DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROJECT REPORT

WEB PHISHING DETECTION

Submitted in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

BY

VIGNESHINI I
MARI SUPRIYA M
ESAKKIAMMAL M
SAKTHILAKSHMI B

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

It is hereby certified that the project work entitled “WEB PHISHING DETECTION” is a bonafide work carried out by Vigneshini I(950819104051)Mari Supriya M(950819104028),Esakkiammal M(950819104707), Sakthilakshmi B (950819104703) in partial fulfilment for the award of Bachelor of Engineering in COMPUTER SCIENCE AND ENGINEERING of the New Horizon College of Engineering during the year 2019-2020. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said Degree.

ABSTRACT

The criminals, who want to obtain sensitive data, first create unauthorized replicas of a real website and e-mail.

The nature of website creation is one of the reasons that the Internet has grown so rapidly as a communication medium.
Phisher then send the "spoofed" e-mails to as many people as possible in an attempt to lure them into the scheme.

When these e-mails are opened or when a link in the mail is clicked, the consumers are redirected to a spoofed website, appearing to be from the legitimate entity.

We discuss the methods used for detection of phishing Web sites based on url importance properties.

CONTENT

INTRODUCTION

1.1 Project Overview:

There are a number of users who purchase products online and make payment through various websites. There are multiple websites who ask users to provide sensitive data such as username, password or credit card details etc. often for malicious reasons. This type of website is known as a phishing website. In order to detect and predict phishing websites, we proposed an intelligent, flexible and effective system that is based on using classification Data mining algorithm. We implemented classification algorithms and techniques to extract the phishing data sets criteria to classify their legitimacy. The 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 the website our system will use a data mining algorithm to detect whether the website is a phishing website or not. This application can be used by many E-commerce enterprises in order to make the whole transaction process secure. Data mining algorithm used in this system provides better performance as compared to other traditional classifications algorithms. With the help of this system users can also purchase products online without any hesitation. Admin can add phishing website URL or fake website URL into a system where the system could access and scan the phishing website and by using an algorithm, it will add new suspicious keywords to the database. System uses machine learning techniques to add new keywords into the database.

1.2.Purpose:

The main purpose of the project is to detect the fake or phishing websites who are trying to get access to the sensitive data or by creating the fake websites and trying to get access to the user's personal

credentials. We are using machine learning algorithms to safeguard the sensitive data and to detect the phishing websites who are trying to gain access to sensitive data.

LITERATURE SURVEY

Introduction:

In recent times, Phishing has become an important area of concern for security researchers because it is not difficult to develop a phishing website, which looks so close to legitimate websites. Experts can identify phishing websites but not all the users can identify the phishing website and such users become the victim of phishing attacks. Main aim of the attacker is to steal bank account details and personal information. In United States businesses, there is a loss of US$2billion per year because their clients become victims of phishing. As per Index Report released in 2020, it was estimated that the annual worldwide impact of phishing could be as high as $1.6 million. Phishing attacks are becoming successful because of lack of user awareness. Since phishing attacks exploit the weaknesses found in users, it is very difficult to reduce them but it is very important to enhance phishing detection techniques. In recent times, Phishing has become an important area of concern for security researchers because it is not difficult to develop the phishing website, which looks so close to legitimate websites. Experts can identify phishing websites but not all the users can identify the phishing website and such users become the victim of phishing attacks. Main aim of the attacker is to steal bank account details and personal information. In United States businesses, there is a loss of US$2billion per year because their clients become victims of phishing. As per Index Report released in 2020, it was estimated that the annual worldwide impact of phishing could be as high as $1.6 million. Phishing attacks are becoming successful because of lack of user awareness. Since phishing attacks exploit the weaknesses found in users, it is very difficult to reduce them but it is very important to enhance phishing detection techniques.

Literature review:

Phishing detection and protection scheme (1):

Developing with the anti-phishing methods, phishers use various phishing methods and more complex and hard-to-detect approaches. The most straightforward way for a phisher to swindle people is to make the phishing web page similar to their target. However, many distinctive features can distinguish the original legitimate website from the clone phishing website like the spelling error, image alteration, long URL address and abnormal DNS records. The full list is revealed in Table 3 which is used later in our analysis and classification study. If an attacker clones a legitimate website as a whole or designed to look similar as they usually do in most attacks in recent times,our approach is that similar looking phishing web page con-tent is not left for the users to check for the indicator or the authenticity attentively, but can detect by automated methods. Our approach is based on website phishing detection using the features of the site, content and their appear-ance. These properties are stored in a local database (Excel table) as a knowledge model and first compared with the newly loaded site at the time of loading against the dangerous web page offline. After the comparison was unable to detect the similarity, then the critical approach was to compare the legitimate and fake using the features of the website with machine learning for an intelligent decision. The critical contribution of our approach includes

Result:

The output is determined by the classifier, in the phishing detection stage which predicts if the web page is suspicious, legiti-mate or phishing. The knowledge model and plug-in development will be developed at a later stage.

System detection related work (2):

Nowadays most people use the internet for various purposes such as online shopping like purchasing or selling products, chatting with friends, sending mail. Internet users now spend more time on social networking sites. Information can spread very fast and easily within the social media networks. Social media systems depend on users for content contribution and sharing.Facebook had over 1.3 billion active users as of June 2014. There are over 1.3 billion (the number is keep growing) pages from various categories, such as company, product/service, musician/band, local business, politician, government, actor/director, artist, athlete, author, book, health, beauty, movie, cars,clothing,

community. Fans not only can see information submitted by the page, but also can post comments, photos and videos to the page.

Result:

Domain anomaly features are used to identify possible malicious domains based on lexical and reputation factors, whereas social anomaly features represent anomalous user behaviours in social communications.

Learning to Detect Phishing Emails (3):

An alternative for detecting these attacks is a relevant process of reliability of machine on a trait intended for the reflection of the besieged deception of user by means of electronic communication.This approach can be used in the detection of phishing websites, or the text messages sent through emails that are used for trapping the victims. Approximately, 800 phishing mails and 7,000 non-phishing mails are traced till date and are detected accurately over 95% of them along with the categorization on the basis of 0.09% of the genuine emails.

Result:

We can just wrap up with the methods for identifying the deception, along with the progression nature of attacks.

Phishing websites machine learning (4):

Phishing URL is a widely used and common technique for cybersecurity attacks. Phishing is a cybercrime that tries to trick the targeted users into exposing their private and sensitive information to the attacker. The motive of the attacker is to gain access to personal information such as usernames,login credentials, passwords, financial account details, social networking data, and personal addresses.These private credentials are then often used for malicious activities such as identity theft, notoriety,financial gain, reputation damage, and many more illegal activities. This paper aims to provide a comprehensive and comparative study of various existing free service systems and research-based systems used for phishing website detection. The systems in this survey range from different detection techniques and tools used by

many researchers. The approach included in these researched papers ranges from Blacklist and Heuristic features to visual and content-based features. The studies presented here use advanced machine learning and deep learning algorithms to achieve better precision and higher accuracy while categorising websites as phishing or benign. This article would provide a better understanding of the current trends and existing systems in the phishing detection domain.

Result:

Phishing URL detection plays a pivotal role for many cybersecurity software and applications. In this paper, we researched and reviewed works based on the advanced machine learning techniques and approaches that promise a fresh approach in this domain.

Support vector machine (5):

The existing anti-phishing approaches use the blacklist methods or features based machine learning techniques. Blacklist methods fail to detect new phishing attacks and produce high false positive rate. Moreover, existing machine learning based methods extract features from the third party, search engine, etc. Therefore, they are complicated, slow in nature, and not fit for the real-time environment. To solve this problem, this paper presents a machine learning based novel anti-phishing approach that extracts the features from client side only. Below architecture diagram as shown in Fig. 1. represents mainly the flow of training phase to Detection phase. First data needs to be pre-processed and feature extraction using different feature sets and later we need to train this dataset with the corresponding algorithms and the output is displayed.

Result: In future we can use a combination of any other two or more classifiers to get maximum accuracy.

We can also explore various phishing techniques that use Lexical features.

2.2:REFERENCES:

TITLE	PROBLEM IDENTIFIED	METHODOLOGY	STRENGTH	WEAKNESS
Phish Shield: A desktop application to detect phishing webpages through heuristic approach (Rao & ali,2015)	To detect URL and website content of phishing pages	Heuristic approach	Ability to detect zero hour phishing attacks & increased speed in detecting speed in detecting phishing attack	High computational cost,inability to immediately update the whitelist & blacklist
Mitigating cyber identity fraud advanced multi anti phishing technique(Yusuf et al,2013)	To tackle loopholes in electronic payment system security challenges in online banking transaction	Semantic content analysis,Earth mover Distance(EMD) & biometric authentication with fingerprint	Detecting of phishing webpages & preventing unauthorised online banking transfer & withdrawal	It achieved 80% true negative
Efficient prediction of phishing website using supervised learning algorithms,Santhana Laksmi.v & vijaya MS,2011	Phishers are using new techniques to break all antiphishing mechanisms	Supervised learning algm,ie,.multi layer perceptron ,decision tree induction & machine learning techniques to model the prediction task & naive Bayes classification to explore result	It can predict whether a given website is legitimate or phishing website	Time taken to build the model & predictions accuracy is high in the case of decision tree induction
Anti phishing based on	Blacklist is not completely	Naive bayesian classifier	It keeps a whitelist of users	It requires gathering the

automated individual whitelist (AIWL)YE cao,Weli han & Yueran le,2008cao,Weli han & Yueran le,2008	effective in detecting phishing URL because of partial list of global phishing sites		all familiar login user interface (LUIs)of website .it guides against pharming attacks & low false positive	website IP & this is time consulting as IP needs to be changed

2.3.PROBLEM STATEMENT:

Customer Problem Statement Template:

Create a problem statement to understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.

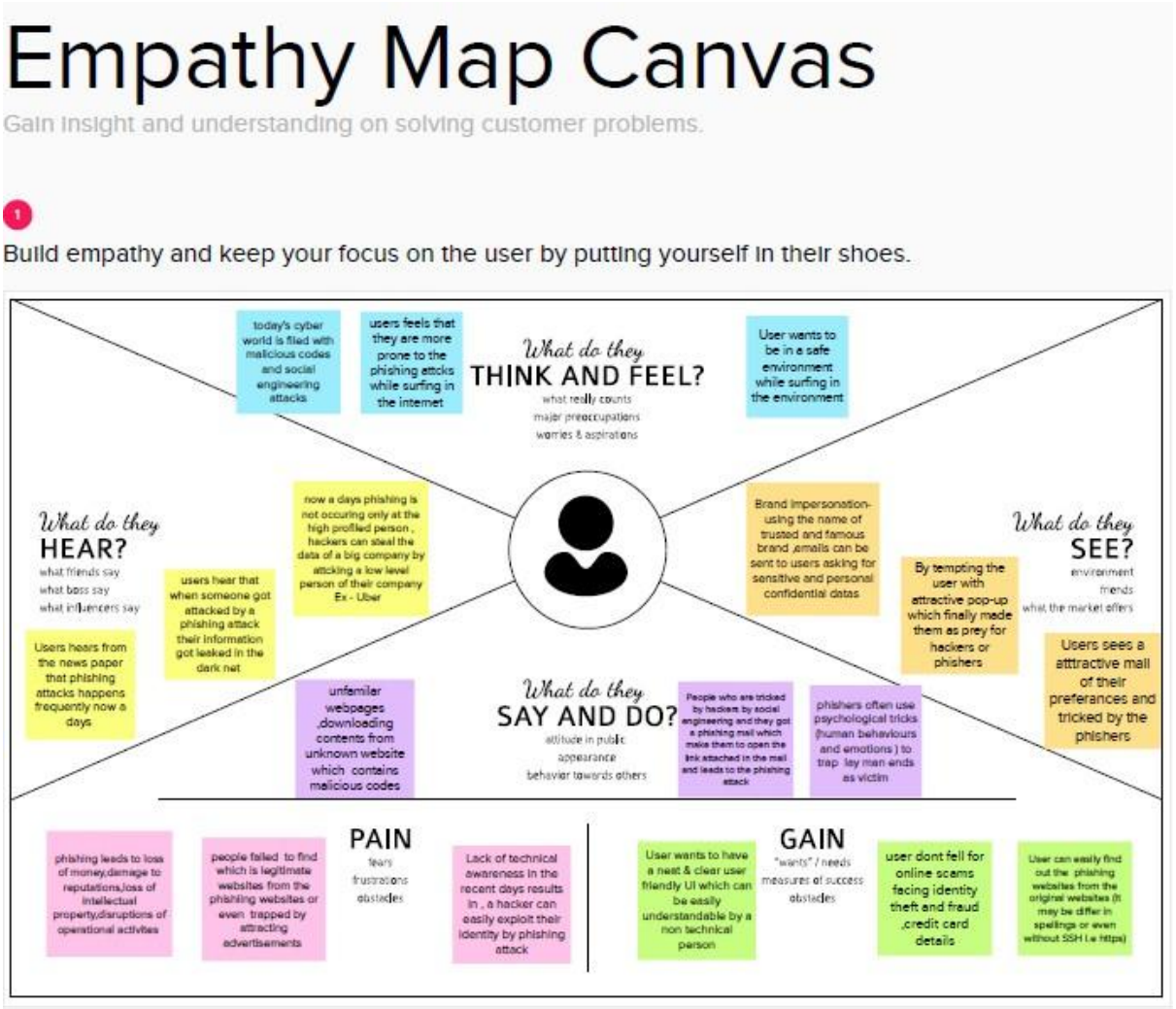
A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you'll also be able to empathise with your customers, which helps you better understand how they perceive your product or service.

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

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	A model	purchase latest design boutique for event	i feel security concern	the website is not completely secured	disappointed

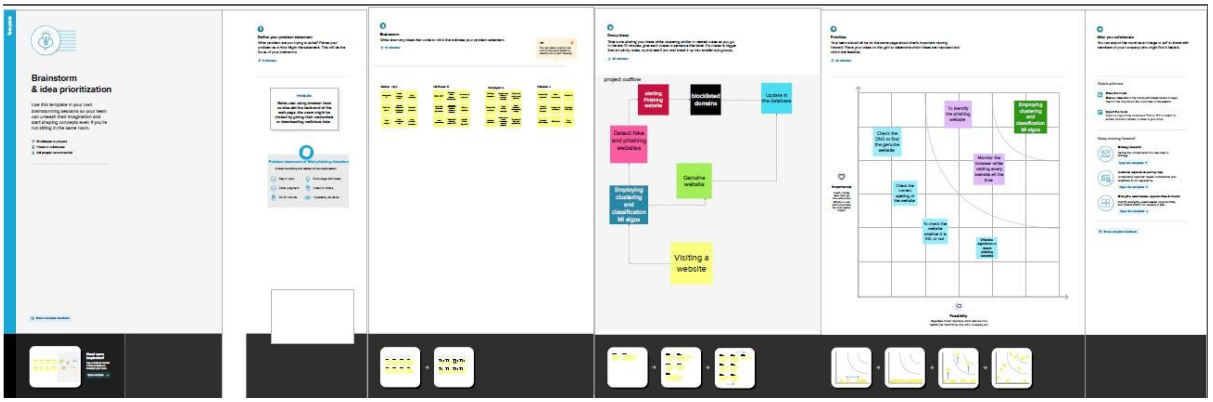
IDEATION & PROPOSED SOLUTION

3.1. Empathy Map Canvas:



3.2. Ideation & Brainstorming:

Brainstorming:

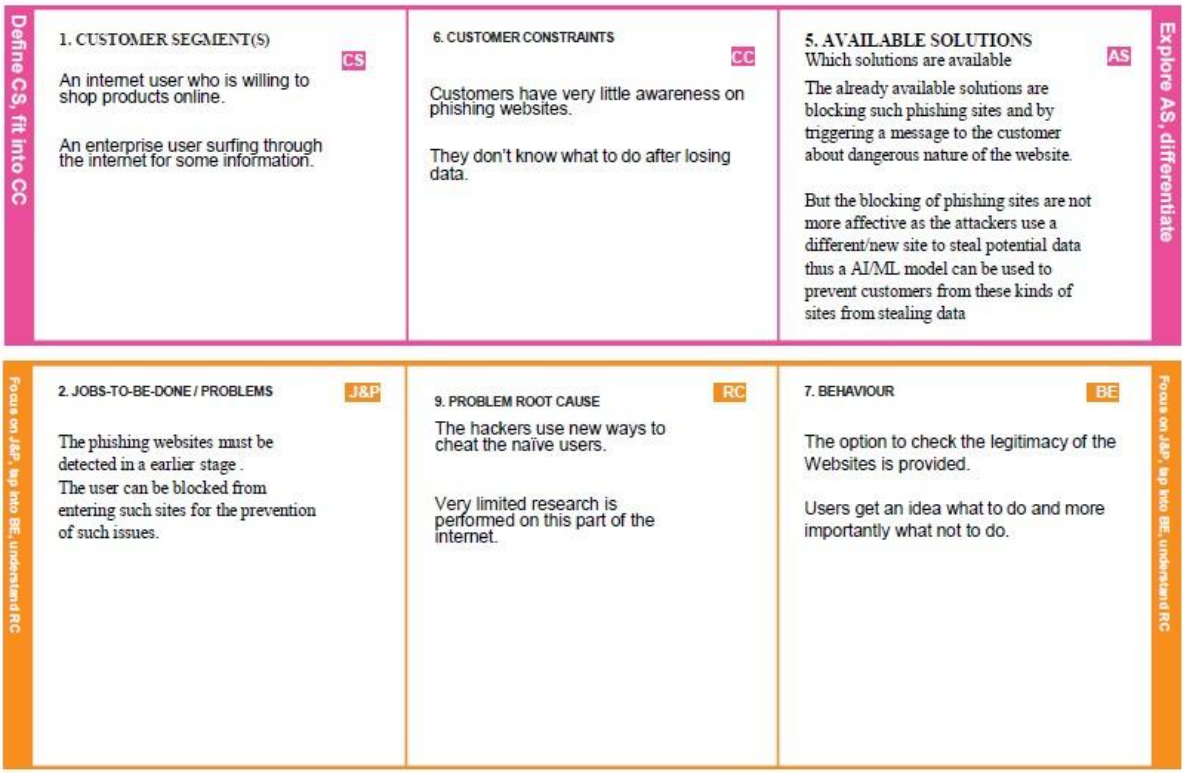


3.3. Proposed Solution:

Sno.	Parameter	Description
1.	Problem Statement	An online user needed to purchase something online. So he entered the online website through the internet. It takes some time to display the product. He started to see all the products. He searches for the necessary things on an online website. At last he found the needed products. After that he entered all the credit card details, username and password for purchasing the things online. Then he received the message "Your order is placed and the transaction is successfully completed. You will receive the ordered product within 2 days". After that, within 24 hours he got a message in mobile and the bank account was empty, then the customer was shocked . Then only he realised that it was a fake website and his bank account details were stolen by a hacker .To avoid this scenario. We need to solve this problem by using the Web Phishing Detection.
2.	Solution description	To overcome the problem of phishing website whenever we are clicking on one website it must show an alert box like it is a secure website or it is not a secure website Then another way is that we can scan the website in order to prevent our system or mobile from the phishing attack. Even though technologies are there we as the user have to be aware of the websites whether it is secure or not. We should not click any unwanted websites.

3.	Uniqueness	The proposed approach has divided the hyperlink specific features into 12 different categories and used these features to train the machine learning algorithms.We have evaluated the performance of our proposed phishing detection approach on various classification algorithms using the phishing and non-phishing websites dataset.
4.	Customer Satisfaction	As we are using some websites but while clicking that website it display an alert box which leads to an aware of the customer which results in satisfaction of the user while using the websites, And another way is that we can scan the website in order to prevent the hacking of the information which makes even more satisfaction to the customer.

3.4. Problem Solution fit:



REQUIREMENT ANALYSIS

4.1 Functional requirement:

A function of a software system is defined in functional requirement and the behaviour of the system is evaluated when presented with specific inputs or conditions which may include calculations, data manipulation and processing and other specific functionality.

- Our system should be able to load air quality data and preprocess data.
- It should be able to analyse the air quality data.
- It should be able to group data based on hidden patterns.
- It should be able to assign a label based on its data groups.
- It should be able to split data into train sets and testset.
- It should be able to train models using a train set.
- It must validate a trained model using testset.
- It should be able to display the trained model accuracy.
- It should be able to accurately predict the air quality on unseen data.

3.2. NON-FUNCTIONAL REQUIREMENTS:

Nonfunctional requirements describe how a system must behave and establish constraints of its functionality. This type of requirement is also known as the system's *quality attributes*. Attributes such as performance, security, usability, compatibility are not the feature of the system, they are a required characteristic. They are "developing" properties that emerge from the whole arrangement and hence we can't compose a particular line of code to execute them. Any attributes required by the customer are described by the specification. We must include only those requirements that are appropriate for our project. Some Non-Functional Requirements are as follows:

- Reliability
- Maintainability
- Performance
- Portability
- Scalability
- Flexibility

Some of the quality attributes are as follows:

3.2.1 ACCESSIBILITY:

Availability is a general term used to depict how much an item, gadget, administration, or condition is open by however many individuals as would be prudent.

In our venture individuals who have enrolled with the cloud can get to the cloud to store and recover their information with the assistance of a mystery key sent to their email ids.

UI is straightforward and productive and simple to utilise.

3.2.2 MAINTAINABILITY:

In programming designing, viability is the simplicity with which a product item can be altered so as to:

- ➢ Correct absconds
- ➢ Meet new necessities

New functionalities can be included in the task based the client necessities just by adding the proper documents to existing venture utilising ASP.net and C# programming dialects. Since writing computer programs is extremely straightforward, it is simpler to discover and address the imperfections and to roll out the improvements in the undertaking.

3.2.3 SCALABILITY:

Framework is fit for taking care of incrementing all out throughput under an expanded burden when assets (commonly equipment) are included.

Framework can work ordinarily under circumstances, for example, low data transfer capacity and substantial number of clients.

3.2.4 PORTABILITY:

Convey ability is one of the key ideas of abnormal state programming. Convenient is the product code base component to have the capacity to reuse the current code as opposed to making new code while moving programming from a domain to another. Venture can be executed under various activity conditions to meet its base setups. Just framework records and dependent congregations would need to be designed in such a case.

The functional requirements for a system describe what the system should do.

Those requirements depend on the type of software being developed,the expected users of the software. These are the statements of services the system should provide,how the system should react to particular inputs and how the system should behave in particular situations.

- Extracting data from CSV files
- Cleaning the data.
- Vector Representation.

Non-functional requirements are not about functionality or behaviour of a system, but rather are used to specify the capacity of a system. They are more related to properties

of the system such as quality, reliability and quick response time. Non- functional requirements come up via customer needs, because of budget, interoperability need such as software and hardware requirement, organisational policies or due to some external factors such as:-

- Basic Operational Requirement
- Organisational Requirement
- Product Requirement
- User Requirement

☐ **Basic Operational Requirement:**

The four primary functions of systems engineering are all performed by the end users, which are the customers. Operational requirements which are given by:

- **Mission profile or scenario:** It is a map which describes the procedures and leads us to the final goal/ objective. The goal of the proposed system is to predict the crop yield prediction for future year using previous year dataset.
- · **Performance:** It basically gives system parameters to reach our goal. Parameters for the proposed system are an accurate predicted value which is compared to the existing system.
- · **Utilisation environments:** It enlists the different permutations and combinations a system can be reused in many other applications which gives better prediction, as well as gives a new approach to prediction techniques
- · **Life cycle:** It discusses the lifespan of a system. As the number of data increases the number of iterations increases, which will give more accuracy to the output.

☐ **Organisational Requirement:**

The Organisational requirement consists of the following types:

- **Process Standards:** To make sure the system is a quality product, IEEE standards have been used during system development.
- **Design Methods:** Design is an important step, on which all other steps in the engineering process are based on.

- · It takes the project from a theoretical idea to an actual product. It gives us the basis of our solution.Because all the steps after designing are based on the design itself, this step affects the quality of the product and is a major player in how the testing and maintenance of a project take place and how successful they are. Following the design to the 'T' is of utmost importance.

☐ **Product Requirement:**

- **Portability:** As the system is Python based, it will run on a platform which is supported by ANACONDA.
- **Correctness:** The system has been put through rigorous testing after it has followed strict guidelines and rules. The testing has validated the data.
- **Ease of Use:** The user interface allows the user to interact with the system at a very comfortable level with no hassles.
- **Modularity:** The many different modules in the system are neatly defined for ease of use and to make the product as flexible as possible with different permutations and combinations.
- **Robustness:** During the development of the system special care is being taken to make sure that the end results are optimised to the highest level and the results are relevant and validated. Python language is used for the development, itself provides robustness to the system and thus makes it highly unlikely to fail.

'System quality' and 'Non-functional requirements' are interchangeable terms. These qualities mainly consist of two things i.e. evolution and execution. Evolution includes scalability, maintainability and testability whereas execution includes usability and privacy of the system.

User Requirement:

- The user should be able to have a User Interface Window with Visualise Graphics.
- The user should be able to configure with a neat GUI all the parameters.

Resource Requirement:

Anaconda 3-5.0.3: Anaconda is a free and open source distribution of the Python and R programming languages for data science, machine learning and other applications. Anaconda distribution comes with 1400 packages as well as the conda package and virtual environment manager, called Anaconda Navigator. Packages can be made using the conda build command. **Anaconda Navigator**is a desktop graphical user interface that allows user to manage conda packages. The following applications are available by default in navigator: Jupyter lab, Jupyter netbook, Spyder, Orange, Rstudio etc. conda is an open source, cross platform, language-agnostic package manager and environment management system. It instals, runs and updates packages and their dependencies.

1. **Jupyter Notebook**: The code is fully written in Python language using Jupyter notebook. It is the spin-off project from the IPyton project, which used to have an IPython Notebook project itself. IPython kernel, which allows you to write your programs in Python. We can install Jupyter Notebook using command $pip installJupyter. It has several menus that you can use to interact with your notebook they are listed as

   - File
   - Edit
   - View
   - · Insert
   - Cell
   - Kernel, Widgets, Help

The kernel cell is for working with the kernel that is running in the background. Here we can restart the kernel, reconnect to it, shut it down, or even change with kernel your notebook is using.

Hardware Requirements:

The following is the hardware requirements of the system for the proposed system:

- Processor:Any Processor above 500 MHz
- RAM :8 GB
- Hard Disk :1 TB
- Input device: Standard keyboard and mouse

Software Requirements:

The following is the software requirements of the system for the proposed system:

- OS : Windows 10
- Platform : Jupyter Notebook
- Language : Python
- IDE/tool : Anaconda 3-5.0.3

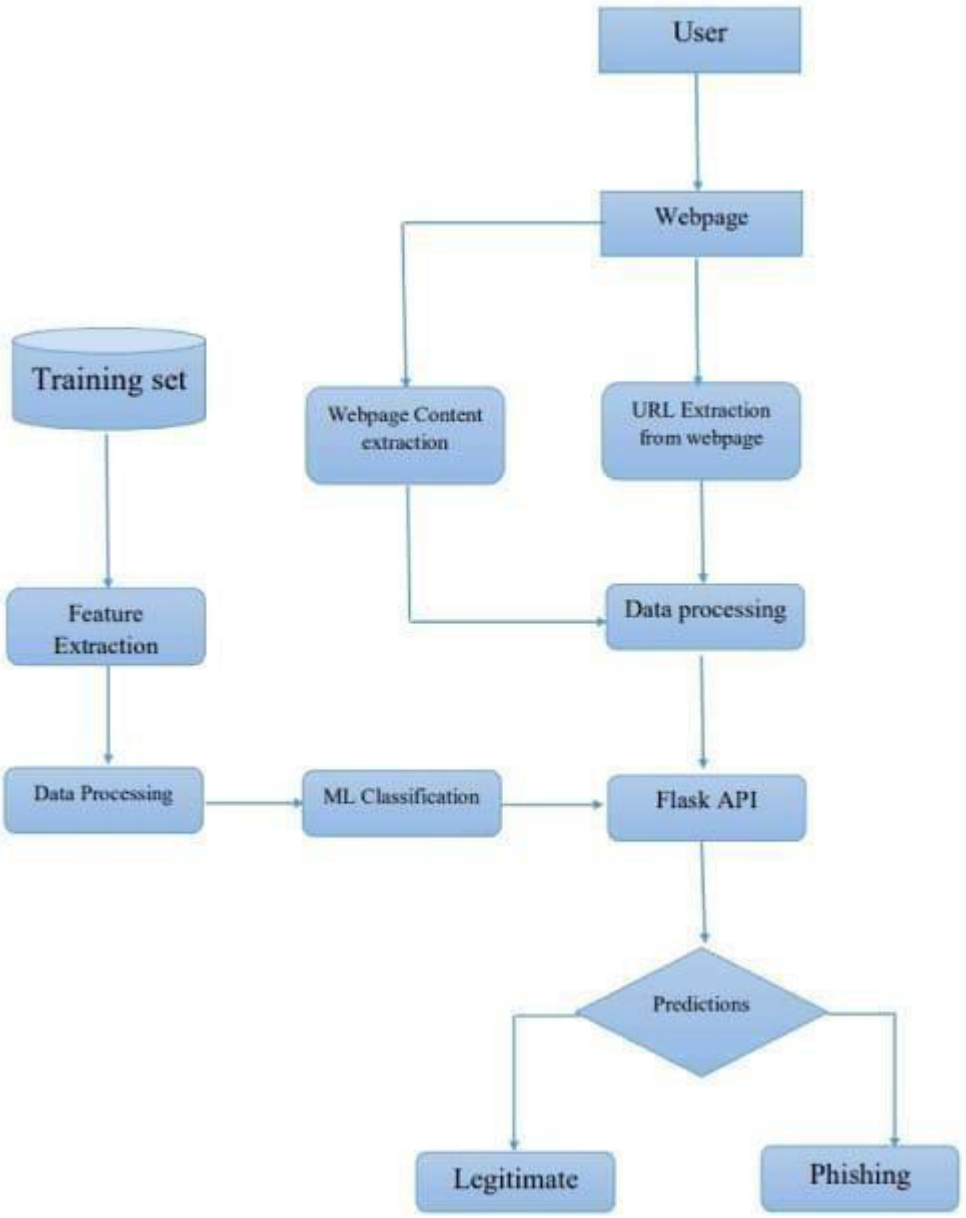
PROJECT DESIGN

4.1 Data Flow Diagrams:

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

Project Design Phase-II
Data Flow Diagram & User Stories

Date	26 October 2022
Team ID	PNT2022TMID33808
Project Name	Project – Web phishing Detection
Maximum Marks	4 Marks

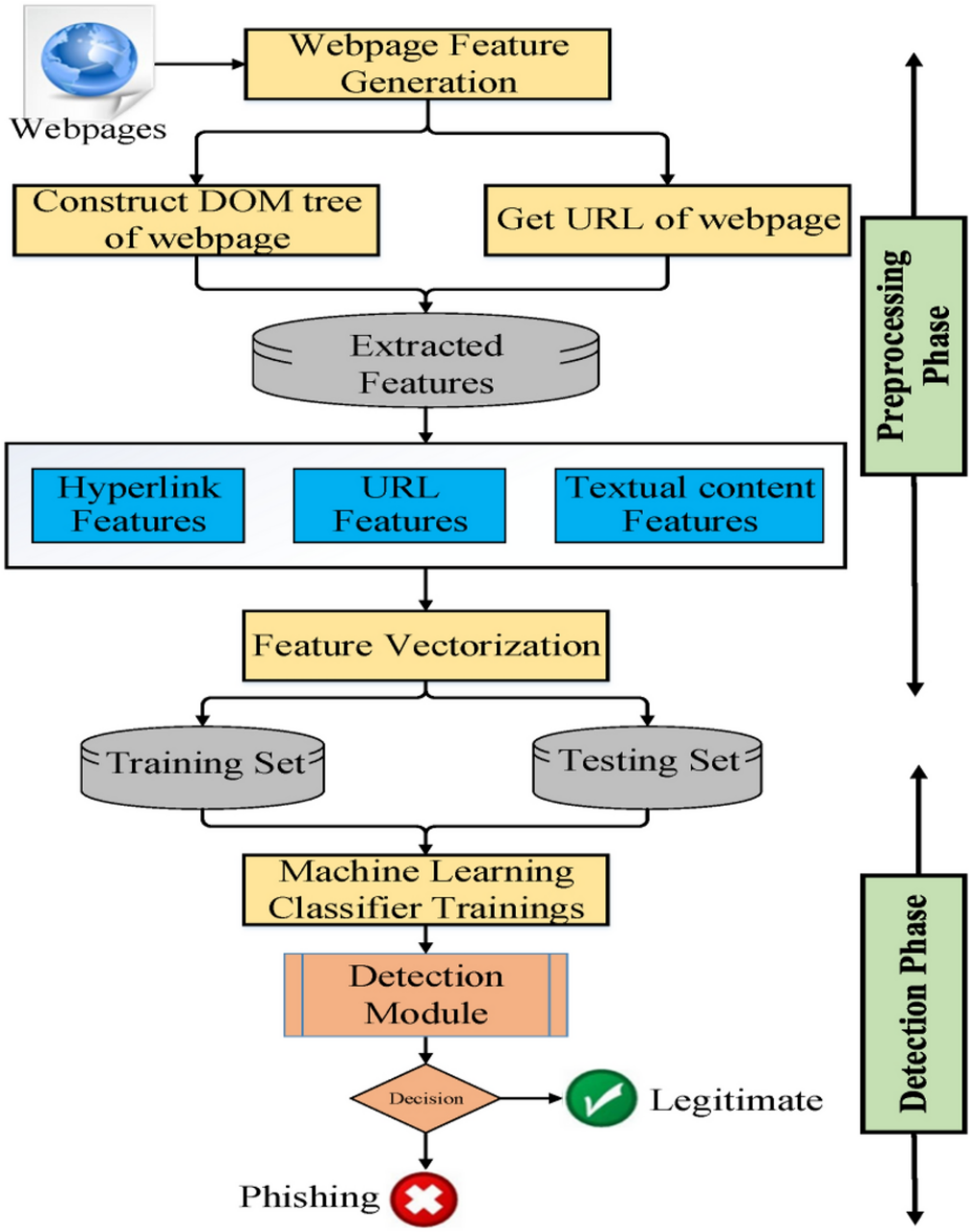

User
Webpage
Training set
Webpage Content extraction
URL Extraction from webpage
Feature Extraction
Data processing
Data Processing
ML Classification
Flask API
Predictions
Legitimate
Phishing

USER STORIES:

	USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
	USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
Dashboard					
User input	USN-1	As a user i can input the particular URL in the required field and waiting for validation.	I can go access the website without any problem	High	Sprint-1
Feature extraction	USN-1	After i compare in case if none found on comparison then we can extract feature using heuristic and visual similarity approach.	As a User i can have comparison between websites for security.	High	Sprint-1
Prediction	USN-1	Here the Model will predict the URL websites using Machine Learning algorithms such as Logistic Regression, KNN	In this i can have correct prediction on the particular algorithms	High	Sprint-1
Classifier	USN-2	Here i will send all the model output to classifier in order to produce final result.	I this i will find the correct classifier for producing the result	Medium	Sprint-2

Solution &Technical Architecture

The overall architecture of the proposed approach is divided into three phases. In the first phase, all the essential features are extracted and HTML source code will be crawled. The second phase applies feature vectorization to generate a particular feature vector for each webpage. The third phase identifies if the given webpage is phishing. Figure shows the system structure of the proposed approach. Details of each phase are described as follows.


Webpages
Webpage Feature Generation
Construct DOM tree of webpage
Get URL of webpage
Extracted Features
Hyperlink Features
URL Features
Textual content Features
Feature Vectorization
Training Set
Testing Set
Machine Learning Classifier Trainings
Detection Module
Decision
Legitimate
Phishing
Preprocessing Phase
Detection Phase

1. PROJECT PLANNING & SCHEDULING

1.1 Sprint Planning & Estimation

Project Planning Phase
Milestone and Activity List

Date	09 November 2022
Team ID	PNT2022TMID33808
Project Name	Web Phishing Detection

TITLE	DESCRIPTION	DATE
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the, technical papers,research publications etc.	16 OCTOBER 2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, Prepare list of problem statements	16 OCTOBER 2022
Ideation	List the by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	16 OCTOBER 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.	03 November2022
Problem Solution Fit	Prepare problem - solution fit document.	03November 2022

Solution Architecture	Prepare solution architecture document.	05November 2022
Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application.	05November2022
Data Flow Diagrams	Draw the data flow diagrams and submit for review.	07November2022
Technology Architecture	architecture diagram.	07November2022
Prepare Milestone & Activity List	Prepare the milestones & activity list of the project.	09 November2022
Project Development - Delivery of Sprint-1, 2, 3 & 4	Develop & submit the developed code by testing it.	IN PROGRESS..

1.1 Sprint Delivery Schedule

Project Tracker : (4 Marks)

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

1.1 Reports from JIRA

CODING & SOLUTIONING

APP.PY

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

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

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


ans = ""
bns = ""
@app.route('/y_predict', 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('index.html',bns=pred)

    else:
        pred="You are on the wrong site. Be cautious!"
        return render_template('index.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)

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

Inputscript.py

```
import ipaddress
import re
import urllib.request
from bs4 import BeautifulSoup
import socket
import requests
from googlesearch import search
import whois
from datetime import date, datetime
from dateutil.parser import parse as date_parse
from urllib.parse import urlparse
import favicon

import regex
from tldextract import extract
import ssl
import socket
from bs4 import BeautifulSoup
import urllib.request

import datetime
import requests

import re



"""
Check if URL contains any IP address. Returns -1 if contains else returns 1
"""
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])\\.([0
1]?\\d\\d?|2[0-4]\\d|25[0-5])\\/)|'  #IPv4

'((0x[0-9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\.(0x[0-9a-fA-F]{1,2})\\/)'
#IPv4 in hexadecimal
                '(?:[a-fA-F0-9]{1,4}:){7}[a-fA-F0-9]{1,4}',url)
   #Ipv6
    if match:
      #print match.group()
      return -1
    else:
      #print 'No matching pattern found'
      return 1
```

```
"""
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\.im|is\.gd|cli\.gs|'
                       'yfrog\.com|migre\.me|ff\.im|tiny\.cc|url4\.eu|twit\.ac|su\.pr|twurl\.nl|snipurl\.com|'
                       'short\.to|BudURL\.com|ping\.fm|post\.ly|Just\.as|bkite\.com|snipr\.com|fic\.kr|loopt\.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\.org|'

'x\.co|prettylinkpro\.com|scrnch\.me|filoops\.info|vzturl\.com|qr\.net|1url\.com|tweez\.me|v\.gd|tr\.im|li
nk\.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\.40|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\.83\.43\.69|52\.69\.166\.231|216\.58\.192\.225|118\.184\.25\.86|67\.208\.74\.71|23\.253\.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\.104\.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|103\.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|216\.38\.62\.18|104\.130\.124\.96|47\.89\.58\.141|78\.46\.211\.158|54\.86\.225\.156|54\.82\.156\.19|37\.157\.192\.102|204\.11\.56\.48|110\.34\.231\.42',ip_address)
    except:
        return -1

    if url_match:
        return -1
    else:
        return 1

#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),SSLfinal_State(url),

Domain_registeration_length(url),Favicon(url),port(url),HTTPS_token(url),Request_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
```

TESTING

8.1 Test Cases

TESTCASES REPORT

Data	15-Nov-22
Team ID	PNT2022TMID33808
Project Name	Project - Web Phishing Detection
Maximum Marks	4marks

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
LoginPage_TC_OO1	Functional	Home Page	Verify user is able to see the Landing Page when user can type the URL in the box		1.Enter URL and click go 2.Type the URL 3.Verify whether it is processing or not.	https://phishing-shield.herokuapp.com/	Should Display the Webpage	Working as expected	Pass		N		S Balaji
LoginPage_TC_OO2	UI	Home Page	Verify the UI elements is Responsive		1. Enter URL and click go 2. Type or copy paste the URL 3. Check whether the button is responsive or not 4. Reload and Test Simultaneously	https://phishing-shield.herokuapp.com/	Should Wait for Response and then gets Acknowledge	Working as expected	Pass		N		R Abishek
LoginPage_TC_OO3	Functional	Home page	Verify whether the link is legitimate or not		1. Enter URL and click go 2. Type or copy paste the URL 3. Check the website is legitimate or not 4. Observe the results	https://phishing-shield.herokuapp.com/	User should observe whether the website is legitimate or not.	Working as expected	Pass		N		TS Aswin
LoginPage_TC_OO4	Functional	Home Page	Verify user is able to access the legitimate website or not		1. Enter URL and click go 2. Type or copy paste the URL 3. Check the website is legitimate or not 4. Continue if the website is legitimate or be cautious if it is not legitimate.	https://phishing-shield.herokuapp.com/	Application should show that Safe Webpage or Unsafe.	Working as expected	Pass		N		Balajee A V
LoginPage_TC_OO5	Functional	Home Page	Testing the website with multiple URLs		1. Enter URL (https://phishing-shield.herokuapp.com/) and click go 2. Type or copy paste the URL to test 3. Check the website is legitimate or not 4. Continue if the website is secure or be cautious if it is not secure	1. https://avbalajee.github.io/welcome 2. totalpad.com 3. https://www.kince.edu 4. salescript.info 5. https://www.google.com/ 6. delgets.com	User can able to identify the websites whether it is secure or not	Working as expected	Pass		N		Balajee A V

8.2 User Acceptance Testing

Acceptance Testing
UAT Execution & Report Submission

Date	15 November 2022
Team ID	PNT2022TMID33808
Project Name	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	10	2	4	20	36
Not Reproduced	0	0	1	0	1
Skipped	0	0	0	0	0
Won't Fix	0	0	2	1	3
Totals	23	9	12	25	60

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	10	0	0	10
Client Application	50	0	0	50
Security	5	0	0	4
Outsource Shipping	3	0	0	3

RESULTS

ADVANTAGES

Blacklists -

- Requiring low resources on host machine
- Effective when minimal FP rates are required.

Heuristics and visual similarity

- Mitigate zerohour attacks.

Machine Learning -

- Mitigate zerohour attacks.
- Construct own classification models.

DISADVANTAGE

- Mitigation of zero-hour phishing attacks.
- Can result in excessive queries with heavily loaded servers
- Higher FP rate than blacklists
- . High computational cost.
- Time consuming. -Costly.
- Huge number of rules.

CONCLUSION

It is outstanding that a decent enemy of the phishing apparatus ought to anticipate the phishing assaults in a decent timescale. We accept that the accessibility of a decent enemy of phishing devices at a decent time scale is additionally imperative to build the extent of anticipating phishing sites. This apparatus ought to be improved continually through consistent retraining. As a matter of fact, the accessibility of crisp and cutting-edge preparing dataset which may be gained utilising our very own device [30, 32] will help us to retrain our model consistently and handle any adjustments in the highlights, which are influential in deciding the site class. Albeit neural system demonstrates its capacity to tackle a wide assortment of classification issues, the procedure of finding the ideal structure is very difficult, and much of the time, this structure is controlled by experimentation.Our model takes care of this issue via computerising the way toward organising a neural system conspire; hence, on the off chance that we construct an enemy of phishing model and for any reasons we have to refresh it, at that point our model will encourage this procedure, that is, since our model will mechanise the organising procedure and will request scarcely any client defined parameters.

REFERENCES

Liu J, Ye Y (2001) Introduction to E-business operators: commercial center arrangements, security issues, and market interest. In: E-business specialists, commercial centerarrangements, security issues, and market interest, London, UK • APWG, Aaron G, Manning R (2013) APWG phishing reports. APWG, 1 February 2013. [Online]. Accessible: http://www. antiphishing.org/assets/apwg-reports/. Gotten to 8 Feb2013 • Kaspersky Lab (2013) Spam in January 2012: love, governmental issues and game. [Online]. Available:http://www.kaspersky.com/about/news/spam/2012
Spam_in_January_2012_Love_Politics_and_ Sport. Gotten to 11 Feb 2013 • Seogod (2011) Black Hat SEO. Search engine optimization Tools. [Online]. Accessible:http://www.seobesttools.com/dark cap website optimization/. Gotten to 8 Jan 2013 • Dhamija R, Tygar JD, Hearst M (2006) Why phishing works. In: Proceedings of the SIGCHImeeting on human factors in figuring frameworks, Cosmopolitan Montre 'al, Canada • Cranor LF (2008) A system for thinking about the human tuned in. In: UPSEC'08 Proceedings of the first meeting on ease of use, brain science, and security, Berkeley, CA,USA • Miyamoto D, Hazeyama H, Kadobayashi Y (2008) An assessment of AI based techniquesfor recognition of phishing destinations. Aust J Intell Inf Process Syst 10(2):54–6 • Xiang G, Hong J, Rose CP, Cranor L (2011) CANTINA?:a include rich AI structure foridentifying phishing sites. ACM Trans Inf Syst Secur 14(2):1–28

APPENDIX

GitHub & Project Demo Link:

IBM-EPBL/IBM-Project-36555-1660296011

https://github.com/IBM-EPBL/IBM-Project-36555-1660296011