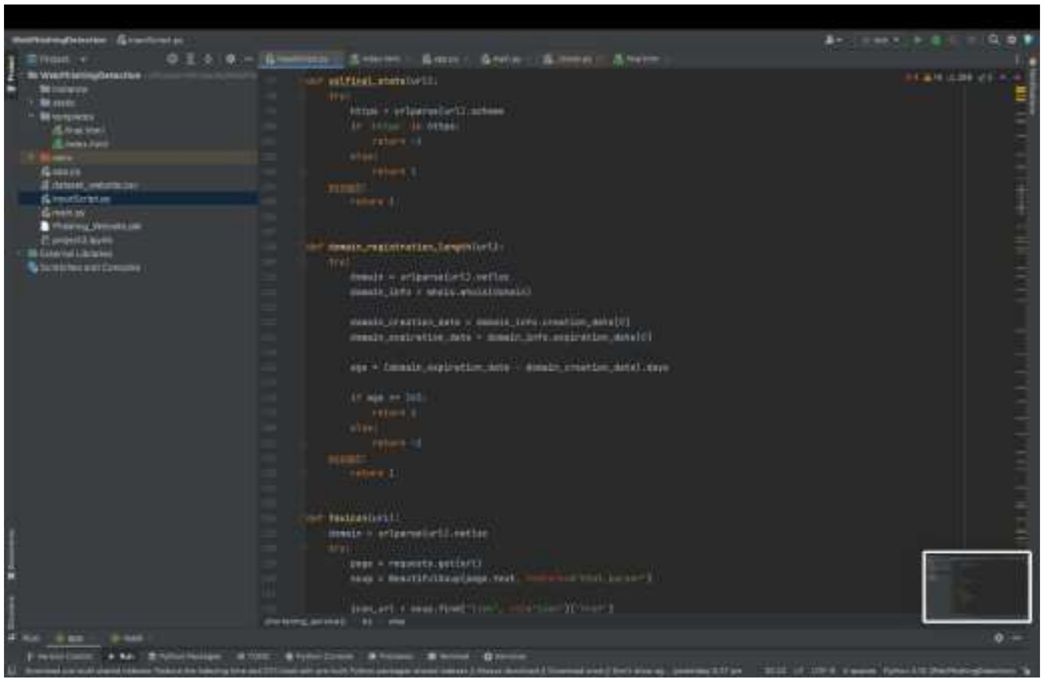
```
def url_length(url):
    length = len(url)
    if(length < 54):
        return -1
    elif(54 <= length <= 75):
        return 0
    else:
        return 1


def shortening_service(url):
    try:
        session = requests.Session()
        resp = session.head(url, allow_redirects=True)
        if resp.url != url:
            return 1
        else:
            return -1
    except:
        return 1


def having_at_symbol(url):
    if "@" in url:
        return 1
    else:
        return -1


def double_slash_redirecting(url):
    try:
        double_slash_index = url.rindex('//')
        if double_slash_index > 7:
            return 1
        else:
            return -1
```

7.4.2 URL Feature Checking Code b)



```
def double_slash_redirecting(url):
    try:
        double_slash_index = url.rindex('//')
        if double_slash_index > 7:
            return 1
        else:
            return -1
    except:
        return 1


def prefix_suffix(url):
    try:
        domain = urlparse(url).netloc
        if '-' in domain:
            return 1
        else:
            return -1
    except:
        return 1

def having_sub_domain(url):
    try:
        domain = urlparse(url).netloc
        dot_count = domain.count('.')
        if dot_count == 1:
            return -1
        elif dot_count == 2:
            return 0
        else:
            return 1
    except:
        return 1


def sslfinal_state(url):
```

7.4.3 URL Feature Checking Code c)

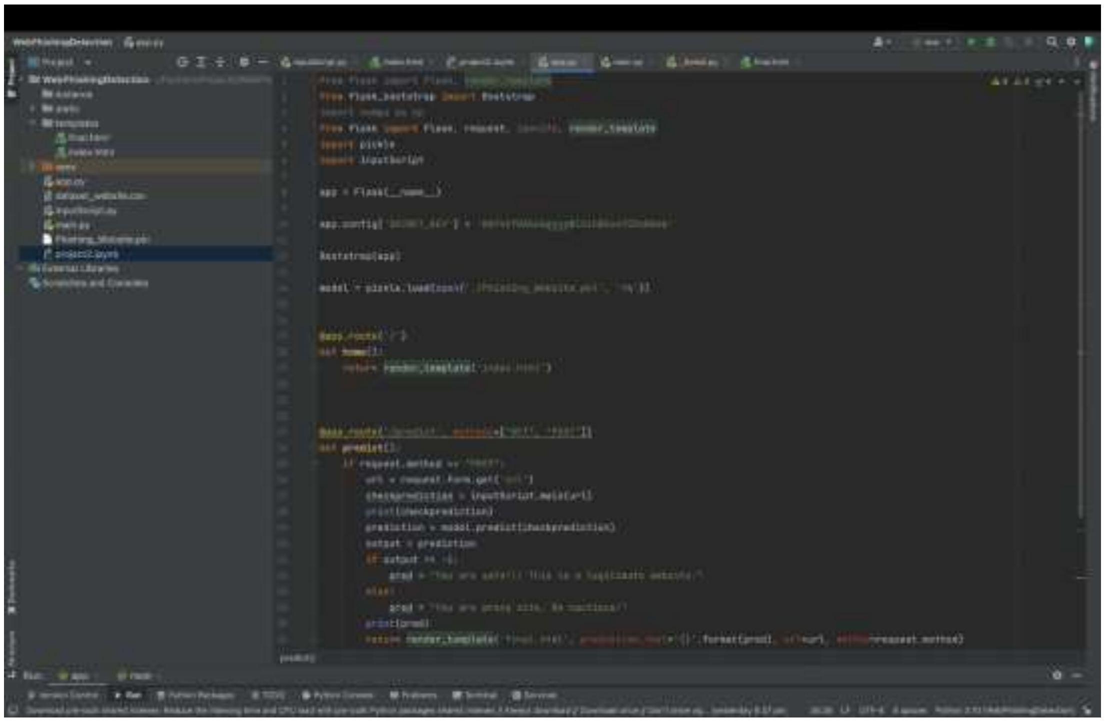


```
def url_final_state(url):
    try:
        https = urlparse(url).scheme
        if 'https' in https:
            return -1
        else:
            return 1
    except:
        return 1


def domain_registration_length(url):
    try:
        domain = urlparse(url).netloc
        domain_info = whois.whois(domain)

        domain_creation_date = domain_info.creation_date[0]
        domain_expiration_date = domain_info.expiration_date[0]

        age = (domain_expiration_date - domain_creation_date).days

        if age >= 365:
            return 1
        else:
            return -1
    except:
        return 1


def favicon(url):
    domain = urlparse(url).netloc
    try:
        page = requests.get(url)
        soup = BeautifulSoup(page.text, features="html.parser")

        icon_url = soup.find("link", rel="icon")["href"]
```

7.4.4 URL Feature Checking Code d)

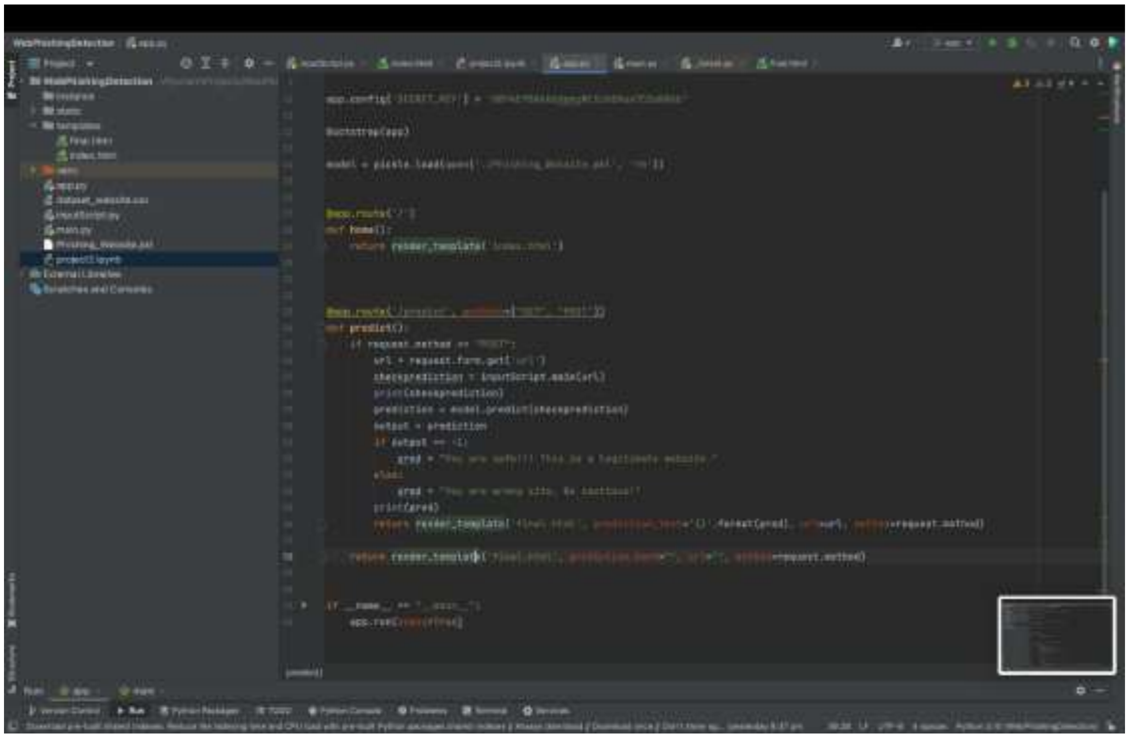
7.5 Website Rendering

Acts as an interface between the HTML webpage and the model built.



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

app = Flask(__name__)

app.config['SECRET_KEY'] = '[illegible]'

Bootstrap(app)

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


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



@app.route('/predict', methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        url = request.form.get('url')
        checkprediction = inputScript.main(url)
        print(checkprediction)
        prediction = model.predict(checkprediction)
        output = prediction
        if output == -1:
            pred = "You are safe!! This is a legitimate website."
        else:
            pred = "You are prone site. Be cautious!"
        print(pred)
        return render_template('final.html', prediction_text='{}'.format(pred), url=url, method=request.method)
```

7.5.1 Website Render Code a)



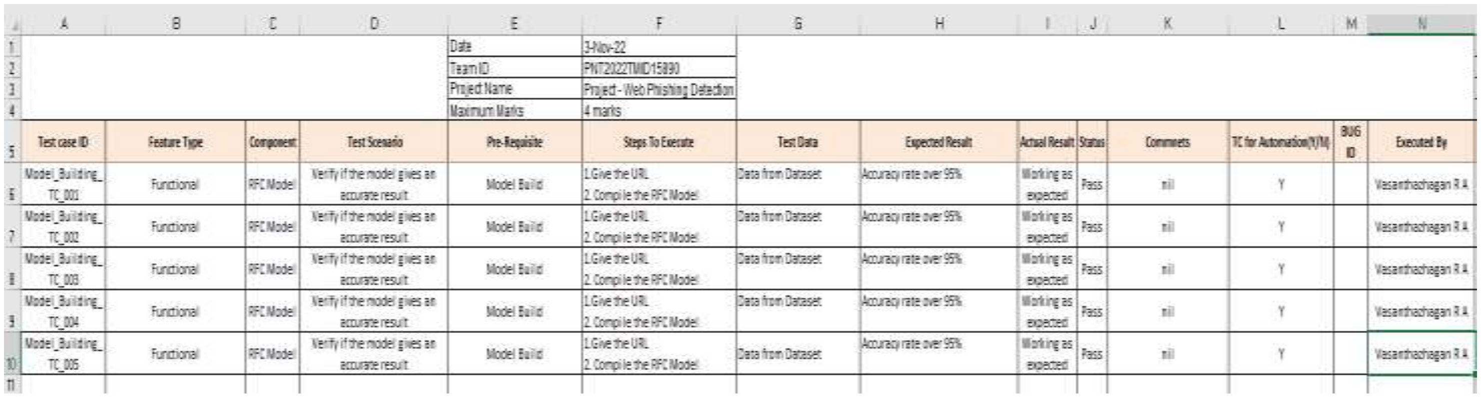
```
app.config['SECRET_KEY'] = '[illegible]'

Bootstrap(app)

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


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



@app.route('/predict', methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        url = request.form.get('url')
        checkprediction = inputScript.main(url)
        print(checkprediction)
        prediction = model.predict(checkprediction)
        output = prediction
        if output == -1:
            pred = "You are safe!!! This is a legitimate website."
        else:
            pred = "You are wrong site. Be cautious!"
        print(pred)
        return render_template('final.html', prediction_text='{}'.format(pred), url=url, method=request.method)

    return render_template('final.html', prediction_text="", url="", method=request.method)


if __name__ == '__main__':
    app.run(debug=True)
```

7.5.2 Website Render Code b)

8. TESTING

8.1. Test Cases

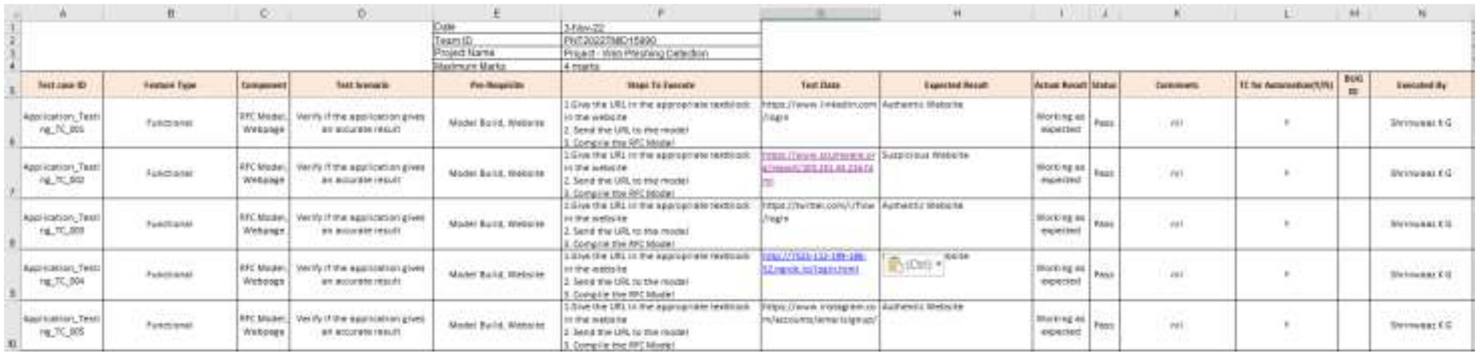
Test cases are sample data input that are used to obtain the efficiency of the application.



Date	3-Nov-22
Team ID	PNT2022TMID15890
Project Name	Project - Web Phishing Detection
Maximum Marks	4 marks

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
Model_Building_TC_001	Functional	RFC Model	Verify if the model gives an accurate result	Model Build	1.Give the URL 2. Compile the RFC Model	Data from Dataset	Accuracy rate over 95%	Working as expected	Pass	nil	Y		Vasanthazhagan R A
Model_Building_TC_002	Functional	RFC Model	Verify if the model gives an accurate result	Model Build	1.Give the URL 2. Compile the RFC Model	Data from Dataset	Accuracy rate over 95%	Working as expected	Pass	nil	Y		Vasanthazhagan R A
Model_Building_TC_003	Functional	RFC Model	Verify if the model gives an accurate result	Model Build	1.Give the URL 2. Compile the RFC Model	Data from Dataset	Accuracy rate over 95%	Working as expected	Pass	nil	Y		Vasanthazhagan R A
Model_Building_TC_004	Functional	RFC Model	Verify if the model gives an accurate result	Model Build	1.Give the URL 2. Compile the RFC Model	Data from Dataset	Accuracy rate over 95%	Working as expected	Pass	nil	Y		Vasanthazhagan R A
Model_Building_TC_005	Functional	RFC Model	Verify if the model gives an accurate result	Model Build	1.Give the URL 2. Compile the RFC Model	Data from Dataset	Accuracy rate over 95%	Working as expected	Pass	nil	Y		Vasanthazhagan R A

8.1.1 Test cases for the build model



Date	3-Nov-22
Team ID	PNT2022TMID15890
Project Name	Project - Web Phishing Detection
Maximum Marks	4 marks

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
Application_Testing_TC_001	Functional	RFC Model, Webpage	Verify if the application gives an accurate result	Model Build, Website	1.Give the URL in the appropriate textblock in the website 2. Send the URL to the model 3. Compile the RFC Model	https://www.linkedin.com/login	Authentic Website	Working as expected	Pass	nil	Y		Shrinivaas K G
Application_Testing_TC_002	Functional	RFC Model, Webpage	Verify if the application gives an accurate result	Model Build, Website	1.Give the URL in the appropriate textblock in the website 2. Send the URL to the model 3. Compile the RFC Model	[illegible]	Suspicious Website	Working as expected	Pass	nil	Y		Shrinivaas K G
Application_Testing_TC_003	Functional	RFC Model, Webpage	Verify if the application gives an accurate result	Model Build, Website	1.Give the URL in the appropriate textblock in the website 2. Send the URL to the model 3. Compile the RFC Model	https://twitter.com/i/flow/login	Authentic Website	Working as expected	Pass	nil	Y		Shrinivaas K G
Application_Testing_TC_004	Functional	RFC Model, Webpage	Verify if the application gives an accurate result	Model Build, Website	1.Give the URL in the appropriate textblock in the website 2. Send the URL to the model 3. Compile the RFC Model	http://1533-132-189-186-52.ngrok.io/login.html	[illegible]	Working as expected	Pass	nil	Y		Shrinivaas K G
Application_Testing_TC_005	Functional	RFC Model, Webpage	Verify if the application gives an accurate result	Model Build, Website	1.Give the URL in the appropriate textblock in the website 2. Send the URL to the model 3. Compile the RFC Model	https://www.instagram.com/accounts/emailsignup/	Authentic Website	Working as expected	Pass	nil	Y		Shrinivaas K G

8.1.2 Test cases for the application

8.2. User Acceptance Testing

1. 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	4	2	1	0	20
URL	3	0	3	4	6
External	2	4	0	1	6
Model	8	2	4	2	16
Skipped	0	5	1	0	1
Prediction	4	0	1	1	2
Won't Fix	1	2	2	1	8
Totals	24	14	13	26	77

8.2.1 Table: Defect Analysis

2. Test Case Analysis

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

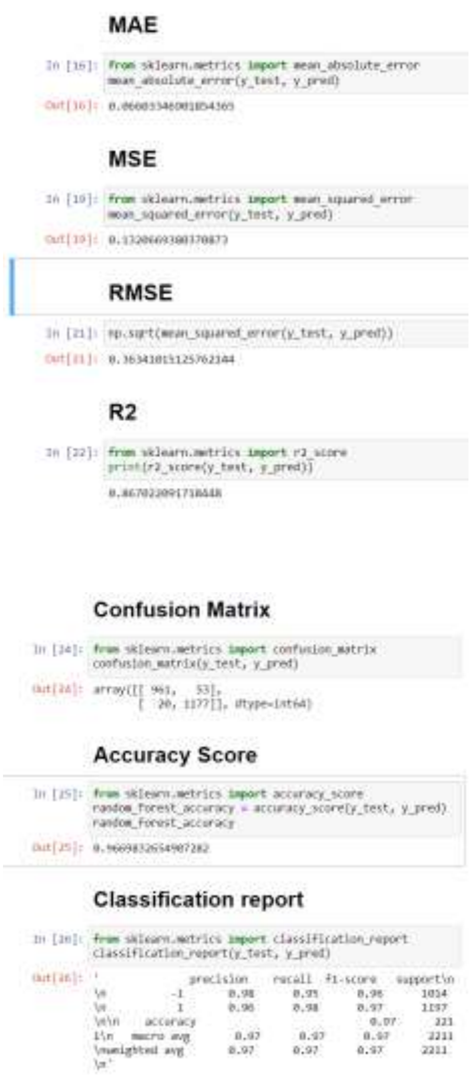
Section	Total Cases	Not Tested	Fail	Pa ss
By Design	7	0	0	7
URL	51	0	0	51
External	2	0	0	2
Model	3	0	0	3
Skipped	9	0	0	9
Prediction	4	0	0	4
Won't Fix	2	0	0	2

8.2.2 Table: Test Case Analysis

9. RESULTS

9.1. Performance Metrics

Performance metrics determines the performance of the application by checking the model for its accuracy, confusion matrix, etc.

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE – 0.066 MSE – 0.132 RMSE – 0.363 R2 score – 0.867 **Classification Model:** Confusion Matrix – array([[961,53], [20, 1177]], dtype=int64) Accuracy Score – 96.69% Classification Report – precision recall f1-score support -1 0.98 0.95 0.96 1014 1 0.96 0.98 0.97 1197 Accuracy 0.97 2211 Macro avg 0.97 0.97 0.97 2211 Nweighted avg 0.97 0.97 0.97 2211	
2.	Tune the Model	Hyperparameter Tuning - GridSearchCV Validation Method	

9.1.1 Table: Performance Testing

MAE

```
In [16]: from sklearn.metrics import mean_absolute_error
         mean_absolute_error(y_test, y_pred)
Out[16]: 0.06603346001854365
```

MSE

```
In [19]: from sklearn.metrics import mean_squared_error
         mean_squared_error(y_test, y_pred)
Out[19]: 0.1320669380370873
```

RMSE

```
In [21]: np.sqrt(mean_squared_error(y_test, y_pred))
Out[21]: 0.3634101512576214
```

R2

```
In [22]: from sklearn.metrics import r2_score
         print(r2_score(y_test, y_pred))
         0.8670220917184448
```

Confusion Matrix

```
In [24]: from sklearn.metrics import confusion_matrix
         confusion_matrix(y_test, y_pred)
Out[24]: array([[ 961,   53],
                [  20, 1177]], dtype=int64)
```

Accuracy Score

```
In [25]: from sklearn.metrics import accuracy_score
         random_forest_accuracy = accuracy_score(y_test, y_pred)
         random_forest_accuracy
Out[25]: 0.9669832654907282
```

Classification report

```
In [26]: from sklearn.metrics import classification_report
         classification_report(y_test, y_pred)
Out[26]: '              precision    recall  f1-score   support\n
         \n          -1       0.98      0.95      0.96      1014
         \n           1       0.96      0.98      0.97      1197
         \n\n    accuracy                           0.97      221
         1\n   macro avg       0.97      0.97      0.97      2211
         \nweighted avg       0.97      0.97      0.97      2211
         \n'
```

10. ADVANTAGES & DISADVANTAGES

Advantages:

- Easy and simple to use
- User-friendly website
- Error accuracy is 3.6×10^{-16}.

Disadvantages:

- Prediction accuracy is not completely accurate (96.69%).
- Takes more time to process the URL.
- Requires really good internet connectivity.

11. CONCLUSION

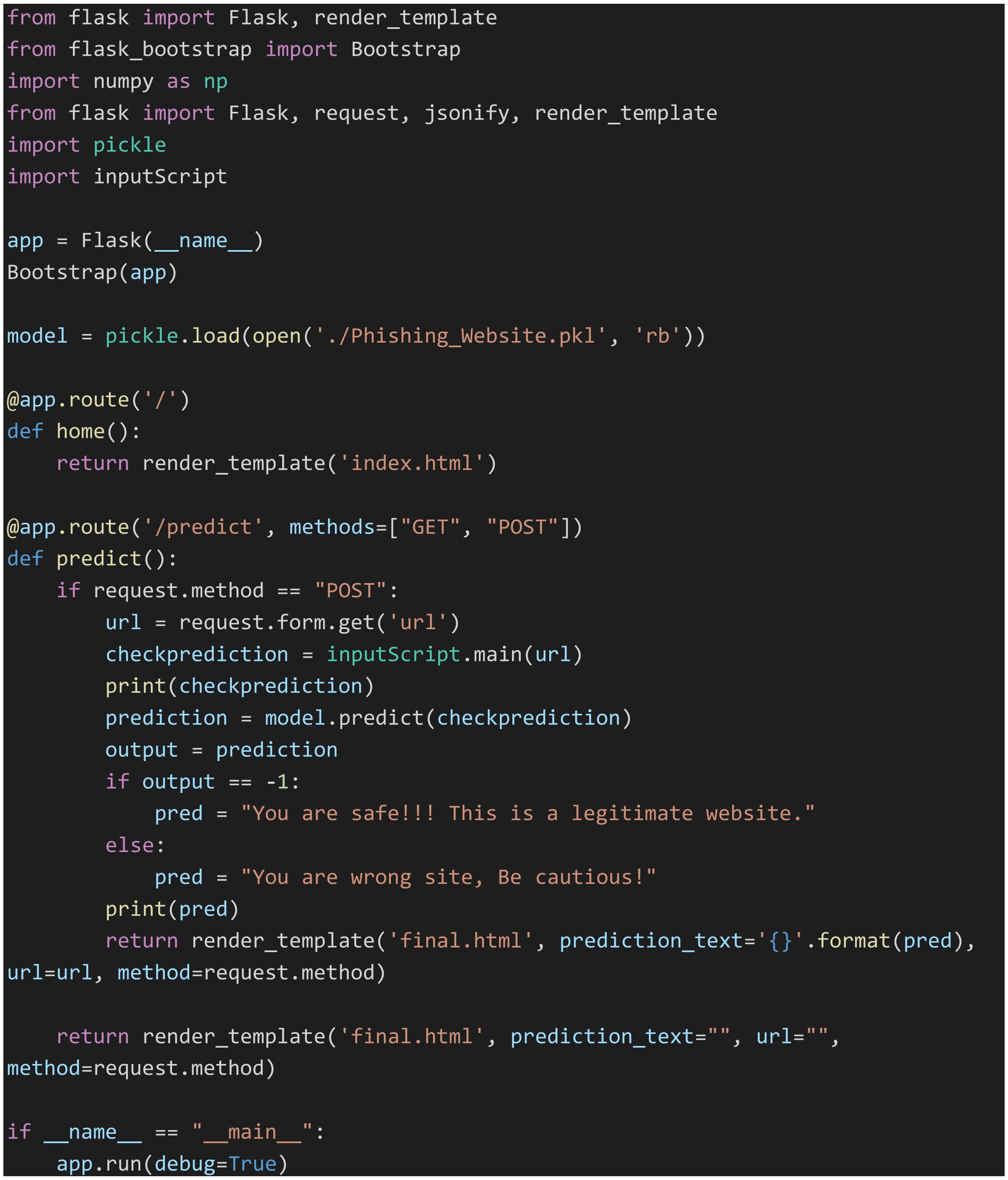
The application thus developed uses a webpage as an interface to get the URL of a website as input from the user. It then predicts the fraudulence of the website using the model built and produces the result to the user as output. The model is developed using the Random Forest Classifier. It uses 80 percentage of the dataset as Training set and the remaining 20 percentage as Testing set. The model thus developed, produces the result with an accuracy of 96.69% and an error accuracy of 3.6×10^{-16}.

12. FUTURE SCOPE

- The model currently used is fed with around 11,000 data samples to be built, which is not really sufficient in the current times where there is a lot of cases. So, the model must be trained with more data samples.
- The current system uses a website as an interface to interact with the user. In future, the system can be developed into a Browser Extension.

13. APPENDIX

13.1. Source Code



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

app = Flask(__name__)
Bootstrap(app)

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

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

@app.route('/predict', methods=["GET", "POST"])
def predict():
    if request.method == "POST":
        url = request.form.get('url')
        checkprediction = inputScript.main(url)
        print(checkprediction)
        prediction = model.predict(checkprediction)
        output = prediction
        if output == -1:
            pred = "You are safe!!! This is a legitimate website."
        else:
            pred = "You are wrong site, Be cautious!"
        print(pred)
        return render_template('final.html', prediction_text='{}'.format(pred),
url=url, method=request.method)

    return render_template('final.html', prediction_text="", url="",
method=request.method)

if __name__ == "__main__":
    app.run(debug=True)
```

13.1.1 Source Code

13.2. GitHub Link

https://github.com/IBM-EPBL/IBM-Project-25999-1668681377

13.3. Project Demo Link

https://drive.google.com/file/d/1IpTDXoYWCE8qXtVvHYoCVfsjxrIw5B4A/view?usp=share_l
ink