WEB PHISHING DETECTION

A PROJECT REPORT

Submitted by

BOOMIKA R

BOWTHARA G

MANJU S

VINOTHINI S

Of

COMPUTER SCIENCE AND ENGINEERING

SSM COLLEGE OF ENGINEERING

KOMARAPALAYAM

Project Report Format

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**
Sprint Planning & Estimation
6.1 Sprint Delivery Schedule
6.2 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 Link

INTRODUCTION

Project Overview:

WebPhish is a website which is used to detect phishing sites to improve the customer’s sense of safety whenever he/she attempts to provide any sensitive information to a site. Also, by which people won’t access them which will reduce the revenue of malicious site owners. This application can be accessed online without paying instead, can be accessed via any browser of the customer’s choice to detect any site with high accuracy. This system uses machine learning algorithm which implements classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy.

The design and implementation of a comprehensive web phishing detection system instils a cyber security culture which prevents the need for the deployment of targeted anti-phishing solutions in a corporate to meet industry’s compliance obligations.

Purpose:

Web phishing is a threat in various aspects of security on the internet, which might involve scams and private information disclosure. Some of the common threats of web phishing are:

- Attempt to fraudulently solicit personal information from an individual or organization.
- Attempt to deliver malicious software by posing as a trustworthy organization or entity.
- Installing those malwares infects the data that cause a data breach or even nature’s forces that takes down your company’s data headquarters, disrupting access.

For this purpose, the objective of our project involves building an efficient and intelligent system to detect such websites by applying a machine-learning algorithm which implements classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy and as a result of which whenever a user makes a transaction online and 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.

LITERETURE SURVEY

1.Web phishing detection using a deep learning framework

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. 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.

2.Spam detection using machine learning techniques

Online reviews are often the primary factor in a customer's decision to purchase a product or service, and are a valuable source of information that can be used to determine public opinion on these products or services. Because of their impact, manufacturers and retailers are

highly concerned with customer feedback and reviews. Reliance on online reviews gives rise to the potential concern that wrongdoers may create false reviews to artificially promote or devalue products and services. This practice is known as Opinion (Review) Spam, where spammers manipulate and poison reviews (i.e., making fake, untruthful, or deceptive reviews) for profit or gain. Since not all online reviews are truthful and trustworthy, it is important to develop techniques for detecting review spam. By extracting meaningful features from the text using Natural Language Processing (NLP), it is possible to conduct review spam detection using various machine learning techniques. Additionally, reviewer information, apart from the text itself, can be used to aid in this process. In this paper, we survey the prominent machine learning techniques that have been proposed to solve the problem of review spam detection and the performance of different approaches for classification and detection of review spam. The majority of current research has focused on supervised learning methods, which require labeled data, a scarcity when it comes to online review spam. Research on methods for Big Data are of interest, since there are millions of online reviews, with many more being generated daily. To date, we have not found any papers that study the effects of Big Data analytics for review spam detection. The primary goal of this paper is to provide a strong and comprehensive comparative study of current research on detecting review spam using various machine learning techniques and to devise methodology for conducting further investigation.

3. A machine learning based approach for phishing detection using hyperlinks information

This paper presents a novel approach that can detect phishing attack by analysing the hyperlinks found in the HTML source code of the website. The proposed approach incorporates various new outstanding hyperlink

specific features to detect phishing attack. 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 data set. The proposed approach is an entirely client-side solution, and does not require any services from the third party. Moreover, the proposed approach is language independent and it can detect the website written in any textual language. Compared to other methods, the proposed approach has relatively high accuracy in detection of phishing websites as it achieved more than 98.4% accuracy on logistic regression classifier

4.Deep Learning for Phishing Detection: Taxonomy, Current Challenges and Future Directions

Phishing has become an increasing concern and captured the attention of end-users as well as security experts. Despite decades of development and improvement, existing phishing detection techniques still suffer from the deficiency in performance accuracy and the inability to detect unknown attacks. Motivated to solve these problems, many researchers in the cybersecurity domain have shifted their attention to phishing detection that capitalizes on machine learning techniques. In recent years, deep learning has emerged as a branch of machine learning that has become a promising solution for phishing detection. As a result, this study proposes a taxonomy of deep learning algorithms for phishing detection by examining 81 selected papers using a systematic literature review approach. The paper first introduces the concept of phishing and deep learning in the context of cybersecurity. Then, phishing detection and deep learning algorithm taxonomies are provided to classify the existing literature into various categories. Next, taking the proposed taxonomy as a baseline, this study comprehensively reviews the state-of-the-art deep learning techniques and analyzes their advantages as well as disadvantages. Subsequently, the paper discusses various issues deep learning faces in phishing detection and proposes future research directions to overcome these challenges. Finally, an empirical analysis is conducted to evaluate the performance of various deep learning techniques in a practical context and highlight the related issues that motivate researchers in their future works. The results obtained from the empirical experiment showed that the common issues among most of the state-of-the- art deep learning algorithms are manual parameter-tuning, long training time, and deficient detection accuracy.

5.Machine learning based phishing detection from URLs

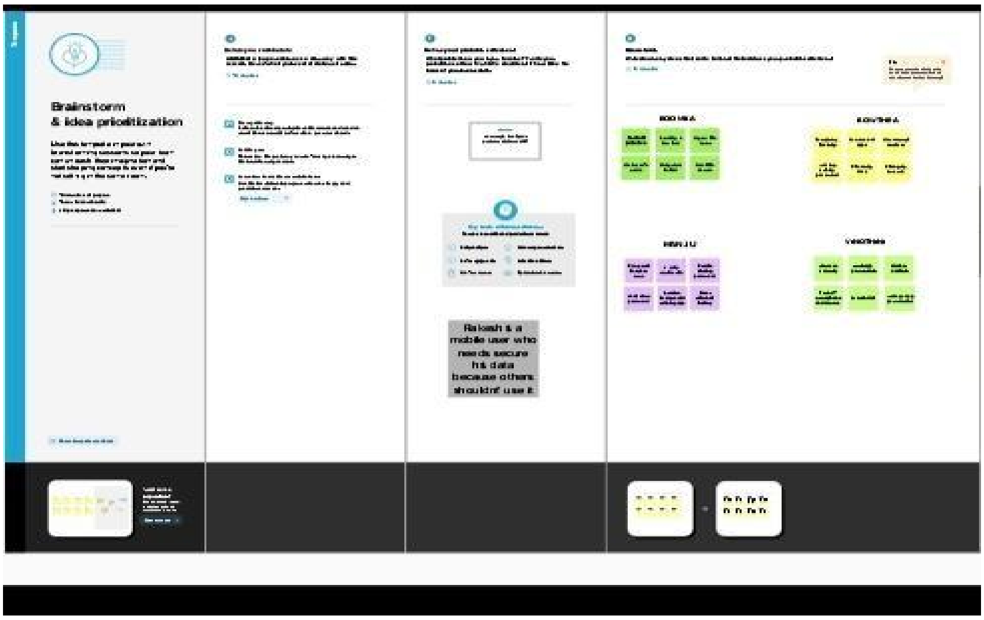
Due to the rapid growth of the Internet, users change their preference from traditional shopping to the electronic commerce. Instead of bank/shop robbery, nowadays, criminals try to find their victims in the cyberspace with some specific tricks. By using the anonymous structure of the Internet, attackers set out new techniques, such as phishing, to deceive victims with the use of false websites to collect their sensitive information such as account IDs, usernames, passwords, etc. Understanding whether a web page is legitimate or phishing is a very challenging problem, due to its semantics-based attack structure, which mainly exploits the computer users' vulnerabilities. Although software companies launch new anti-phishing products, which use blacklists, heuristics, visual and machine learning-based approaches, these products cannot prevent all of the phishing attacks. In this paper, a real-time anti-phishing system, which uses seven different classification algorithms and natural language processing (NLP) based features, is proposed. The system has the following distinguishing properties from other studies in the literature: language independence, use of a huge size of phishing and legitimate data, real-time execution, detection of new websites, independence from third-party services and use of feature-rich classifiers. For measuring the performance of the system, a new data set is constructed, and the experimental results are tested on it. According to the experimental and comparative results from the implemented classification algorithms, Random Forest algorithm with only NLP based features gives the best performance with the 97.98% accuracy rate for detection of phishing URLs.

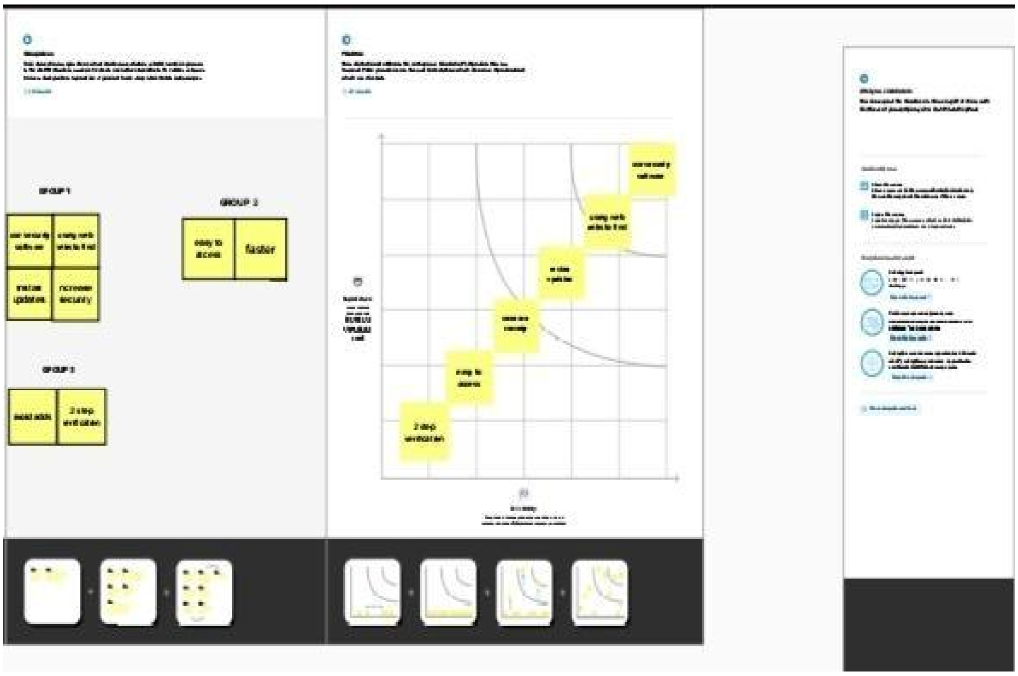
CHAPTER 3

IDEATION & PROPOSED SOLUTION

EMPATHY MAP

IDEATION & BRAINSTROMING





Proposed SolutionTemplate:

Project team shall fill the following information in proposed

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The purpose of this web phishing detection is to help individually identify these phishing URLs and prevent the Data(Or)device from phishing websites.
2.	Idea / Solution description	If user open the phishing URLs then web phishing detection will find out the phishing websites and notified to the user with detail description of phishing websites to prevent the data (or) device of user. The user can view the status of web phishing.
3.	Novelty / Uniqueness	Occupies less storage User can view the table (date, time, phishing URL and type of phishing) of phishing websites.
4.	Social Impact / Customer Satisfaction	User satisfaction does have a positive effect on an organisation's profitability. when the users are happy with the service they receive, they are more likely to trust and be loyal to the company.
5.	Business Model (Revenue Model)	It prevents the data. It avoids web phishing URLs. It has 2-step verification method. Fast and easy to access. We can access on all devices (desktop, android). Reduce the customer problems.
6.	Scalability of the Solution	An environment where they will be able to spend less time on grunt work and more time on actually resolving critical customer issue. User does not need much to worry about web phishing..

PROBLEM SOLUTION FIT

1. CUSTOMER SEGMENT(S) — CS

Who is your customer?

The user who wants to detect the phishing links and prevents the data.

6. CUSTOMER CONSTRAINTS — CC

What constraints prevent your customers from taking action or limit their choices of solutions?

1. Network connection
2. have a device

5. AVAILABLE SOLUTIONS — AS

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have?

Its finding out the phishing links .

2. JOBS-TO-BE-DONE / PROBLEMS — J&P

Which jobs-to-be-done (or problems) do you address for your customers?

1. People want to do the most of things with only one device.
2. They want automatically detect the phishing websits.
3. They want prevent their data from unknown person.

9. PROBLEM ROOT CAUSE — RC

What is the real reason that this problem exists? What is the back story behind the need to do this job?

1. Sometimes the user doesn't know about real and phishing links.
2. The unknown person sand the phishing link to get the user's data

7. BEHAVIOUR — BE

What does your customer do to address the problem and get the job done?

Detect phishing links and the user have network connection.

3. TRIGGERS — TR

What triggers customers to act?

1. People want to make their life easier, for connected anytime and anywhere.
2. Phishing detection act prevent the data

10. YOUR SOLUTION — SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.

If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.

These days data often needs to be visually presented in the form of interactive graphs or charts to be impact full and understand.

8. CHANNELS of BEHAVIOUR — CH

8.1 ONLINE

What kind of actions do customers take online?

online for detecting phishing websites.

8.2 OFFLINE

What kind of actions do customers take offline?

1. Table of phishing links.
2. offline videos and photos and some data.

4. EMOTIONS: BEFORE / AFTER — EM

How do customers feel when they face a problem or a job and afterwards?

1. They shouldn't want to others access their data.
2. User doesn't want much worry about web phishing

CHAPTER 4

REQUIREMENT ANALYSIS

Funtional Requirements

Following are the functional requirements of the proposed solution.

FR NO	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User uploads	User enter the URL to know about the detail of phishing
FR-4	User accessibility	User can get the notification about the phishing web page. User enter the URL to know about the detail of phishing. They can access table of history which detail description of phishing URL.
FR-5	User benefits	They can prevent the their data from the unknown persons like hackers.
FR-6	notification	The notification displays whether website is a legal site or phishing site.

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

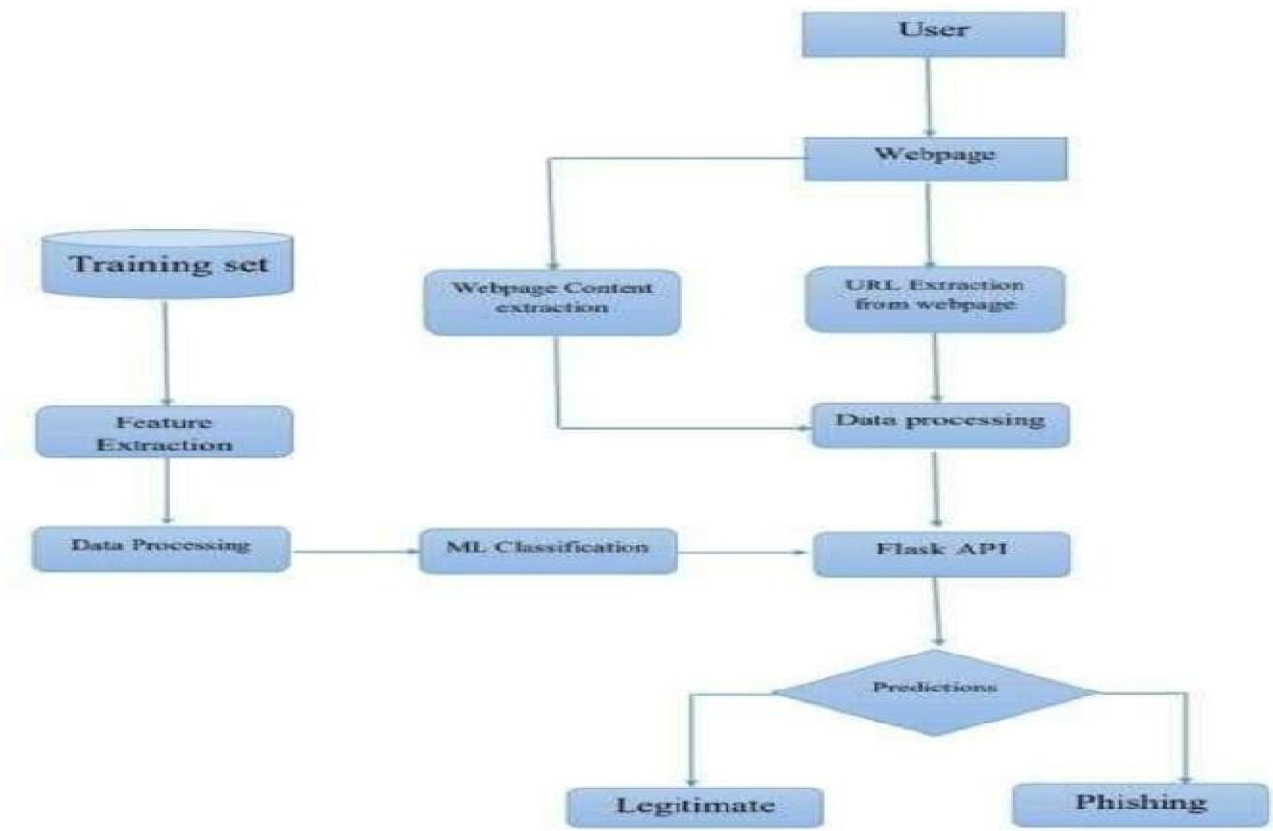
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	An user friendly web application, dynamic and attractive user interface ,maintaining the table of history and easily satisfies the user needs.
NFR-2	**Security**	Users information in the web application are protected by advanced security system. Implementation of proper logging. Authentication.
NFR-3	**Reliability**	Fault less. A piece of software operating without failure. The website's load time is not more than one second for user
NFR-4	**Performance**	Fast and quick analyzation of phishing URLs, is done as a GPU used for the model is 10% more fast in analysing and uploading the user uploaded the URL. Occupation of less storage space.
NFR-5	**Availability**	Available in a google play store, in security and production.
NFR-6	**Scalability**	It works in high speed authentication of phishing URLs, quick response for queries from user, highly reliable.

CHAPTER 5

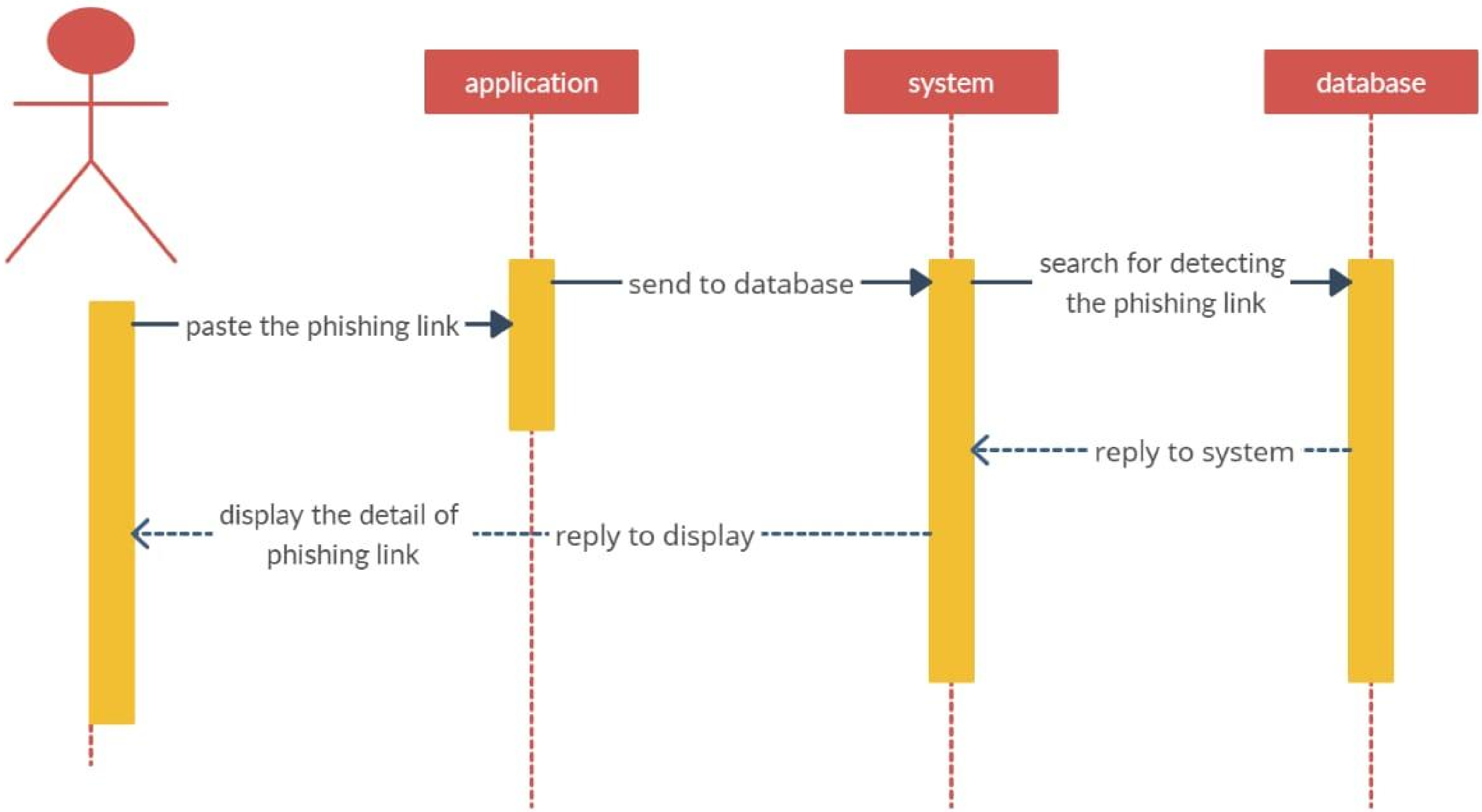
PROJECT DESIGN

Data Flow Diagram

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.



SOLUTION & TECHNICAL ARCHITECTURE



TECHNICAL ARCHITECTURE

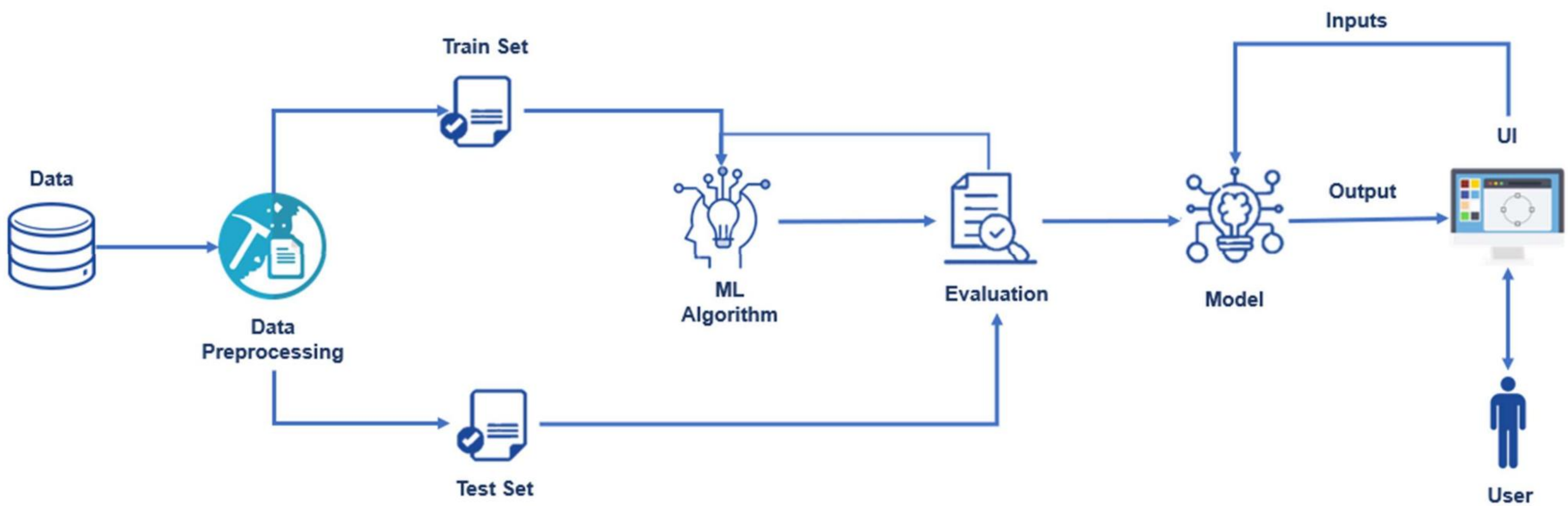


Table-1 : Components & Technologies:

S. No	Component	Description	Technology
1.	User Interface	Mobile application	HTML, CSS, JavaScript ,Python.
2.	Application Logic-1	Data pre-processing	Python.
3.	Application Logic-2	Model creating.	Keras, numpy, pandas.
4.	Application Logic-3	Web application(UI)	IBM Watson Assistant
5.	Database	data	MySQL, No SQL, etc.
6.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local File system
7.	External API	Purpose of External API used in the application	Data processing API.
8.	Machine Learning Model	Binary, multi class, regression classification.	Object Recognition Model.
9.	Infrastructure (Server / Cloud)	Application Deployment on web server.	Flask – python HTTP server.

Table-2: Application Characteristics:

S. No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask	Technology of Opensource framework
2.	Security Implementations	Secure flag for cookies.	Flask WTF Session_cookie_secure.
3.	Scalable Architecture	Micro-services	Micro web applications.
4.	Availability	Integrated supporting for unit testing.	SQL, Sinatra rubby, framework, jinja 2
5.	Performance	HTTP request handling High flexibility.	SQL, Sinatra rubby, framework,jinja 2.

User Stories

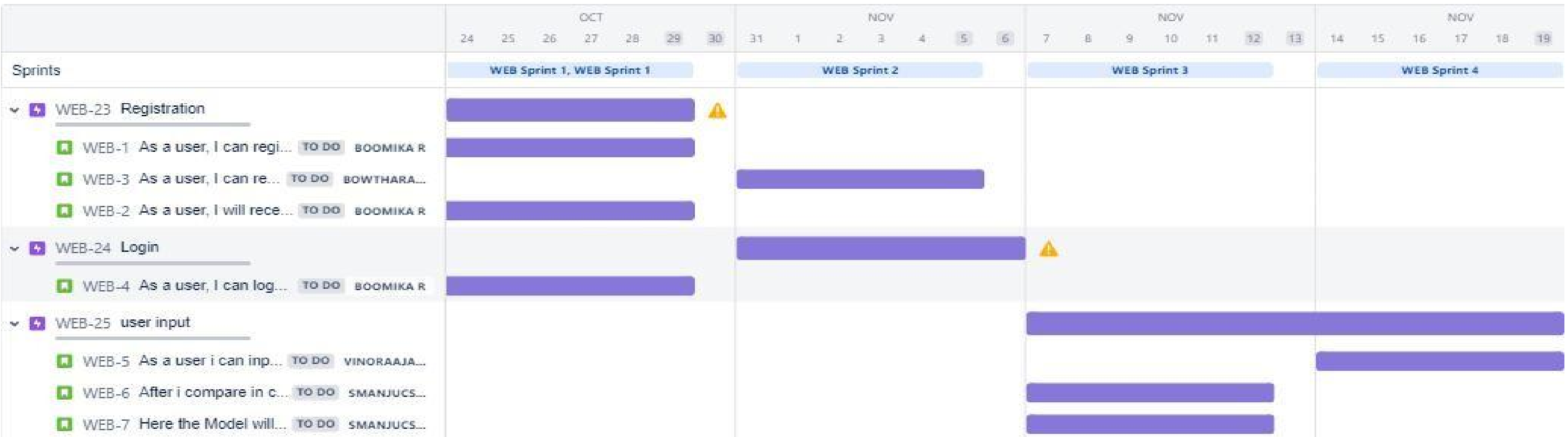
Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application By entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		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					
Customer (Web user)	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
Customer Care Executive	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
Administrator	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

SPRINT PLANNING & ESTIMATION

Title	Description	Date
Literature Survey and Information Gathering	Gathering Information by referring the technical papers, research publications etc	10 September 2022
Prepare Empathy Map	To capture user pain and gains Prepare List of Problem Statement	17 September 2022
Ideation	Prioritise a top 3 ideas based on feasibility and Importance	18 September 2022
Proposed Solution	Solution include novelty, feasibility, business model, social impact and scalability of solution	1 October 2022
Problem Solution Fit	Solution fit document	1 October 2022
Solution Architecture	Solution Architecture	1 October 2022
Customer Journey	To Understand User Interactions and experiences with application	8 October 2022
Functional Requirement	Prepare functional Requirement	9 October 2022
Data flow Diagrams	Data flow diagram	11 October 2022
Technology Architecture	Technology Architecture diagram	15 October 2022
Milestone sprint & delivery plan	Activity what we done &further plans	21 October 2022
Project Development- Delivery of sprint 1,2,3 &4	Develop and submit the developed code by testing it	24 October 2022 – 19 November 2022

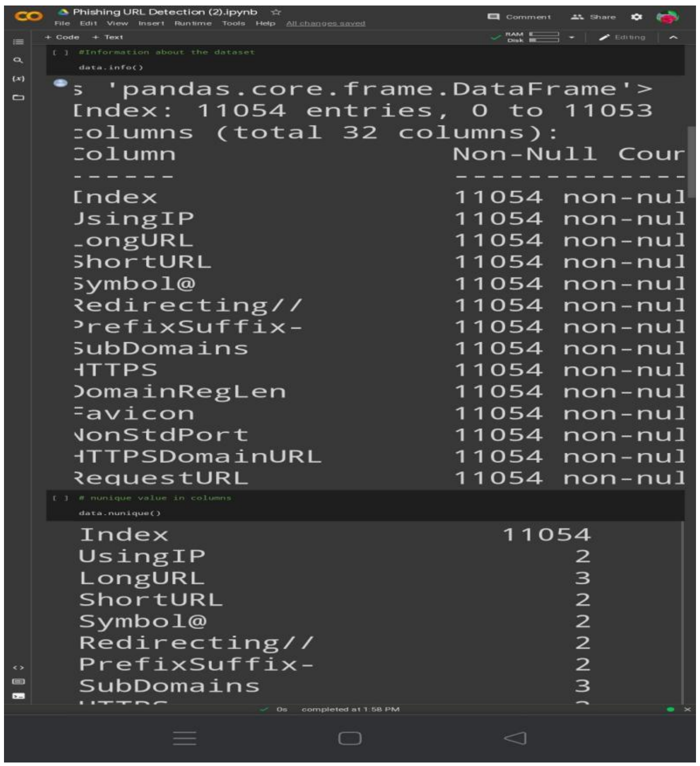
REPORTS FROM JIRA

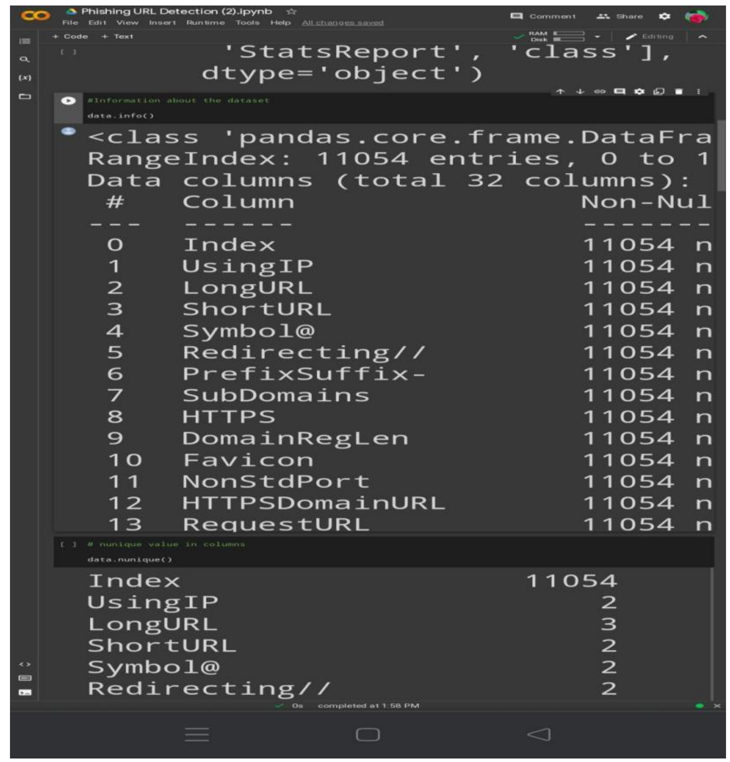


CODING & SOLUTIONING

FEATURE 1

Phishing URL Detection





Phishing URL Detection (2).ipynb

File Edit View Insert Runtime Tools Help All changes saved

```
dtypes: int64(32)
memory usage: 2.7 MB
```

```
# nunique value in columns
data.nunique()
```

```
AnchorURL              3
LinksInScriptTags      3
ServerFormHandler      3
InfoEmail              2
AbnormalURL            2
WebsiteForwarding      2
StatusBarCust          2
DisableRightClick      2
UsingPopupWindow       2
IframeRedirection      2
AgeofDomain            2
DNSRecording           2
WebsiteTraffic         3
PageRank               2
GoogleIndex            2
LinksPointingToPage    3
StatsReport            2
class                  2
dtype: int64
```

```
#droping index column
data = data.drop(['Index'],axis = 1)
```

```
#description of dataset
data.describe().T
```

	count	mean	std	min	25%	50%	75%	max
UsingIP	11054.0	0.313914	0.949495	-1.0	-1.0	1.0	1.0	1.0
LongURL	11054.0	-0.633345	0.765973	-1.0	-1.0	-1.0	-1.0	1.0
ShortURL	11054.0	0.738737	0.674024	-1.0	1.0	1.0	1.0	1.0
Symbol@	11054.0	0.700561	0.713625	-1.0	1.0	1.0	1.0	1.0
Redirecting//	11054.0	0.741632	0.670837	-1.0	1.0	1.0	1.0	1.0
PrefixSuffix-	11054.0	-0.734938	0.678165	-1.0	-1.0	-1.0	-1.0	1.0
SubDomains	11054.0	0.064049	0.817492	-1.0	-1.0	0.0	1.0	1.0

0s completed at 1:58 PM

Phishing URL Detection (2).ipynb

File Edit View Insert Runtime Tools Help All changes saved

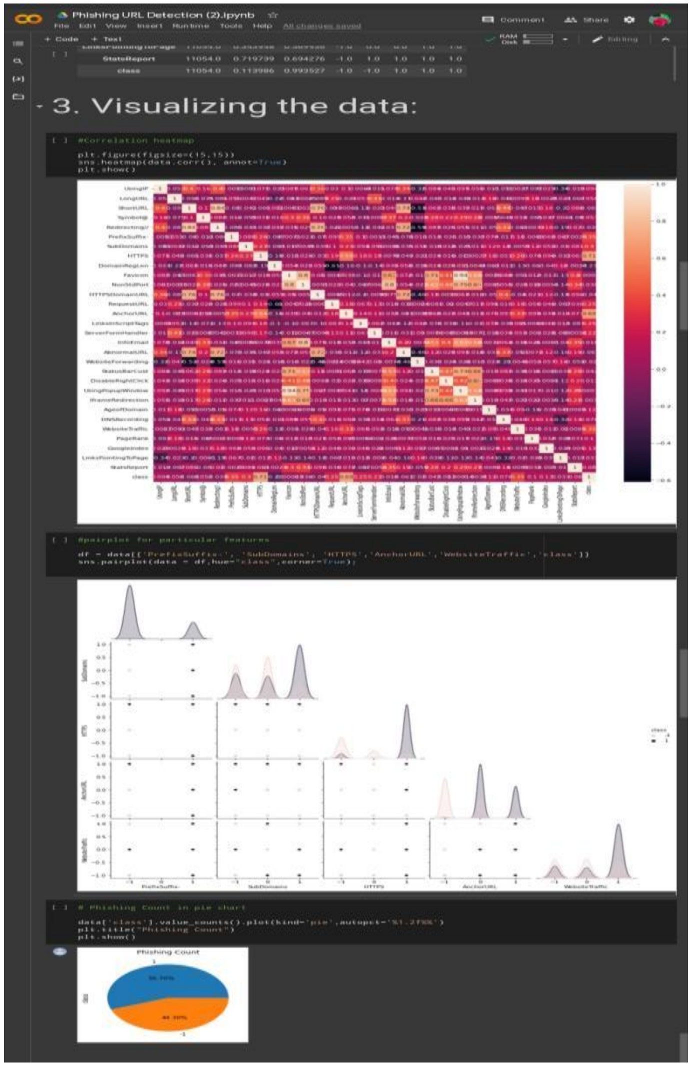

```
GoogleIndex            2
LinksPointingToPage    3
StatsReport            2
class                  2
dtype: int64
```

```
#droping index column
data = data.drop(['Index'],axis = 1)
```

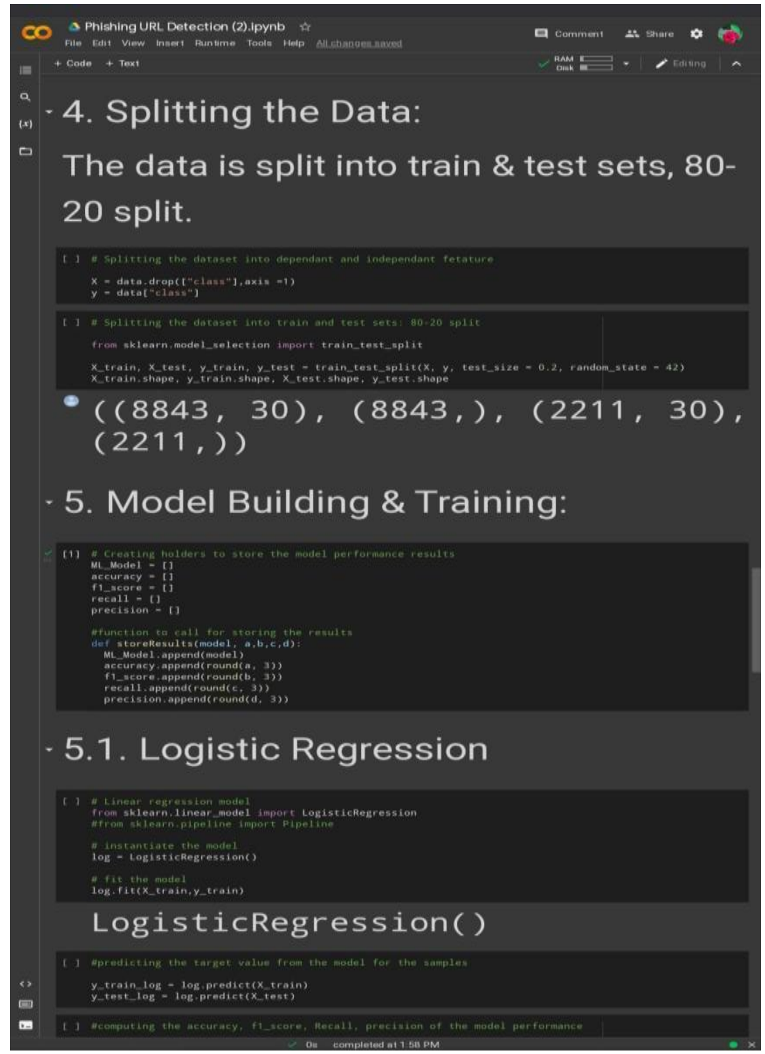
```
#description of dataset
data.describe().T
```


	count	mean	std	min	25%	50%	75%	max
UsingIP	11054.0	0.313914	0.949495	-1.0	-1.0	1.0	1.0	1.0
LongURL	11054.0	-0.633345	0.765973	-1.0	-1.0	-1.0	-1.0	1.0
ShortURL	11054.0	0.738737	0.674024	-1.0	1.0	1.0	1.0	1.0
Symbol@	11054.0	0.700561	0.713625	-1.0	1.0	1.0	1.0	1.0
Redirecting//	11054.0	0.741632	0.670837	-1.0	1.0	1.0	1.0	1.0
PrefixSuffix-	11054.0	-0.734938	0.678165	-1.0	-1.0	-1.0	-1.0	1.0
SubDomains	11054.0	0.064049	0.817492	-1.0	-1.0	0.0	1.0	1.0
HTTPS	11054.0	0.251040	0.911856	-1.0	-1.0	1.0	1.0	1.0
DomainRegLen	11054.0	-0.336711	0.941651	-1.0	-1.0	-1.0	1.0	1.0
Favicon	11054.0	0.628551	0.777804	-1.0	1.0	1.0	1.0	1.0
NonStdPort	11054.0	0.728243	0.685350	-1.0	1.0	1.0	1.0	1.0
HTTPSDomainURL	11054.0	0.675231	0.737640	-1.0	1.0	1.0	1.0	1.0
RequestURL	11054.0	0.186720	0.982458	-1.0	-1.0	1.0	1.0	1.0
AnchorURL	11054.0	-0.076443	0.715116	-1.0	-1.0	0.0	0.0	1.0
LinksInScriptTags	11054.0	-0.118238	0.763933	-1.0	-1.0	0.0	0.0	1.0
ServerFormHandler	11054.0	-0.595712	0.759168	-1.0	-1.0	-1.0	-1.0	1.0
InfoEmail	11054.0	0.635788	0.771899	-1.0	1.0	1.0	1.0	1.0
AbnormalURL	11054.0	0.705446	0.708796	-1.0	1.0	1.0	1.0	1.0
WebsiteForwarding	11054.0	0.115705	0.319885	0.0	0.0	0.0	0.0	1.0
StatusBarCust	11054.0	0.762077	0.647516	-1.0	1.0	1.0	1.0	1.0
DisableRightClick	11054.0	0.913877	0.406009	-1.0	1.0	1.0	1.0	1.0
UsingPopupWindow	11054.0	0.613353	0.789845	-1.0	1.0	1.0	1.0	1.0
IframeRedirection	11054.0	0.816899	0.576807	-1.0	1.0	1.0	1.0	1.0
AgeofDomain	11054.0	0.061335	0.998162	-1.0	-1.0	1.0	1.0	1.0
DNSRecording	11054.0	0.377239	0.926158	-1.0	-1.0	1.0	1.0	1.0
WebsiteTraffic	11054.0	0.287407	0.827680	-1.0	0.0	1.0	1.0	1.0
PageRank	11054.0	-0.483626	0.875314	-1.0	-1.0	-1.0	1.0	1.0
GoogleIndex	11054.0	0.721549	0.692395	-1.0	1.0	1.0	1.0	1.0
LinksPointingToPage	11054.0	0.343948	0.569936	-1.0	0.0	0.0	1.0	1.0
StatsReport	11054.0	0.719739	0.694276	-1.0	1.0	1.0	1.0	1.0
class	11054.0	0.113986	0.993527	-1.0	-1.0	1.0	1.0	1.0

Visualizing the data

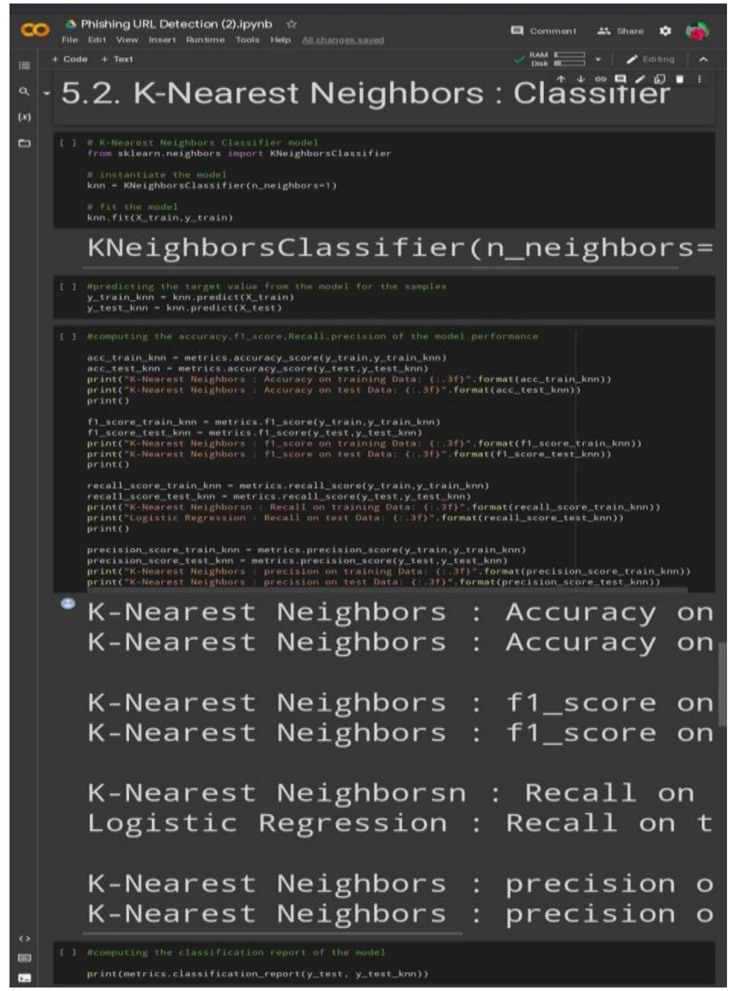


Splitting the data



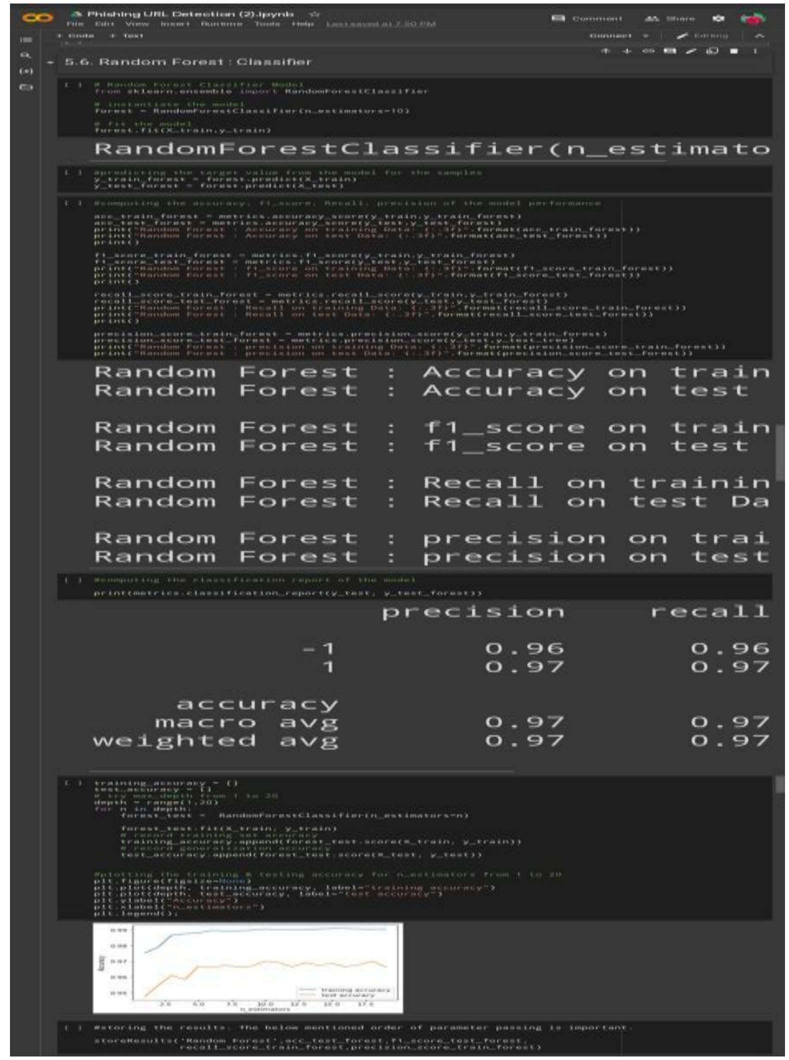
5.Model Building & Trainning

K-Nearest Neighbors: Classifier

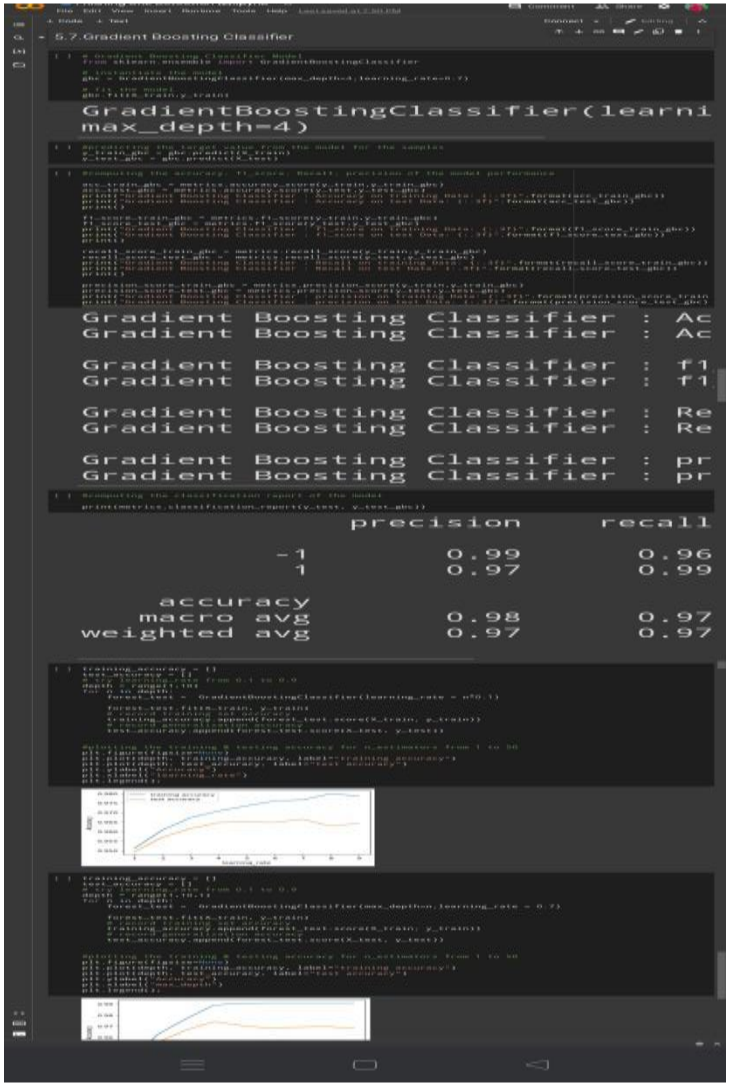


Decision Trees: Classifier

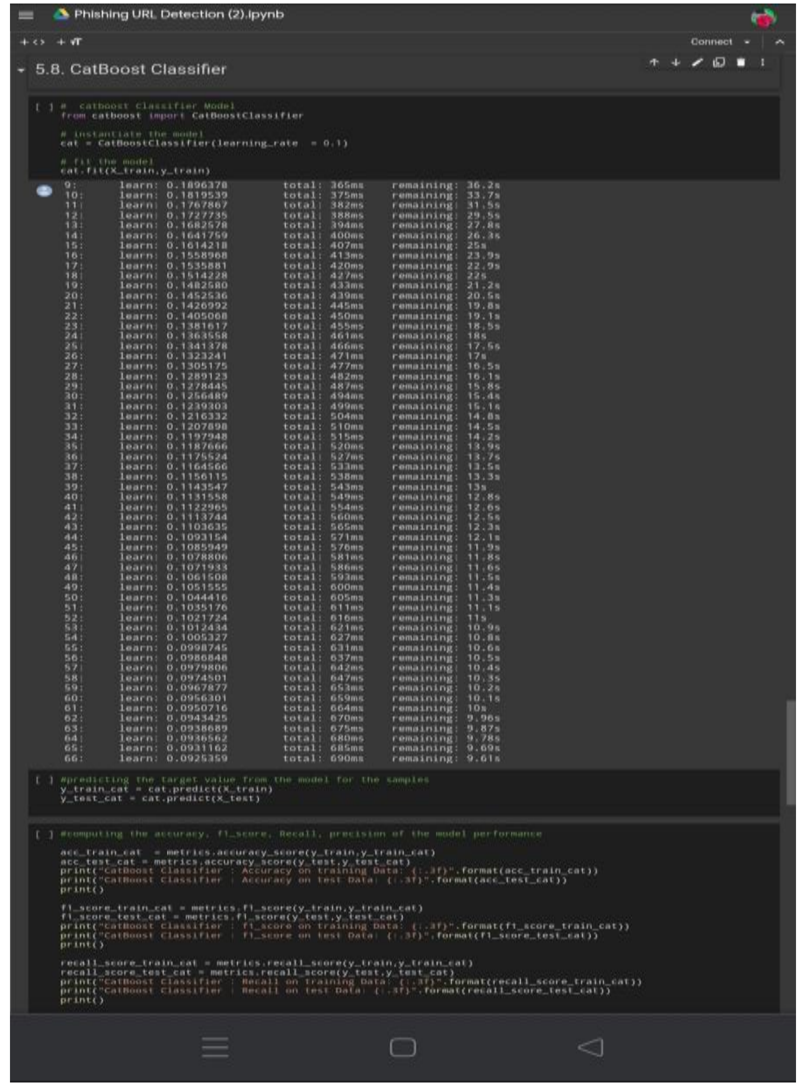
Random Forest : Classifier



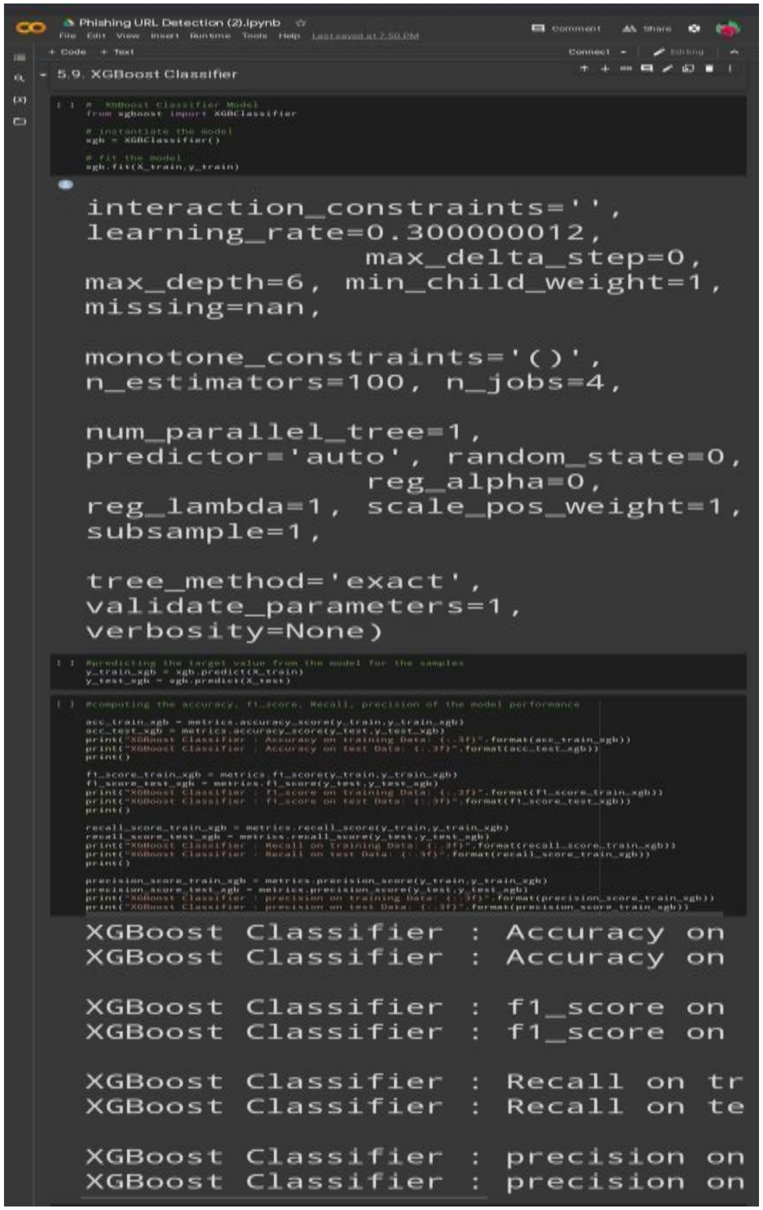
Gradient Boosting Classifier



CatBoost Classifier

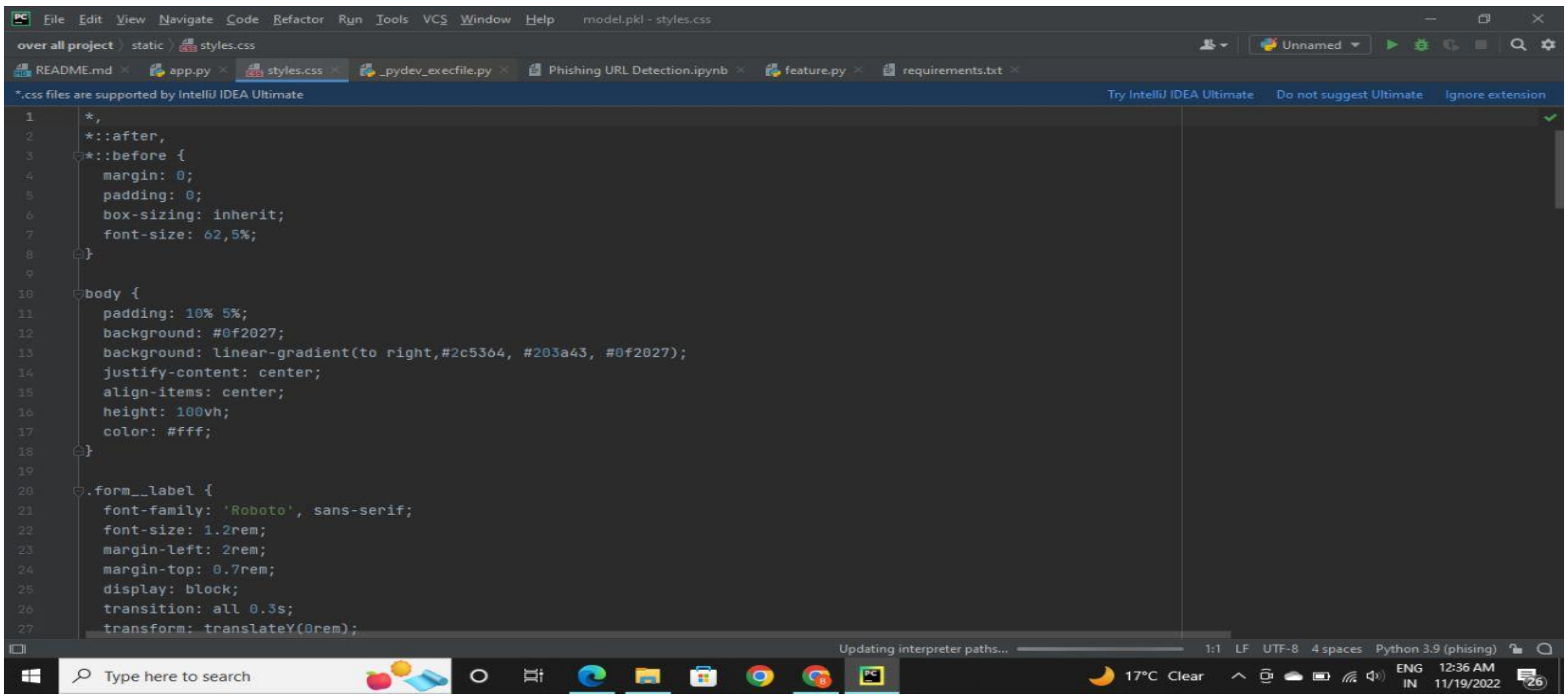


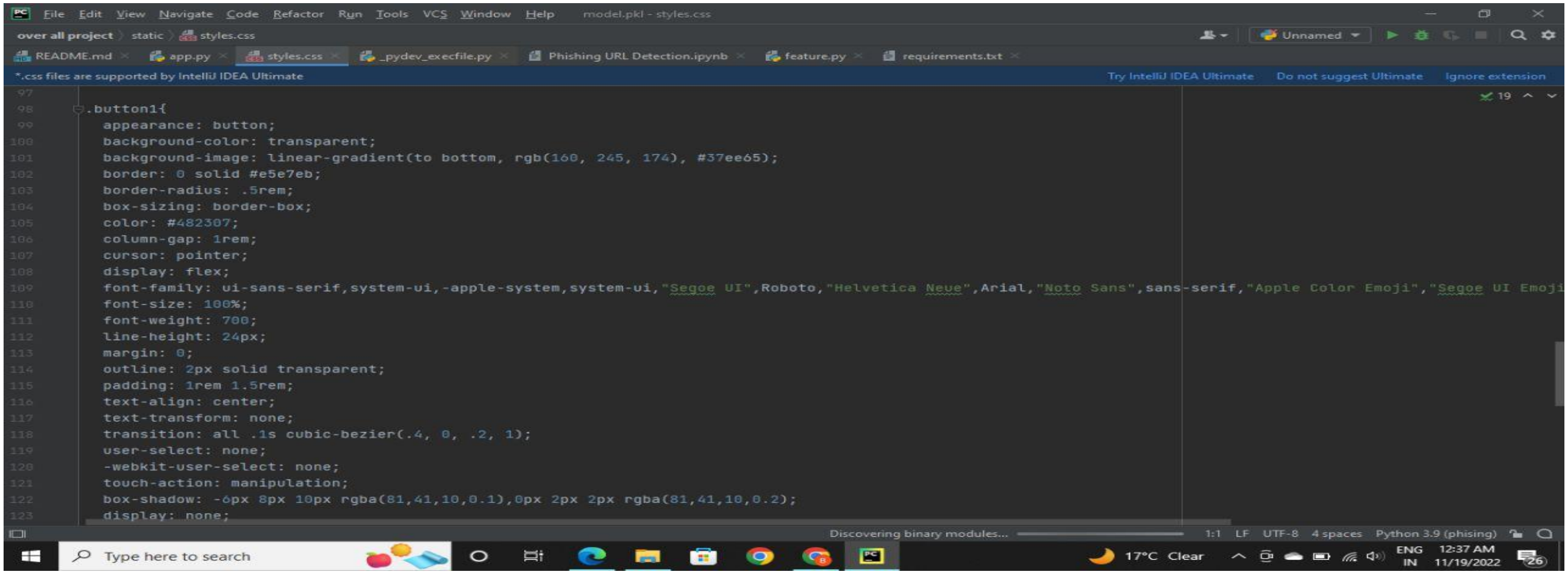
XGBoost Classifier

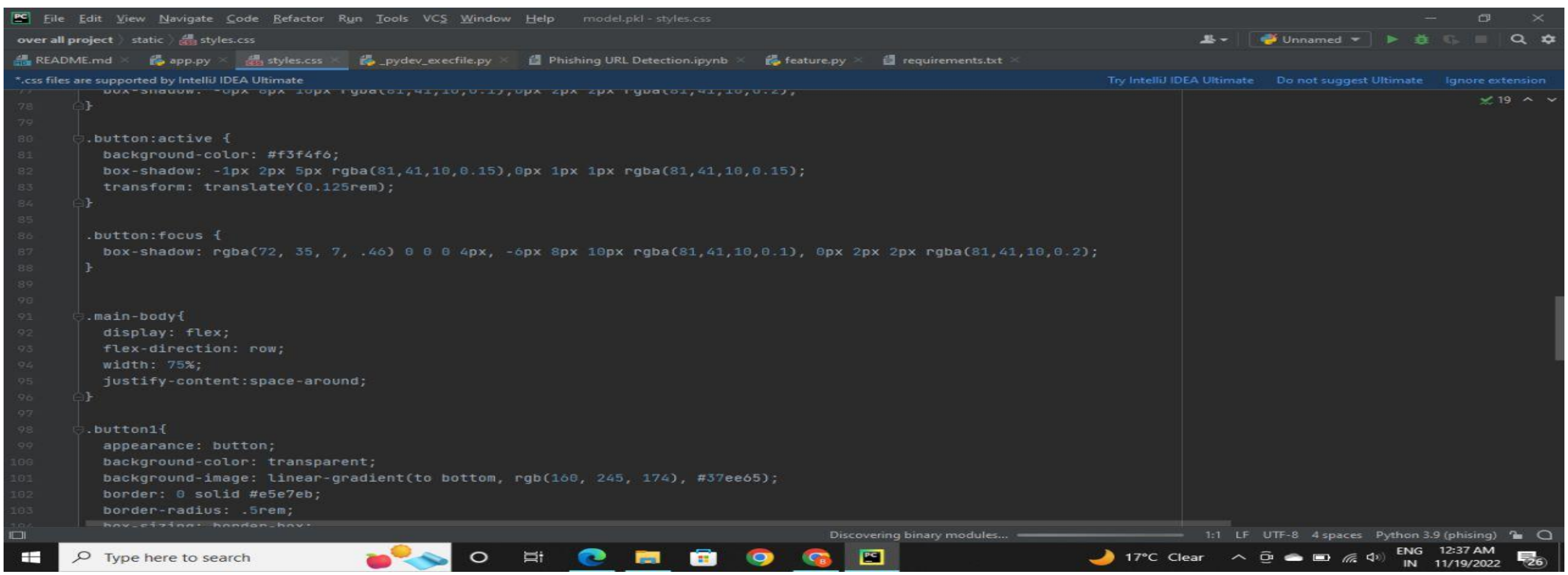


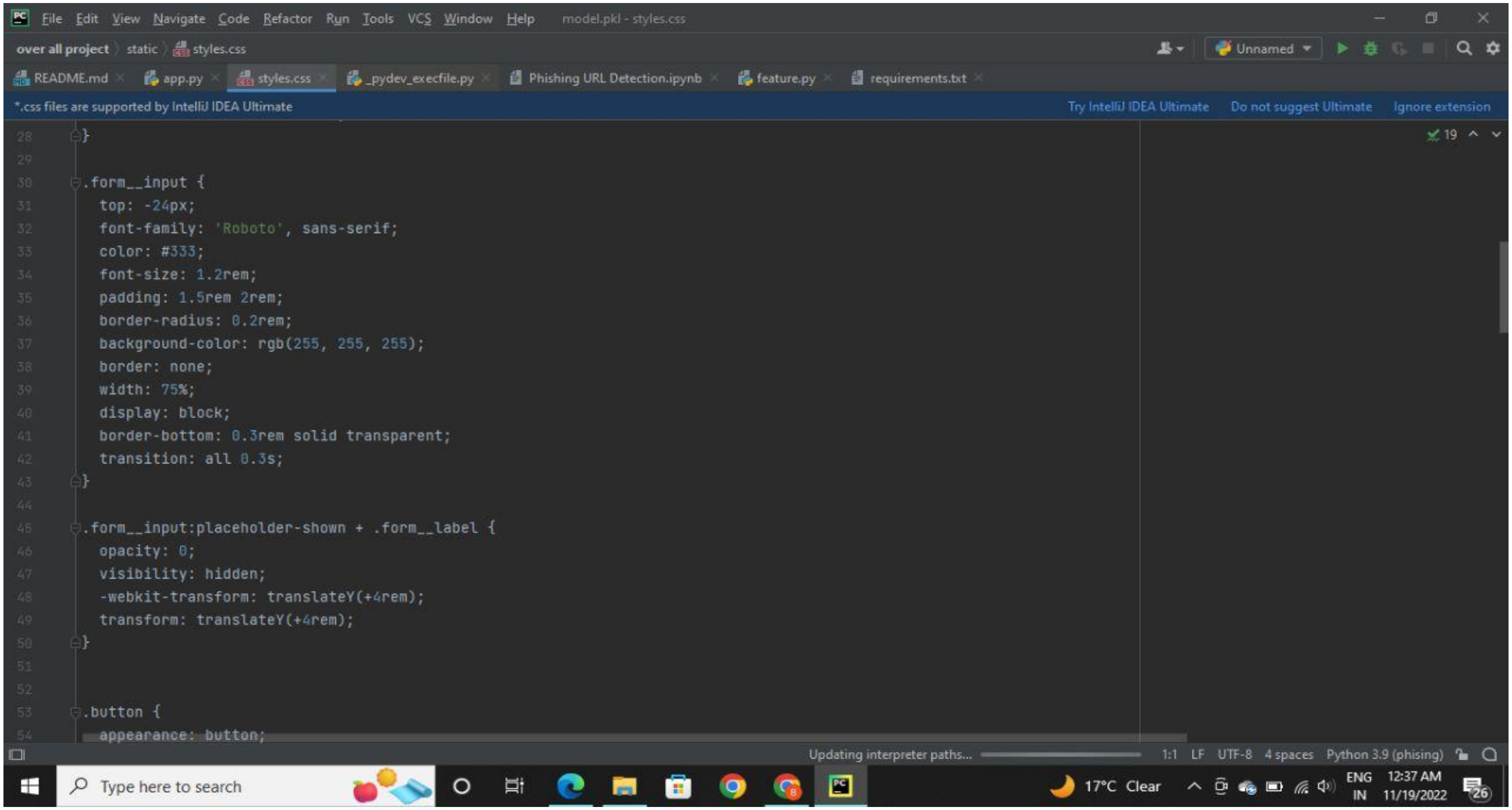
FEATURE 2

Styles.css

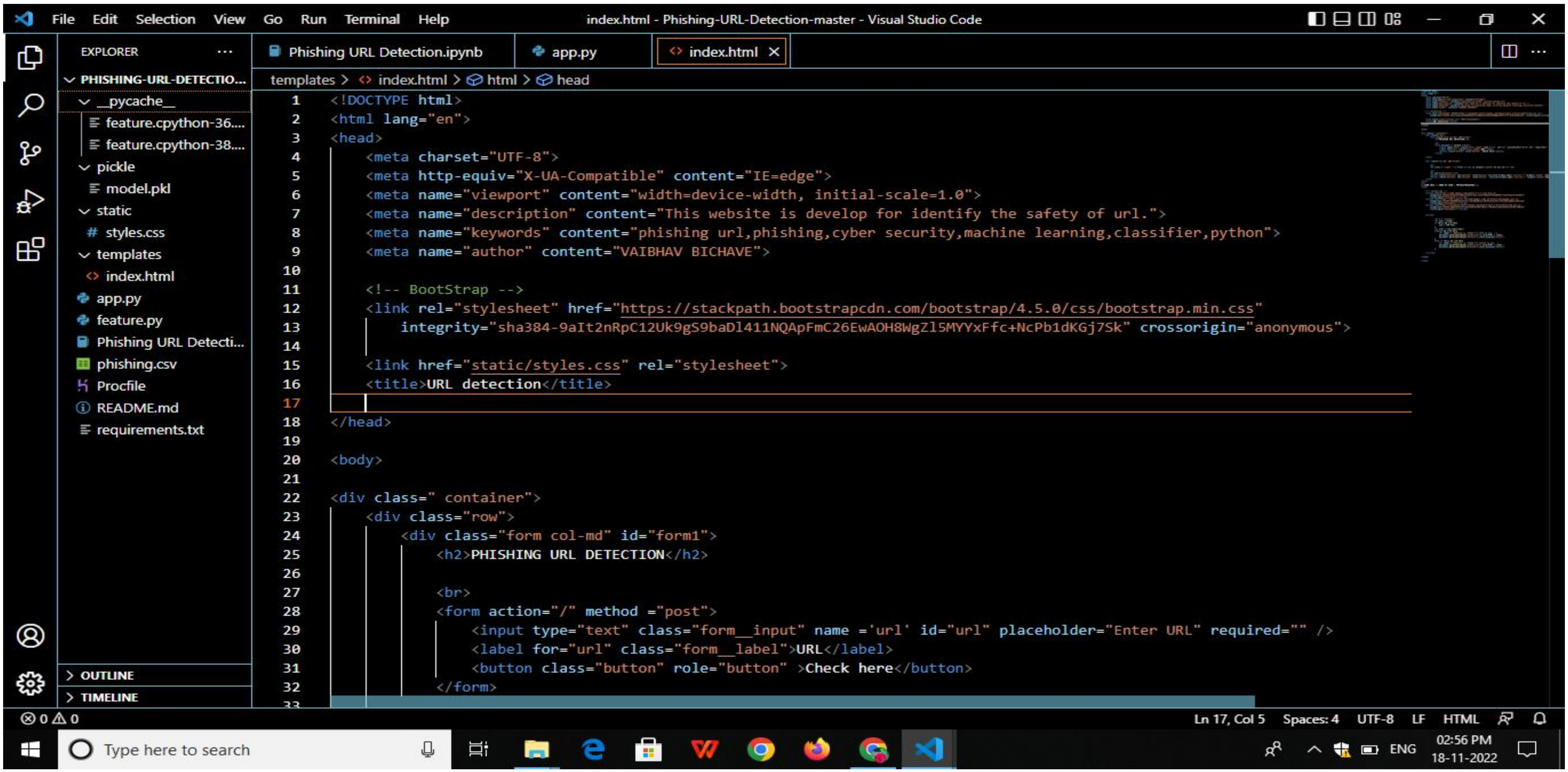



File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - styles.css
over all project static styles.css
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
*.css files are supported by IntelliJ IDEA Ultimate
Try IntelliJ IDEA Ultimate Do not suggest Ultimate Ignore extension
.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: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji
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;
Discovering binary modules... 1:1 LF UTF-8 4 spaces Python 3.9 (phising)
Type here to search
17°C Clear
ENG IN 12:37 AM 11/19/2022

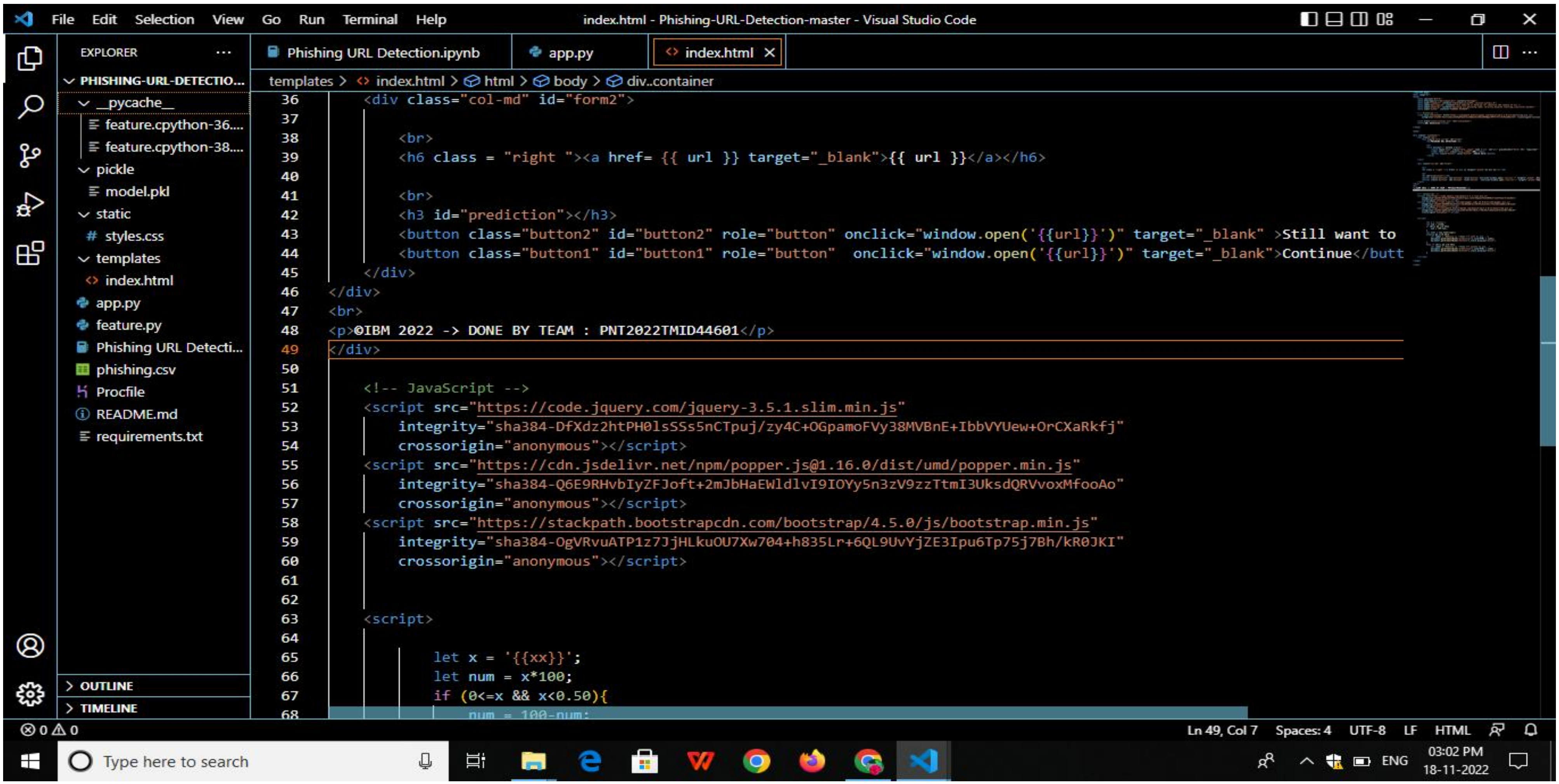

File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - styles.css
over all project static styles.css
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
*.css files are supported by IntelliJ IDEA Ultimate
Try IntelliJ IDEA Ultimate Do not suggest Ultimate Ignore extension
}
.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;
width: 75%;
justify-content:space-around;
}
.button1{
appearance: button;
background-color: transparent;
background-image: linear-gradient(to bottom, rgb(160, 245, 174), #37ee65);
border: 0 solid #e5e7eb;
border-radius: .5rem;
Discovering binary modules... 1:1 LF UTF-8 4 spaces Python 3.9 (phising)
Type here to search
17°C Clear
ENG IN 12:37 AM 11/19/2022


File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - styles.css
over all project static styles.css
Unnamed
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
*.css files are supported by IntelliJ IDEA Ultimate
Try IntelliJ IDEA Ultimate Do not suggest Ultimate Ignore extension
}
.form__input {
top: -24px;
font-family: 'Roboto', sans-serif;
color: #333;
font-size: 1.2rem;
padding: 1.5rem 2rem;
border-radius: 0.2rem;
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;
Updating interpreter paths...
1:1 LF UTF-8 4 spaces Python 3.9 (phising)
Type here to search
17°C Clear
ENG IN
12:37 AM 11/19/2022

Index.html


File Edit Selection View Go Run Terminal Help
index.html - Phishing-URL-Detection-master - Visual Studio Code
EXPLORER
Phishing URL Detection.ipynb
app.py
index.html
PHISHING-URL-DETECTIO...
__pycache__
feature.cpython-36....
feature.cpython-38....
pickle
model.pkl
static
styles.css
templates
index.html
app.py
feature.py
Phishing URL Detecti...
phishing.csv
Procfile
README.md
requirements.txt
OUTLINE
TIMELINE
templates > index.html > html > head
<!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 develop for identify the safety of url.">
<meta name="keywords" content="phishing url,phishing,cyber security,machine learning,classifier,python">
<meta name="author" content="VAIBHAV BICHAVE">
<!-- BootStrap -->
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"
integrity="sha384-9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk" crossorigin="anonymous">
<link href="static/styles.css" rel="stylesheet">
<title>URL detection</title>
</head>
<body>
<div class=" container">
<div class="row">
<div class="form col-md" id="form1">
<h2>PHISHING URL DETECTION</h2>

<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" >Check here</button>
</form>
Ln 17, Col 5 Spaces: 4 UTF-8 LF HTML
Type here to search
ENG
02:56 PM
18-11-2022

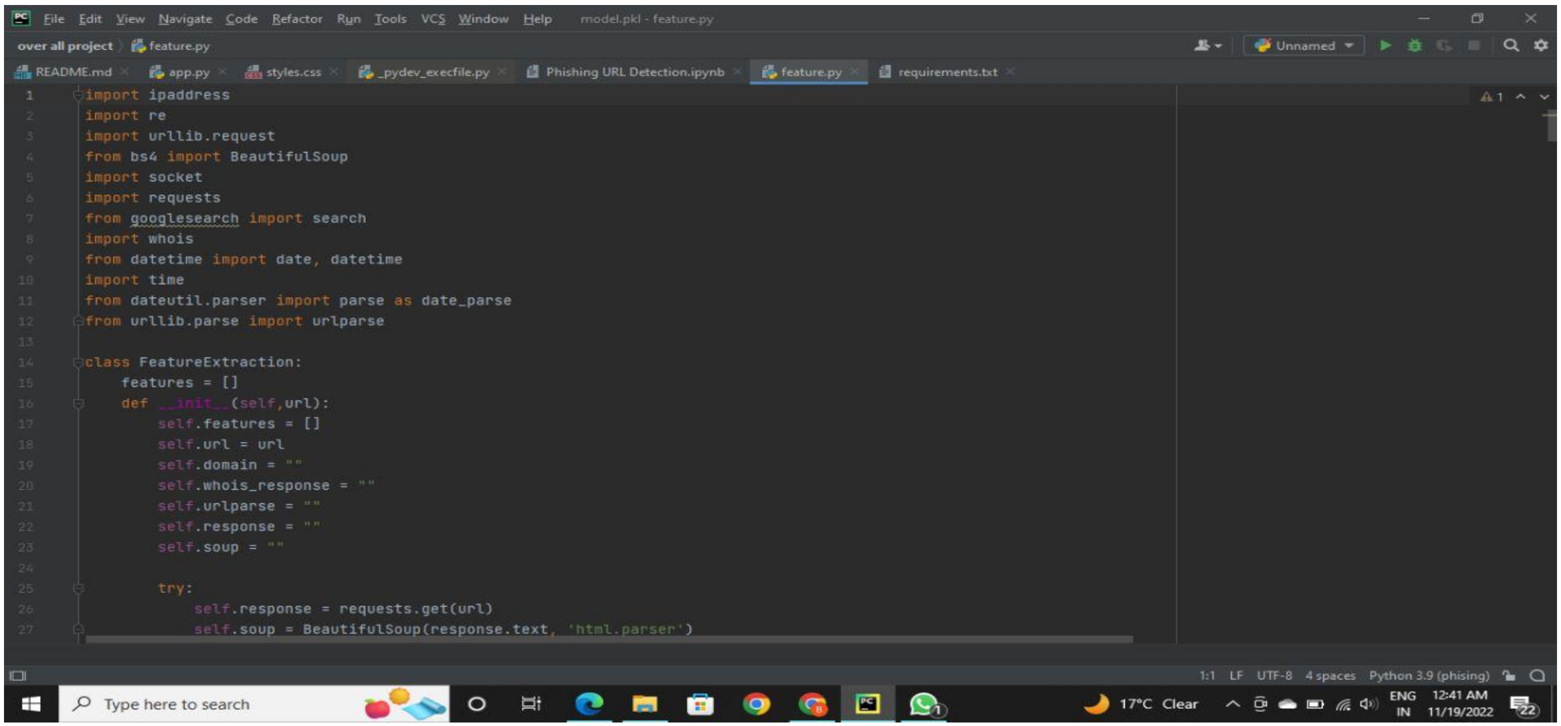

File Edit Selection View Go Run Terminal Help
index.html - Phishing-URL-Detection-master - Visual Studio Code
EXPLORER
PHISHING-URL-DETECTIO...
pycache
feature.cpython-36....
feature.cpython-38....
pickle
model.pkl
static
styles.css
templates
index.html
app.py
feature.py
Phishing URL Detecti...
phishing.csv
Procfile
README.md
requirements.txt
OUTLINE
TIMELINE
Phishing URL Detection.ipynb
app.py
index.html
templates > index.html > html > body > div..container
<div class="col-md" id="form2">

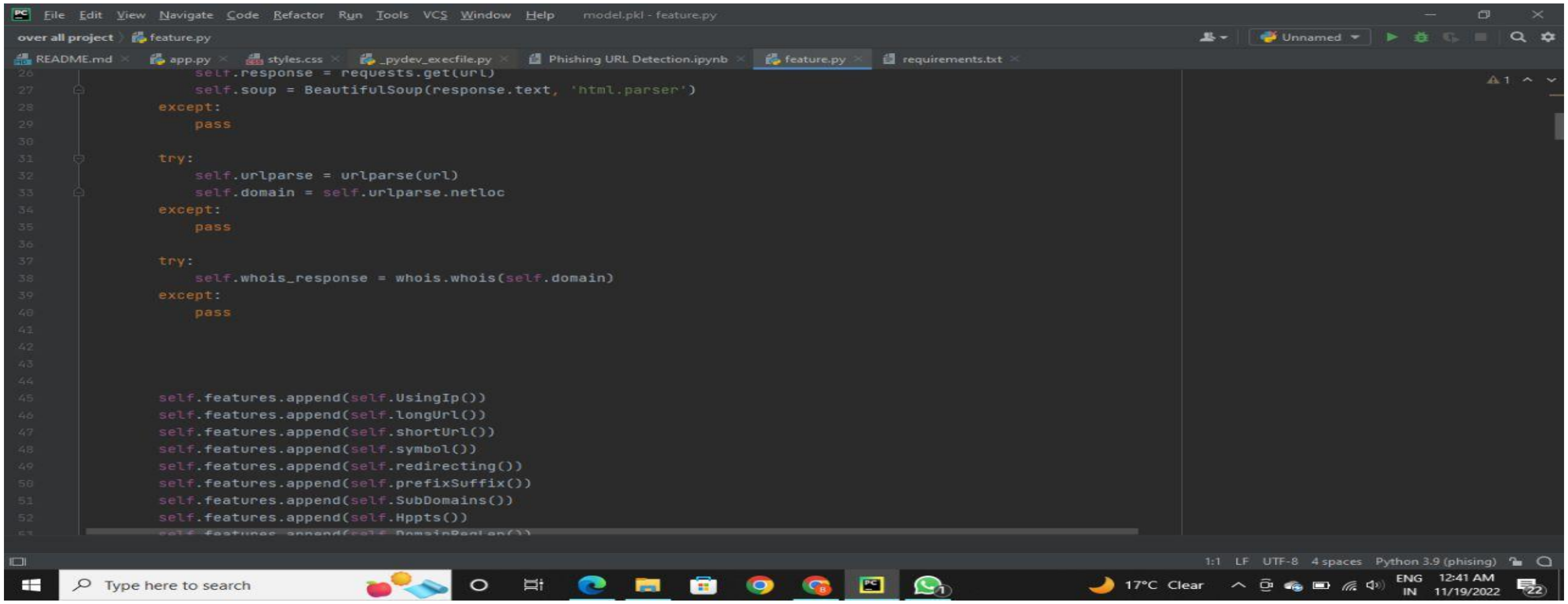
<h6 class = "right ">{{ url }}</h6>

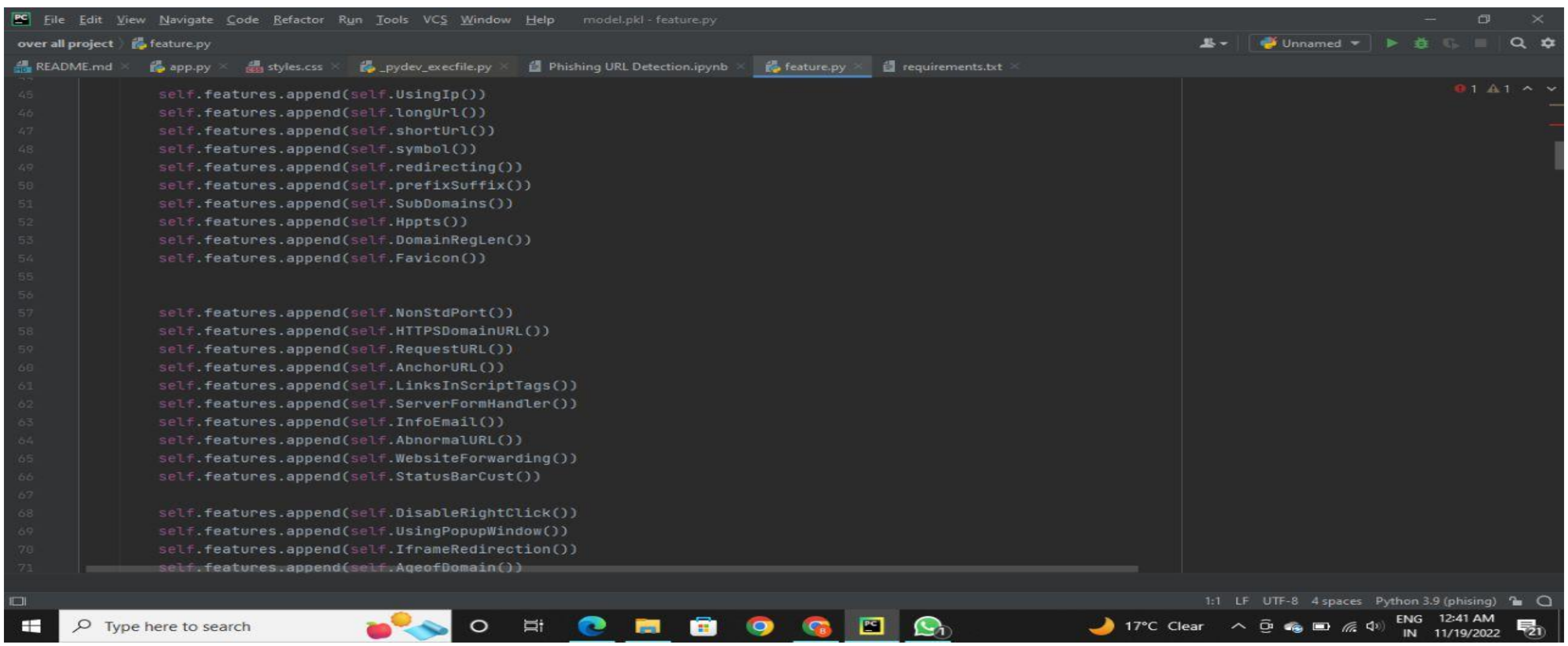
<h3 id="prediction"></h3>
<button class="button2" id="button2" role="button" onclick="window.open('{{url}}')" target="_blank" >Still want to
<button class="button1" id="button1" role="button" onclick="window.open('{{url}}')" target="_blank">Continue</butt
</div>
</div>

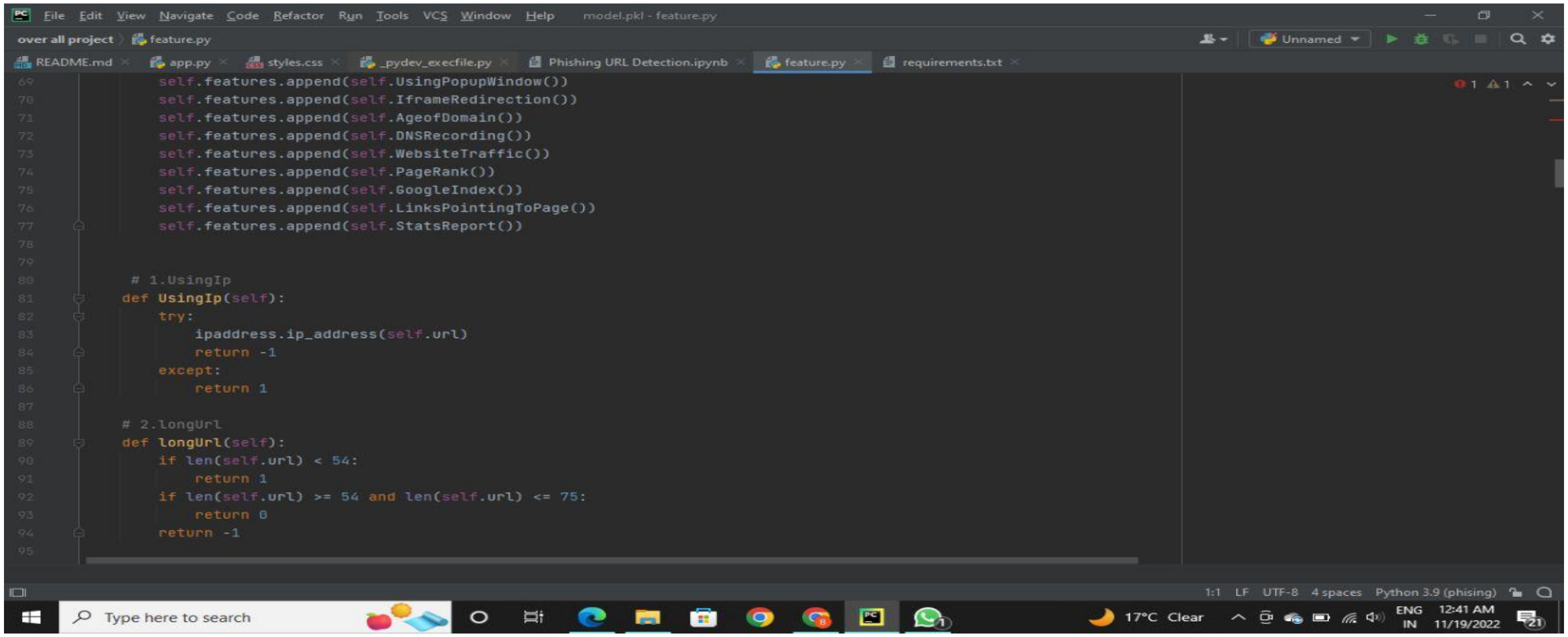
<p>©IBM 2022 -> DONE BY TEAM : PNT2022TMID44601</p>
</div>
<!-- JavaScript -->
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"
integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
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}}';
let num = x*100;
if (0<=x && x<0.50){
num = 100-num;
Ln 49, Col 7 Spaces: 4 UTF-8 LF HTML
Type here to search
ENG
03:02 PM
18-11-2022

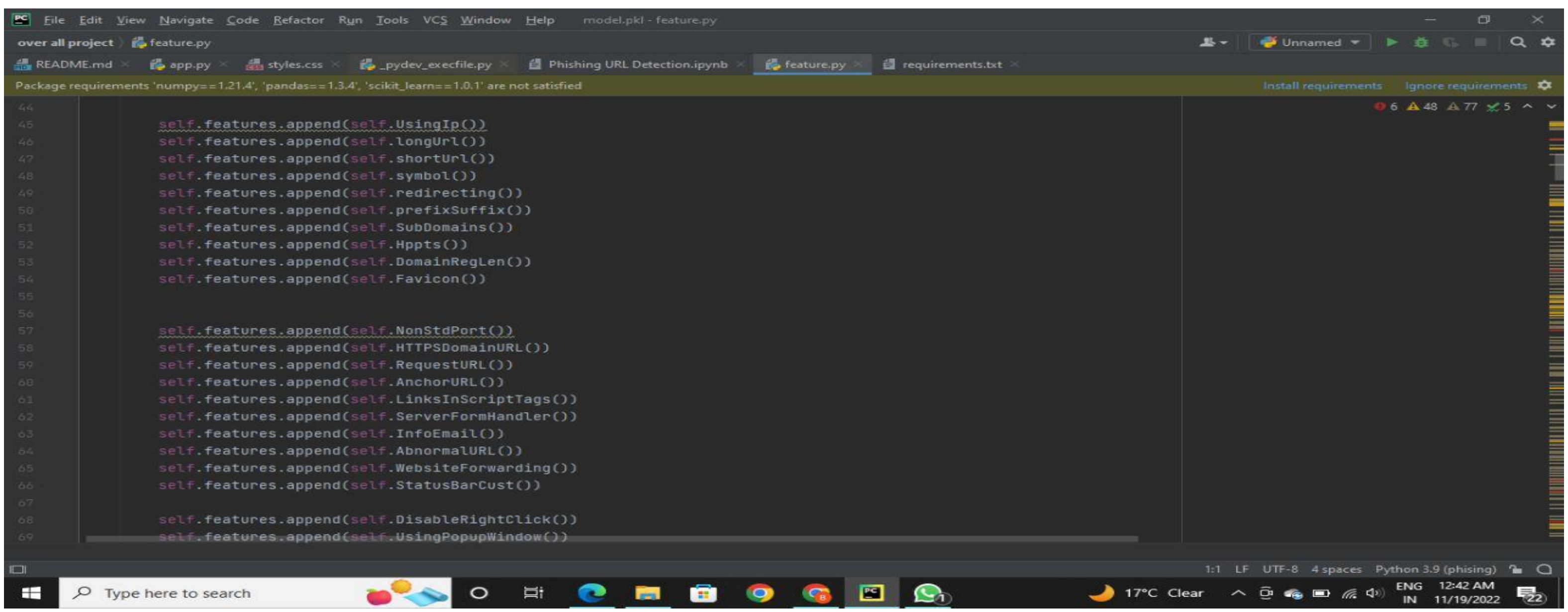
Feauture.py

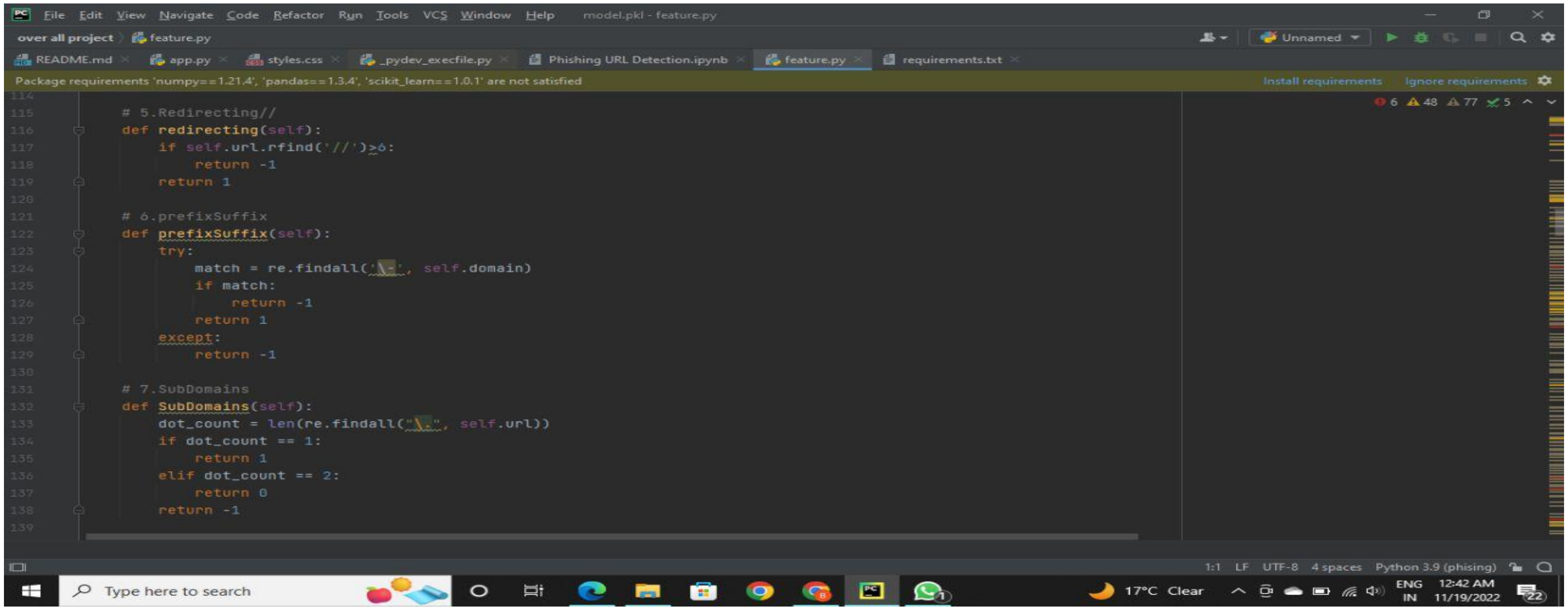

File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - feature.py
over all project feature.py
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
import ipaddress
import re
import urllib.request
from bs4 import BeautifulSoup
import socket
import requests
from googlesearch import search
import whois
from datetime import date, datetime
import time
from dateutil.parser import parse as date_parse
from urllib.parse import urlparse
class FeatureExtraction:
features = []
def __init__(self,url):
self.features = []
self.url = url
self.domain = ""
self.whois_response = ""
self.urlparse = ""
self.response = ""
self.soup = ""
try:
self.response = requests.get(url)
self.soup = BeautifulSoup(response.text, 'html.parser')
1:1 LF UTF-8 4 spaces Python 3.9 (phising)
Type here to search
17°C Clear
ENG IN 12:41 AM 11/19/2022


File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - feature.py
over all project feature.py
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
self.soup = BeautifulSoup(response.text, 'html.parser')
except:
pass
try:
self.urlparse = urlparse(url)
self.domain = self.urlparse.netloc
except:
pass
try:
self.whois_response = whois.whois(self.domain)
except:
pass
self.features.append(self.UsingIp())
self.features.append(self.longUrl())
self.features.append(self.shortUrl())
self.features.append(self.symbol())
self.features.append(self.redirecting())
self.features.append(self.prefixSuffix())
self.features.append(self.SubDomains())
self.features.append(self.Hppts())
Type here to search
17°C Clear
12:41 AM
11/19/2022

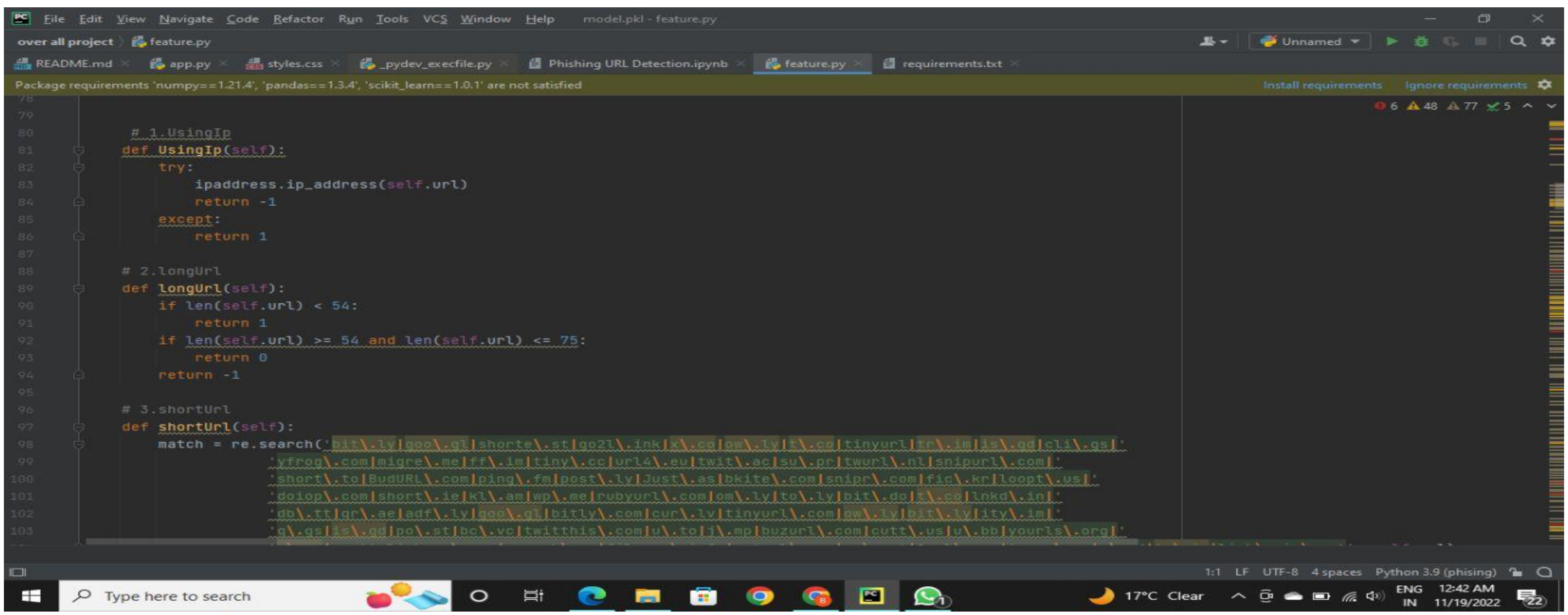

File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - feature.py
over all project feature.py
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
self.features.append(self.UsingIp())
self.features.append(self.longUrl())
self.features.append(self.shortUrl())
self.features.append(self.symbol())
self.features.append(self.redirecting())
self.features.append(self.prefixSuffix())
self.features.append(self.SubDomains())
self.features.append(self.Hppts())
self.features.append(self.DomainRegLen())
self.features.append(self.Favicon())
self.features.append(self.NonStdPort())
self.features.append(self.HTTPSDomainURL())
self.features.append(self.RequestURL())
self.features.append(self.AnchorURL())
self.features.append(self.LinksInScriptTags())
self.features.append(self.ServerFormHandler())
self.features.append(self.InfoEmail())
self.features.append(self.AbnormalURL())
self.features.append(self.WebsiteForwarding())
self.features.append(self.StatusBarCust())
self.features.append(self.DisableRightClick())
self.features.append(self.UsingPopupWindow())
self.features.append(self.IframeRedirection())
self.features.append(self.AgeofDomain())
Type here to search
17°C Clear
12:41 AM
11/19/2022


File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - feature.py
over all project feature.py
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
self.features.append(self.UsingPopupWindow())
self.features.append(self.IframeRedirection())
self.features.append(self.AgeofDomain())
self.features.append(self.DNSRecording())
self.features.append(self.WebsiteTraffic())
self.features.append(self.PageRank())
self.features.append(self.GoogleIndex())
self.features.append(self.LinksPointingToPage())
self.features.append(self.StatsReport())
# 1.UsingIp
def UsingIp(self):
try:
ipaddress.ip_address(self.url)
return -1
except:
return 1
# 2.longUrl
def longUrl(self):
if len(self.url) < 54:
return 1
if len(self.url) >= 54 and len(self.url) <= 75:
return 0
return -1
1:1 LF UTF-8 4 spaces Python 3.9 (phising)
Type here to search
17°C Clear
ENG IN 12:41 AM 11/19/2022

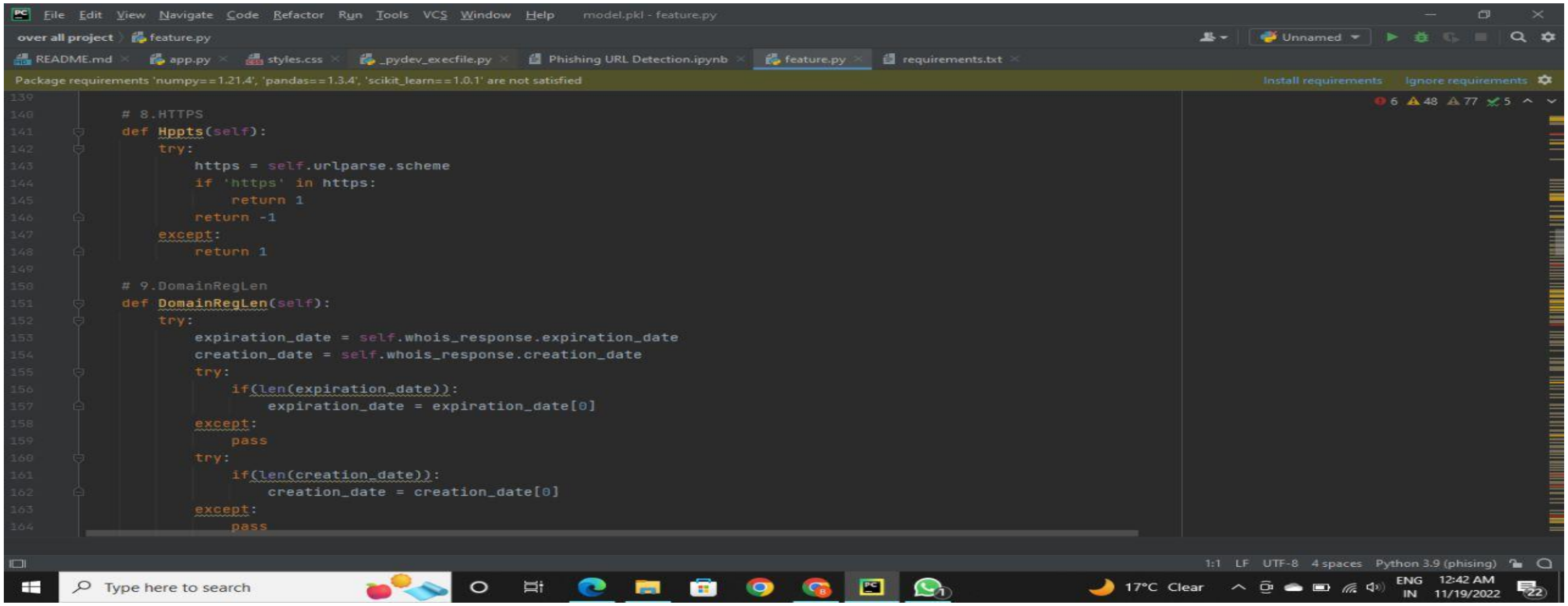

File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - feature.py
over all project feature.py
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
Package requirements 'numpy==1.21.4', 'pandas==1.3.4', 'scikit_learn==1.0.1' are not satisfied
Install requirements Ignore requirements
self.features.append(self.UsingIp())
self.features.append(self.longUrl())
self.features.append(self.shortUrl())
self.features.append(self.symbol())
self.features.append(self.redirecting())
self.features.append(self.prefixSuffix())
self.features.append(self.SubDomains())
self.features.append(self.Hppts())
self.features.append(self.DomainRegLen())
self.features.append(self.Favicon())
self.features.append(self.NonStdPort())
self.features.append(self.HTTPSDomainURL())
self.features.append(self.RequestURL())
self.features.append(self.AnchorURL())
self.features.append(self.LinksInScriptTags())
self.features.append(self.ServerFormHandler())
self.features.append(self.InfoEmail())
self.features.append(self.AbnormalURL())
self.features.append(self.WebsiteForwarding())
self.features.append(self.StatusBarCust())
self.features.append(self.DisableRightClick())
self.features.append(self.UsingPopupWindow())
1:1 LF UTF-8 4 spaces Python 3.9 (phising)
Type here to search
17°C Clear
ENG IN 12:42 AM 11/19/2022

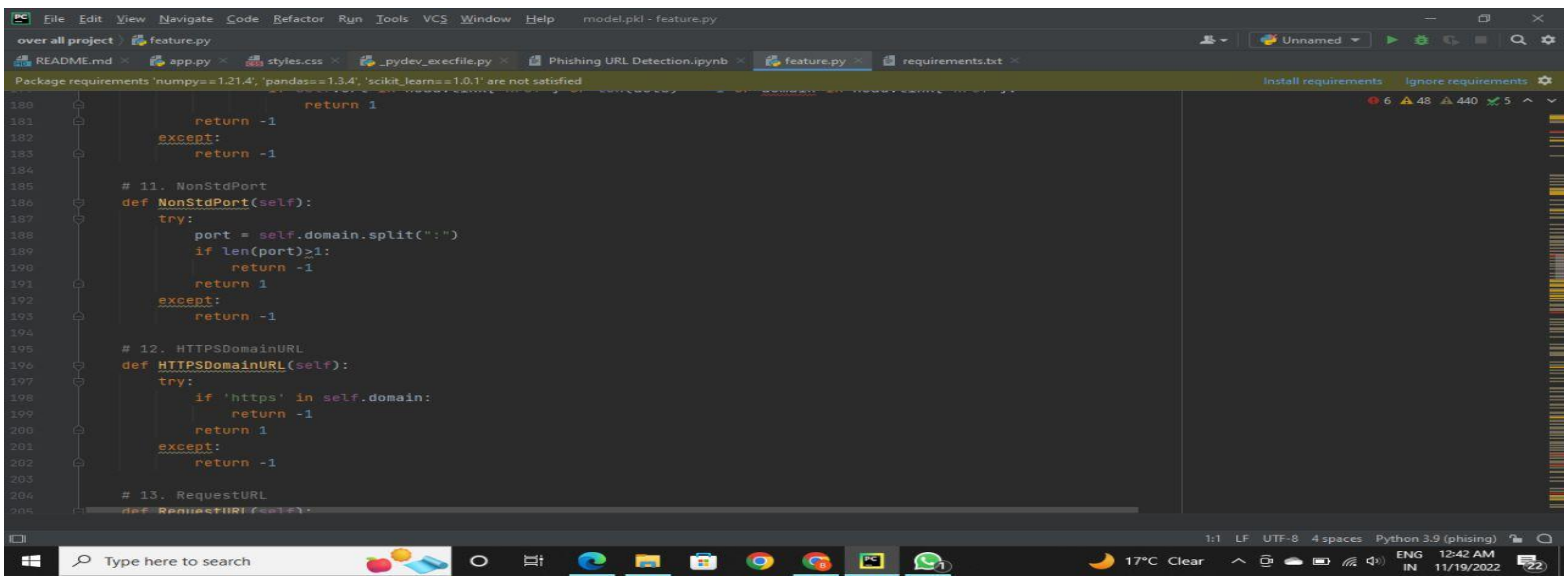


File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - feature.py
over all project feature.py
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
Package requirements 'numpy==1.21.4', 'pandas==1.3.4', 'scikit_learn==1.0.1' are not satisfied
Install requirements Ignore requirements
# 5.Redirecting//
def redirecting(self):
if self.url.rfind('//')>6:
return -1
return 1
# 6.prefixSuffix
def prefixSuffix(self):
try:
match = re.findall('\-', self.domain)
if match:
return -1
return 1
except:
return -1
# 7.SubDomains
def SubDomains(self):
dot_count = len(re.findall("\.", self.url))
if dot_count == 1:
return 1
elif dot_count == 2:
return 0
return -1
Type here to search
17°C Clear
12:42 AM
11/19/2022



File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - feature.py
over all project feature.py
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
Package requirements 'numpy==1.21.4', 'pandas==1.3.4', 'scikit_learn==1.0.1' are not satisfied
Install requirements Ignore requirements
# 1.UsingIp
def UsingIp(self):
try:
ipaddress.ip_address(self.url)
return -1
except:
return 1
# 2.longUrl
def longUrl(self):
if len(self.url) < 54:
return 1
if len(self.url) >= 54 and len(self.url) <= 75:
return 0
return -1
# 3.shortUrl
def shortUrl(self):
match = re.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|'
Type here to search
17°C Clear
12:42 AM
11/19/2022


File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - feature.py
over all project feature.py
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
Package requirements 'numpy==1.21.4', 'pandas==1.3.4', 'scikit_learn==1.0.1' are not satisfied
Install requirements Ignore requirements
# 8.HTTPS
def Hppts(self):
try:
https = self.urlparse.scheme
if 'https' in https:
return 1
return -1
except:
return 1
# 9.DomainRegLen
def DomainRegLen(self):
try:
expiration_date = self.whois_response.expiration_date
creation_date = self.whois_response.creation_date
try:
if(len(expiration_date)):
expiration_date = expiration_date[0]
except:
pass
try:
if(len(creation_date)):
creation_date = creation_date[0]
except:
1:1 LF UTF-8 4 spaces Python 3.9 (phising)
Type here to search
17°C Clear
ENG IN 12:42 AM 11/19/2022


File Edit View Navigate Code Refactor Run Tools VCS Window Help model.pkl - feature.py
over all project feature.py
README.md app.py styles.css _pydev_execfile.py Phishing URL Detection.ipynb feature.py requirements.txt
Package requirements 'numpy==1.21.4', 'pandas==1.3.4', 'scikit_learn==1.0.1' are not satisfied
Install requirements Ignore requirements
return 1
return -1
except:
return -1
# 11. NonStdPort
def NonStdPort(self):
try:
port = self.domain.split(":")
if len(port)>1:
return -1
return 1
except:
return -1
# 12. HTTPSDomainURL
def HTTPSDomainURL(self):
try:
if 'https' in self.domain:
return -1
return 1
except:
return -1
13. RequestURL
1:1 LF UTF-8 4 spaces Python 3.9 (phising)
Type here to search
17°C Clear
ENG IN 12:42 AM 11/19/2022

```python
    # 15. RequestURL
    def RequestURL(self):
        try:
            for img in self.soup.find_all('img', src=True):
                dots = [x.start(0) for x in re.finditer('\.', img['src'])]
                if self.url in img['src'] or self.domain in img['src'] or len(dots) == 1:
                    success = success + 1
                i = i+1

            for audio in self.soup.find_all('audio', src=True):
                dots = [x.start(0) for x in re.finditer('\.', audio['src'])]
                if self.url in audio['src'] or self.domain in audio['src'] or len(dots) == 1:
                    success = success + 1
                i = i+1

            for embed in self.soup.find_all('embed', src=True):
                dots = [x.start(0) for x in re.finditer('\.', embed['src'])]
                if self.url in embed['src'] or self.domain in embed['src'] or len(dots) == 1:
                    success = success + 1
                i = i+1

            for iframe in self.soup.find_all('iframe', src=True):
                dots = [x.start(0) for x in re.finditer('\.', iframe['src'])]
                if self.url in iframe['src'] or self.domain in iframe['src'] or len(dots) == 1:
                    success = success + 1
                i = i+1
```

TESTING

TEST CASES

	Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status
6	LoginPage_TC_OO1	Functional	Home Page	user can access this page like they can enter URL and can click the buttons.		1.Enter the phishing URLs 2.Click here to find out the url	https://drive.google.com/drive/folders/1CDFC0g7ryUh8RF-esIICZGL3JLF74SYK?usp=share_link	some massage display like the url is either safe or not safe with persentage.	Working as expected	Pass
7										

User Acceptance Testing

1.Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] 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	14	7	4	3	28
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	15	5	3	20	43
Not Reproduced	0	0	1	0	1
Skipped	0	2	1	1	4
Won't Fix	0	1	1	2	4
Totals	32	18	13	27	90

3.Test Case Analysis

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

Section	Total Cases	Not Tested	Fa il	Pas s
Print Engine	6	0	0	6
Client Application	40	0	2	38
Security	4	0	0	4
Outsource Shipping	3	1	0	2
Exception Reporting	7	1	0	6
Final Report Output	5	0	0	5
Version Control	3	0	1	2

RESULTS

Performance Matrics

Model Performance Testing:

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

S.No.	Parameter	Values
1.	Metrics	**Random Forest Classifier Accuracy score-96.653**
2.	Tune the Model	**Hyperparameter Tuning - Validation Method -**

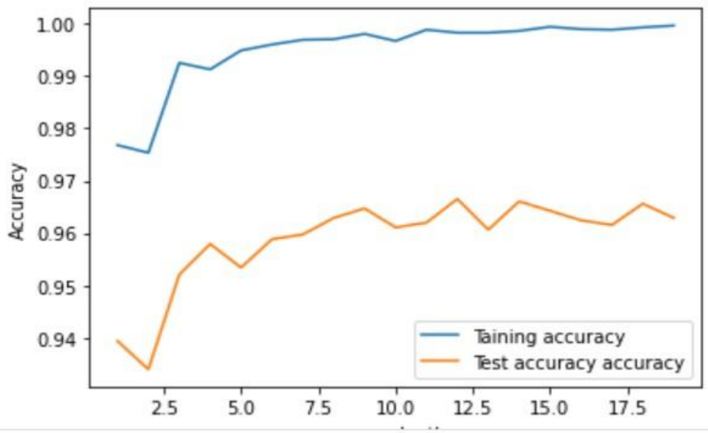
14. METRICS

Classification Report:

```
[ ] #classification report of Randomm Forest model
    print(metrics.classification_report(y_test,y_test_rf))

              precision    recall  f1-score   support

          -1       0.98      0.95      0.96      1014
           1       0.96      0.98      0.97      1197

    accuracy                           0.97      2211
   macro avg       0.97      0.97      0.97      2211
weighted avg       0.97      0.97      0.97      2211
```

Peformance

```
training_accuracy=[]
test_accuracy=[]
depth=range(1,20)
for n in depth:
  rf_test=RandomForestClassifier(n_estimators=n)
  rf_test.fit(x_train,y_train)
  training_accuracy.append(rf_test.score(x_train,y_train))
  test_accuracy.append(rf_test.score(x_test,y_test))
plt.figure(figsize=None)
plt.plot(depth,training_accuracy,label="Taining accuracy")
plt.plot(depth,test_accuracy,label="Test accuracy accuracy")
plt.ylabel("Accuracy")
plt.xlabel("max_depth")
plt.legend();
```



✓ 1s completed at 1:37 PM

:

2.Tune the model

[ ]

	ML Model	Accuracy	f1_score	Recall	Precision
0	Logistic Regression	91.814	92.567	94.496	94.496
1	Random Forest	96.653	96.942	100.000	100.000
2	XgbClassifier	94.754	95.207	96.714	96.714
3	Decision tree	95.206	95.605	100.000	100.000

[ ]

```
sorted_result=result.sort_values(by=['Accuracy', 'f1_score'],ascending=False).reset_index(drop=True)
sorted_result
```

	ML Model	Accuracy	f1_score	Recall	Precision
0	Random Forest	96.653	96.942	100.000	100.000
1	Decision tree	95.206	95.605	100.000	100.000
2	XgbClassifier	94.754	95.207	96.714	96.714
3	Logistic Regression	91.814	92.567	94.496	94.496

ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- **Increases user alertness to phishing risks** Whenever the user navigates into the website and provide the URL of the website that needs to be verified for legitimacy, the system detects phishing sites by applying a machine learning algorithm which implements classification algorithms and techniques to extract the phishing datasets criteria to classify their legitimacy which in turn helps the customers to eliminate the risks of cyber threat and protect their valuable corporate or personal data.

- **Users will also be able to pose any query to the admin through the report page designed** Our system is also provided with an option for the clients to report to the administrator which helps them to ask their questions significantly improving their experience on our site.

DISADVANTAGES:

- Not a generalized model
- Huge number of rules
- Needs feed continuously

CHAPTER 11

CONCLUSION

Phishing detection is now an area of great interest among the researchers due to its significance in protecting privacy and providing security. There are many methods to perform phishing detection. Our system aims to enhance the detection method to detect phishing websites using machine learning technology. We achieved a high detection accuracy, and the results show that the classifiers give better performance when we use more data as training data.

In future, hybrid technology will be implemented to detect

phishing websites more accurately.

FUTURE SCOPE

In future we intend to build an add-ons for our system and if we get a structured dataset of phishing, we can perform phishing detection much faster than any other technique. We can also use a combination of any two or more classifiers to get maximum accuracy. We plan to explore various phishing techniques which use Network based features, Content based features, Webpage based features and HTML and JavaScript features of web pages which will improve the performance of the system. In particular, we extract features from URLs and pass it through the various classifiers.

APPENDIX

Source Code