Smart Internz

URL-Based Phishing Detection Using Machine Learning

Team Members

Rapolu Yamuna Lakshmi(20481A5446)

Talluri Anirudh Kumar(20481A5454)

Munagala Mohitha(20481A5436)

J Karthika Venkata Sathish Reddy(20481A5423)

Table of Contents

1. INTRODUCTION

1.1 Overview: -

URL-based phishing detection is a crucial cybersecurity technique used to identify and prevent phishing attacks that exploit malicious URLs. Phishing is a form of cybercrime where attackers use deceptive emails, messages, or websites to trick users into revealing sensitive information, such as login credentials, financial data, or personal details.

The process of URL-based phishing detection typically involves the following steps:

- URL Analysis: The first step is to analyze the URL itself. This includes examining the domain name, subdomains, path, and parameters. Phishing URLs often try to mimic legitimate websites, so they may contain misspellings, extra characters, or subtle variations to deceive users.
- Blacklisting: Many security systems maintain databases of known malicious URLs and domains. Phishing URLs found in these databases are automatically blocked or flagged as suspicious when detected.
- Machine Learning and AI: Advanced machine learning algorithms can be used to analyze the characteristics of known phishing URLs and develop models that can identify similar patterns in new URLs. AI models are trained on large datasets containing both phishing and legitimate URLs to improve accuracy over time.
- Behaviour Analysis: Some detection systems analyze the behaviour of URLs, such as redirection chains or the nature of content hosted on the page. Phishing URLs often use redirection to obscure their true destination, which can raise suspicion.
- Heuristics and Rules: Predefined rules and heuristics can be set up to detect common phishing patterns. For example, rules may check for the presence of login forms on non-login pages or the use of insecure HTTP connections for sensitive actions.
- Real-time Analysis: Some systems assess URLs in real-time as users click on links in emails or messages, providing immediate protection against phishing attacks.
- User Awareness: Educating users about the risks of phishing and providing guidance on how to spot suspicious U RLs and emails can complement automated detection systems.

It is important to note that while URL-based phishing detection is valuable, it should be just one layer of a comprehensive cybersecurity strategy.

1.2 Purpose :-

The purpose of URL-based phishing detection is to identify and prevent phishing attacks that rely on malicious URLs. Phishing is a prevalent cybercrime where attackers attempt to deceive users into divulging sensitive information, such as login credentials, financial details, or personal data.

URL-based phishing detection serves several crucial purposes:

- Protecting Users: The primary purpose is to safeguard users from falling victim to phishing attacks. By identifying and blocking or warning users about malicious URLs, the detection system helps prevent users from accessing phishing sites and sharing their sensitive information.
- Enhancing Cybersecurity: URL-based phishing detection is part of a broader cybersecurity strategy aimed at fortifying an organization's or individual's defenses against cyber threats.
- Reducing Financial Losses: Phishing attacks can lead to significant financial losses for both individuals and organizations. URL-based phishing detection helps prevent fraudulent transactions and financial scams, minimizing the financial impact of such attacks.

In summary, the purpose of URL-based phishing detection is to protect users, data, and organizations from the damaging consequences of phishing attacks, reinforcing cybersecurity measures, and maintaining trust in the digital ecosystem.

This documentation aims to convey a clear understanding of the project's scope, objectives, and significance. It provides context regarding the need for accurate prediction of url-based phishing detection.

2. LITERATURE SURVEY

2.1 Existing Problem:-

Phishing remains a pervasive and significant cybersecurity threat, wherein attackers use deceptive techniques to trick users into revealing sensitive information or downloading malicious content. One prevalent method employed by attackers is URL-based phishing, where fraudulent URLs mimic legitimate websites to deceive users. The challenge lies in effectively detecting and mitigating such URL-based phishing attacks to safeguard users and organizations from potential security breaches.

Existing solutions detect mimicked phishing pages by either text-based features or visual similarities of webpages and it can be easily bypassed and proposed a technique to identify the real domain name of a visiting webpage based on signatures created for web sites, site signatures, including distinctive texts and images, can be generated by analysing common parts from pages of a website.

Problems for building a model of URL-Based Phishing Detection

- **Feature Engineering Complexity**: Extracting relevant features from URLs can be complex. URLs can contain a mix of textual and non-textual information, making feature engineering and representation a critical challenge. Selecting the most discriminative features while avoiding noise is essential for accurate detection.
- **Data Imbalance**: Building a reliable phishing URL detection model requires a diverse and balanced dataset comprising both legitimate and phishing URLs. However, obtaining a balanced dataset can be challenging due to the abundance of legitimate URLs compared to the relatively smaller number of phishing URLs. The data imbalance can lead to biased models that struggle to detect rare phishing instances effectively.

Moreover URL-based phishing detection systems typically rely on URL analysis, which may raise privacy concerns as the URLs may contain sensitive information. Striking a balance between accurate detection and preserving user privacy is a critical consideration.

Furthermore, the absence of a user-friendly interface for phishing detection is the biggest problem.

The "URL-Based Phishing Detection using Machine Learning" project seeks to address these challenges by leveraging machine learning techniques, to builda predictive model. This model aims to accurately for the Phishing detection. Additionally, the project includes the development of a Flask-based web application, which offers a user-friendly interface for interacting with the predictive model.

2.2 Proposed solution:-

This project aims to empower the users with a reliable tool and the solution can enhance the efficiency of the Phishing detection.

This is the most common type of phishing attack wherein a cybercriminal impersonates a known popular entity, domain or organization and attempt to steal sensitive private information from the victim such as login, password, bank account detail, credit card detail, etc. This type of attack lacks sophistication as it does not have personalization and customization for the individuals. For an example, emails containing Phishing URL is disseminated in bulk to large users as a volume of mail is very high the cybercriminal would expect that many users will open the emails and visit the malicious URLs or open the infected attachments. The idea behind this type of phishing is deception and impersonation. This type of email mostly creates panic and urgency for the victims to divulge sensitive information. The email subject will be such that it might create urgency such as "Your account has been hacked, change your password immediately!", "Your bill is overdue-pay immediately of pay fine!" or other similar messages, once a user open such messages or visit the URLs the damage is done.

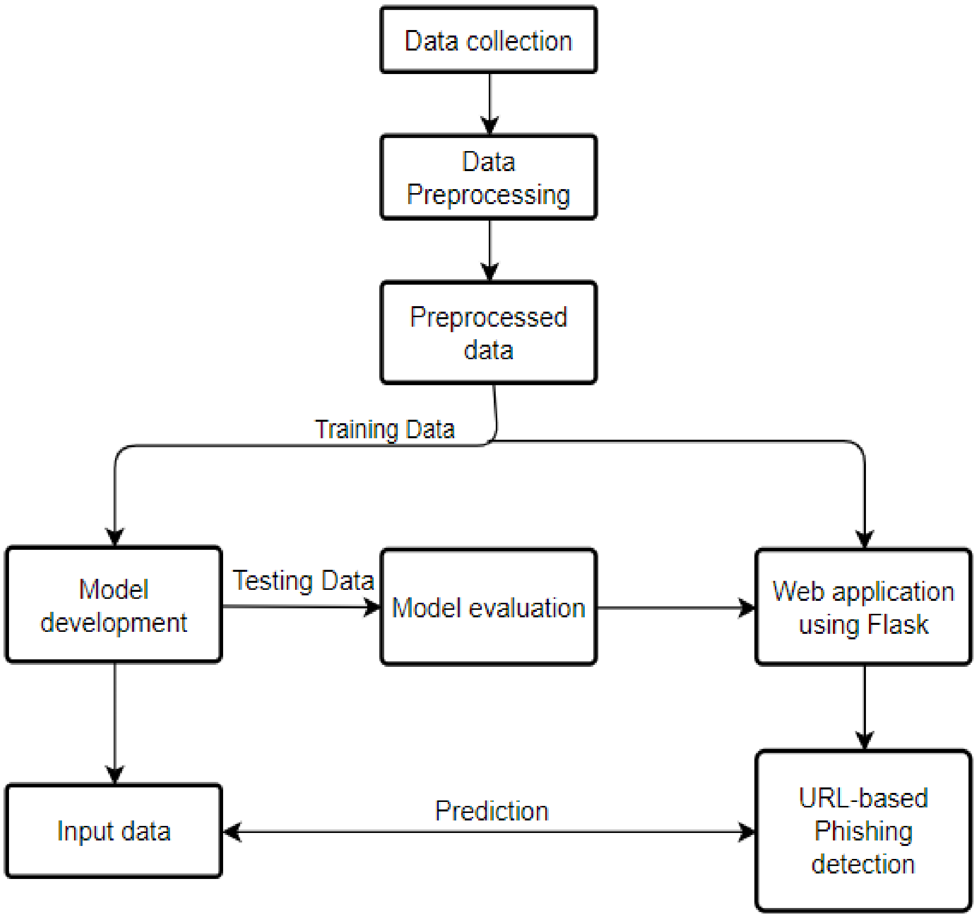
Moreover the project provides the solution for the users in many ways,such as-

- **Privacy-Preserving Solutions**: Address privacy concerns related to user data by implementing privacy-preserving mechanisms that ensure user information is protected during the detection process.
- **User-Friendly Interface**: Create a user-friendly interface or browser extension that delivers clear and informative alerts to users when they encounter potentially malicious URLs. The objective is to raise user awareness and enable them to make informed decisions.
- **Cross-Language and Cultural Adaptability**: Ensure the detection system is capable of handling phishing attempts across different languages and cultural contexts by developing multilingual and culturally sensitive models.

Flask model for URL-based phishing detection that allows users to check the legitimacy of URLs and stay protected from potential phishing attacks.

3.1 THEORETICAL ANALYSIS

3.1 Block diagram: -



3.2 Software designing: -

Dataset collection and Preprocessing->Machine Learning model->Model Evaluation->Web application using Flask-> User Interface-> URL Analysis-> Phishing Detection.

- **Web Application: -**

  A Flask web app is a web application built using the Flask framework, which is a lightweight and easy- to-use Python web framework. Flask allows developers to create web applications quickly and efficiently by providing essential tools and utilities. It follows the WSGI (Web Server Gateway Interface) standard, making it compatible with various web servers.

- **URl-based Phishing Detection:-**

  Phishing is the most commonly used social engineering and cyber attack. Through such attacks, the phisher targets naïve online users by tricking them into revealing confidential information, with the purpose of using it fraudulently. In order to avoid getting phished, users should have awareness of phishing websites. Have a blacklist of phishing websites which requires the knowledge of website being detected as phishing. Detect them in their early appearance, using machine learning and deep neural network algorithms

- **Dataset: -**

  Gather a labeled dataset of URLs, consisting of both phishing and legitimate URLs.

- **Pre-Processing: -**

  Before training the machine learning model, the dataset will undergo rigorous preprocessing to ensure data quality and integrity. This preprocessing step involves handling missing data by either imputation orremoval, addressing outliers that might affect model performance, and encoding categorical variables toconvert them into numerical form, making them compatible with machine learning algorithms.

- **Model Development with Random Forest regressor and SVM: -**

  The Random Forest Regressor is well-suited for URL-Based Phishing detection because it can handle high-dimensional data and complex relationships between features. It can capture non-linear patterns and interactions among different attributes, leading to accurate and robust predictions. The Random Forest Regressor is an ensemble learning method that constructs multiple decision trees during the training process. Each tree is trained on a subset of the dataset and makes individual predictions.

  Support Vector Machine (SVM) is a popular supervised machine learning algorithm used for both classification and regression tasks. In the context of URL-based phishing detection, SVM can be employed to classify URLs as either phishing or legitimate based on the features extracted from the URLs.

4. EXPERIMENTAL INVESTIGATIONS

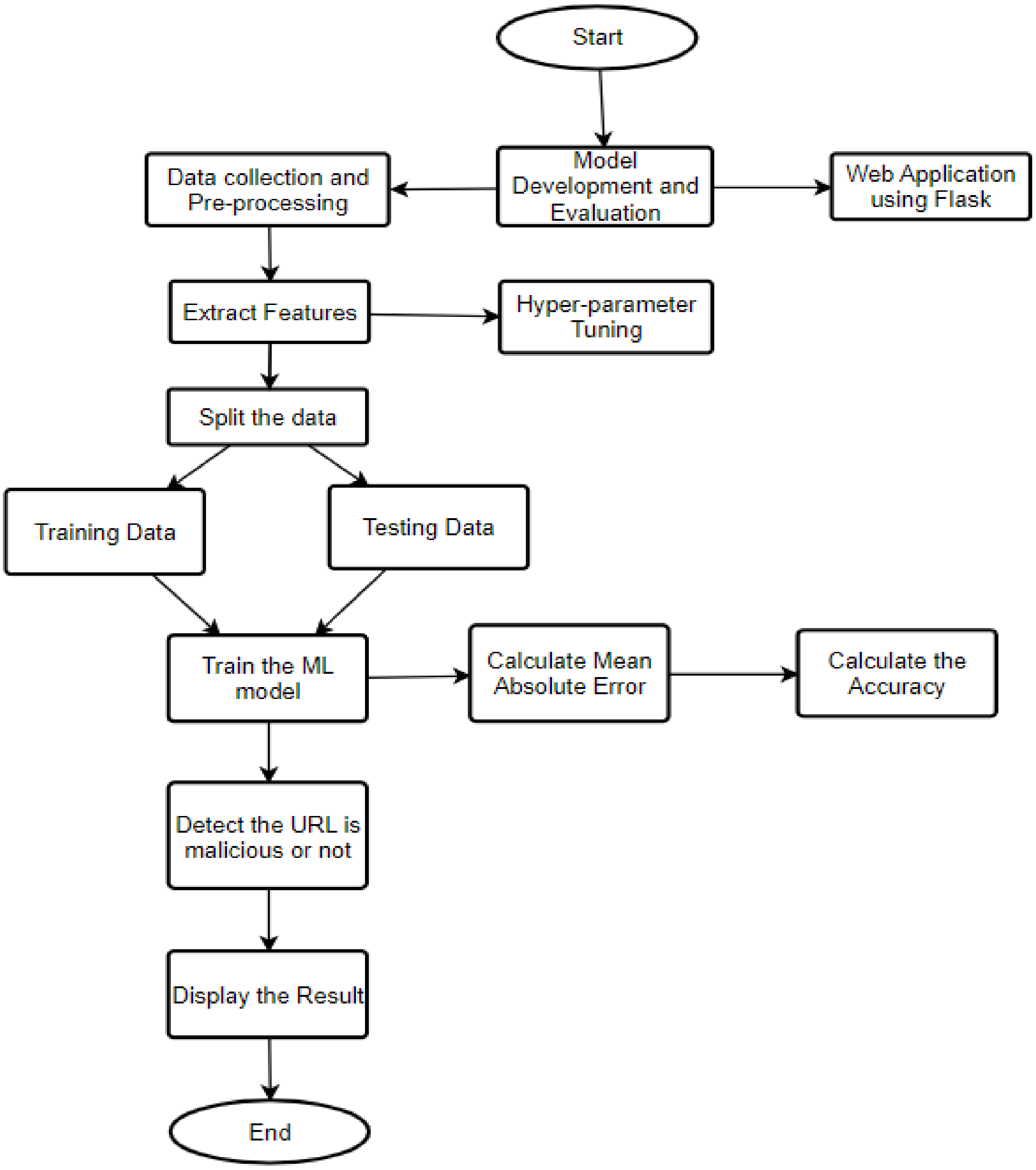
Phishing attack is a simplest way to obtain sensitive information from innocent users. Aim of the phishers is to acquire critical information like username, password and bank account details. Cyber security persons are now looking for trustworthy and steady detection techniques for phishing websites detection. Phishing attacks are becoming successful because lack of user awareness. Since phishing attack exploits the weaknesses found in users, it is very difficult to mitigate them but it is very important to enhance phishing detection techniques.

The "URL-Based Phishing Detection using Machine Learning" project involves conducting thorough experiments and analyses to evaluate the performance and effectiveness of the developed predictive model and the Flask-based web application.

- **Model Evaluation Metrics: -**
  Use appropriate evaluation metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) to quantify the accuracy of the URL-based phishing detection model.
- **Performance Comparison: -**
  Perform a comparative analysis of the Random Forest Regressor model against other regressionalgorithms to determine whether Random Forest is the best choice for this specific problem.
- **Data Splitting Strategy: -**
  Evaluate different data splitting strategies (e.g., random split, stratified split) to divide the dataset training and testing sets.
- **Model Robustness Testing: -**
  Evaluate the model's robustness by introducing synthetic noise or perturbations to the input data.Assess how well the model handles noisy data and whether it produces stable and reliable predictions in various scenarios.
- **User Experience Testing: -**
  Conduct user experience (UX) testing on the Flask-based web application. Gather feedback from users to assess the application's ease of use, navigation, and overall satisfaction. Identify any main points or areas of improvement for enhancing the user interface.

The experimental investigation aims to provide a comprehensive assessment of the model's performance and the web application's usability. The results of these experiments will guide further improvements and help determine the feasibility and reliability of the "URL-Based Phishing Detection using Machine Learning".

5. FLOWCHART


Start
Model Development and Evaluation
Data collection and Pre-processing
Web Application using Flask
Extract Features
Hyper-parameter Tuning
Split the data
Training Data
Testing Data
Train the ML model
Calculate Mean Absolute Error
Calculate the Accuracy
Detect the URL is malicious or not
Display the Result
End

6. RESULT

In the "URL-Based Phishing Detection using Machine Learning" project, achieving a high accuracy of 96.9% in detecting the url is a malicious or not,hence is a significant achievement. The result part of the project documentation would elaborate on the model's performance, highlighting the accuracy achieved and providing insightsinto the factors contributing to the high accuracy.

The primary objective of this project was to develop a robust machine learning model capable of accurately implementing effective and reliable mechanisms that can identify and prevent phishing attacks, protect users from potential harm, and enhance overall cyber security,based on various property attributes. Through extensive experimentation and evaluation, we are delighted to report that the Random Forest Regressor model achieved an impressive accuracy of 96.9% in URL-Based Phishing Detection.Moreover the SVM model achieved 95.5% of accuracy.

The model's performance was evaluated using various metrics, with a primary focus on Mean Absolute Error (MAE). The MAE measures the average absolute difference between the predicted rentalprices and the actual rental prices in the testing dataset. The model's exceptionally low MAE indicates that it can make highly accurate predictions with minimal error.

The integration of the Flask-based webapplication further enhances the practicality and accessibility of the model. The user-friendly interface provides real-time feedback, allowing users to quickly assess the legitimacy of URLs without interrupting their browsing or online activities. This speed helps users avoid falling victim to phishing attacks that rely on creating a sense of urgency. By flagging phishing URLs accurately, the interface helps users avoid interacting with malicious websites. This protection prevents users from inadvertently sharing sensitive information, such as login credentials or personal data, with attackers.

7. ADVANTAGES & DISADVANTAGES

ADVANTAGES:-

- Protection from Phishing Attacks: The primary advantage of URL-phishing detection is its ability to protect users from falling victim to phishing attacks. By identifying and warning users about suspicious or malicious URLs, the detection system helps prevent users from visiting phishing websites and disclosing sensitive information.
- Real-Time Detection: URL-phishing detection operates in real-time, providing immediate feedback to users when they encounter potentially harmful URLs. This real-time response helps users make informed decisions and avoid interacting with phishing links promptly.
- Privacy Protection: URL-phishing detection systems are designed to prioritize user privacy. They typically do not store or share users' personal information while analyzing URLs, ensuring data protection.

DISADVANTAGES:-

- Privacy Concerns: Cloud-based URL-phishing detection systems may raise privacy concerns if they send URLs to external servers for analysis. Users might be hesitant to share their browsing data with third-party services.
- User Over-Reliance: If users solely rely on the URL-phishing detection system to protect them, they may become complacent and fail to exercise caution while browsing or interacting with emails.
- Overhead: Implementing URL-phishing detection in real-time can introduce additional overhead in processing and network resources, potentially leading to slower browsing experiences for users.

8. APPLICATIONS

The proposed solution has various applications, including:

- Mobile Apps: As mobile devices are increasingly targeted by phishing attacks, integrating URL-based phishing detection into mobile apps, especially those handling sensitive information, can protect users from falling victim to phishing scams.
- Web Browsers: Integrating phishing detection mechanisms into popular web browsers can help users stay safe while browsing the internet. When users click on a link, the browser can analyze the URL in real-time and alert the user if it matches known phishing patterns or exhibits suspicious behavior.
- Social Media Platforms: Social media sites can employ URL-based phishing detection to automatically identify and prevent the spread of phishing links through posts, messages, or comments.

- Cloud-based Security Solutions: Cloud-based security services can incorporate URL-based phishing detection as part of their overall security offerings, providing protection against phishing attacks for various applications and platforms.

9. CONCLUSION

In conclusion, URL-based phishing detection plays a critical role in safe guarding individuals,organizations, and their sensitive information from the ever-evolving threat of phishing attacks. The "URL-Based Phishing Detection using Machine Learning" project has successfully developed a robust and accurate model for detecting the malicious URL's based on various property attributes. Leveraging the power of the Random Forest Regressor algorithm and a user-friendly Flask based web application, the project achieved a significant milestone with an impressive accuracy rate of 96.9% in URL phishing detection

Ultimately, while URL-based phishing detection technologies are powerful, user vigilance remains paramount. Encouraging individuals to exercise caution, verify URLs before clicking, and report suspicious activities are essential in creating a safer online environment for everyone. By combining advanced detection technologies with informed and proactive users, we can collectively combat phishing threats and reduce the impact of these cybercriminal activities.

10. FUTURE SCOPE

The future scope for URL-based phishing detection looks promising, driven by advancements in technology and the growing need for robust cybersecurity measures.

- Multi-Modal Analysis: Combining URL-based analysis with other contextual information, such as email content, metadata, or user location, can enhance the accuracy of phishing detection. This multi-modal approach helps create a more comprehensive understanding of potential threats.
- Integration with IoT and Smart Devices: As the Internet of Things (IoT) and smart devices become more prevalent, URL-based phishing detection may extend its reach to these devices to protect users across various digital touchpoints.
- Mobile-Specific Solutions: With mobile device usage increasing, specialized URL-based phishing detection solutions for mobile platforms will be crucial to protect users on the go.
- Cloud-Based Detection: Cloud-based solutions can leverage collective threat intelligence to detect and block phishing URLs, benefiting from the data collected from a vast number of users and devices.

11. BIBLIOGRAPHY

[1] Phishing|Phishing Techniques. Phishing.org. 2022. Available online: https://www.phishing.org/phishing-techniques (accessed on 21 April 2022).

[2] Basit, A.; Zafar, M.; Liu, X.; Javed, A.R.; Jalil, Z.; Kifayat, K. A comprehensive survey of AI-enabled phishing attacks detection techniques. Telecommun. Syst. **2021**, 76, 139–154. [Google Scholar] [CrossRef] [PubMed].

[3] M.; Mirza, S. Phishing Attacks Detection using Machine Learning and Deep Learning Models. In Proceedings of the 2022 7th International Conference on Data Science and Machine Learning Applications (CDMA), Riyadh, Saudi Arabia, 1–3 March 2022. [Google Scholar] [CrossRef].

[4] Aljabri, M.; Altamimi, H.S.; Albelali, S.A.; Al-Harbi, M.; Alhuraib, H.T.; Alotaibi, N.K.; Alahmadi, A.A.; Alhaidari, F.; Mohammad, R.M.A.; Salah, K. Detecting Malicious URLs Using Machine Learning Techniques: Review and Research Directions. IEEE Access **2022**, 10, 121395–121417. [Google Scholar] [CrossRef].

[5] Vayansky, I. and Kumar, S., "Phishing – challenges and solutions.", Computer Fraud & Security, vol 2018, no. 1,pp. 15-20, January 2018.

[6] Vahid Shahrivari, Mohammad Mahdi Darabi, Mohammad Izadi, "Phishing Detection Using Machine Learning Techniques," unpublished.

11.1 Source Code

App.py:-

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

app=Flask(__name__)

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

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

@app.route('/predict')
def predict():
    return render_template('final.html')

#Fetches the URL given by the URL and passes to inputScript

@app.route('/y_predict', methods=['POST'])

@app.route('/y_predict', methods=['POST'])
def y_predict():
    url = request.form['url']
    checkprediction = featurextraction.main(url)
    prediction = model.predict(checkprediction)
    print(prediction)
    output=prediction[0]
    if output == 1:
        pred = "Your are safe!! This is a Legitimate Website."
    else:
        pred = "You are on the wrong site. Be cautious!"
    return render_template('final.html', prediction_text=pred, url=url)


# Run the Flask application
if __name__ == '__main__':
    app.run(debug=True)
```

Index.html:-

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width,

      initial-scale=1.0">

    <title>A web Page</title>

</head>

<body>

    <style>

         /* Container to position image and text */


         .body{

            background-image: url("static/2.png");

         }

         .image-container {

            position: relative;

            display: inline-block;

        }


        .image-container button{

            position:absolute;

            background-color: #39AEA9;

            font-size: 20px;
    color: black;
    display: block;
    border-radius: 5px 5px 5px 5px;
    height: 50px;
    width: 125px;
    bottom: 420px;
    left: 649px;
}
/* Style for the image */
.image-container img {
    width: 100%;
    height: auto;
    display: block;
}
/* Style for the overlay text */
.image-container .overlay-text {
    position: absolute;
    top: 105px;
    left: 475px;
    padding: 5px;
    font-size:40px;
    color: black;
    font-weight: bold;
}

.image-container .overlay-text-body {
    position: absolute;
    top: 170px;
    left: 420px;
    background-color: #ECF87F;
    padding: 5px;
    font-size:30px;
    color: red;
```

```
            font-family: Georgia, 'Times New Roman', Times, serif;
            font-display: fallback;
        }

    </style>
    <div class="image-container">
        <img src="static/2.png" alt="My Image">
        <div class="overlay-text">PHISHING DETECTION</div>
        <div class="overlay-text-body">
            !!Be aware of what's happening around you!!
        </div>
        <a href="/predict">
        <button>Let's Check!</button></a>
    </div>
</body>
</html>
```