**Web Phishing Detection**

**Team ID: PNT2022TMID22501**

**Team Members:**

**VIGNESH.M – 113119UG04111**

**MOHAMED SALMAN.R – 113119UG04059**

**UKENDIRAN.J – 113119UG04109**

**SABAREESH.G K – 113119UG04085**

**1. INTRODUCTION**

1.1 Project Overview

1.2 Purpose

**2. LITERATURE SURVEY**

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

**3. IDEATION & PROPOSED SOLUTION**

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

**4. REQUIREMENT ANALYSIS**

4.1 Functional requirement

4.2 Non-Functional requirements

**5. PROJECT DESIGN**

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

**6. PROJECT PLANNING & SCHEDULING**

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

**7. CODING & SOLUTIONING (Explain the features added in the project along with code)**

7.1 Feature 1

7.2 Feature 2

7.3 Database Schema (if Applicable)

**8. TESTING**

8.1 Test Cases

8.2 User Acceptance Testing

**9. RESULTS**

9.1 Performance Metrics

**10. ADVANTAGES & DISADVANTAGES**

**11. CONCLUSION**

**12. FUTURE SCOPE**

**13. APPENDIX**

Source Code GitHub & Project Demo

# **INTRODUCTION**

* 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.

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.

We have come up with a solution to detect if a website is safe or not by using machine learning for prediction. This helps the user to predict the legitimacy of a website beforehand and thus prevents the user from entering their personal information.

# Purpose

Phishing is a type of social engineering attack often used to steal user data, including login credentials and credit card numbers. It occurs when an attacker, masquerading as a trusted entity, dupes a victim into opening an email, instant message, or text message. The recipient is then tricked into clicking a malicious link, which can lead to the installation of malware, the freezing of the system as part of a ransomware attack or the revealing of sensitive information.

An attack can have devastating results. For individuals, this includes unauthorized purchases, the stealing of funds, or identify theft.

# LITERATURE SURVEY

* 1. **Existing problem**

Junaid Rashid et al[1] used machine learning based phishes detection gadget relies upon efficiently on the aspects of accuracy. The most of antiphishers researchers center of attention on optimizing new feature proposals or classification algorithms, where developing proper features analysis and

selection techniques is not the important plan. The paper involved phishing enabled, reaching an effective positive rate of 97% and a false positive rate of

4%. The features are obtained by META tagging, web pages content, URLs, hyperlinks, TF-IDF, and more. Ping Yi[2] et al used deep learning frameworks to detect web phishing. This paper mainly focuses on applying a deep learning framework to detect

phishing websites. This paper first designs two types of features for web phishing: original features and interaction features. A detection model based on Deep Belief Networks (DBN) is then presented. The test using real IP flows from ISP (Internet Service Provider) shows that the detecting model based on DBN can achieve an approximately 90% true positive rate and 0.6% false positive rate.

Jain, A.K. et al [3] described anti-phishing technology that removes 19 features on the buyer's side to determine phishing websites from approved sites using machine learning. They used 2,141 phishing pages as well as the famous Alexa website, some online debit gateways, and some great banking websites.

Chiew[4] et al proposed to use probability minimization standard and Monte Carlo algorithm using a new neural network-based classification technique for detecting phishing net pages. The thirty points were used to categorize the four main areas, especially around the bar-based, anomaly-based, HTML and JavaScript.

Zhang, W., et al[5] extract features towards URL, text, and web content and utilize Extreme Machine Learning (ELM) technology. The first step in this method is to write the text content of the classifier to determine the content

of the label text through ELM. In this case, OCR software is used to retrieve the text from the image. It is a second-stage-based hybrid that combines text and other function classifiers

# References

[1]Junaid Rashid;Toqeer Mahmood;Muhammad Wasif Nisar;Tahira Nazir; (2020). Phishing Detection Using Machine Learning Technique . 2020 First International Conference of Smart Systems and Emerging Technologies (SMARTTECH), (), –. doi:10.1109/smart-tech49988.2020.00026

[2] Yi, Ping; Guan, Yuxiang; Zou, Futai; Yao, Yao; Wang, Wei; Zhu, Ting (2018). Web Phishing Detection Using a Deep Learning Framework. Wireless Communications and Mobile Computing, 2018(), 1–9.

doi:10.1155/2018/4678746

[3] Jain, A.K. and B.B. Gupta, Towards detection of phishing websites on clientside using machine learning based approach. Telecommunication Systems,

2018. 68(4): p. 687-700.

[4] Chiew, K.L., et al., Utilisation of website logo for phishing detection. Computers & Security, 2015. 54: p. 16-26.

[5] Zhang, W., et al., Two-stage ELM for phishing Web pages detection using hybrid features. World Wide Web, 2017. 20(4): p. 797-813

# Problem Statement Definition

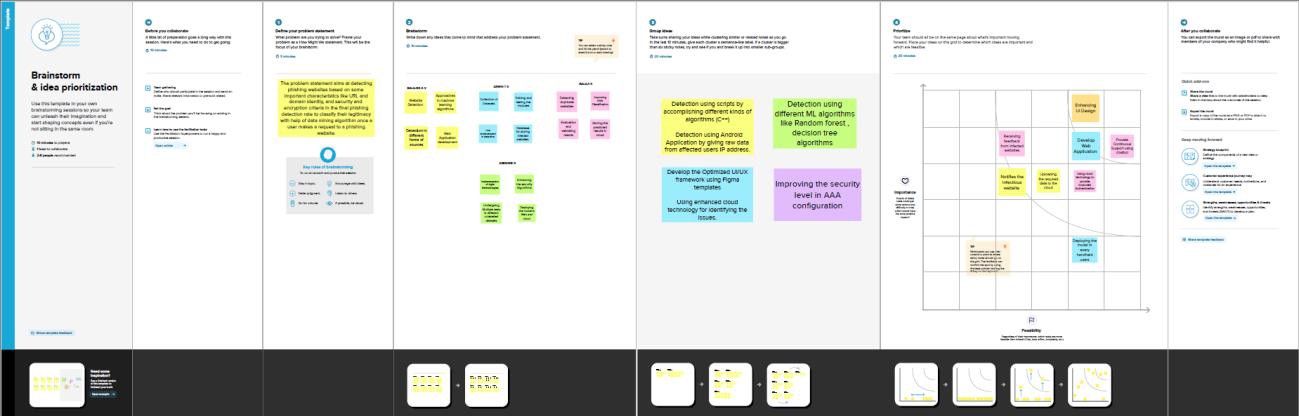
To detect and predict the website URL and check if the website is safe or unsafe to use using machine learning algorithm.

# IDEATION & PROPOSED SOLUTION

* 1. **Empathy Map Canvas**



* 1. **Ideation & Brainstorming**



* 1. **Proposed Solution**

IDEA / SOLUTION DESCRIPTION:

The main objective of the project is applying a machine-learning algorithm to detect Phishing websites. It is a web application to detect good and malicious URLs i.e web phishing sites. We employ HTML, CSS, and JavaScript for the website and the web application is deployed using Flask framework. Different ML models are run and the one with the high accuracy will be selected for final model. The dataset is downloaded and then run-on notebooks. Necessary preprocessing techniques are implemented by applying various statistical methods and encodings are done. Then it is split to train test model and different algorithms are tried.

UNIQUENESS/NOVELTY:

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 like logistic regression, SVM, KNN 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.

SOCIAL IMPACT/ CUSTOMER SATISFACTION:

It helps customers to reduce threats that happen during e-banking. Customer can enter the

correct URL and all the passwords, usernames, private information, credit card details are kept safe. It will prevent from information disclosure and property damage. It will increase customer satisfaction without fearing of malicious sites.

BUSINESS MODEL (FINANCIAL BENEFIT):

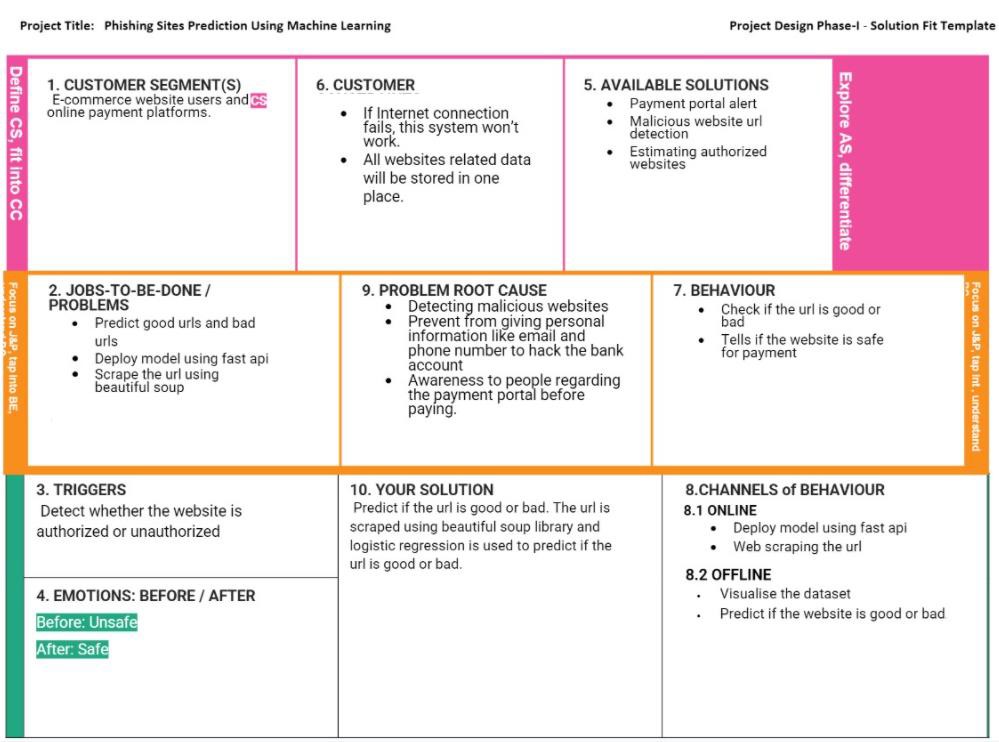
Phishing detection helps in preventing from falling for malicious websites and traps. Thus ensuring the safety of one's personal data and other private information. Doing this beforehand by detecting through ML models can save time. The proposed solution is also a low-cost model and the customers are not charged for the service they receive.

SCALABILITY OF SOLUTION:

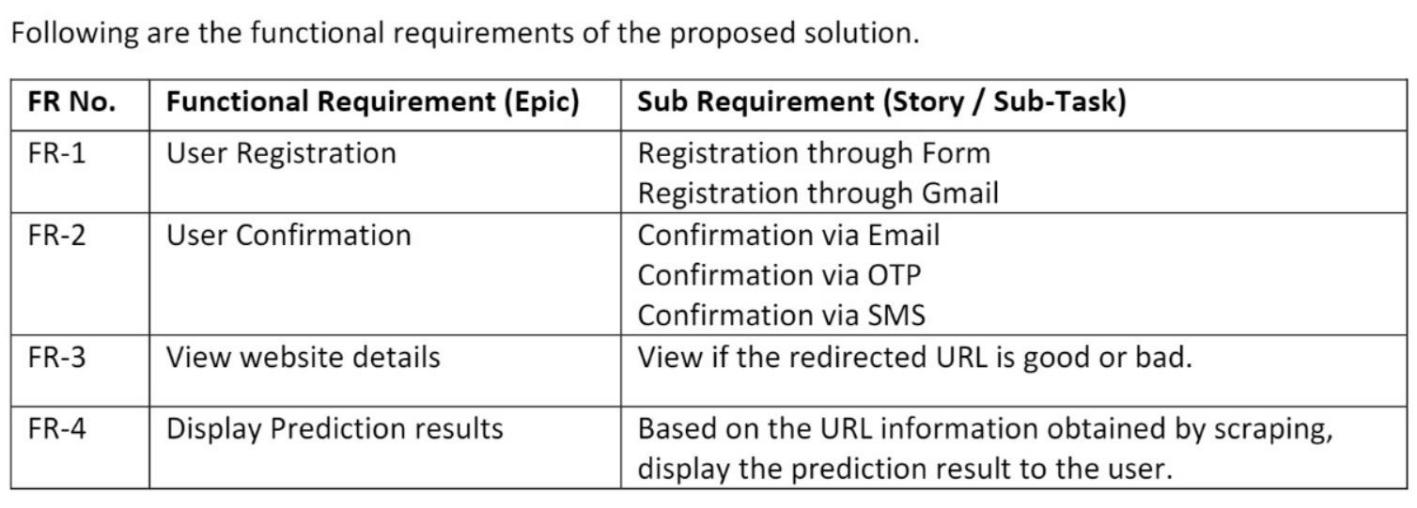
The model’s performance is increased by building it more accurate model with the use of several classification algorithms and selecting the best accurate model among the different models run. Also, through integration of these models, an optimized hybrid model can be

obtained in order to result in more scalability. Deploying the ML model into cloud also makes it easy for enterprises to experiment with the model capabilities and scale up. Placing a finished flight prediction model into a live environment can be used for its intended purpose and it is integrated with Flask, so that they can be accessed by end users.

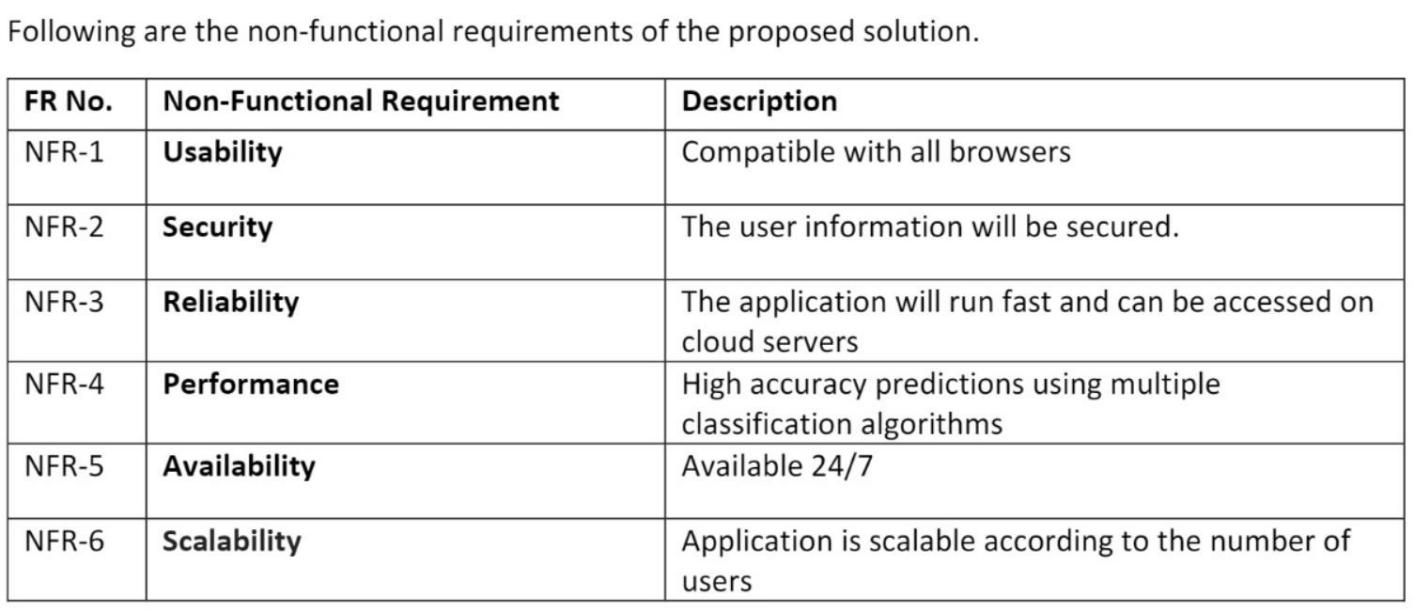
# Problem Solution fit



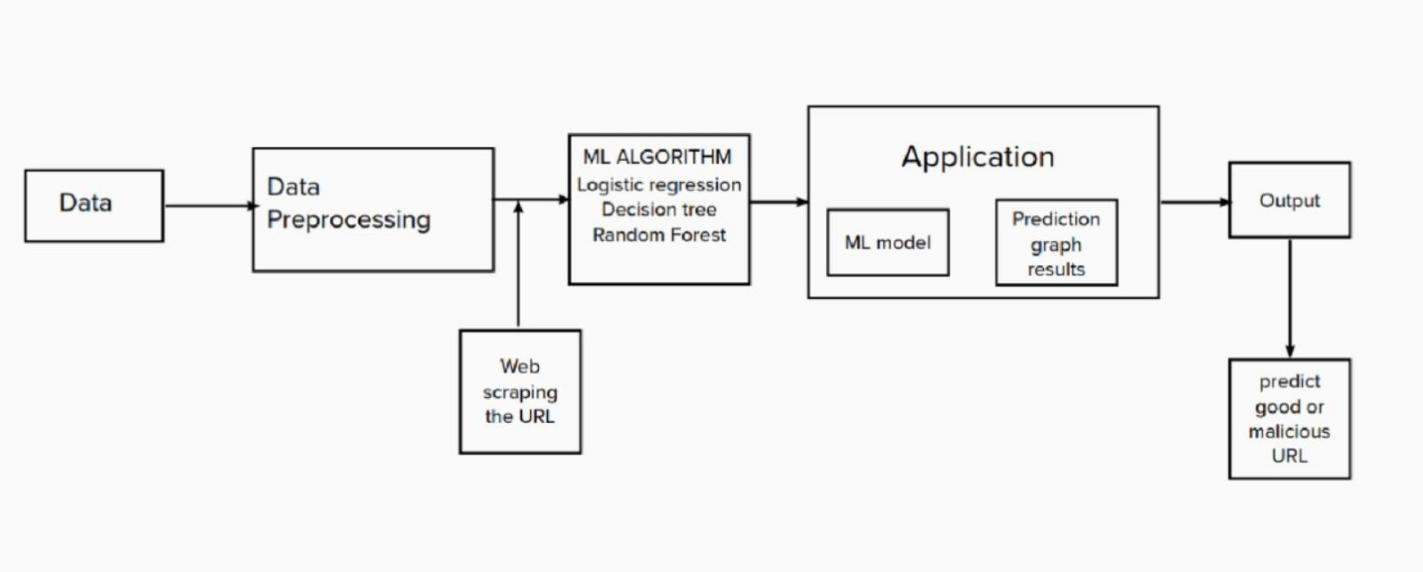
1. **REQUIREMENT ANALYSIS**
   1. **Functional requirement**



* 1. **Non-Functional requirements**

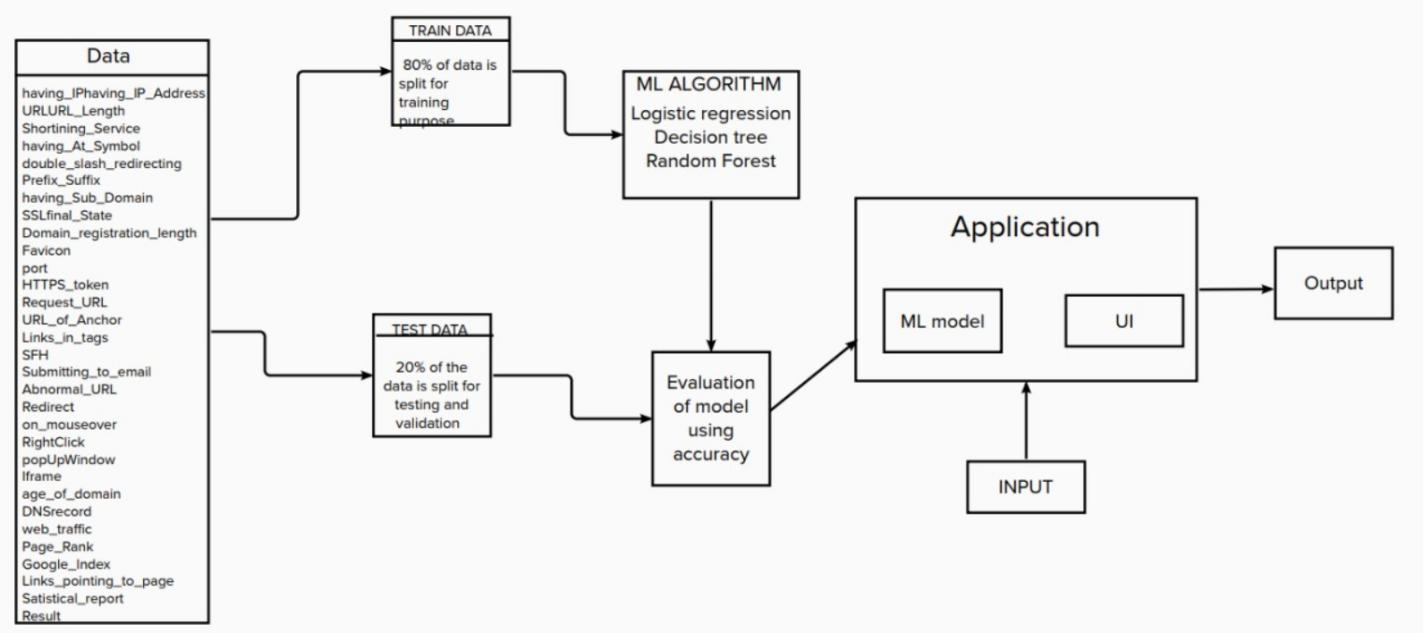


1. **PROJECT DESIGN**
   1. **Data Flow Diagrams**

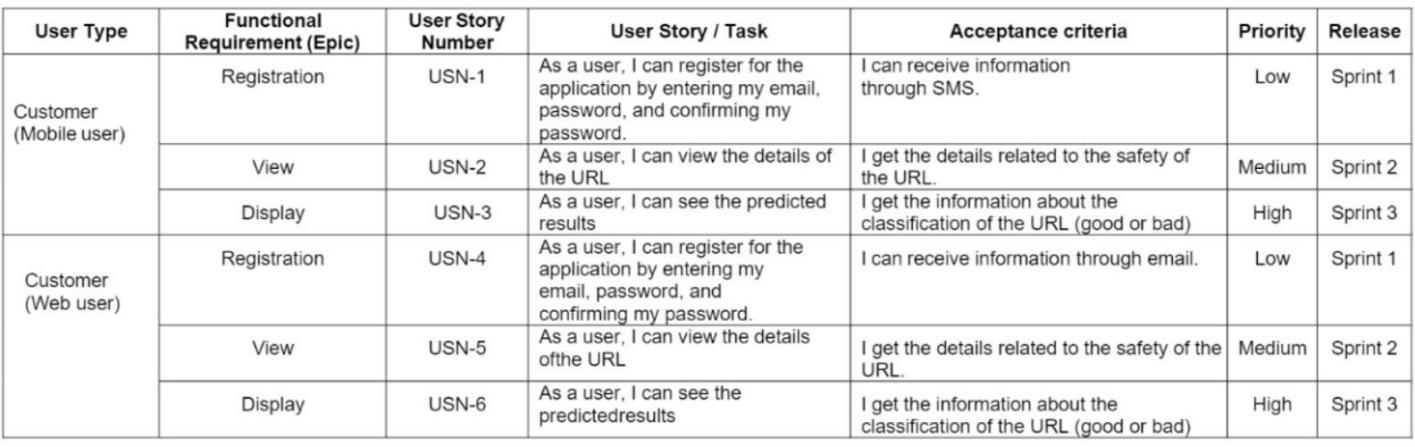


* 1. **Solution & Technical Architecture**

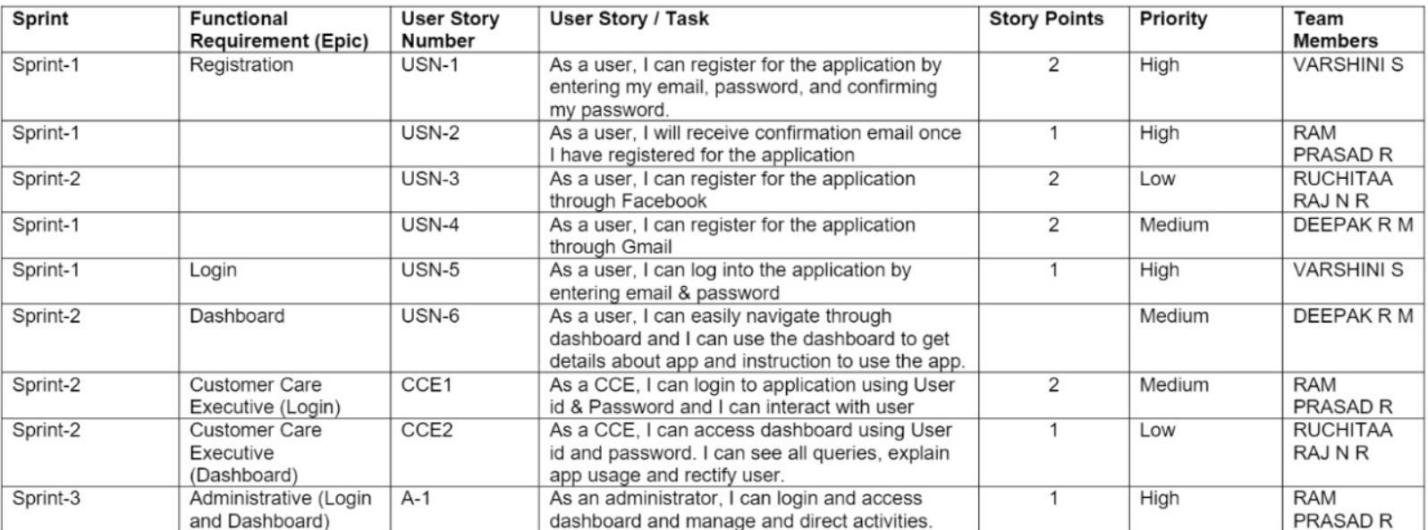
The below diagram illustrates the architecture of the solution proposed. The programming language that’ll be used to analyses data and build the machine language is Python as it is easier to work on and has several built-in functions that is user friendly and fast enough to give the results. The application will be built using Flask as it is a simple Python framework that can be used to build web applications. It easily encapsulates the trained machine learning model. The user interface will be built using HTML, CSS and JavaScript. The attributes mentioned are the actual attributes that’ll be taken into account for prediction in case of input data. The percentage of train and test data split will depend on the performance of the model during testing.

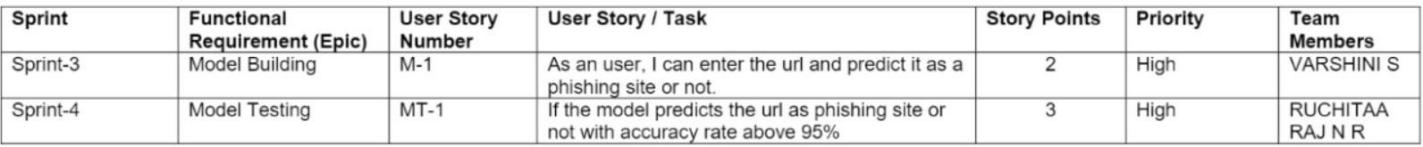


# User Stories

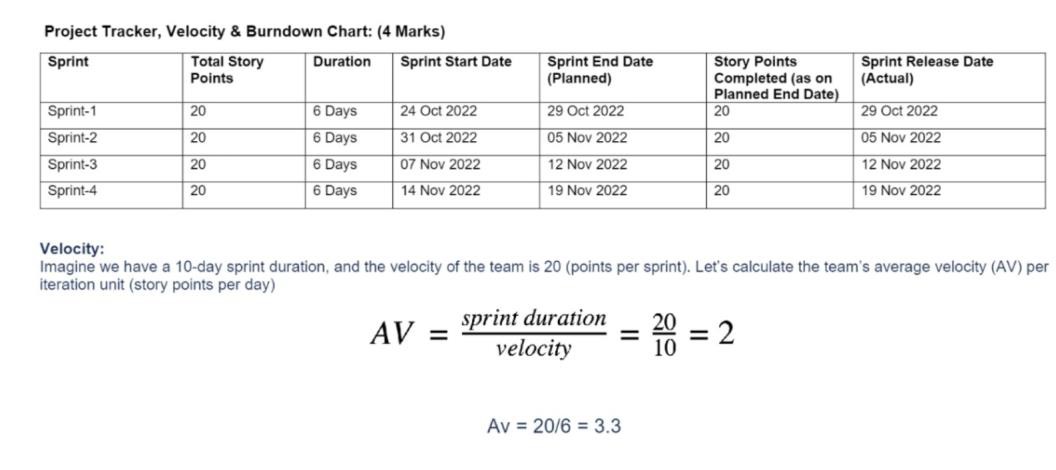


1. **PROJECT PLANNING & SCHEDULING**
   1. **Sprint Planning & Estimation**

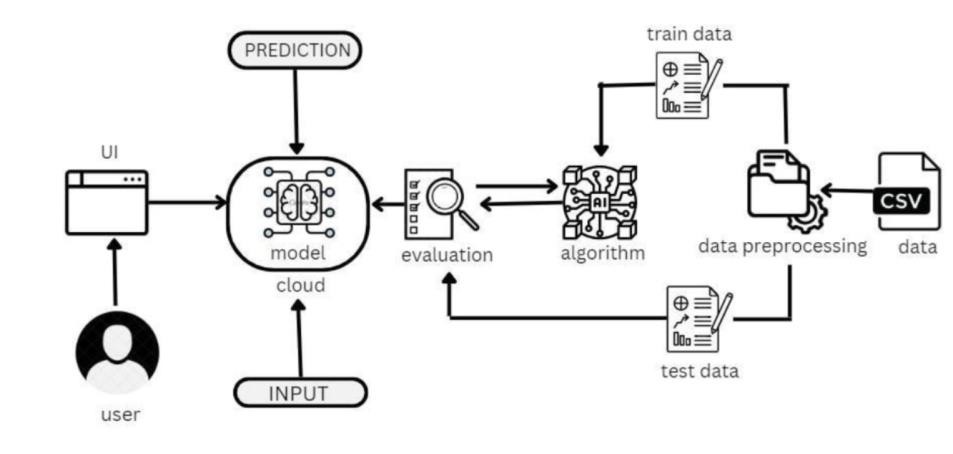




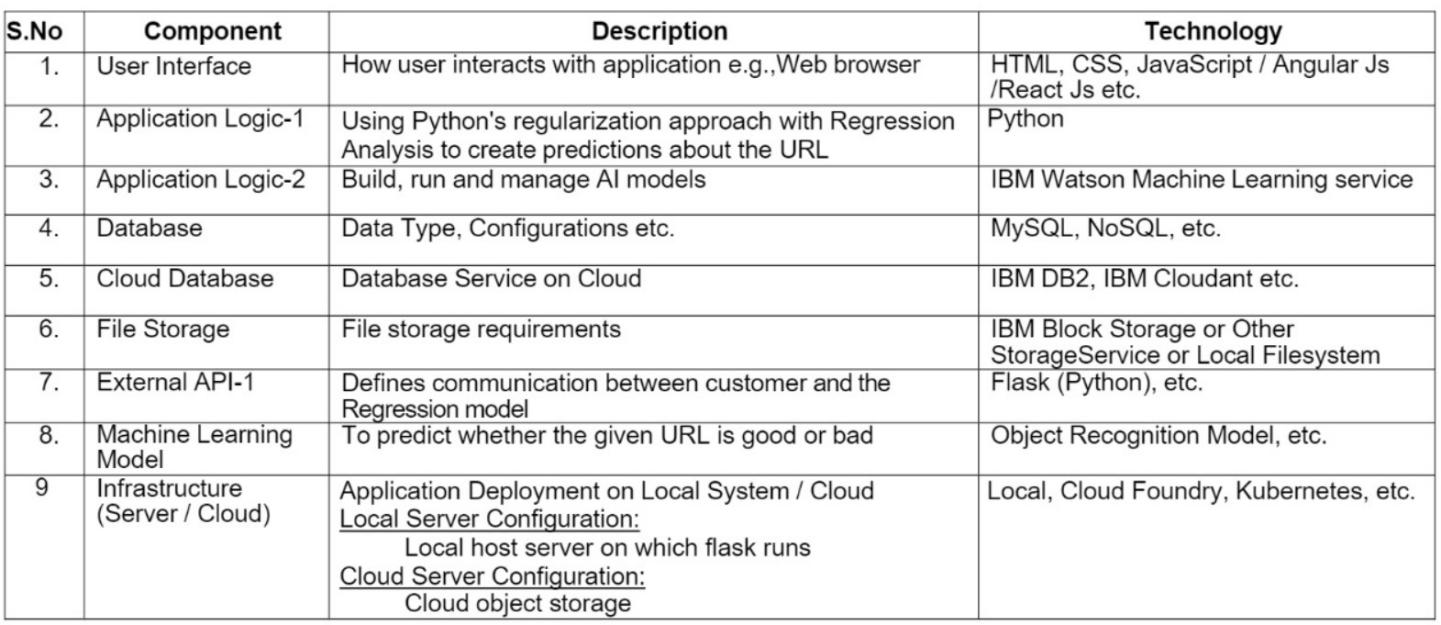
* 1. **Sprint Delivery Schedule**



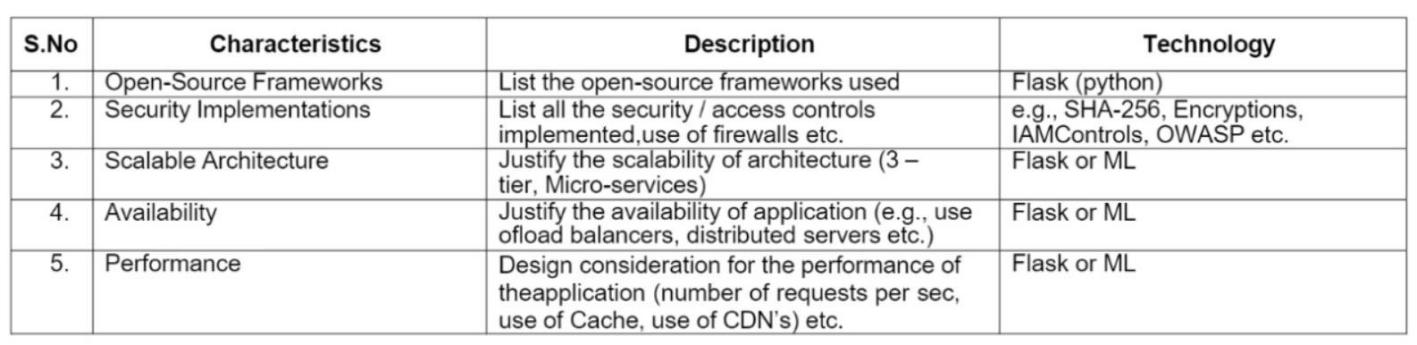
1. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
   1. **Feature 1**



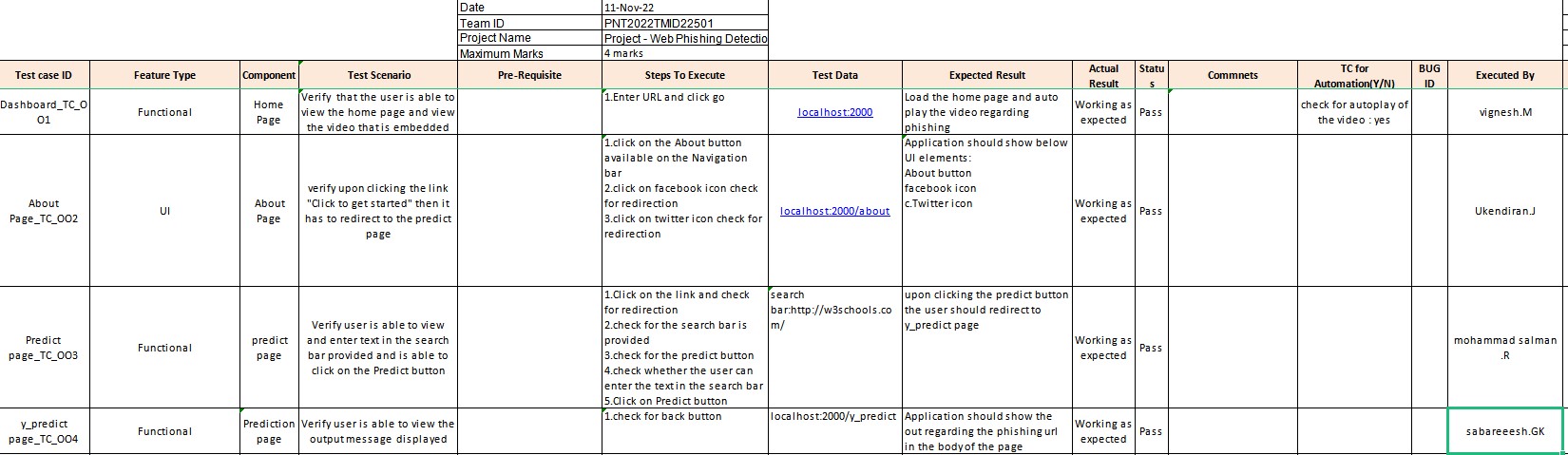
Technological Stack

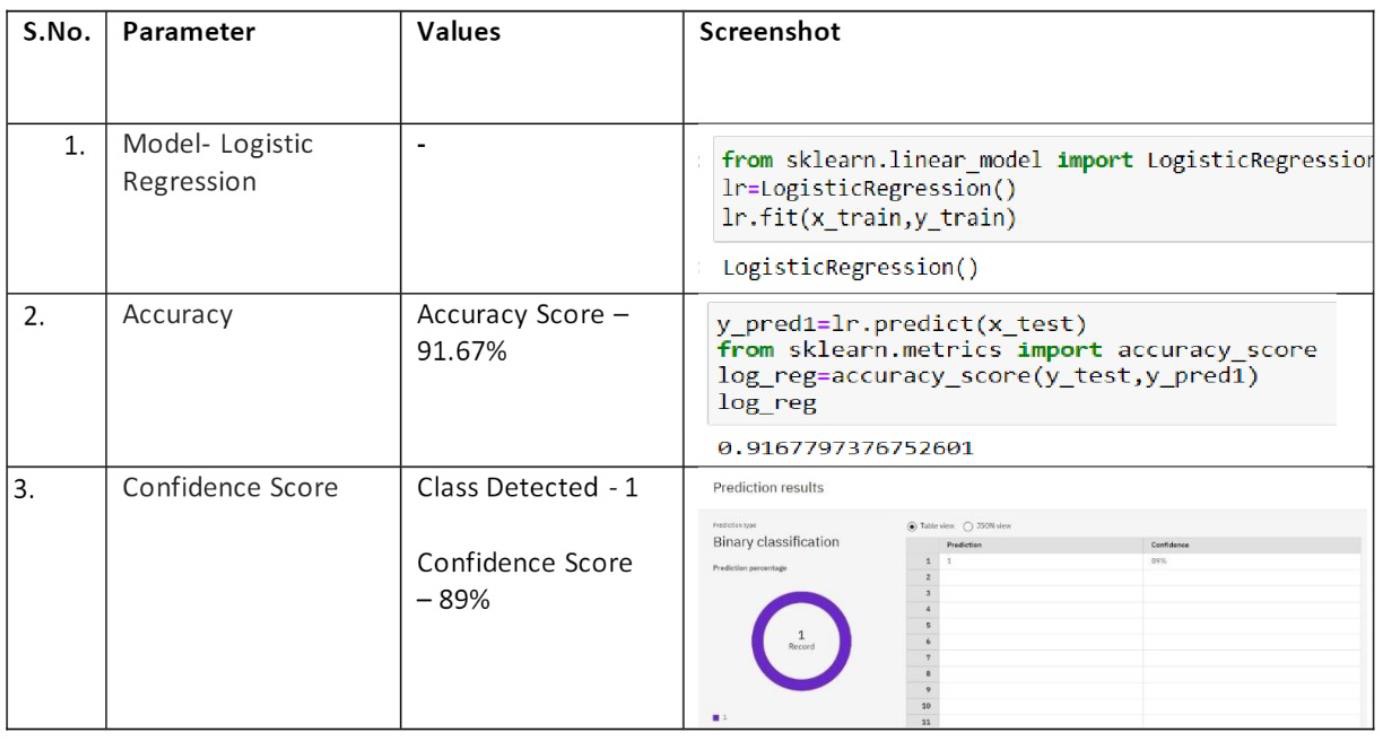


# Feature 2



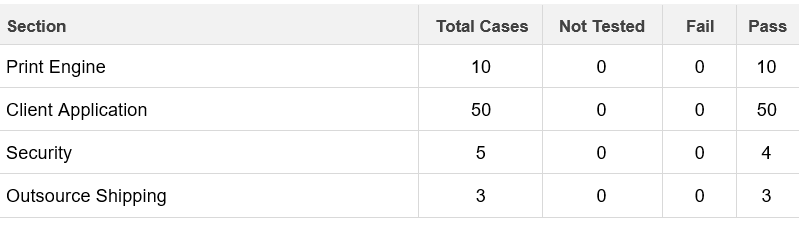
1. **TESTING**
   1. **Test Cases**

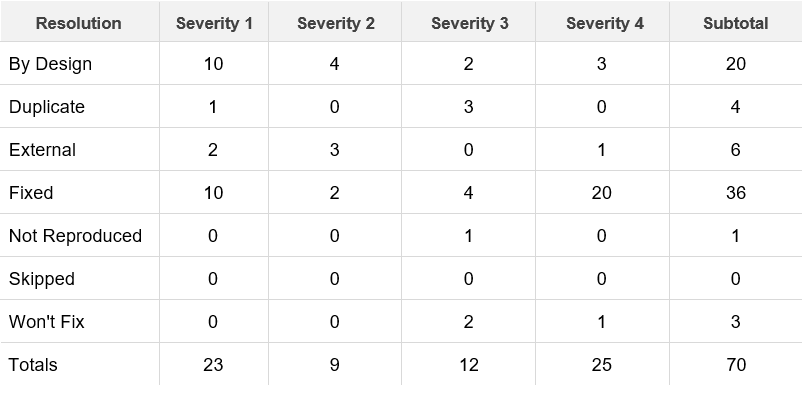




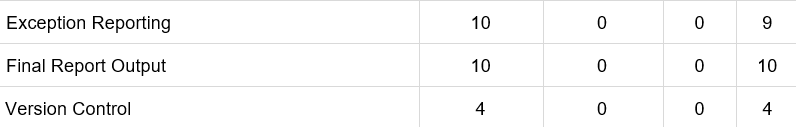
* 1. **User Acceptance Testing**

Defect Analysis



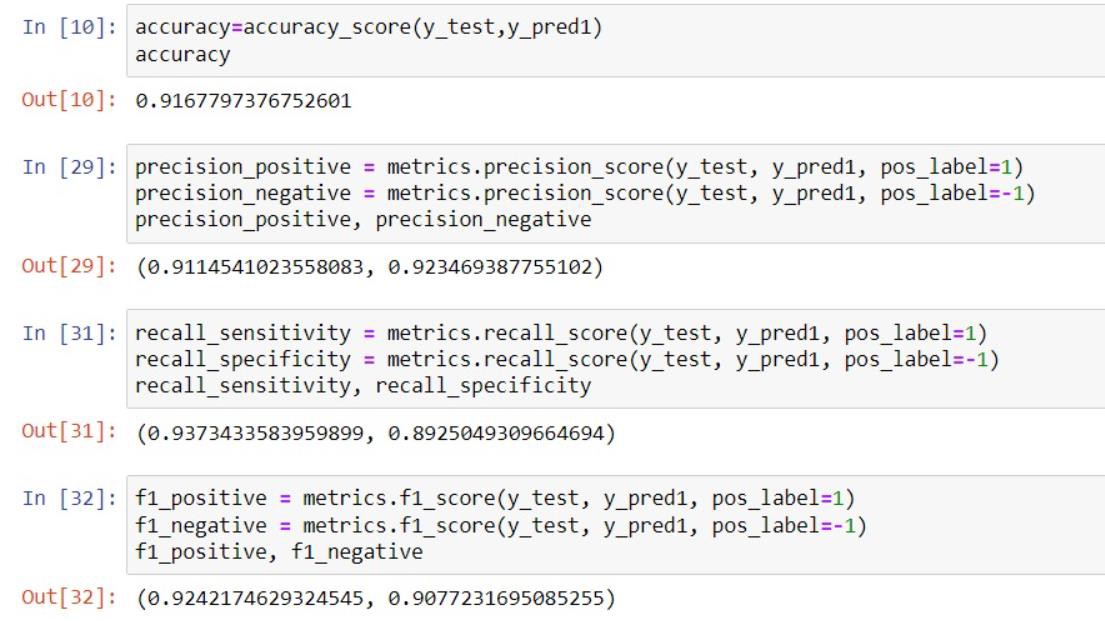


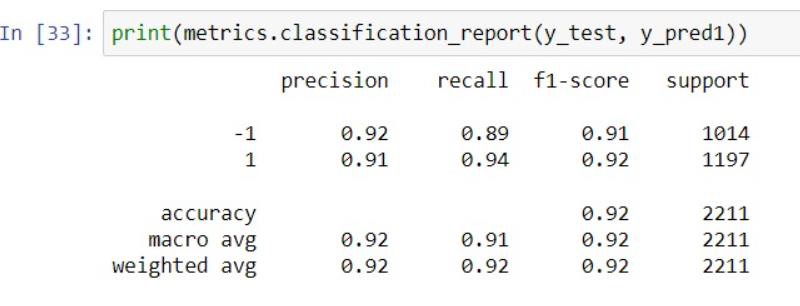
Test Case Analysis



# RESULTS

* 1. **Performance Metrics**





1. **ADVANTAGES & DISADVANTAGES**

Advantages:

1. Build secure connection between uses male transfer agent (MTA) in male user agent (MUA)
2. Provide clear idea about the effective levels of each classifier on phishing email detection
3. High level of accuracy may take the advantages of many classifiers
4. High level of accuracy
5. Create new type of features like Markov features
6. Fast in classification process
7. Fast, less consuming memory, high accuracy, evolving with time and online working Disadvantages:
8. Time consuming, huge number of features and consumes memory
9. Non standard classifier
10. Time consuming because this technique has many layers to make the final result.
11. Huge number of features, many algorithms for classification which mean time consuming.
12. Expensive and need large mail server and high memory requirement.
13. Less accuracy because it depends on unsupervised learning and needs feed continuously.

# CONCLUSION

The system designed is used to prevent valuable information from leak out, produce better control mechanism and alerts user to keep the private information safe. Like any other program, there are improvements which could be made into the system. The proposed system has been identified and chosen to address the web phishing. The application is designed to show awareness, features that can be displayed, safety of the website. Its unique features such as capturing blacklisted URLs from the browser directly to verify the validity of the website, notifying the user on unsafe websites by entering the URL and checking its safety. In our project, we used machine learning classification algorithm to differentiate whether the site is safe or unsafe. With the higher accuracy of the model, a web application was build using the flask framework. It was also deployed in IBM cloud as an extension.

# FUTURE SCOPE

Like any other program, there are improvements which could be made into the system. Based on the capabilities which the current system processes, a pop up could be displayed when accessing the phishing site. Further notification through email can also be sent to assist the used to be alerted when they are trying to access a blacklisted website. A text message integration would be a grater recommendation that could improve the program in future. The future version of the application could also implement an option to directly notify the phishing website with a text message. The program could be made to access the list as an attachment. This text message integration would further enhance the usability of the application. This could be further improved to be added as a chrome extension.

# APPENDIX

Code:

1. model.ipynb

import pandas as pd

import numpy as np

from sklearn.preprocessing import MinMaxScaler

from sklearn.metrics import confusion\_matrix,accuracy\_score

# Reading the dataset

# Importing Dataset

ds = pd.read\_csv("data\_website.csv") ds.head()

# Handling null values ds.info() ds.isnull().any()

# Splitting the data

# removing index column in independent dataset x = ds.iloc[:,1:31].values

y = ds.iloc[:,-1].values print(x,y)

# splitting data into train and test

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)

# Model Building

from sklearn.linear\_model import LogisticRegression lr=LogisticRegression()

lr.fit(x\_train,y\_train) y\_pred1=lr.predict(x\_test)

from sklearn.metrics import accuracy\_score log\_reg=accuracy\_score(y\_test,y\_pred1) log\_reg

import pickle

pickle.dump(lr,open('Phishing\_Website.pkl','wb'))

1. app.py

from flask import Flask, request, render\_template import numpy as np

import pandas as pd

from sklearn import metrics import warnings

import pickle warnings.filterwarnings('ignore') from feature import FeatureExtraction import math

file = open("model.pkl","rb") gbc = pickle.load(file) file.close()

app = Flask( name ,*template\_folder*="templates")

@app.route("/", *methods*=["GET", "POST"]) def index():

if request.method == "POST":

url = request.form["url"] obj = FeatureExtraction(url)

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

y\_pred =gbc.predict(x)[0]

*#0 - unsafe*

*#1 - safe*

y\_pro\_phishing = gbc.predict\_proba(x)[0,0] y\_pro\_non\_phishing = gbc.predict\_proba(x)[0,1] print("phi ",y\_pro\_phishing)

print("non phi ",y\_pro\_non\_phishing)

x = math.floor(y\_pro\_non\_phishing\*1000)/10 print(x)

return render\_template('index.html',*xx* =x,*url*=url )

*#home page render*

return render\_template("index.html", *xx* =-1)

if name == " main ": app.run(*debug*=True,*port*=2002)

1. index.html

<!DOCTYPE *html*>

<html *lang*="en">

<head>

<meta *charset*="UTF-8">

<meta *http-equiv*="X-UA-Compatible" *content*="IE=edge">

<meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

<meta *name*="description" *content*="This website is developed to identify the safety of url.">

<meta *name*="keywords" *content*="phishing url,phishing,cyber security,machine learning,classifier,python">

*<!-- BootStrap -->*

<link *rel*="stylesheet" *href*="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"

*integrity*="sha384- 9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk" *crossorigin*="anonymous">

<link *type*="text/css" *href*="{{ url\_for('static', filename='styles/styles.css') }}" *rel*="stylesheet">

<link *href*=["https://fonts.googleapis.com/css2?family=Changa:wght@700&display=swap"](https://fonts.googleapis.com/css2?family=Changa%3Awght%40700&amp;display=swap) *rel*="stylesheet">

<title>URL phishing detection</title>

</head>

<body>

<*center*> <h1> WEB PHISHING URL DETECTION </h1> </*center*>

<*center*><h3>Check if the site is safe or not!</h3></*center*>

<br>

<*center*> <img *class*="image image-contain" *src*="https://[www.managedsolution.com/wp-content/uploads/2019/08/employee-](http://www.managedsolution.com/wp-content/uploads/2019/08/employee-) awareness-of-phishing-social-engineering-attacks.jpg" *alt*="MDN logo" /> </*center*>

<div *class*=" container">

<div *class*="row">

<div *class*="form col-md" *id*="form1">

<br>

<form *action*="/" *method* ="post">

<input *type*="text" *class*="form input" *name* ='url' *id*="url" *placeholder*="Enter URL" *required*="" />

<label *for*="url" *class*="form label">URL</label>

<button *class*="button" *role*="button" >Click here</button>

</form>

</div>

<div *class*="col-md" *id*="form2">

<br>

<h6 *class* = "right "><a *href*= {{ *url }} target*="\_blank">{{ url

}}</a></h6>

<br>

<h3 *id*="prediction"></h3>

<button *class*="button2" *id*="button2" *role*="button" *onclick*="window.open**(**'{{url}}'**)**" *target*="\_blank" >Still want to Continue</button>

<button *class*="button1" *id*="button1" *role*="button" *onclick*="window.open**(**'{{url}}'**)**" *target*="\_blank">Continue</button>

</div>

</div>

<br>

</div>

*<!-- JavaScript -->*

<script *src*="https://code.jquery.com/jquery-3.5.1.slim.min.js" *integrity*="sha384-

DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"

*crossorigin*="anonymous"></script>

<script

*src*=["https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"](https://cdn.jsdelivr.net/npm/popper.js%401.16.0/dist/umd/popper.min.js) *integrity*="sha384-

Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"

*crossorigin*="anonymous"></script>

<script

*src*="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js" *integrity*="sha384-

OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI"

*crossorigin*="anonymous"></script>

<script>

let x = '{{xx}}'; console.log**(**x**)**; let num = x;

if **(**0<=x && x<50**)**{ num = 100-num;

}

let txtx = num.toString**()**; if**(**x<=100 && x>=50**)**{

var label = "Website is "+txtx +"% safe to use"; document.getElementById**(**"prediction"**)**.**innerHTML** = label; document.getElementById**(**"button1"**)**.style.**display**="block";

}

else if **(**0<=x && x<50**)**{

var label = "Website is "+txtx +"% unsafe to use" document.getElementById**(**"prediction"**)**.**innerHTML** = label ; document.getElementById**(**"button2"**)**.style.**display**="block";

}

</script>

</body>

</html>

1. styles.css

\*,

\**::after*,

\**::before* { margin: 0;

padding: 0;

box-sizing: inherit; font-size: 62,5%;

}

*.image* {

width: 1000px; height: 500px; border-radius: 50%;

}

h1 {

font-family: 'Changa', sans-serif;

}

*.image-contain* { object-fit: contain;

object-position: center;

}

*.image-cover* { object-fit: cover;

object-position: center;

}

body {

padding: 5% 5%; background: #0f2027;

background: linear-gradient(to right,#b2ca71, #a6ce7d, #1d210e); justify-content: center;

align-items: center;

height: 100vh; color: #fff;

}

*.form label* {

font-family: 'Changa', sans-serif; font-size: 1.2rem;

margin-left: 2rem; margin-top: 0.7rem; display: block; transition: all 0.3s;

transform: translateY(0rem);

}

*.form input* { top: -24px;

font-family: 'Changa', sans-serif; color: #333;

font-size: 1.2rem; padding: 1.5rem 2rem; border-radius: 0.5rem;

background-color: rgb(255, 255, 255); border: none;

width: 75%; display: block;

border-bottom: 0.3rem solid transparent; transition: all 0.3s;

}

*.form input*:placeholder-shown + *.form label* { opacity: 0;

visibility: hidden;

-webkit-transform: translateY(+4rem); transform: translateY(+4rem);

}

*.button* { appearance: button;

background-color: transparent;

background-image: linear-gradient(to bottom, #fff, #f8eedb); border: 0 solid #020712;

border-radius: .5rem; box-sizing: border-box; color: #110801;

column-gap: 1rem; cursor: pointer;

display: flex;

font-family: 'Changa', sans-serif; font-size: 100%;

font-weight: 700; line-height: 24px; margin: 0;

outline: 2px solid transparent; padding: 1rem 1.5rem;

text-align: center; text-transform: none;

transition: all .1s cubic-bezier(.4, 0, .2, 1); user-select: none;

-webkit-user-select: none; touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);

}

*.button:active* {

background-color: #f3f4f6;

box-shadow: -1px 2px 5px rgba(81,41,10,0.15),0px 1px 1px rgba(81,41,10,0.15); transform: translateY(0.125rem);

}

*.button:focus* {

box-shadow: rgba(72, 35, 7, .46) 0 0 0 4px, -6px 8px 10px rgba(81,41,10,0.1), 0px 2px 2px rgba(81,41,10,0.2);

}

*.main-body*{ display: flex;

flex-direction: row;

}

*.button1*{ appearance: button;

background-color: transparent;

background-image: linear-gradient(to bottom, rgb(160, 245, 174), #37ee65); border: 0 solid #e5e7eb;

border-radius: .5rem; box-sizing: border-box; color: #482307;

column-gap: 1rem; cursor: pointer; display: flex;

font-family: 'Changa', sans-serif; font-size: 100%;

font-weight: 700; line-height: 24px; margin: 0;

outline: 2px solid transparent; padding: 1rem 1.5rem;

text-align: center; text-transform: none;

transition: all .1s cubic-bezier(.4, 0, .2, 1); user-select: none;

-webkit-user-select: none; touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2); display: none;

}

*.button2*{ appearance: button;

background-color: transparent;

background-image: linear-gradient(to bottom, rgb(252, 162, 162), #ee3737); border: 0 solid #e5e7eb;

border-radius: .5rem; box-sizing: border-box; color: #482307;

column-gap: 1rem; cursor: pointer; display: flex;

font-family: 'Changa', sans-serif; font-size: 100%;

font-weight: 700; line-height: 24px; margin: 0;

outline: 2px solid transparent; padding: 1rem 1.5rem;

text-align: center; text-transform: none;

transition: all .1s cubic-bezier(.4, 0, .2, 1); user-select: none;

-webkit-user-select: none; touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2); display: none;

}

*.right* { right: 0px;

width: 300px;

}

@media (max-width: 576px) {

*.form* {

width: 100%;

}

}

*.abc*{

width: 50%;

}

v. Scoring\_Endpoint.py

from flask import Flask, request, render\_template import numpy as np

import pandas as pd

from sklearn import metrics import warnings

import pickle import requests

warnings.filterwarnings('ignore') from feature import FeatureExtraction import math

file = open("model.pkl","rb") gbc = pickle.load(file) file.close()

API\_KEY = "<YOUR\_API\_KEY>"

token\_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":

API\_KEY, "grant\_type": 'urn:ibm:params:oauth:grant-type:apikey'}) mltoken = token\_response.json()["access\_token"]

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

app = Flask(\_\_name , template\_folder="templates")

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

if request.method == "POST": url = request.form["url"]

obj = FeatureExtraction(url)

x = np.array(obj.getFeaturesList()).reshape(1,30) y\_pred =gbc.predict(x)[0]

#0 - unsafe

#1 - safe

y\_pro\_phishing = gbc.predict\_proba(x)[0,0] y\_pro\_non\_phishing = gbc.predict\_proba(x)[0,1]

# if(y\_pred ==1 ):

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

# payload\_scoring = {"input\_data": [{"fields": [array\_of\_input\_fields], "values": [array\_of\_values\_to\_be\_scored, another\_array\_of\_values\_to\_be\_scored]}]} payload\_scoring = {"input\_data": [{"fields": ["UsingIP","LongURL","ShortURL","Symbol@","Redirecting//","PrefixSuffix- ","SubDomains","HTTPS","DomainRegLen","Favicon","NonStdPort","HTTPSDomainURL","Re questURL","AnchorURL","LinksInScriptTags","ServerFormHandler","InfoEmail","Abnorm alURL","WebsiteForwarding","StatusBarCust","DisableRightClick","UsingPopupWindow"

,"IframeRedirection","AgeofDomain","DNSRecording","WebsiteTraffic","PageRank","Go ogleIndex","LinksPointingToPage","StatsReport"

], "values": [1,1,1,1,1,-1,-1,-1,-1,1,1,1,1,-1,-1,1,1,1,0,1,1,1,1,-1,-1,-1,-

1,1,0,1]}]}

response\_scoring = requests.post('https://us- south.ml.cloud.ibm.com/ml/v4/deployments/27c47874-fd3f-4c1c-aefa- afa3d1738374/predictions?version=2022-11-17', json=payload\_scoring, headers={'Authorization': 'Bearer ' + mltoken})

print("Scoring response for given input") print(response\_scoring.json()) predictions=response\_scoring.json()

x = math.floor(y\_pro\_non\_phishing\*1000)/10 pred=print(predictions['predictions'][0]['values'][0][0]) if(pred == -1):

print("The Website is unsafe") else:

print("The Website is safe2020)

GitHub link:

<https://github.com/IBM-EPBL/IBM-Project-20590-1659755673>