IBM Project Report

Project Title	Web phishing Detection
Team ID	PNT2022TMID16590
Team Members	Sripathi C Venkatesan B Sanjay S Sanjay K

1.INTRODUCTION

1.1 PROJECT OVERVIEW

This project describes the machine learning based Over the last decade, many cyber-attacks start with a poisoned link in a seemingly harmless email. When you click on the link, it could be a phishing or malicious site. Phishing websites try to hook Internet surfers into revealing their sensitive information including credentials, bank account, and other personal information and malicious sites try to install malware onto your devices. These new, short-lived phishing URLs can easily bypass signature-based detectors. To combat this problem, researchers have also used machine learning methods to detect phishing websites. Nevertheless, there is still no definitive solution with machine learning or another approach.

The main objective of the web phising process consists of

1. To create a dataset and apply necessary preprocessing followed by feature selection.
2. To apply various machine learning models and compare them based on different metrics.
3. To run all the algorithms in the selected cloud platform using IBM Cloud's AutoAI feature to validate the choose obtained previously.
4. To implement and deploy the selected machine learning model onto a cloud-based platform.
5. To predict the probability of a website being legitimate or phishing based on the URL of the website.

1.2 PURPOSE

The main purpose of the web phising project Phishing is a form of fraud in which an attacker masquerades as a reputable entity or person in email or other forms of communication. Attackers will commonly use phishing emails to distribute malicious links or attachments that can perform a variety of functions. Some will extract login credentials or account information from victims.

2.LITERATURE SURVEY

Construction of Phishing Site. In the first step attacker identifies the target as a well-known organization. Afterward, attacker collects the detailed information about the organization by visiting their website. The attacker then uses this information to construct the fake website URL Sending. In this step, attacker composes a bogus e-mail and sends it to the thousands of users. Attacker attached the URL of the fake website in the bogus e-mail. In the case of spear phishing attack, an attacker sends the e-mail to selected users. An attacker can also spread the link of phishing website with the help of blogs, forum, and so forth Stealing of the Credentials. When user clicks on attached URL, consequently, fake site is opened in the web browser. The fake website contains a fake login form which is used to take the credential of an innocent user. Furthermore, attacker can access the information filled by the user Identity Theft. Attacker uses this credential of malicious purposes. For example, attacker purchases something by using credit card details of the user. Although attacks use different techniques to create phishing websites to deceive users, most have similarly designed phishing website features. Therefore, researchers have conducted extensive anti-phishing research using phishing website features. Current methods for phishing detection include black and whitelists, heuristics, visual similarity, and machine learning, among which heuristics and machine learning are more widely used. The following is an introduction to the aforementioned phishing detection techniques Black and whitelist To prevent phishing attack threats, many anti-phishing methods have been proposed. Blacklisting methods are the most straightforward ways to prevent phishing attacks and are widely used in the industry. Google Safe Browsing uses a

blacklist-based phishing detection method to check if the URL of the matching website exists in the blacklist. If it does, it is considered a phishing website.

2.1 EXISTING WORK PROBLEM

Many researchers have been working on phishing website detection for more than a decade now. Phishing site detection can be achieved from many perspectives using different sets of features, i.e., search-based, URL-based, content-based, or hybrid.

An influential search-based framework, CANTINA [48], uses TF-IDF scores of each term on the web page, then generates a lexical signature by taking the five terms with highest TF-IDF weights to feed into a search engine (Google). Detection is based on whether the domain of the current web page matches one of the domains in the top 30 search results

In the real world, there are many types of legitimate and phishing websites. Many new legitimate websites exist, which use very generic terms in their website content, e.g., nonprofit websites do this frequently, and have no logos in the web content. Such domains may not be easy to find, if the corresponding websites are not popular. Therefore, such methods tend to have a relatively high false positive rate, and must be complemented with other 9 features. Although our search-based features are inspired by [3, 48], they are novel, since we look for domain emails and subdomains rather than keywords from the content.

A typical content-based phishing detector. The system will get HTML source code and URL of input webpage first. URL features normally just check

internal and external links from HTML source code based on domain name. In HTML source code, there normally are four types of features that will be investigated and extracted, namely login forms, hyperlinks, CSS and JavaScript, and web identity features.

2.2 PROBLEM STATEMENT DEFINITION

The problem of web phising system Phishing is a major problem, which uses both social engineering and technical deception to get users' important information such as financial data, emails, and other private information. Phishing exploits human vulnerabilities; therefore, most protection protocols cannot prevent the whole phishing attacks.Phishing is a major threat to all Internet users and is difficult to trace or defend against since it does not present itself as obviously malicious in nature. In today's society, everything is put online and the safety of personal credentials is at risk. Phishing can be seen as one of the oldest and easiest ways of stealing information from people and it is used for obtaining a wide range of personal details. It also has a fairly simple approach – send an email, email sends victim to a site, site steals information.

2.3 REFERENCES

1. J. Alamelu Mangai, V. Santhosh Kumar, and S. Appavu alias Balamurugan. A novel feature selection framework for automatic web page classification. International Journal of Automation and Computing, 9(4):442–448, Aug 2012

2. Ankesh Anand, Kshitij Gorde, Joel Ruben Antony Moniz, Noseong Park, Tanmoy Chakraborty, and Bei-Tseng Chu. Phishing URL detection with oversampling based on text generative adversarial networks. In 2018 IEEE

International Conference on Big Data (Big Data), pages 1168–1177, Dec 2018

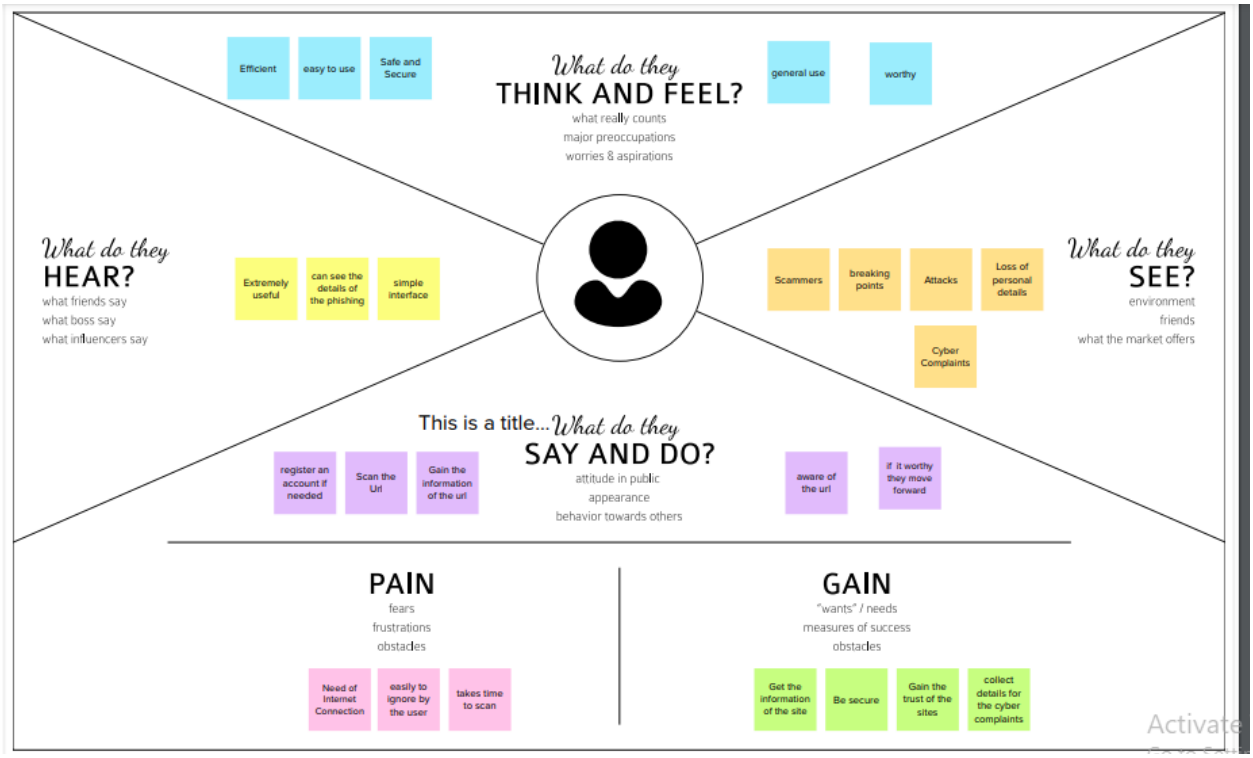
3. Choon Lin Tan andKang Leng Chiew and San Nah Sze. Phishing website detection using URL-assisted brand name weighting system. In 2014 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), pages 054–059, Dec 2014.

4. Mehdi Babagoli, Mohammad Pourmahmood, Aghababa, and Vahid Solouk. Heuristic nonlinear regression strategy for detecting phishing websites. In Soft Computing, pages 4315–4327. Springer Berlin Heidelberg, June 2019.

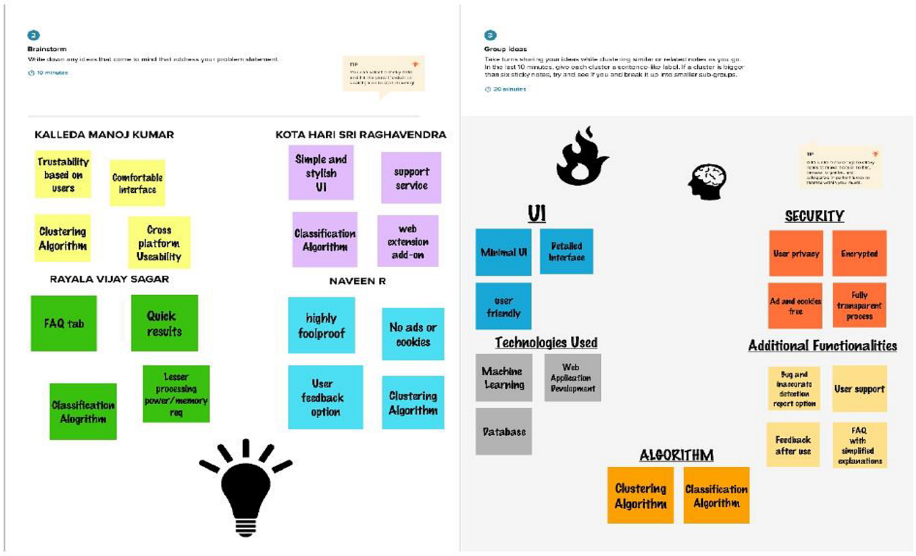
5. Alejandro Correa Bahnsen, Eduardo Contreras Bohorquez, Sergio Villegas, Javier Vargas, and Fabio A. Gonz´alez. Classifying phishing URLs using recurrent neural networks. In 2017 APWG Symposium on Electronic Crime Research (eCrime). IEEE, 2017.

3.IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION & BRAINSTORMING



3.3 PROPOSED SOLUTION

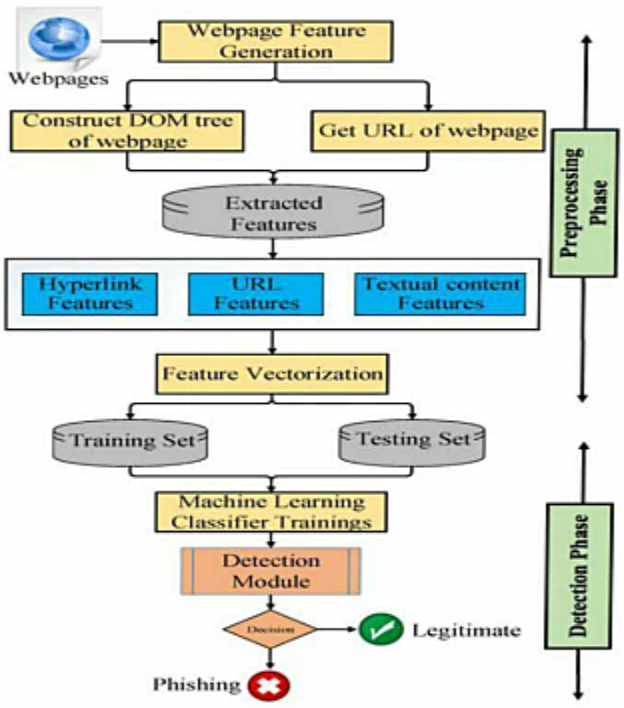
4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

Solution Architecture:

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

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



4.2 FUNCTIONAL REQUIREMENT

A characteristic of a quality SRS is that in addition to describing the functional requirements of a system, It will also provide detailed coverage of the non-functional requirements. In practice, this would entail detailed analysis of issues such as availability, security, usability and maintainability. However, as this document is only an outline specification, it does not contain the same degree of rig our that would normally be expected in a formal SRS. Therefore, the sections below should be seen as indicative rather than providing specific (l.e. testable) requirements.

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

Use the below template to create product backlog and sprint schedule

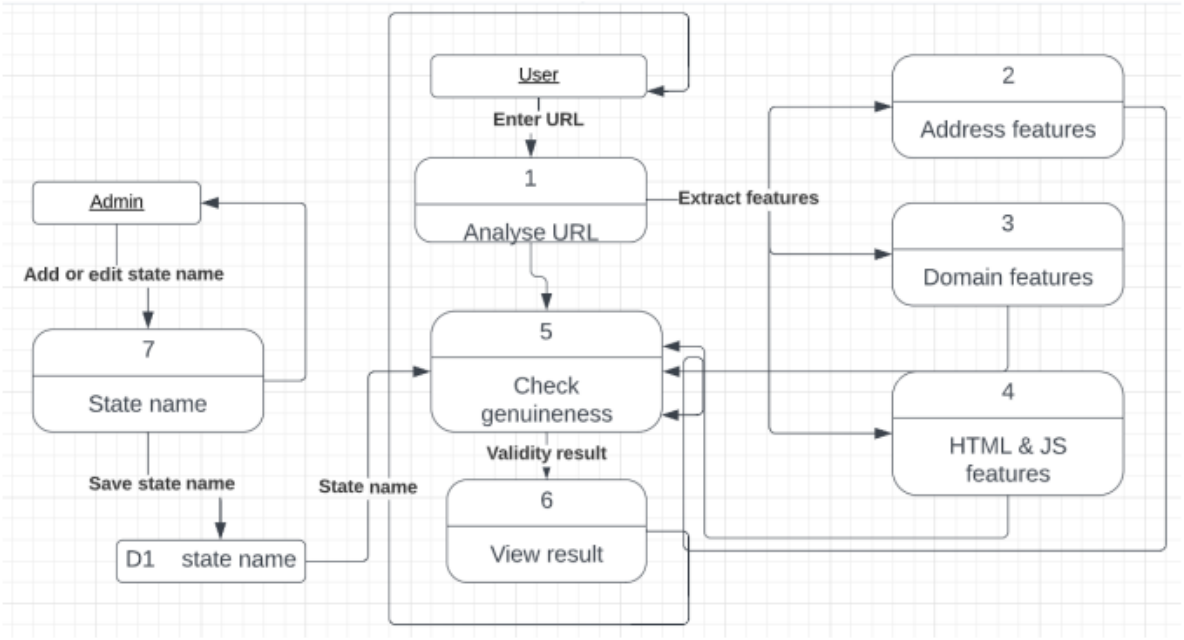
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Login	USN-1	As a user, I can navigate into the website.	1	High	Amala
Sprint-1	Dashboard	USN-2	As a user, I will input any site's URL in the form to check its genuineness.	1	High	Annie
Sprint-1		USN-3	As a user, I can see the output.	2	High	Akshaya
Sprint-2	Backend	USN-4	As an admin, if a new URL is found, I can add the new state into the database.	3	Medium	Shekinah
Sprint-3	Report	USN-5	As a user, I can ask my queries and report suspicious sites in the report box.	1	Low	Akshaya
Sprint-4		USN-6	As an admin, I can take actions to the queries asked by the user.	2	Low	Shekinah

5. PROJECT DESIGN

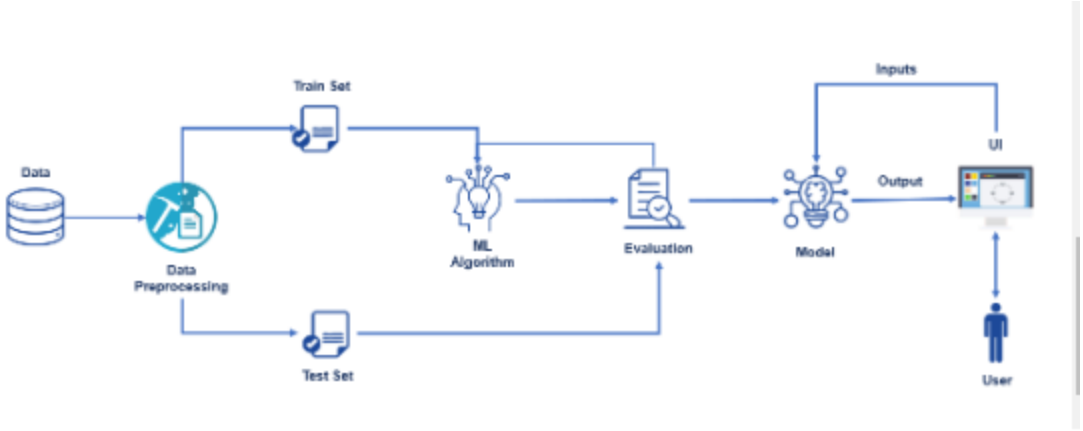
5.1 Data Flow Diagrams

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.

DFD level 0



5.2 Solution & Technical Architecture



5.3 User Stories

User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Login	USN-1	As a user, I can navigate into the website	I can access the page	High	Sprint-1
	Dashboard	USN-2	As a user, I will paste the URL that needs to be checked if it's a phishing website or not	I can paste the URL in the text box	High	Sprint-1
		USN-3	As a user, I can see the output	I can see if it's a safe site	High	Sprint-1
Administrator		USN-4	If a new URL is found, I can add the new state into the database	I can add the new URL	Medium	Sprint-2

6.PROJECT PLANNING & SCHEDULING

Milestones	Activities
Project development phase	Delivery of sprint- 1,2,3,4
Create and configure and IBM cloud services	Create IBM Watson
Create and access deep learning	Create v1 to interact with app deploy
	Create IBM and connect with python
Create & database in cloud and DB	Launch the cloudant DB and Create database
Develop the python flask	Install the python software
	Develop python code
Create the web application	Develop the web application

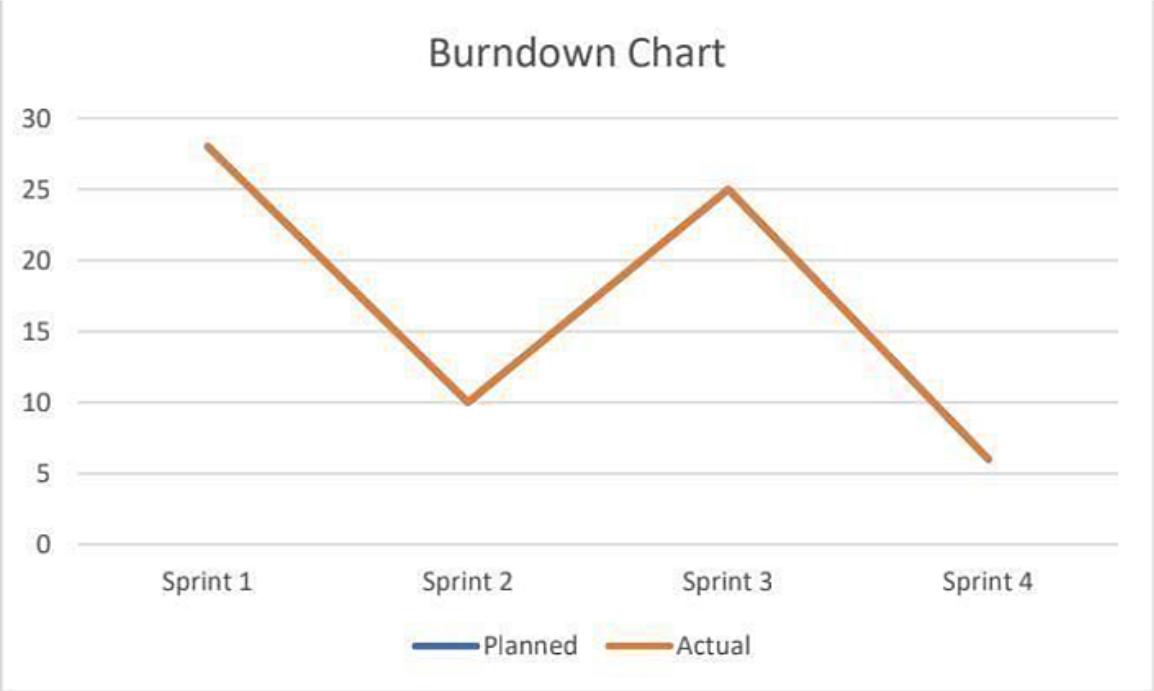
6.1 Sprint Planning & Estimation

Sprint	Functional requirement	User story/task	Story priority points	Team member
Sprint 1	Registration	User enter the details can register with details	20 high	**S**ripathi.C Sanjay.K Sanjay.S Venkatesan.B
Sprint 2	Training of dataset	We can collect the dataset train the model using data	20 high	**S**ripathi.C Sanjay.K Sanjay.S Venakatesan.B
Sprint 3	Prediction	Based on the model we can build the model	20 high	Sripathi.C Sanjay.K Sanjay.S Venkatesan.B

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	25 Oct 2022	31 Oct 2022	20	31 Oct 2022
Sprint-2	20	6 Days	01 Nov 2022	06 Nov 2022	18	06 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	13 Nov 2022	20	13 Nov 2022
Sprint-4	20	6 Days	13 Nov 2022	19 Nov 2022	19	19 Nov 2022

6.3 Reports from JIRA



7. CODING & SOLUTIONING

7.1 Feature 1

In feature one,we implement a design for getting URL for web phishing detection.We gives the input URL for Detection in feature one we can explore the data we collected dataset for train the model we designed HTML page design for recomment the Web Phishing Detection.

Layout.html :

```
<!DOCTYPE html>
<html>
<head>
  <title>phishcoop</title>
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootst
rap.min.css">
  <link rel="stylesheet" href="{{
url_for('static',filename='cover.css') }}">
</head>
<body>
  <div class="site-wrapper">
    <div class="site-wrapper-inner">
      <div class="cover-container">
        <div class="masthead clearfix">
          <div class="inner">
            <h3 class="masthead-brand">phishcoop</h3>
            <nav>
              <ul class="nav masthead-nav">
                <li><a href="/">Home</a></li>
                <li><a href="/about">About</a></li>
              </ul>
            </nav>
```

```
          </div>
        </div>
        {% block body %} {% endblock %}
        <br>
        {% if label %} {{ label }} {% endif %}
        <div class="mastfoot">
          <div class="inner">
            <p>detect phishing websites, by Sripathi.</p>
          </div>
        </div>
      </div>
    </div>
  </div>
  </script>
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.
min.js"></script>
  <script>window.jQuery || document.write('<script
src="https://code.jquery.com/jquery-
1.12.4.min.js"><\/script>')</script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstra
p.min.js"></script>
  <script language="javascript" type="text/javascript">
    $(window).load(function () {
      $('#loading').hide();
    });
    $(".btn").click(function () {
      $("#loading").show();
    });
  </script>
</body>
```

```
</html>
```

7.2 Feature 2

In feature 2 after appling the URL in the input box, the data transfer into the dataset and check the URL have any vulnerability in the website.If any Vulnerability in the website URL the output show as "Website is not legitimate",It means "NOT ACCEPTABLE" , this can be check by api.py file which is used as backend file for checking the URL.

api.py

```
import flask
from flask import Flask, render_template, request
import joblib
import inputScript
import regex
import sys
import logging
app = Flask(__name__)
app.logger.addHandler(logging.StreamHandler(sys.stdout))
app.logger.setLevel(logging.ERROR)
@app.route('/')
@app.route('/index')
def index():
    return flask.render_template('home.html')
@app.route('/about')
def about():
    return flask.render_template('about.html')
```

```
@app.route('/predict', methods = ['POST'])
def make_prediction():
    classifier = joblib.load('rf_final.pkl')
    if request.method=='POST':
        url = request.form['url']
        if not url:
            return render_template('home.html', label = 'Please
input url')
        elif(not(regex.search(r'^(http|ftp)s?://', url))):
            return render_template('home.html', label = 'Please
input full url, for exp- https://facebook.com')
        checkprediction = inputScript.main(url)
        prediction = classifier.predict(checkprediction)
        if prediction[0]==1 :
            label = 'website is not legitimate'
        elif prediction[0]==-1:
            label ='website is legitimate'
        return render_template('home.html', label=label)
if __name__ == '__main__':
    classifier = joblib.load('rf_final.pkl')
    app.run()
```

8 TEST CASE

8. SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type

addresses a specific testing requirement.

8.1 TYPES OF TESTS

8.1.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform Basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

8.1.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.
Systems/Procedures: interfacing systems or procedures must be invoked.
Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

8.1.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

8.1.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

8.1.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

8.2 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

8.2.1 Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

8.2.2 Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

8.2.3 Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

8.3 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error. Test Results: All the test cases mentioned above passed successfully. No defects encountered.

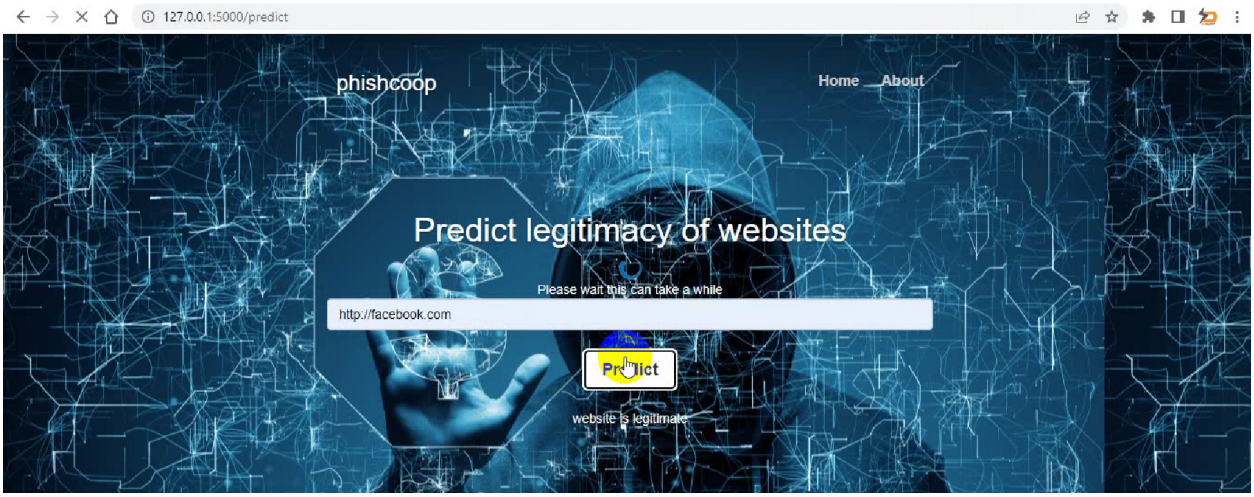
8.4 Acceptance Testing

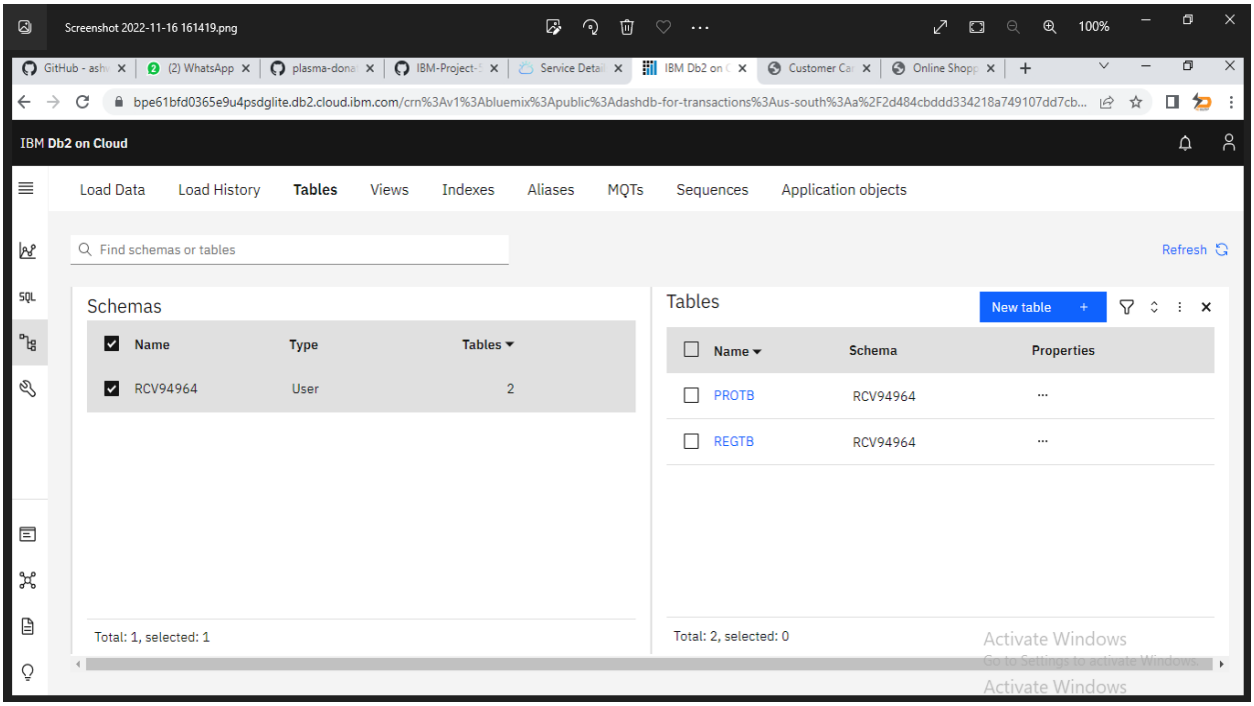
User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

9.RESULT

Prediction result:





10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

- High Accuracy
- High Prediction Rate
- We can prevent the attacks
- High reliability

DISADVANTAGES

- High redutancy
- We can store less amount of data
- Reduced reliability

11. CONCLUSION

The phishing detection process using our model from the user prospective can be explained in the following steps: The end-user clicks on a link within an email or browses the internet.He will be directed to a website that could be legitimate or phishy. This website is basically the test data. A script written in PHP that is embedded within the browser starts processing to extract the features of the test data (current website) and saves them in a data structure. Now, the intelligent model will be active within the browser to guess the type of the website based on

rules learnt from historical websites (previous data collected). The rules of the classifier are utilised to predict the type of the test data based on features similarity. When the browsed website is identified as legitimate no action will be taken. On the other hand, when the website turned to be phishy, the user will be warned by the intelligent method that he is under risk.

12. FUTURE SCOPE

- In future work the web phising system is implemented with new algorithm High detection efficiency: To provide high detection efficiency, incorrect classification of benign sites as phishing (false-positive) should be minimal and correct classification of phishing sites (true-positive) should be high.

- Real-time detection: The prediction of the phishing detection approach must be provided before exposing the user's personal information on the phishing website.

- Target independent: Due to the features extracted from both URL and HTML the proposed approach can detect new phishing websites targeting any benign website (zero-day attack).

- Third-party independent: The feature set defined in our work are lightweight and client-side adaptable, which do not rely on third-party services such as blacklist/whitelist, Domain Name System (DNS) records, WHOIS record (domain age), search engine indexing, network traffic measures, etc. Though third-party services may raise the effectiveness of the detection approach, they might misclassify benign websites if a benign website is newly

registered. Furthermore, the DNS database and domain age record may be poisoned and lead to false negative results (phishing to benign).

Hence, a light-weight technique is needed for phishing websites detection adaptable at client side. The major contributions in this paper are itemized as follows.

- We propose a phishing detection approach, which extracts efficient features from the URL and HTML of the given webpage without relying on third-party services. Thus, it can be adaptable at the client side and specify better privacy.

- We proposed eight novel features including URL character sequence features (F1), textual content character level (F2), various hyperlink features (F3, F4, F5, F6, F7, and F14) along with seven existing features adopted from the literature.

- We conducted extensive experiments using various machine learning algorithms to measure the efficiency of the proposed features. Evaluation results manifest that the proposed approach precisely identifies the legitimate websites as it has a high true negative rate and very less false positive rate.

- We release a real phishing webpage detection dataset to be used by other researchers on this topic.

13. APPENDIX

SOURCE CODE:

Layout.html:

```
<!DOCTYPE html>
<html>
<head>
  <title>phishcoop</title>
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootst
rap.min.css">
  <link rel="stylesheet" href="{{
url_for('static',filename='cover.css') }}">
</head>
<body>
  <div class="site-wrapper">
    <div class="site-wrapper-inner">
      <div class="cover-container">
        <div class="masthead clearfix">
          <div class="inner">
            <h3 class="masthead-brand">phishcoop</h3>
            <nav>
              <ul class="nav masthead-nav">
                <li><a href="/">Home</a></li>
                <li><a href="/about">About</a></li>
              </ul>
            </nav>
          </div>
        </div>
        {% block body %} {% endblock %}
```

```
        <br>
        {% if label %} {{ label }} {% endif %}
        <div class="mastfoot">
          <div class="inner">
            <p>detect phishing websites, by Sripathi.</p>
          </div>
        </div>
      </div>
    </div>
  </div>
  </script>
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.
min.js"></script>
  <script>window.jQuery || document.write('<script
src="https://code.jquery.com/jquery-
1.12.4.min.js"><\/script>')</script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstra
p.min.js"></script>
  <script language="javascript" type="text/javascript">
    $(window).load(function () {
      $('#loading').hide();
    });
    $(".btn").click(function () {
      $("#loading").show();
    });
  </script>
</body>
</html>
```

Home.html:

```
{% extends 'layout.html' %}
{% block body %}
    <div class="inner cover">
        <h1 class="cover-heading" >Predict legitimacy of websites</h1>
        <div id="loading" style="height:25px">
        <img   src="/static/ajax-loader.gif" height="25" width="25"
alt="Loading..." /><p>Please wait this can take a while</p>
        </div> <br>
        <form action ='/predict' method='post'>
           <input class="form-control" type = 'text' name = 'url'>
            <br>
           <input class="btn btn-lg btn-default" type="submit"
value="Predict">
        </form>
    </div>
{% endblock %}
```

About.html:

```
{% extends 'layout.html' %}
{% block body %}
    <div class="inner cover">
        <p class="lead">
            Detection of phishing websites is a really important
safety measure for most of the online platforms. So, as to save a
platform with malicious requests from such websites, it is important
to have a robust phishing detection system in place.
        </p>
        <p class="lead">
            PhishCoop is an intelligent, flexible and effective system
that uses UCI datasets and supervised machine learning algorithms to
classify legitimacy of websites. The main objective of this system is
to distinguish the phishing websites from the legitimate websites and
```

```
ensure secure transactions to users.
        </p>
    </div>
{% endblock %}
```

api.py

```
import flask
from flask import Flask, render_template, request
import joblib
import inputScript
import regex
import sys
import logging
app = Flask(__name__)
app.logger.addHandler(logging.StreamHandler(sys.stdout))
app.logger.setLevel(logging.ERROR)
@app.route('/')
@app.route('/index')
def index():
    return flask.render_template('home.html')
@app.route('/about')
def about():
    return flask.render_template('about.html')
@app.route('/predict', methods = ['POST'])
def make_prediction():
    classifier = joblib.load('rf_final.pkl')
    if request.method=='POST':
        url = request.form['url']
        if not url:
            return render_template('home.html', label = 'Please
input url')
        elif(not(regex.search(r'^(http|ftp)s?://', url))):
```

```
            return render_template('home.html', label = 'Please
input full url, for exp- https://facebook.com')
        checkprediction = inputScript.main(url)
        prediction = classifier.predict(checkprediction)
        if prediction[0]==1 :
            label = 'website is not legitimate'
        elif prediction[0]==-1:
            label ='website is legitimate'
        return render_template('home.html', label=label)
if __name__ == '__main__':
    classifier = joblib.load('rf_final.pkl')
    app.run()
```

GITHUB & PROJECT DEMO LINK:

GitHub Link : https://github.com/IBM-EPBL/IBM-Project-17108-1659628418

Demo Link :
https://drive.google.com/file/d/1ZMd4SsbG8Y4bgc0D4qldPBLehgOX6YLj/view?usp=share_link