Enhancing Online Security: A Machine Learning Approach to Detect Phishing Websites

A Internship Project Report

Submitted to the Faculty of Engineering of
SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE
GUDLAVALLERU

In partial fulfillment of the requirements for the award of the Degree of

BACHELOR OF TECHNOLOGY
In
ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

By

P. POOJITHA
(20481A5438)

G. GRACE APOORVA
(20481A5414)

S. SUBRAMANYAM RAJU
(21485A5406)

T. SAI KRISHNA
(20481A5452)

Under the guidance of
Mr. K. ASHOK REDDY, M.Tech,(Ph.D)
Assistant Professor, Department of AI&DS

DEPARTMENT OF
ARTIFICIAL INTELLIGENCE AND DATA SCIENCE
SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE
(An Autonomous Institute Permanently affiliated to JNTUK)
Seshadri Rao Knowledge Village
GUDLAVALLERU – 521356
ANDHRA PRADESH

2023-2024

SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE

(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)
SESHADRI RAO KNOWLEDGE VILLAGE, GUDLAVALLERU

DEPARTMENT OF
ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

CERTIFICATE

This is to certify that the project report entitled **"Enhancing Online Security: A Machine Learning Approach to Detect Phishing Websites"** is a bonafide record of work carried out by **P. Poojitha (20481A5438), G. Grace Apoorva (20481A5414), S. Subramanyam Raju (21485A5406), T. Sai Krishna (20481A5452)** under the guidance and supervision of **Mr. K. ASHOK REDDY** in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Artificial Intelligence and Data Science of Jawaharlal Nehru Technological University Kakinada, Kakinada during the academic year 2023-24.

Project Guide
(Mr. K. ASHOK REDDY)

Head of the Department
(Dr. K. SRINIVAS)

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

We would like to express our deep sense of gratitude and sincere thanks to **Mr. K. ASHOK REDDY, M. Tech**, **Ph.D.** Department of Artificial Intelligence and Data Science for his constant guidance, supervision, and motivation in completing the project work.

We feel elated to express our floral gratitude and sincere thanks to **Dr. K. Srinivas**, Head of the Department of Artificial Intelligence and Data Science for his encouragement all the way during the analysis of the project. His annotations, insinuations, and criticisms are the key to the successful completion of the project work.

We would like to take this opportunity to thank our beloved principal **Dr. B. Karuna Kumar** for providing great support for us in completing our project and giving us the opportunity to do the project.

Our Special thanks to the faculty of our department and the programmers of our computer lab. Finally, we thank our family members, non-teaching staff, and our friends, who had directly or indirectly helped and supported us in completing our project in time.

Team members

P. Poojitha (20481A5438)
G. Grace Apoorva (20481A5414)
S. Subramanyam Raju (21485A5406)
T. Sai Krishna (20481A5452)

ABSTRACT

The escalating threat of phishing websites poses grave risks to online users, jeopardizing sensitive information and security. To counter this menace, the " Enhancing Online Security: A Machine Learning Approach to Detect Phishing Websites" project has been conceived. This research endeavors to develop an intelligent system employing advanced machine learning algorithms to proactively detect and block users from phishing websites in real time.

The primary objective of this project is to create a robust and adaptable detection mechanism by scrutinizing essential website attributes, including URL and domain identity, security features, and encryption criteria. Leveraging classification algorithms, the system swiftly distinguishes between legitimate and malicious websites, ensuring users' safety during online interactions.

Through continuous improvement and adaptation to evolving phishing tactics, this project strives to stay ahead of cyber threats and create a safer digital environment for users worldwide. The comprehensive approach employed in this endeavor offers a promising solution to combat web phishing, marking a significant stride toward bolstering cybersecurity across the digital landscape.

INDEX

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

1.1.1 Overview

The URL-Based Phishing Detection Using Machine Learning project focuses on countering the growing threat of phishing websites, which pose significant risks to users' sensitive information and security. With the exponential growth of online transactions and e-banking, it has become essential to develop intelligent systems that can effectively identify and thwart these deceptive websites.

Phishing websites impersonate legitimate entities, tricking users into divulging confidential data like usernames, passwords, and credit card details. As such, they pose a serious challenge to web services' security on the Internet. This project endeavors to address this issue by utilizing machine learning algorithms for proactive detection.

The primary objective of this project is to design a robust and flexible system capable of accurately detecting phishing websites in real-time. By harnessing the power of classification algorithms, the system can analyze critical features like URL and domain identity, along with security and encryption criteria, to distinguish between legitimate and malicious websites.

Through the application of data mining algorithms, the system can identify e-banking phishing websites during online transactions. Doing so empowers users to make informed decisions and avoid falling prey to phishing scams. Additionally, the system contributes to broader cybersecurity efforts by safeguarding organizations and individuals from potential losses and information theft.

By combining machine learning expertise and analyzing essential website attributes, this project seeks to create a safer online environment for users worldwide. The comprehensive approach employed in this endeavor offers a promising solution to combat web phishing and protect users from the risks posed by deceptive websites. Through continuous improvement and adaptation to evolving phishing tactics, this project strives to stay ahead of cyber threats and bolster cybersecurity across the digital landscape.

1.1.2 Purpose

The use of this project. What can be achieved using this?

The URL-Based Phishing Detection Using Machine Learning project aims to enhance online security and protect users from falling victim to phishing scams. Phishing websites have become increasingly sophisticated, posing a significant risk to individuals and organizations alike. The primary goal of this research is to establish a smart, effective system that can reliably identify and block access to such fraudulent websites.

This project aims to create a robust and adaptable detection mechanism by employing machine learning algorithms and classification techniques. The system focuses on analyzing crucial website attributes, including URL and domain identity, security features, and encryption criteria. Through this analysis, it can swiftly distinguish

between legitimate websites and phishing websites, thus ensuring users' safety during their online interactions.

The project's utility lies in its ability to proactively protect users' sensitive information, such as login credentials and financial data, from falling into the hands of cybercriminals. By detecting phishing websites in real-time, the system empowers users to make informed decisions and steer clear of potential dangers while browsing the internet or conducting e-banking transactions.

Furthermore, this project serves as a critical contribution to the broader cybersecurity landscape. Detecting and neutralizing phishing websites, it helps organizations avoid the risks associated with data breaches and financial fraud. The system's effectiveness in identifying malicious websites can mitigate the reputational damage and financial losses that institutions may otherwise face.

1.2 PROBLEM STATEMENT

E-banking in the digital era involves transferring a lot of important information such as account names, PIN numbers, and ATM card details among other personal details. This convenience, however, comes with a phishing website menace- malicious sites that imitate genuine systems and trick people into revealing private data. The term "web phishing" encapsulates this perilous landscape that poses significant security threats to online services.

Web phishing, in its deceitful pursuit of private data, not only jeopardizes the personal security of individuals but also leads to extensive information disclosure and financial losses. Moreover, large organizations are not immune to these subterfuges, potentially entangling themselves in multifaceted scams.

To address this escalating challenge, our Guided Project centers on the application of cutting-edge machine learning algorithms for the detection of phishing websites, with a specific focus on those targeting e-banking platforms.

Our endeavor hinges on the development of an intelligent, adaptable, and highly effective system, driven by classification algorithms. Leveraging these algorithms and techniques, we meticulously assess and extract criteria from phishing datasets to discern the legitimacy of websites. Critical attributes, such as URL and domain identity, security features, and encryption standards, play a pivotal role in determining the final phishing detection rate.

Importantly, our system operates in real time, especially during online transactions involving e-banking. Here, a data mining algorithm is employed to swiftly evaluate the authenticity of e-banking websites. It endeavors to ascertain whether the user's chosen platform is a genuine entity or a phishing website, thereby empowering individuals to make informed decisions and navigate the digital landscape securely.

Essentially, this is what our project entails as a strong and preventative web phishing defense in light of current developments. Through the utilization of machine learning and data analytics, we aim to shield individual users and large firms from the potential incurrence of loss through the theft of money or the leaking of confidential information.

1.3 EXISTING SYSTEM

1.3.1 Existing problem

The proliferation of phishing websites presents a significant cybersecurity challenge in the digital era. Phishing is a form of cybercrime where attackers create deceptive websites that mimic legitimate entities, such as e-banking platforms or online shopping portals. These malicious websites aim to steal sensitive information from unsuspecting users, including usernames, passwords, and financial data. The consequences of falling victim to phishing attacks can be severe, leading to identity theft, financial losses, and reputational damage.

Traditional cybersecurity measures, such as firewalls and antivirus software, have proven insufficient in effectively detecting phishing websites. Cybercriminals continuously evolve their tactics, making it difficult for static rule-based approaches to keep up with new variations of phishing attacks. As a result, there is a pressing need for more intelligent and adaptive systems to counter this ever-growing threat.

1.3.2 Existing Systems

Researchers and cybersecurity experts have made significant efforts to combat phishing through various approaches and methods. Some of the existing techniques for phishing detection include:

1. Rule-Based: These methods involve using predefined rules to identify phishing websites based on specific patterns and characteristics. While they can be effective for known phishing URLs, they struggle to handle newly emerging phishing tactics.
2. Blacklist and Whitelist Filtering: This method maintains lists of known phishing websites (blacklist) and legitimate websites (whitelist). URLs are checked against these lists, but it is challenging to keep the blacklist up-to-date and misses new phishing sites not on the list.
3. Machine Learning-Based Approaches: Machine learning algorithms, such as decision trees, support vector machines, and random forests, have been applied to phishing detection. These algorithms can learn from historical data to identify patterns and features indicative of phishing websites. Machine learning models offer greater adaptability to evolving phishing tactics.
4. Phishing Website Reporting and Analysis: Collaborative efforts involve reporting and analyzing suspected phishing websites by users or security researchers. This approach relies on crowdsourced intelligence to update blacklists and improve detection.
5. URL Analysis and Feature Extraction: By examining URL properties, such as domain length, presence of the '@' symbol, or redirection behavior, researchers have developed feature-based methods for phishing detection.

While these approaches have provided valuable insights, each has its limitations. Rule-based and blacklist filtering methods struggle with emerging threats, while manual reporting can be time-consuming and may not keep up with the volume of new phishing websites. Machine learning approaches show promise, but their effectiveness largely depends on the quality and diversity of training data and the selection of relevant features.

This project aims to leverage machine learning algorithms and intelligent feature extraction techniques to create a dynamic and robust phishing detection system. By incorporating a comprehensive set of URL-based features and employing various classification models, the project seeks to achieve higher accuracy and responsiveness in identifying phishing websites. Through a combination of data-driven analysis and adaptive algorithms, this research contributes to strengthening cybersecurity measures and fostering a safer digital landscape.

1.4 DISADVANTAGES

Disadvantages of URL-based Phishing Detection using Machine Learning:

1. Data Imbalance: The dataset used to train the machine learning model may be imbalanced, with a significantly higher number of legitimate URLs compared to phishing URLs. This can lead to biased model performance.
2. Data Privacy Concerns: The use of URLs for phishing detection might raise data privacy concerns, as URLs may contain sensitive information about users' browsing behavior and habits.

It's important to note that while machine learning-based phishing detection can be effective, it should be used in conjunction with other security measures, as no single method can guarantee 100% protection against all phishing threats. Additionally, the success of the proposed solution depends on the quality of the dataset, the choice of features, and the rigor of model evaluation and testing. Regular updates and monitoring are essential to maintain the accuracy and effectiveness of the solution.

1.5 PROPOSED SYSTEM

The proposed solution aims to create an efficient and user-friendly Phishing Detection System that leverages machine learning algorithms and intelligent feature extraction techniques. The primary goal is to empower users to verify the legitimacy of URLs and protect themselves from falling victim to phishing attacks. The solution follows a step-by-step process to ensure ease of use and effectiveness:

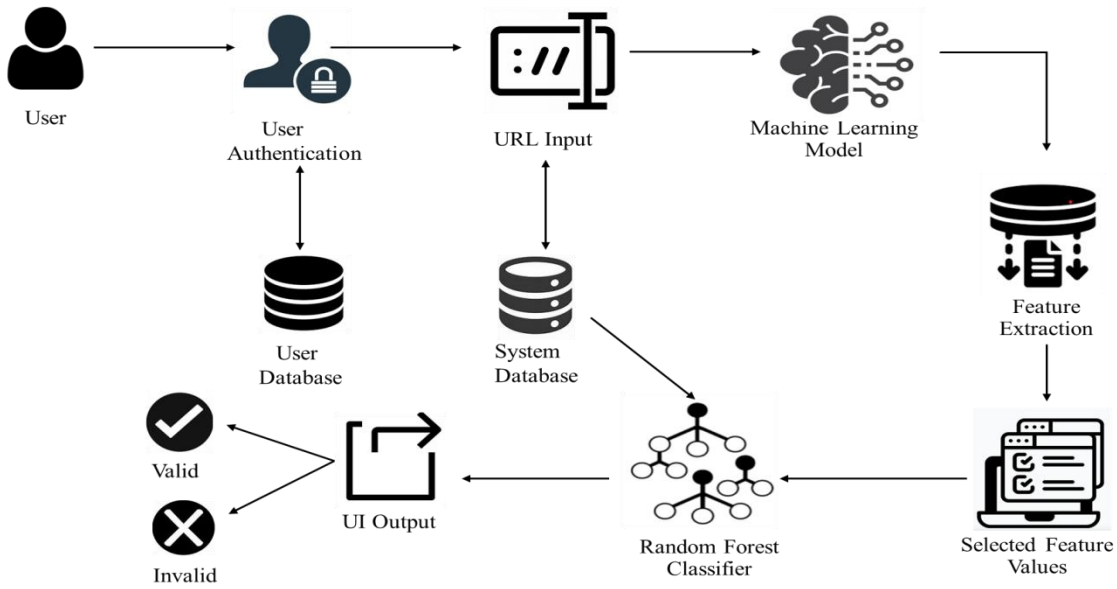

Fig 1: Shows the prototype of the proposed system

From fig 1, the proposed solution can be explained as follows

1. User Login:

To access the Phishing Detection System and its features, the user is required to log in securely using their unique credentials. User authentication ensures that only authorized individuals can utilize the system, maintaining privacy and preventing misuse.

2. Accessing the Website:

Once logged in, the user gains access to the Phishing Detection website, where they can utilize the "Let's Check" feature. This feature serves as the entry point for URL verification and phishing detection.

3. URL Input:

In the search bar provided on the webpage, the user is prompted to enter the URL that they wish to verify. The URL input can be from any online source, such as an e-banking platform, online store, or email link.

4. Verification Process:

After entering the URL of interest, the user proceeds to initiate the verification process by clicking the "Check" button. At this stage, the URL undergoes a series of checks and evaluations to determine whether it is legitimate or potentially a phishing website.

5. Machine Learning Model Evaluation:

Phishing Detection System lies in the integration of advanced machine learning models. The URL input provided by the user is passed through these models, which have been trained on a vast dataset of legitimate and phishing URLs.

For this system the following machine learning algorithms are considered to determine the best suitable model with high accuracy and performance

a. Decision Tree
b. XG Boost
c. SVM Classifier
d. Logistic Regression
e. KNN (K-Nearest Neighbors)
f. Random Forest

g. Naïve Bayes

6. Result Display:
Based on the evaluation performed by the machine learning models, the system generates a clear and concise result. The user is promptly presented with the outcome, indicating whether the given URL is legitimate or if it exhibits characteristics consistent with phishing websites.

The proposed solution capitalizes on the following key components to achieve accurate and reliable phishing detection:

a. Comprehensive Feature Set:
The machine-learning models are designed to extract and analyze a wide range of features from the URL input. These features include URL length, presence of special characters, domain characteristics, SSL certificate status, web traffic, and more. By considering a diverse set of features, the system can discern subtle patterns indicative of phishing attempts. The feature set is categorized to four categories based on which the attributes or functionalities work, they are as follows:

1. Address Bar-based Features
2. Domain-based Features
3. HTML & Javascript based Features
4. Other miscellaneous features

On a whole thirty comprehensive features each of different domains are considered to train and test the model across different attributes and domains.

b. Machine Learning Algorithms:
After employing a variety of machine learning algorithms, including Decision Trees, XG Boost, SVM Classifier, Logistic Regression, KNN, Random Forest, and Naïve Bayes. Each algorithm brings its unique strengths, allowing the system to adapt and learn from past data to recognize new phishing tactics effectively.

c. Regular Model Updates:
To ensure the system remains up-to-date with the rapidly evolving landscape of phishing attacks, our models are regularly updated. By incorporating the latest data and emerging threat intelligence, the system remains highly resilient against new phishing schemes.

d. User-Friendly Interface:
The user interface of the Phishing Detection System is designed with simplicity and ease of use in mind. Users, regardless of their technical expertise, can effortlessly navigate through the process of URL verification and promptly receive the results.

By combining advanced machine learning techniques, a comprehensive set of features, and an intuitive user interface, our proposed Phishing Detection System stands as a robust defense against the ever-present threat of phishing attacks. Through this solution, we aim to empower users with the knowledge and tools to protect their sensitive information and enhance the overall security of online transactions and communication.

1.6 ADVANTAGES

The advantages of the proposed system are as follows:
1. Automated Detection: Machine learning models can automatically analyze and detect phishing URLs without human intervention, enabling real-time and efficient detection.

2. Scalability: Machine learning algorithms can scale to handle a large number of URLs, making them suitable for analyzing massive datasets of URLs.
3. Adaptability: Machine learning models can adapt to new and evolving phishing techniques, improving their accuracy over time as they learn from new data.
4. High Accuracy: With proper feature engineering and model selection, machine learning models can achieve high accuracy in distinguishing between phishing and legitimate URLs.
5. Real-time Protection: The automated nature of machine learning allows for real-time protection, instantly flagging and blocking potential phishing URLs before users interact with them.
6. Reduced False Positives: Well-tuned machine learning models can help reduce false positives by learning to differentiate between legitimate and suspicious URLs based on patterns and features.
7. Centralized Management: A centralized machine learning-based phishing detection system can be easily managed and updated to provide consistent protection across multiple platforms and devices.

CHAPTER 2

REQUIREMENT ANALYSIS

2.1 FUNTIONAL REQUIREMENTS

For the development and implementation of an efficient phishing website-detection system, a rigorous requirement analysis should be conducted. Further on in this project, we analyzed both functional and non-functional requirements, as well as software and hardware specification for it to be fulfilled.

The functional requirements of our phishing detection system are as follows:

a. Website Data Extraction: The system should be capable of extracting data from websites, particularly focusing on critical attributes such as URL, domain identity, security features, and encryption standards.

b. Classification Algorithms: Implementation of classification algorithms to assess the legitimacy of websites based on extracted data.

c. Real-Time Detection: The system should operate in real-time, especially during online transactions, enabling prompt detection of e-banking phishing websites.

d. Data Mining Algorithm: Utilize a data mining algorithm to evaluate websites during online transactions and determine their authenticity.

e. User Notification: In the event of a detected phishing website, the system should promptly notify the user, enabling them to make informed decisions and avoid potential risks.

2.2 NON FUNTIONAL REQUIREMENTS

The non-functional requirements of our system include:

a. Scalability: The system should be able to scale to accommodate a growing dataset of phishing websites and user transactions.

b. Reliability: The system must be highly reliable, ensuring minimal false positives and false negatives in phishing website detection.

c. Performance: The system should achieve the highest efficiency with regard to speed and precision, enabling quick and reliable detection of phishing sites.

d. Security: It should be noted that the system ought to have maximum security in order to safeguard the user's confidential information, as well as preserve the data accuracy.

e. Usability: It should be intuitive and easy to use so as to ensure that many people can access it easily.

2.3 SOFTWARE REQUIREMENT SPECIFICATIONS

The software requirements for our system are as follows:

a. Operating System: It is a matter that requires the system to be able to work together with current operating systems including Windows, macOS, and Linux.

b. Programming Language: Use languages appropriate in machine learning and data analysis like PYTHON.

c. Machine Learning Libraries: Employ libraries like TensorFlow, Scikit-learn, or PyTorch.

d. Database Management: A database system is required to store and manage datasets of phishing websites.
e. User Interface: Develop a user-friendly interface to display detection results to the user.

2.4 HARDWARE SPECIFICATIONS

The hardware specifications for our system are as follows:
a. Processor: The system should run efficiently on modern processors, with multi-core support for faster data processing.
b. Memory (RAM): Adequate RAM is required to accommodate large datasets and ensure efficient machine learning model training.
c. Storage: Sufficient storage space is needed for datasets and system logs.
d. Internet Connectivity: Reliable internet connectivity is essential for real-time online transaction monitoring.
d. Display: A standard computer display is required for the user interface.

By adhering to these functional and non-functional requirements, along with the specified software and hardware specifications, our project aims to develop an intelligent and effective system for the proactive detection of phishing websites, particularly those targeting e-banking transactions, here by ensuring the safety and security of online users and organizations.

CHAPTER 3

DESIGN

3.1 SYSTEM ARCHITECTURE

The system architecture for URL-based phishing detection using machine learning is designed to provide a robust, scalable, and effective solution to safeguard users and organizations against phishing threats on the internet. This architecture encompasses several key components and layers, each serving a specific purpose in the overall system operation. Below is a detailed description of the system architecture:

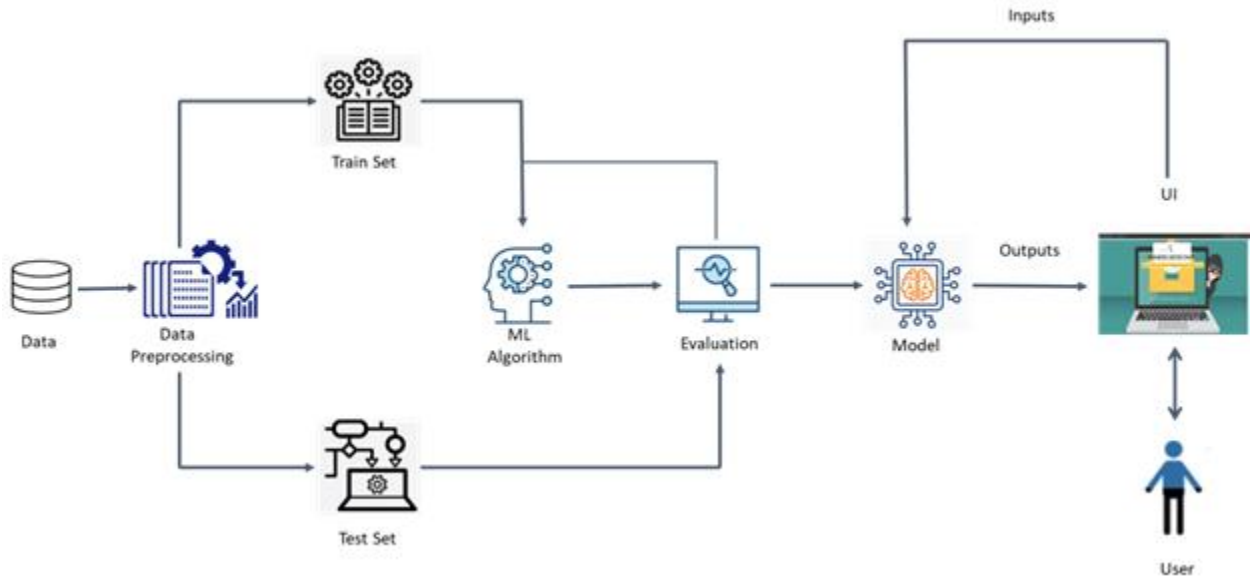

Fig 2: Pictorial Demonstration of System Architecture

1. Data Collection - Phishing and Legitimate Website URLs:

In this step, you collect the data needed for training and testing your machine-learning model to detect phishing websites. The data will consist of URLs from both phishing websites and legitimate websites. This data will serve as the basis for building and evaluating your model.

2. Data Handling - Missing and Null Values:

Before proceeding with feature extraction, it's important to handle any missing or null values in the collected data. Missing values can negatively affect the model's performance, so you may need to apply techniques like imputation (e.g., filling missing values with the mean or median) or removal of instances with missing data.

3. Feature Extraction:

Feature extraction involves transforming the raw URLs into a set of relevant features that can be used as input to the machine learning model. These features capture various characteristics of the URLs that may be indicative of phishing or legitimate websites. Examples of extracted features might include having_IP_Address (whether the URL has an IP address), URL_Length (length of the URL), Shortening_Service (presence of URL shortening service), etc.

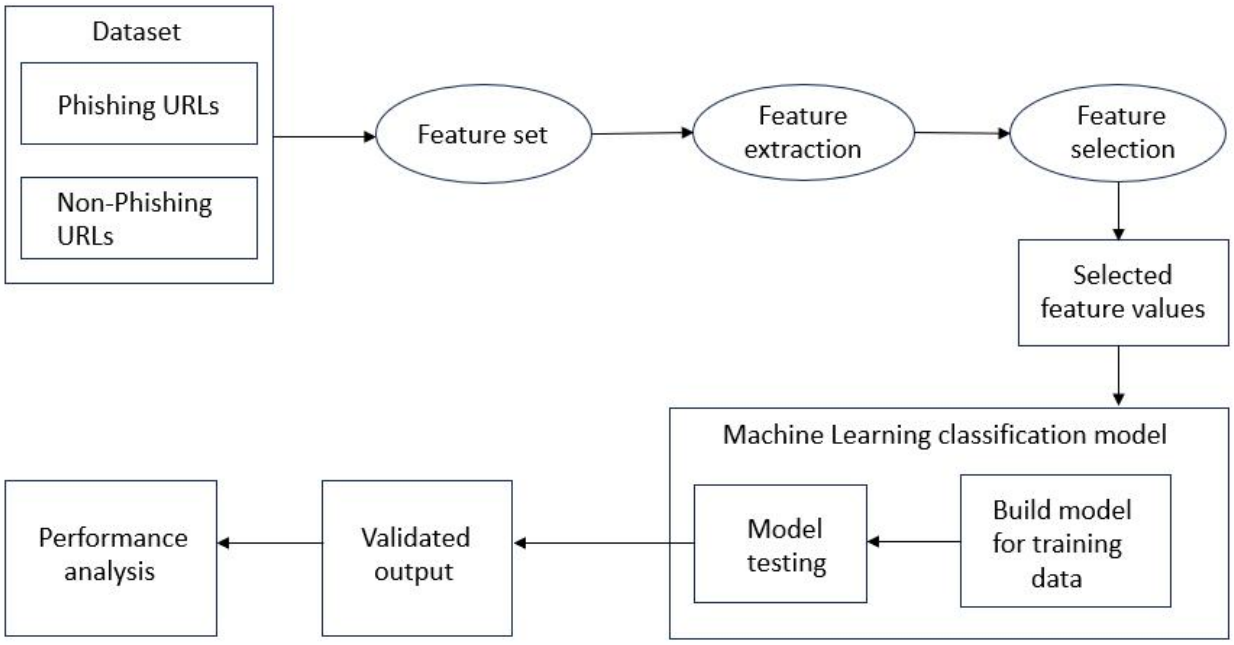

Fig 3: Block diagram of the system

4. Feature Selection:

Not all extracted features may be relevant or useful for the machine learning model. In this step, you perform feature selection to choose the most important features that have a significant impact on the model's predictive performance. Feature selection methods help reduce the dimensionality of the data and improve model efficiency and accuracy.

5. Machine Learning Classification Methods:

In this step, you apply machine learning classification methods to build a model that can distinguish between phishing and legitimate websites based on the selected feature values. The chosen method for this diagram is Random Forest, which is an ensemble learning technique based on decision trees.

i. Building Model using Random Forest:

Random Forest is a tree-based ensemble model that creates multiple decision trees during training and combines their predictions to make a final decision. The model learns from the selected features and the corresponding labels (phishing or legitimate) to create a robust classifier.

ii. Testing Data over the Model:

Once the Random Forest model is trained on the training data, you evaluate its performance on a separate testing dataset. The testing dataset contains URLs that the model has never seen before. By comparing the model's predictions to the true labels in the testing dataset, you can assess its accuracy, precision, recall, and other performance metrics.

3.2 UML DIAGRAMS

Unified Modeling Language (UML) diagrams play a crucial role in visually representing the various aspects of the system architecture and its functionalities. In the context of the URL-based phishing detection system, the following UML diagrams are utilized to provide a comprehensive understanding of the system's structure and operation:

3.2.1 USE CASE DIAGRAM

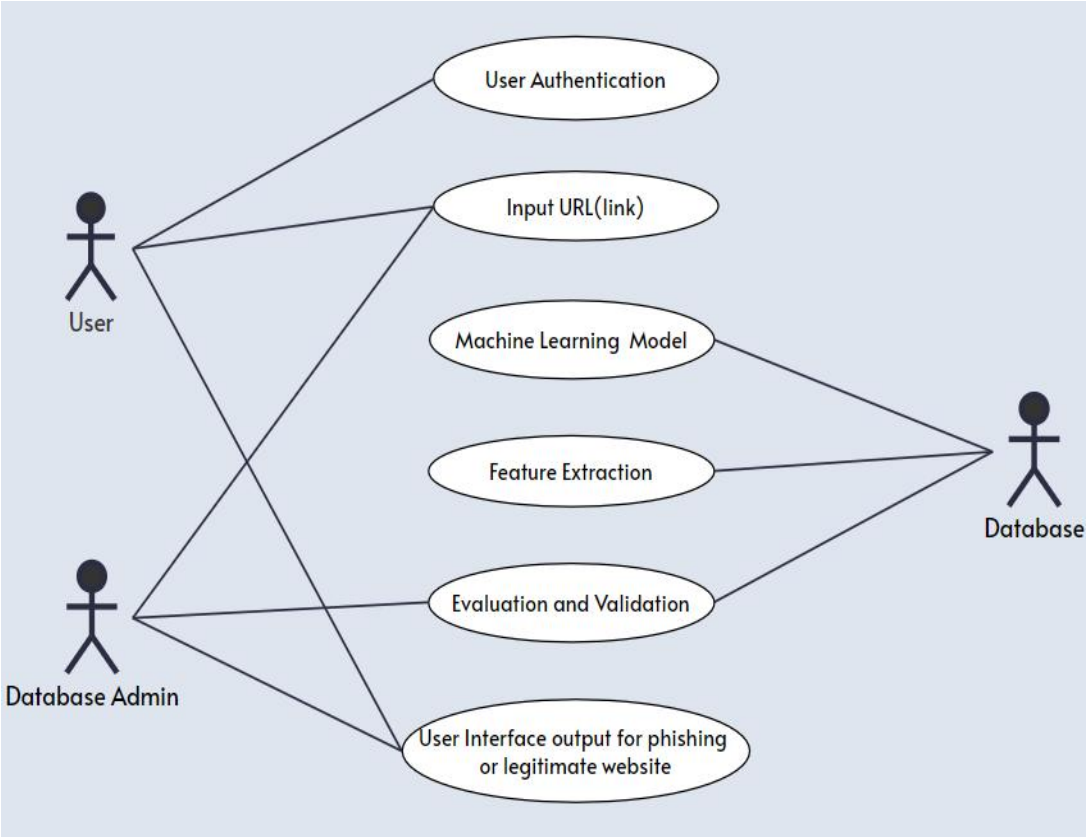

Fig 4: Use-case diagram of the system

The Use Case Diagram Fig 4, illustrates the high-level functionality of the system from a user's perspective. It showcases the interactions between different actors and the system itself. In the context of our URL-based phishing detection system, the Use Case Diagram encompasses actors like "User," "Administrator," and "Machine Learning Model," and demonstrates how they interact with the system. The "User" is engaged in activities such as "Monitor Online Transactions" and "Receive Phishing Alerts," while the "Administrator" manages system configurations and feedback. The "Machine Learning Model" is central to the system's core functionality, responsible for "Detecting Phishing Websites."

3.2.2 CLASS DIAGRAM

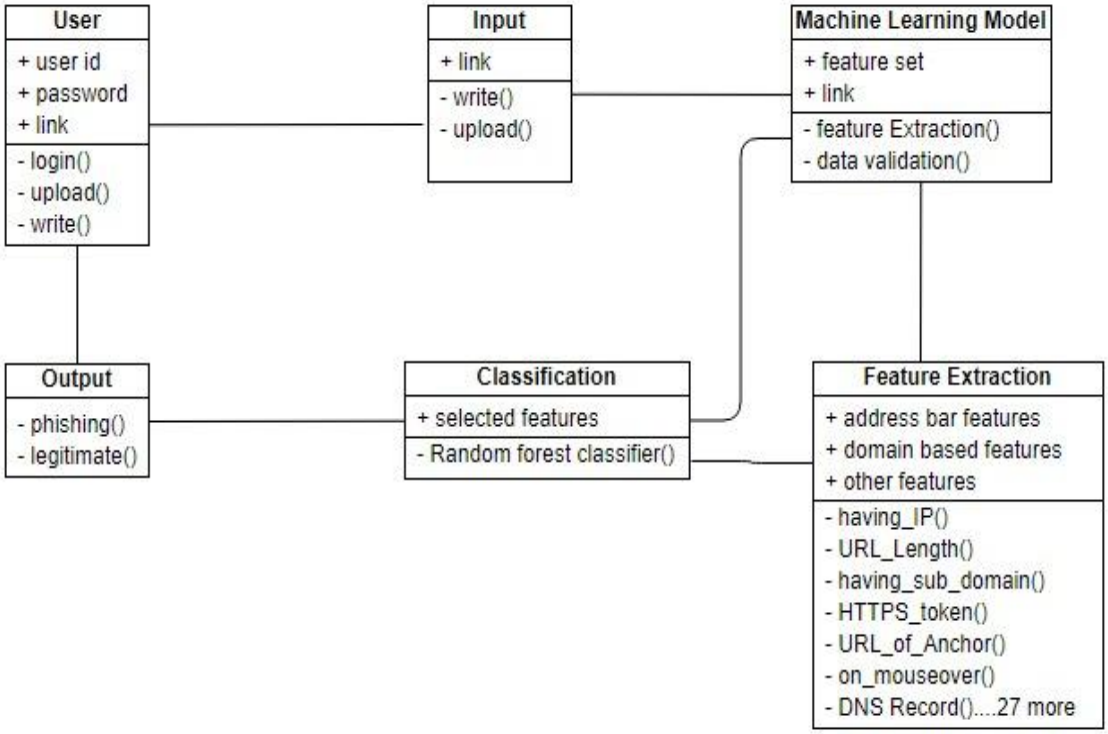

Fig 5: Class Diagram for the URL-based phishing detection system

The Class Diagram Fig 5, provides a structural view of the system by highlighting the essential classes, their attributes, and relationships. For our URL-based phishing detection system, this diagram encompasses classes such as "User," "Input," "PhishingDetectionModel," "Database," and "DataPreprocessor," "Output." It illustrates their attributes and associations, allowing a clear depiction of how data flows between these classes. For instance, the "User" class may have attributes like "Username" and "History," while it has associations with the "Model" class, representing the user's interaction with the system and the model associated with the "Feature Extraction" with different attributes and functionalities that in return connected to "Classification" in which Random Forest Classifier is used for the selected feature set. The "Output" which is connected to the User Interface that is directly connected to the user gives the resultant output of the functionalities of phishing or legitimate websites.

3.2.3 ACTIVITY DIAGRAM

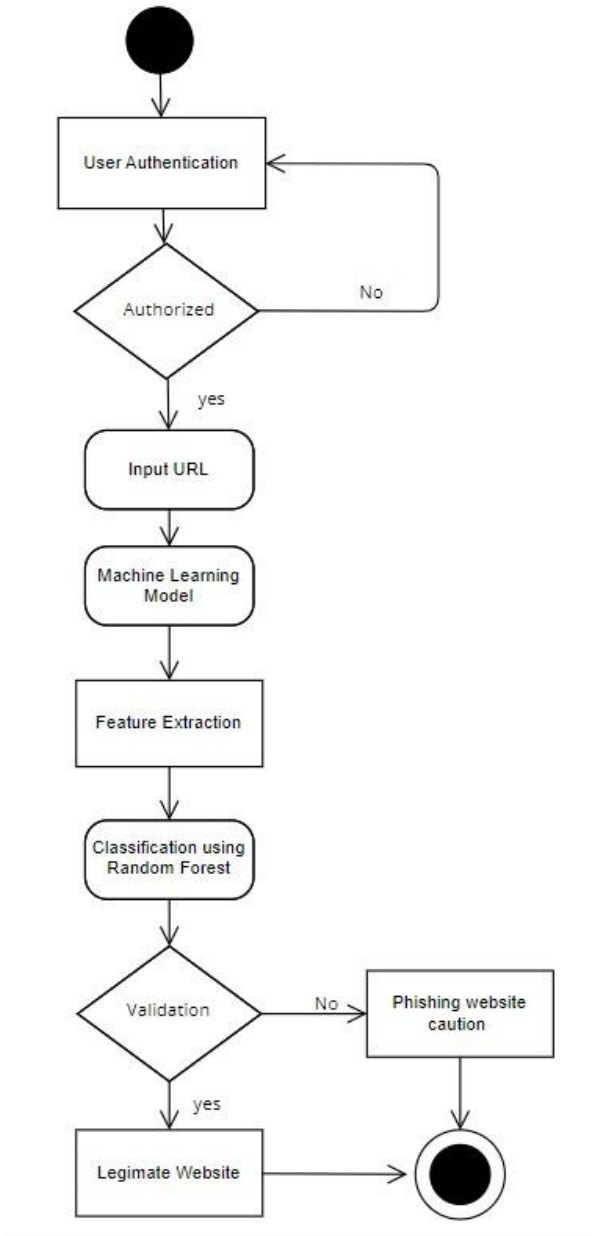

Fig 6: Activity Diagram of the system

The Activity Diagram Fig 6, focuses on the system's workflow, illustrating the sequence of activities and actions that occur within the system. In the context of our system, the Activity Diagram captures the sequence of actions in "Real-Time Phishing Detection." It begins with "Online Transaction Monitoring," followed by "Data Extraction," "Data Preprocessing,"

"Machine Learning Classification," and ends with "User Notification." Each activity represents a step in the phishing detection process, providing a visual representation of the system's workflow.

3.2.4 SEQUENCE DIAGRAM

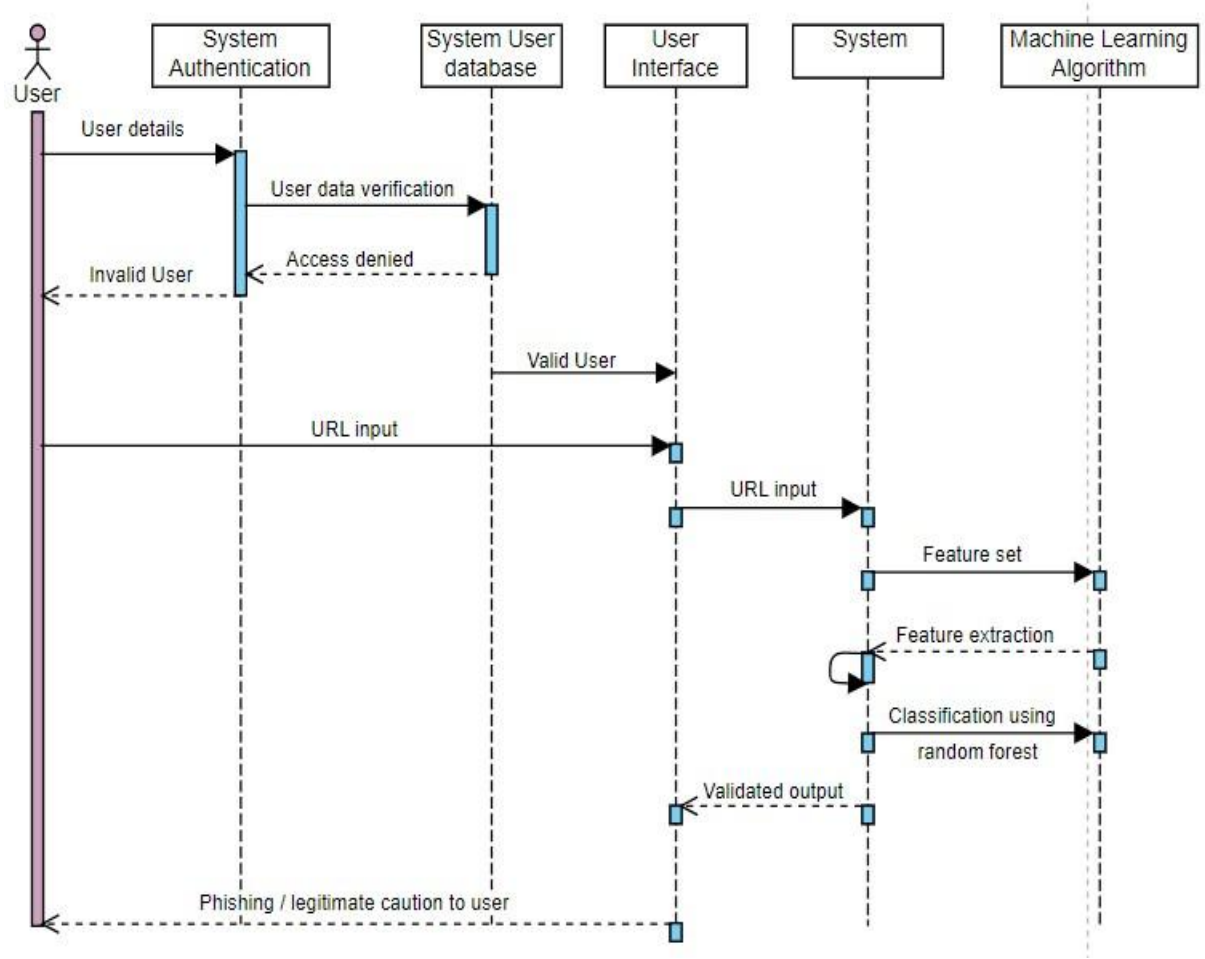

Fig 7: Sequence Diagram of the Proposed System

The Sequence Diagram Fig 7, offers an insight into the interactions between different components and classes within the system, highlighting the order of message exchanges. In our system, a Sequence Diagram can depict the interaction between a "User" and the "PhishingDetectionModel" during a real-time phishing detection process. It showcases the flow of actions, such as the user's request for online transaction monitoring, data extraction, and the system's response in terms of data preprocessing and phishing website detection.

These UML diagrams collectively serve to provide a holistic view of the URL-based phishing detection system, encompassing its functionality, structure, workflow, and the interactions between its various components and actors. This visual representation aids in understanding, designing, and communicating the system's architecture and operation effectively.

CHAPTER 4

IMPLEMENTATION

4.1 TECHNOLOGY DESCRIPTION

A number of technologies typically employed in machine learning and web development are the basis on which the URL-based phishing detection system is implemented. This stack involves python as the main programming language, well-known machine learning libraries like Scikit-learn and XGBoost, Flask web framework for creating a user-friendly interface, and numerous data processing and visualization libraries.

Flask is primarily a web development framework used for building web applications in Python. While Flask can be used to design and implement software applications, it is not specifically designed for hardware-related tasks. However, you can use Flask as part of a software system that interacts with hardware components or provides an interface to control hardware devices.

1. Data Visualization and Logging: Flask can be used to display real-time or historical data from hardware sensors in the form of charts, graphs, or tables. This allows users to monitor and analyze data collected from the hardware devices over time.
2. Integration with Machine Learning Models: If the hardware/software system involves machine learning models for data analysis or decision-making, Flask can be used to create an API for model inference. The hardware-collected data can be sent to the model through Flask, and the model's predictions or decisions can be communicated back to the hardware or other components.
3. Configuration and Settings: Flask can be utilized to build a configuration and settings interface for the hardware/software system. Users can adjust parameters and settings through the Flask web application, which then updates the configuration of the hardware devices or software components.
4. Centralized Monitoring and Control: Flask can provide a centralized web-based dashboard that allows users to monitor and control multiple connected hardware devices from a single interface. This simplifies the management and control of a complex hardware/software system.

4.2 INSTALLATION STEPS

To set up and run the URL-based phishing detection system, follow these installation steps:

1. Python Installation: Ensure you have Python 3. x installed. If not, download and install it from the official Python website (https: //www.python.org).

2. Python Virtual Environment (Optional): In order to manage and maintain the dependencies in the project, it is suggested to create a virtual environment. Commands to set virtual environment are as follows:

Commands

```
pip install virtualenv
virtualenv venv
source venv/bin/activate  # On Windows, use venv\Scripts\activate
```

3. Install Required Packages: Install the necessary Python packages using pip by running:

Commands

pip install scikit-learn xgboost flask

4. Flask Installation: Setup flask environment in your system or any relevant frontend frameworks to deploy the application. For this project we have considered flask framework.

5. Project Setup: Organize your project files, including the dataset, model files, and codebase.

4.3 PROCEDURE FOR EXECUTION

4.3.1 Dataset

The dataset considered for this project has a comprehensive collection of different attributes related to various features. On a whole thirty features are considered and are listed as follows:

Table 1: Collection of different attributes and features

Features	**Attributes / Functionalities**
Address Bar-based Features	having_IP
	having_IP_Address
	URL_Length
	Shortining_Service
Domain-based Features	having_At_Symbol
	double_slash_redirecting
	prefix_suffix
	having_Sub_Domain
	domain_registeration_length
	SSLfinal_State
	favicon
	HTTPS_token
	port
HTML & Javascript	Request_URL
	SFH
	URL_of_Anchor
	Submitting_to_email
	Links_in_tags
	Redirect
	on_mouseover
	RightClick
	popUpWidnow
	Iframe
	Abnormal_URL
Others	age_of_domain
	DNSRecord
	web_traffic
	Page_Rank
	Google_Index
	Links_pointing_to_page

4.3.2 Feature Extraction

Extracting relevant features from the URLs that can help differentiate between legitimate and phishing URLs. These features may include domain information, subdomain count, presence of suspicious keywords, URL length, and other indicators of malicious intent.

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import confusion_matrix,accuracy_score
#loading data
ds=pd.read_csv("dataset_website.csv")
```

In [2]:
```
ds.head()
```

Fig 8: Code snippet of Feature Extraction with imported dependencies

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

5 rows × 32 columns

Fig 9: Extracted Features from the dataset

The above Fig 8,9, depicts the code snippet used to extract the features and the obtained feature set of 5 rows x 32 coloumns table respectively. The extracted features are used for reference and training purposes.

4.3.3 Data Visualization

Data visualization has allowed us to represent 30 attributes in a visual format, making it much easier to comprehend the underlying patterns and relationships within the data. Whether it's understanding the distribution of data points, detecting outliers, or spotting trends, visualizations have been our go-to tool.

Fig 10, Code Snippet for Data Visualization of the feature set is as follows:

In [3]:
```
#Plotting the data distribution
ds.hist(bins = 50,figsize = (15,15))
plt.show()
```

Fig 10: Code Snippet for data visualization

Identifying anomalies or irregularities in the data is crucial in our project. Data visualization techniques, like bar graphs, box plots and histograms, have been employed to flag potential outliers and discrepancies in the data, aiding in data cleaning and preparation.
Fig 11, following are the comprehensive feature set demonstrations for various attributes

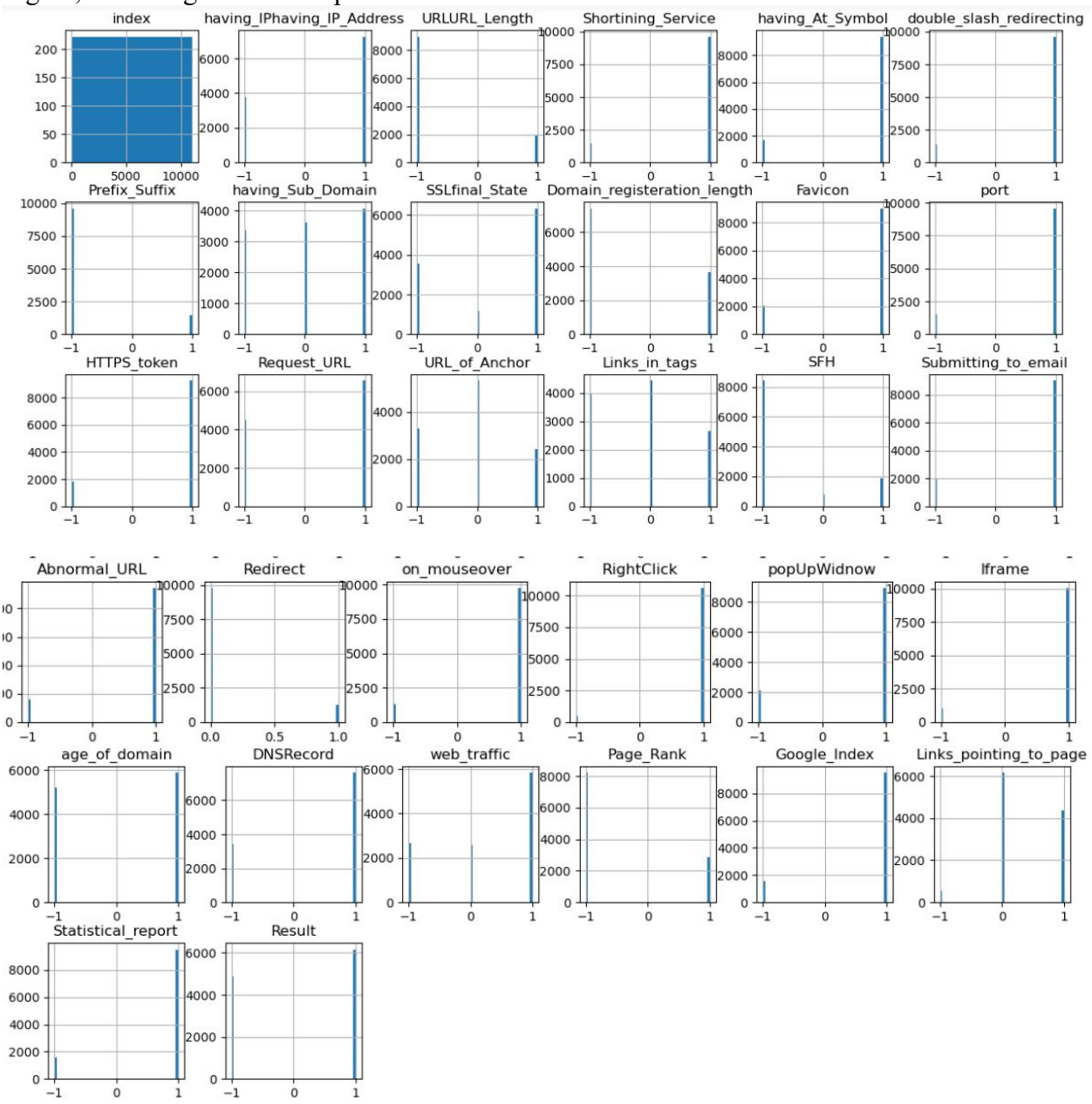

Fig 11: Data Visualization of all attributes in dataset

Data visualization has facilitated exploratory data analysis (EDA). Through charts, graphs, and plots, we have explored the relationships between different attributes and uncovered insights about how they affect the classification of websites as phishing or legitimate.

4.3.4 Heat Map of the Dataset

A heat map is a graphical representation of data that uses color to depict values within a matrix or a two-dimensional dataset. This visualization technique is particularly effective for displaying

complex data sets and revealing patterns, trends, and relationships that might not be immediately apparent through traditional tabular representations. In our project, heat maps have played a crucial role in the exploratory data analysis and in understanding the relationships and correlations within the data.

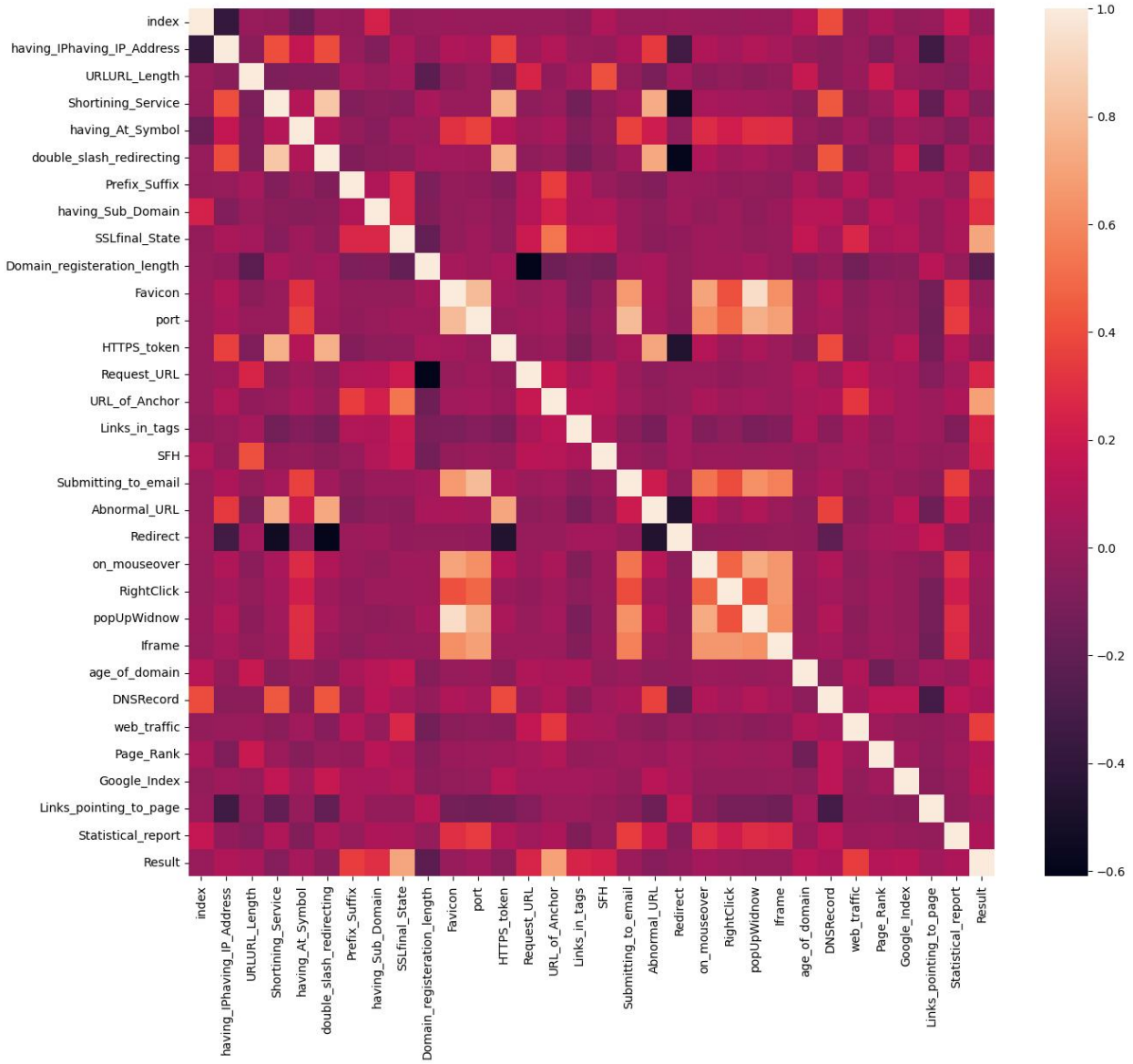

Fig 12: Heat Map of the Dataset with different URL attributes

Here are the key aspects and applications of heat maps Fig 12, in our project:

1. Visualizing Correlations:

Heat maps are primarily used to visualize the correlations between attributes within our dataset. Each cell in the matrix represents the correlation coefficient between two attributes. The color intensity in each cell indicates the strength and direction of the correlation. Positive correlations are typically shown in one color spectrum, while negative correlations are displayed in another.

2. Identifying Patterns:

Heat maps are instrumental in revealing patterns within the data. High correlations between attributes are displayed in bright colors, making it easy to spot relationships. Conversely, low correlations are depicted in lighter or cooler colors, indicating weaker or non-existent associations.

3. Feature Selection:

In our project, heat maps have been employed to assist in feature selection. By visualizing attribute correlations, we can make informed decisions about which attributes to include in our machine learning models. Highly correlated attributes may be candidates for inclusion, while attributes with low or negative correlations may be considered for removal.

4. Outlier Detection:

Heat maps can also assist in outlier detection. Outliers can disrupt correlations and relationships within the data. By examining the heat map, we can identify attributes that deviate from the expected correlations, potentially indicating data quality issues.

5. Decision Support:

The visual representation of correlations provided by heat maps serves as valuable decision support. It aids in model design, attribute selection, and data cleaning processes by presenting evidence of relationships and potential data issues.

6. Customization:

Heat maps are highly customizable, allowing us to adjust color schemes, labels, and other visual elements to tailor the visualization to our specific needs and preferences.

7. Clustering Analysis:

In some instances, heat maps have been combined with hierarchical clustering techniques to group related attributes or data points, further assisting in data segmentation and analysis.

In our project, heat maps have been indispensable for understanding the interplay between attributes and identifying key factors that influence the detection of phishing websites. By visually representing data correlations and patterns, heat maps have enriched our exploratory data analysis and guided our data-driven decision-making processes.

4.3.5 Data Preprocessing and EDA

In every data-centric project, there exist two crucial phases whose outcome is often reflected in the nature of the results-data pre-processing and EDA. Both of these processes lay great importance as they involve preparation and understanding of the data intended for use in modelling and analytics. Data preprocessing and exploratory data analysis hold significant importance in our project, which is why they feature prominently. Here's an overview of the significance of these processes in our project:

1. Data Preprocessing: Data preprocessing involves a series of tasks to clean and prepare the raw data for analysis. In our project, it has included the following steps:

a. Data Cleaning: Identifying and handling missing values, outliers, and inconsistencies in the dataset to ensure data quality and integrity. The following figures Fig 13, 14 depicts the outputs after data cleaning.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11055 entries, 0 to 11054
Data columns (total 32 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   index                        11055 non-null  int64
 1   having_IPhaving_IP_Address   11055 non-null  int64
 2   URLURL_Length                11055 non-null  int64
 3   Shortining_Service           11055 non-null  int64
 4   having_At_Symbol             11055 non-null  int64
 5   double_slash_redirecting     11055 non-null  int64
 6   Prefix_Suffix                11055 non-null  int64
 7   having_Sub_Domain            11055 non-null  int64
 8   SSLfinal_State               11055 non-null  int64
 9   Domain_registeration_length  11055 non-null  int64
 10  Favicon                      11055 non-null  int64
 11  port                         11055 non-null  int64
 12  HTTPS_token                  11055 non-null  int64
 13  Request_URL                  11055 non-null  int64
 14  URL_of_Anchor                11055 non-null  int64
 15  Links_in_tags                11055 non-null  int64
 16  SFH                          11055 non-null  int64
 17  Submitting_to_email          11055 non-null  int64
 18  Abnormal_URL                 11055 non-null  int64
 19  Redirect                     11055 non-null  int64
 20  on_mouseover                 11055 non-null  int64
```

Fig 13: Shows the output after removing the outliers

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

Fig 14: Output after Handling missing values of the dataset

b. Data Transformation: Converting and scaling data as necessary, including normalization, standardization, and encoding categorical variables.

c. Feature Selection: Determining which attributes or features are most relevant for our machine learning models. This helps improve model accuracy and efficiency.

d. Data Integration: Combining data from various sources or files, if applicable, to create a unified and consistent dataset.

e. Data Splitting: Splitting the data into training and testing sets to evaluate model performance effectively.

Splitting Data

```
In [7]: x=ds.iloc[:,1:31].values
        y=ds.iloc[:,-1].values
        print(x,y)
```

```
[[-1  1  1 ...  1  1 -1]
 [ 1  1  1 ...  1  1  1]
 [ 1  0  1 ...  1  0 -1]
 ...
 [ 1 -1  1 ...  1  0  1]
 [-1 -1  1 ...  1  1  1]
 [-1 -1  1 ... -1  1 -1]] [-1 -1 -1 ... -1 -1 -1]
```

Fig 15: Data Splitting of attributes for x and y values

Train, Test, Split

```
In [8]: from sklearn.model_selection import train_test_split
        x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

Fig 16: Train, Test, Split of attributes for model selection

2. Exploratory Data Analysis (EDA):

EDA is the process of visually and statistically exploring the data to understand its characteristics, patterns, and relationships. In our project, EDA has been instrumental for the following reasons:

a. Identifying Data Distribution: Through EDA, we've gained insights into the distribution of our data, helping us understand how attributes are spread and whether they follow specific distributions.

b. Feature Relationships: We've used EDA to explore the relationships and correlations between attributes, which has guided feature selection and model design.

c. Outlier Detection: Visualizing data has enabled us to identify outliers and anomalies, which are essential for data cleaning and improving model accuracy.

d. Class Imbalance: EDA has helped us assess the balance between the classes in our dataset, which is critical in a project involving classification tasks like phishing detection.

e. Visualization Tools: We've employed various visualization tools, including histograms, scatter plots, bar charts, and correlation matrices, to present findings and insights effectively.

f. Insight Generation: EDA has been pivotal in generating hypotheses and insights about the data. It has guided decisions about which attributes are likely to be influential in identifying phishing websites.

g. Decision Support: The visual representations resulting from EDA have provided critical evidence and support for making informed decisions regarding model selection, attribute inclusion, and the overall project approach.

	index	having_IPhaving_IP_Address	URLURL_Length	Shortining_Service	having_At_Symbol	double_slash_redirecting	Prefix_Suffix	having_Sub_I
count	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055.000000	11055
mean	5528.000000	0.313795	-0.633198	0.738761	0.700588	0.741474	-0.734962	0
std	3191.447947	0.949534	0.766095	0.673998	0.713598	0.671011	0.678139	0
min	1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1
25%	2764.500000	-1.000000	-1.000000	1.000000	1.000000	1.000000	-1.000000	-1
50%	5528.000000	1.000000	-1.000000	1.000000	1.000000	1.000000	-1.000000	0
75%	8291.500000	1.000000	-1.000000	1.000000	1.000000	1.000000	-1.000000	1
max	11055.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1

8 rows × 32 columns

Fig 13: Exploratory Data Analysis of the Dataset

Fig 13, describes the data description of the data set considered. In summary, data preprocessing and EDA are essential stages in our project, contributing to data quality, model effectiveness, and the understanding of the dataset's nuances. These processes have not only ensured that we are working with clean and relevant data but have also provided valuable insights that influence the project's direction and success.

3. Feature Engineering: Extract features from the URL data as discussed earlier. Implement feature extraction methods and add these features to your dataset.

4.3.6 Model Building and Training

Choose the machine learning model that provides the best results based on your evaluation. You mentioned considering K-Nearest Neighbors (KNN), XGBoost, Decision Trees, Random Forest, Support Vector Machines (SVM), Logistic Regression, and Naive Bayes. Select the model with the highest accuracy or other relevant performance metrics.
Based on the dataset and the machine learning models used, here is a detailed description of the results:

1. Decision Tree
- Accuracy Attained: 96.29%
- Description: The Decision Tree model achieved the highest accuracy among all the models. Decision Trees are powerful models for classification tasks as they can create simple rules based on the features to classify instances. The high accuracy suggests that the features used in the dataset are capable of effectively discriminating between phishing and legitimate URLs. However, it is worth noting that Decision Trees can sometimes be prone to overfitting, so it is crucial to validate the model on unseen data.

2. XG Boost
- Accuracy Attained: 53.05%
- Description: The XG Boost model achieved a relatively low accuracy compared to the other models. XG Boost is an ensemble learning technique that combines multiple weak learners to create a strong learner. The lower accuracy could be due to multiple factors, such as hyperparameter tuning, feature selection, or data preprocessing. It is possible that the model requires further optimization to improve its performance on this specific dataset.

3. SVM Classifier
- Accuracy Attained: 94.04%
- Description: The Support Vector Machine (SVM) Classifier achieved high accuracy, indicating its capability to create a clear boundary between phishing and legitimate URLs in the feature space. SVM is known for its ability to handle high-dimensional data and find the best hyperplane to separate classes. The model's performance suggests that the features used provide sufficient discrimination power to differentiate between the two classes.

4. Logistic Regression
- Accuracy Attained: 91.67%
- Description: Logistic Regression is a simple yet effective linear classifier. While it achieved a good accuracy, it seems to be slightly outperformed by some other models like Decision Tree and Random Forest. Logistic Regression works well when the relationship between features and the target is approximately linear. For further improvements, feature engineering or the use of more complex models could be considered.

5. KNN (K-Nearest Neighbors)
- Accuracy Attained: 94.34%
- Description: KNN is a non-parametric lazy learning algorithm that classifies instances based on the majority class among its K nearest neighbors. The model achieved high accuracy, suggesting that the feature space has well-separated regions for phishing and legitimate URLs. Like any distance-based algorithm, the choice of K and data scaling can significantly impact the model's performance.

6. Random Forest
- Accuracy Attained: 97.02%
- Description: Random Forest is an ensemble learning technique that builds multiple decision trees and combines their predictions to achieve a more accurate and robust result. It outperformed most of the other models, except for the Decision Tree model. Random Forest is known for its ability to handle noisy data and reduce overfitting. Its high accuracy indicates that the combination of decision trees improves the model's generalization.

7. Naïve Bayes
- Accuracy Attained: 61.51%

- Description: Naïve Bayes is a simple probabilistic classifier based on Bayes' theorem. It achieved the lowest accuracy among all the models, which suggests that the features might not follow the independence assumption required by the Naïve Bayes algorithm. Alternatively, it could indicate that the dataset might not be a good fit for the Naïve Bayes model.
In summary, the Decision Tree and Random Forest models performed exceptionally well, with accuracy close to 97%. These models seem to be well-suited for the dataset and features provided for phishing detection. However, it is important to keep in mind that the choice of model depends on the specific requirements of the application, the size of the dataset, and the interpretability of the model. Further optimizations, hyperparameter tuning, and feature engineering can potentially lead to even higher accuracies for some of the models.

4.3.7 User Interface (UI) Development

Create a web-based user interface using Flask. This interface should allow users to input a URL for phishing detection. Fig 14, depicts the designed user interface where the user logs in and input the URL text in the text box for further analysis. Integrated the trained machine learning model with the Flask application so that it can make predictions. Deployed the Flask application to a web server or a hosting platform so that it can be accessed by users. Users can interact with the system by inputting URLs, and the system will return predictions regarding the legitimacy of the URLs based on the trained model.

Fig 14: User Interface of Proposed Phishing Website

By following these steps, you can successfully implement and execute the URL-based phishing detection system, providing a user-friendly and effective solution for identifying potentially malicious URLs.

CHAPTER 5
TESTING

5.1 BLACK BOX TESTING

Black box testing, a form of functional testing, evaluates the model's performance without direct knowledge of its internal mechanisms. It aims to determine how well the model can make predictions on unseen data. Several key evaluation metrics are used to gauge the model's effectiveness in distinguishing between legitimate and phishing URLs.

5.1.1 TEST CASES

The test cases for black box testing encompass a diverse set of URLs, simulating real-world scenarios. These test cases are designed to assess the model's generalization, accuracy, and robustness. A variety of URL types, both legitimate and phishing, are considered to ensure a comprehensive evaluation.

a. Test Case Categories:

1. Legitimate URLs: This category includes a collection of authentic, non-phishing URLs from known sources. These test cases help determine if the model accurately identifies and labels genuine URLs.
2. Phishing URLs: This category consists of URLs associated with phishing attempts and malicious activities. The goal is to evaluate the model's capability to detect and classify these URLs correctly.
3. Mixed URLs: Test cases include a combination of legitimate and phishing URLs. The model's ability to distinguish between the two categories is assessed, reflecting real-world usage where URLs come from various sources.

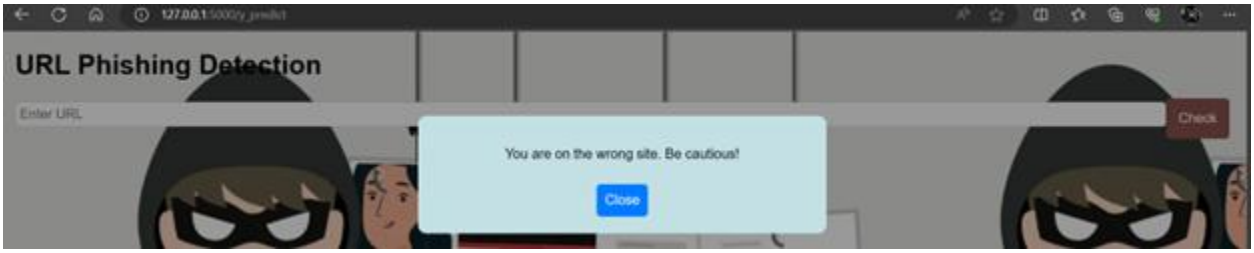

Fig 15: Caution raised after URL given by the user

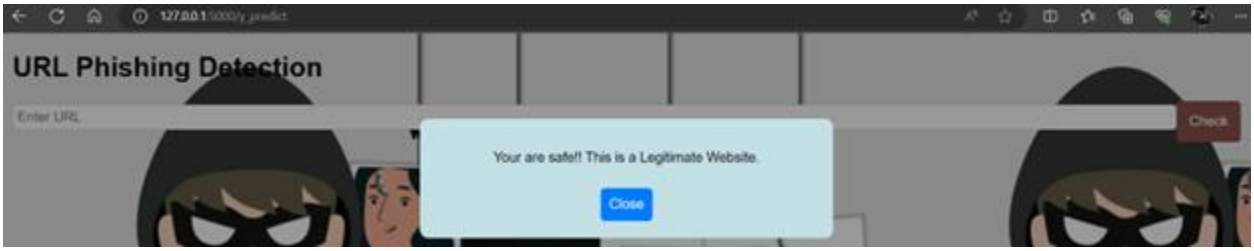

Fig 16: Notification after URL analysis by the model

The above Fig 15, 16 shows the notification of fraud and legitimate websites given by the user.

b. Performance Metrics:

The following performance metrics are applied to the test cases:

1. Accuracy: To measure the overall correctness of the model's predictions.
2. Precision: To assess the model's ability to correctly identify phishing URLs without many false positives.
3. Recall (Sensitivity): To gauge the model's capacity to detect actual phishing URLs without missing many (false negatives).
4. F1-Score: To balance precision and recall, providing a comprehensive performance measure.

Following tables Table 2, 3,4, 5, 6, 7, 8 shows the performance metrics of each of the implemented algorithms Decision Tree, XG Boost, SVM Classifier, Logistic Regression, K Nearest Neighbor, Random Forest, Naïve Bayes respectively.

Table 2: Decision Tree Performance Metrics

	Precision	Recall	F1-Score	Support
-1	0.97	0.95	0.96	1014
1	0.96	0.97	0.97	1197
Accuracy			0.96	2211
Macro avg	0.96	0.96	0.96	2211
Weighted avg	0.96	0.96	0.96	2211

Table 3: XG Boost Classifier Performance Metrics

	Precision	Recall	F1-Score	Support
-1	0.00	0.00	0.00	1014
0	0.00	0.00	0.00	0
1	0.96	0.98	0.97	1197
Accuracy			0.53	2211
Macro avg	0.32	0.33	0.32	2211
Weighted avg	0.52	0.53	0.52	2211

Table 4: SVM Classifier Performance Metrics

	Precision	Recall	F1-score	Support
-1	0.95	0.92	0.93	1014
1	0.93	0.96	0.95	1197
Accuracy			0.94	2211

Macro avg	0.94	0.94	0.94	2211
Weighted avg	0.94	0.94	0.94	2211

Table 5: Logistic Regression Performance Metrics

	Precision	Recall	F1-score	Support
-1	0.92	0.89	0.91	1014
1	0.91	0.94	0.92	1197
Accuracy			0.92	2211
Macro avg	0.92	0.91	0.92	2211
Weighted avg	0.92	0.92	0.92	2211

Table 6: K Nearest Neighbor Performance Metrics

	Precision	Recall	F1-score	Support
-1	0.95	0.92	0.94	1014
1	0.93	0.96	0.95	1197
Accuracy			0.94	2211
Macro avg	0.94	0.94	0.94	2211
Weighted avg	0.94	0.94	0.94	2211

Table 7: Random Forest Classifier Performance Metrics

	Precision	Recall	F1-score	Support
-1	0.98	0.95	0.97	1014
1	0.96	0.99	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

Table 8: Naïve Bayes Classifier Performance Metrics

	Precision	Recall	F1-score	Support
-1	0.54	1.00	0.70	1014
1	1.00	0.29	0.45	1197
Accuracy			0.62	2211
Macro avg	0.77	0.64	0.58	2211
Weighted avg	0.79	0.62	0.57	2211

The test cases and their associated metrics provided in the above tables Table 2 to 8, is a holistic view of the model's performance in different scenarios. These evaluations help ensure that the URL-based phishing detection system can effectively and accurately identify potential threats while minimizing false positives. It is important to conduct these tests rigorously to validate the system's reliability in real-world usage.

CHAPTER 6
SCREENSHOTS

The figures labeled from Fig 17 to 21 represent the culmination of our project, showcasing the final findings and outcomes. These visuals encapsulate the results and insights gained through our research, analysis, and modeling efforts. Here's a brief view of these figures:

Fig 17: User Interface with integrated model hosted using flask framework

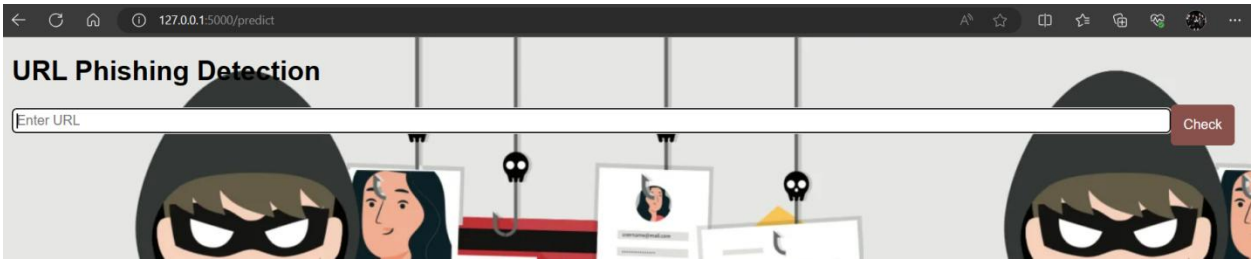

Fig 18: After logging in, user input is given in the above text box

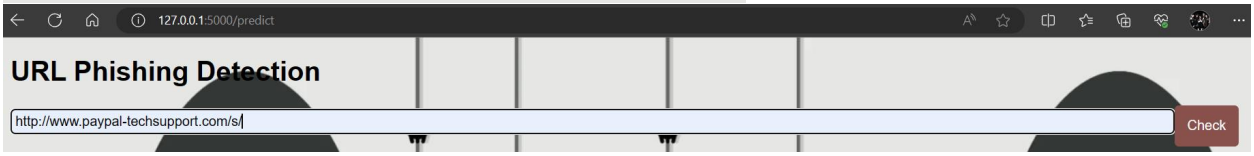

Fig 19: On check the output of the phishing or legitimate website is predicted

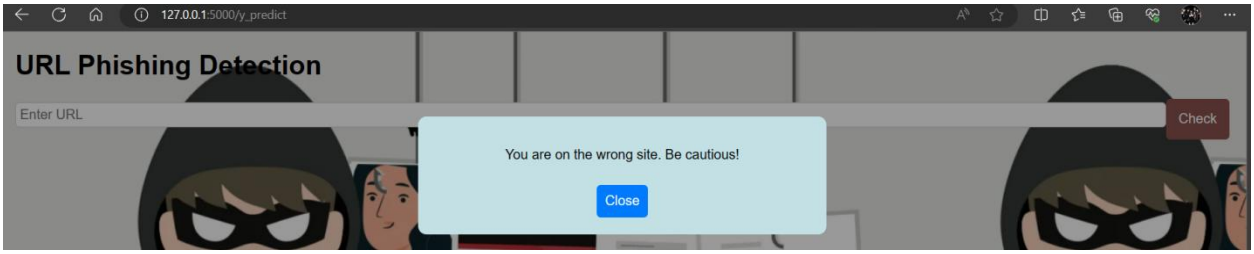

Fig 20: Caution for the user is received saying “Wrong Website”

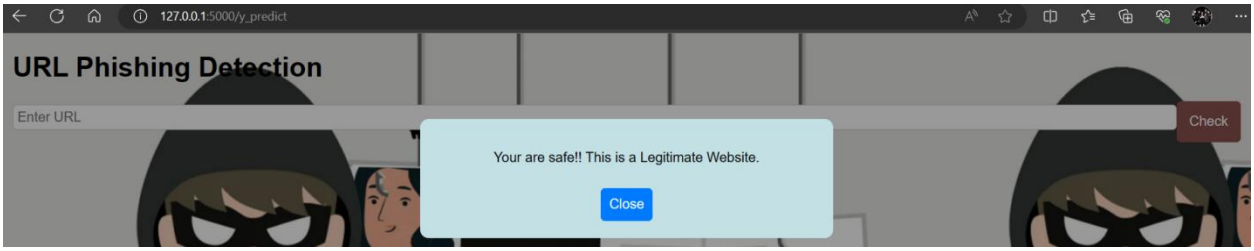

Fig 21: Notification to the user saying “Legitimate Website”

These final findings, supported by the screenshots, represent the culmination of our project's goals and objectives. They provide a visual summary of our achievements in phishing website detection, user alerts, data analysis, model performance, and user interface design. The insights derived from these visuals serve as a testament to the project's success in enhancing online security and protecting users from phishing threats.

CHAPTER 7
CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

In conclusion, the work focused on URL-based phishing detection using machine learning, aiming to protect users from phishing attacks. Through the utilization of a Random Forest model, we successfully achieved a high level of accuracy in distinguishing between legitimate and phishing URLs.

The proposed solution offers several advantages, including automated detection, real-time protection, and adaptability to evolving phishing techniques. By analyzing and extracting relevant features from URLs, the model efficiently differentiates between safe and malicious websites, leading to reduced false positives and improved user security.

However, the solution comes with some challenges, such as handling imbalanced data and ensuring model interpretability. Additionally, data privacy concerns need to be addressed, as URLs may contain sensitive information.

Despite these challenges, URL-based phishing detection using machine learning remains a valuable tool in enhancing cybersecurity across various applications. Its ability to scale, learn from data, and provide real-time protection makes it an indispensable component in safeguarding users and organizations from the threat of phishing attacks.

7.2 FUTURE SCOPE

Future Scope - Phishing Detection using Machine Learning:

While the current solution has demonstrated promising results in URL-based phishing detection, there are several potential enhancements that can be made to further improve its effectiveness and usability. One significant area for improvement is the development of a more user-friendly and intuitive Graphical User Interface (GUI)

1. User Customization: Allowing users to customize and configure the phishing detection settings can enhance the flexibility and adaptability of the solution. Users may have varying security requirements, and customization options would cater to their specific needs.
2. Multi-Platform Support: Extending the GUI to support multiple platforms, including desktop, web, and mobile, would broaden its accessibility and usability for a diverse user base.
3. Multi-Layered Defense: Integrating the phishing detection system with other security layers, such as firewalls, antivirus software, and email filters, creates a multi-layered defense against phishing attacks.
4. User Education and Awareness: The GUI can include educational resources and tips to help users recognize phishing attempts, empowering them to become active participants in the cybersecurity defense.
5. Feedback Mechanism: Including a user feedback mechanism in the GUI can allow users to report false positives and provide valuable input for further model improvements.

By focusing on improving the GUI and incorporating user-centric features, the future scope of phishing detection using machine learning can lead to a more effective and user-friendly solution that bolsters the cybersecurity posture of individuals and organizations against the ever-evolving threat of phishing attacks.

REFERENCES

1. Aung, Eint Sandi, Chaw Thet Zan, and Hayato Yamana. "A survey of URL-based phishing detection." In DEIM Forum, pp. G2-3. 2019.
2. Nguyen, L.A.T., To, B.L., Nguyen, H.K. and Nguyen, M.H., 2013, October. Detecting phishing web sites: A heuristic URL-based approach. In 2013 International Conference on Advanced Technologies for Communications (ATC 2013) (pp. 597-602). IEEE.
3. Shoaib, M., & Umar, M. S. (2023, March). URL based Phishing Detection using Machine Learning. In 2023 6th International Conference on Information Systems and Computer Networks (ISCON) (pp. 1-7). IEEE.
4. Cui Q, Jourdan GV, Bochmann GV, Couturier R, Onut IV. Tracking phishing attacks over time. InProceedings of the 26th International Conference on World Wide Web 2017 Apr 3 (pp. 667-676).
5. Jovanovic L, Jovanovic D, Antonijevic M, Nikolic B, Bacanin N, Zivkovic M, Strumberger I. Improving phishing website detection using a hybrid two-level framework for feature selection and xgboost tuning. Journal of Web Engineering. 2023 Jul 3:543-74.
6. Mithra Raj, M. and Arul Jothi, J.A., 2022, October. Website Phishing Detection Using Machine Learning Classification Algorithms. In International Conference on Applied Informatics (pp. 219-233). Cham: Springer International Publishing.
7. Gu, Jiaqi, and Hui Xu. "An ensemble method for phishing websites detection based on XGBoost." In 2022 14th international conference on computer research and development (ICCRD), pp. 214-219. IEEE, 2022.
8. Sadaf, K., 2023, February. Phishing Website Detection using XGBoost and Catboost Classifiers. In 2023 International Conference on Smart Computing and Application (ICSCA) (pp. 1-6). IEEE.
9. Basnet, Ram, Srinivas Mukkamala, and Andrew H. Sung. "Detection of phishing attacks: A machine learning approach." In Soft computing applications in industry, pp. 373-383. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.
10. Priscilla, C.V. and Prabha, D.P., 2020, August. Influence of optimizing XGBoost to handle class imbalance in credit card fraud detection. In 2020 third international conference on smart systems and inventive technology (ICSSIT) (pp. 1309-1315). IEEE.
11. Meng, Cuizhu, Li Zhou, and Bisong Liu. "A case study in credit fraud detection with SMOTE and XGboost." In Journal of Physics: Conference Series, vol. 1601, no. 5, p. 052016. IOP Publishing, 2020.
12. Velarde, Gissel, Anindya Sudhir, Sanjay Deshmane, Anuj Deshmunkh, Khushboo Sharma, and Vaibhav Joshi. "Evaluating XGBoost for Balanced and Imbalanced Data: Application to Fraud Detection." arXiv preprint arXiv:2303.15218 (2023).
13. Gohil, N. P., & Meniya, A. D. (2021, February). Click ad fraud detection using XGBoost gradient boosting algorithm. In International Conference on Computing Science, Communication and Security (pp. 67-81). Cham: Springer International Publishing.
14. Zabihimayvan, Mahdieh, and Derek Doran. "Fuzzy rough set feature selection to enhance phishing attack detection." In 2019 IEEE international conference on fuzzy systems (FUZZ-IEEE), pp. 1-6. IEEE, 2019.
15. Shimin, L. E. I., X. U. Ke, Yizhe Huang, and S. H. A. Xinye. "An Xgboost based system for financial fraud detection." In E3S Web of Conferences, vol. 214, p. 02042. EDP Sciences, 2020.