

# User-Centric Phishing URL Detection Tool Powered by Interpretable Machine Learning Model

LINKSCOPE

scansearch

INTRODUCING, LINKSCOPE

AI Powered URL Scanner

ManualAutomatic

Manual Mode

Manually input individual URLs that you wish to scan.

https://www.youtube.com/

Add

This URL has already been added

https://www.youtube.com/watch?v=5ZdLvW8RWY

https://www.youtube.com/

Submit

LINKSCOPE | Developed by Senior Synergy

LINKSCOPE

scansearch

https://creations.mtdv.me/articles/QH5UTjOjYZ

Redirected to: https://creations.mtdv.me/articles/QH5UTjOjYZ

Safe

Google Safe Browsing ✓

Wed Apr 24 2024

The website's domain name is managed by NameCheap, Inc. The main IP address is 172.67.179.32. The website's SSL certificate was issued on 7/11/2021, 5:11:45 PM and is valid until 7/11/2024, 5:11:45 PM.

Our model has classified this URL as "safe," with a trust score of 2.71 out of 5. Additionally, Google Safe Browsing has classified the URL as **safe**.

Model Result

URL Scanned

https://creations.mtdv.me/articles/QH5UTjOjYZ

172.67.179.32

More Details >

Probability: 45.71%

Verdict: safe

Trust Score: 2.71

Date & Time Created: Wed Apr 24 2024 12:10:53 GMT+0000 (Coordinated Universal Time)

Extracted Features

Domain Length 

Address Bar

Count the characters in the hostname string.

17

WWW 

Address Bar

If the URL has 'www' as the subdomain, then return 0; otherwise, return 1.

true

LINKSCOPE

scansearch

SEARCH

Explore our database of previously scanned URLs using our Search feature.

Enter search terms...

Apply

Sort By: ☐ Older ☐ Newer ☐ A-Z ☐ Z-A

URL Submission	Status
https://creations.mtdv.me/articles/QH5UTjOjYZ https://creations.mtdv.me/articles/QH5UTjOjYZ	safe
www.google.com https://www.google.com/7gws_rdi-sdl	safe
www.google.com https://www.google.com/7gws_rdi-sdl	safe
www.google.com https://www.google.com/7gws_rdi-sdl	safe
https://next789n.com/?token=dPmIE7FSYG... https://next789n.com/?token=dPmIE7FSYGODag0e	suspicious
www.google.com https://www.google.com/7gws_rdi-sdl	safe
www.google.com https://www.google.com/7gws_rdi-sdl	safe

# Engineering Project Final Report

## User-Centric Phishing URL Detection Tool Powered by Interpretable Machine Learning Model

### Submitted by

PAVARIT WIRIYAKUNAKORN	6338133721
PALIDA YINGWATCHARA	6338140021
THITI SRIKAO	6338052521

### Approved by:

**Project Advisor** : DR. SIRIN NITINAWARAT

**Project Co-Advisor** : ASST. PROF. SUKREE SINTHUPINYO

**Project Committee Member** : DR. AUNG PYAE

**Project Committee Member** : ASST. PROF CHAIYACHET SAIVICHIT

# ABSTRACT

In the ever-evolving landscape of cybersecurity, the proliferation of phishing attacks poses a significant threat to individuals and organizations worldwide. While existing phishing detection tools predominantly focus on technical aspects, there exists a notable void in addressing the informational needs of users beyond numerical data. To bridge this gap, this report represents the development of a user-centric phishing URL detection tool empowered by interpretable machine learning (ML) models. The objective is twofold: to categorize phishing links effectively and to provide users with contextual understanding and educational resources to enhance their online safety. By integrating insights from interdisciplinary fields and leveraging advanced ML techniques, the research endeavors to empower users with actionable insights and foster informed decision-making in the digital realm. Through a methodology encompassing feature engineering, algorithm selection, and model development, we aim to deliver an interpretable model with high performance, thereby contributing to a more secure digital environment. The result of the project represents a user-centric web application phishing detection tool.

# TABLE OF CONTENTS

<b>1. INTRODUCTION</b>	<b>1</b>
1.1 Personal Safety Online	1
1.2 Phishing Threats VS. Existing Tools	1
1.3 Taking Actions	1
<b>2. BACKGROUND</b>	<b>2</b>
<b>3. DESIGN REQUIREMENTS</b>	<b>3</b>
3.1 Model Design Requirements	3
3.2 Web application Requirements	3
3.2.1 Functional Requirements	4
3.2.2 Non-Functional Requirements	4
<b>4. DESIGN DESCRIPTION</b>	<b>5</b>
4.1 Overview	5
4.2 Detailed Description	6
4.2.1 Model Development	6
4.2.2 Web Application	7
4.3 Use	12
4.3.1 Scanning URLs	12
4.3.2 Exploring Previously Scanned URLs	14
4.3.3 Exploring URL Information and Its Associated Results	15
4.3.4 User Interface Responsiveness	15
<b>5. EXPERIMENTAL SETUP AND METHODS</b>	<b>16</b>
5.1 Experimental Setup	16
5.1.1 Model Experimental Setup	16
5.1.2 Web Application Usability Testing Setup	20
5.2 Experimental Methods	21
5.2.1 Model Methodology	21
5.2.2 Web Application Usability Testing	21
<b>6. RESULTS AND DISCUSSION</b>	<b>23</b>
6.1 Results	23
6.1.1 Model Results	23
6.1.2 Web Application Usability Testing Result.	25
6.2 Discussion	27
6.2.1 Model Discussion	27
6.2.2 Web Application Usability Test Discussion	30
<b>7. PROJECT GLOBAL IMPACT</b>	<b>31</b>
<b>8. CONCLUSIONS</b>	<b>31</b>
8.1 Assessment	31
8.2 Next Steps	31
<b>9. REFERENCES</b>	<b>32</b>

# LIST OF FIGURES

Figure 4.1: Sequence diagram shows how URLs are processed from user input.	5
Figure 4.2.1: MLOps process for selecting the model for web application.	6
Figure 4.2.2: Overview of the application structure and technologies used in each part.	7
Figure 4.2.3: A screenshot of the scan page of the application.	8
Figure 4.2.4: Different states the scanner interface can be in.	8
Figure 4.2.5: A screenshot of the search page of the application.	9
Figure 4.2.6: A screenshot of the result page of the application.	9
Figure 4.2.6: Model deployment workflow.	10
Figure 4.2.7: A diagram of the overall structure of the backend application and how it interacts with other parts of the application.	10
Figure 4.2.8: Entity relationship diagram.	11
Figure 4.3.1: The URLs are entered into the list and are ready to be scanned.	12
Figure 4.3.2: An example usage of the automatic URL detector.	12
Figure 4.3.3: Submission results page for summarizing all the results submitted	13
Figure 4.3.4: Result page for displaying information about the URL and the scan results	13
Figure 4.3.5: Search page of the application.	14
Figure 4.3.7: URL information page shows the URL's information and previous scans.	15
Figure 4.3.8: Example of how the UI changes responsively to a narrower screen.	15
Figure 5.2.1: Machine Learning Experiment Process	21
Figure 6.2.1: Feature importance of selected model	28
Figure 6.2.2: ROC curves of selected model	29

## LIST OF TABLES

Table 3.1: Model Design Requirement	3
Table 3.2.1: Functional Requirements	4
Table 3.2.2: Non-Functional Requirements	4
Table 5.1.1 : Address bar based features explored.	16
Table 5.1.2 : HTML/DOM Structure based features explored.	17
Table 5.1.3 : Abnormality based features explored.	18
Table 5.1.4 : List of domain based features explored.	18
Table 6.1.1 : Classification Results of All Extracted Features	23
Table 6.1.2: Classification Results for Feature Selection by RFECV Method	24
Table 6.1.3: Classification Results for Feature Selection by Particle Swarm Optimization	24
Table 6.1.4: Classification Results for Feature Selection by Embed Method	25
Table 6.1.5: Result from the questionnaire.	26
Table 6.2.1: Comparisons of Results With Other Work	29

# **1. INTRODUCTION**

In response to the increasing threat of phishing attacks and the need for user-centric cybersecurity solutions, this project aims to develop a comprehensive phishing detection tool powered by interpretable machine learning models. By prioritizing user understanding and education, the tool seeks to empower individuals to make informed decisions online, enhancing overall cybersecurity awareness and mitigating the risks associated with phishing attacks.

## **1.1 Personal Safety Online**

In today's digitally interconnected world, the prevalence of online threats, particularly phishing attacks, poses significant challenges to individuals' cybersecurity. Phishing attacks, often disguised as legitimate communications, aim to deceive unsuspecting users into divulging sensitive information such as passwords, financial details, or personal data.

## **1.2 Phishing Threats VS. Existing Tools**

While existing tools focus predominantly on technical aspects of phishing link detection, there exists a notable gap in addressing users' informational needs comprehensively. Users, often lacking deep technical knowledge, require more than just numerical data or verdicts; they seek contextual understanding and educational resources to navigate the digital landscape safely.

## **1.3 Taking Actions**

Recognizing this critical gap, our project endeavors to develop a user-centric phishing URL detection tool enhanced with a capability of interpretable machine learning models. By categorizing phishing links and providing relevant informational content tailored to identified threats, the tool not only can enhance users' awareness but also to equip them with the knowledge necessary for making informed decisions online.

Through this interdisciplinary endeavor, drawing on insights from Computer Programming, ICE Capstone, and User Interface Design, we strive to bridge the gap between technical expertise and user education, ultimately contributing to a safer and more secure online environment for all users.

## 2. BACKGROUND

**Phishing Threat Landscape** In the cybersecurity domain, the ongoing threat of phishing requires constant attention and innovative countermeasures. Based on Current research in phishing detection, the navigation aims at the complex balance between malicious tactics and defensive strategies. Researchers utilize advanced computational methods, particularly machine learning techniques to strengthen digital defenses. This involves identifying established phishing patterns and proactively detecting threats. By leveraging these sophisticated tools, the cybersecurity community aims to stay ahead in the battle against cyber threats and contribute to a safer digital environment.

Phishing poses a significant threat online, exploiting various vulnerabilities that make individuals susceptible to deception. These vulnerabilities include a lack of awareness about Uniform Resource Locators (URLs) and their functions, difficulty in distinguishing trustworthy URLs from potential threats, URL concealment or redirection, accidental clicks, and the inability to differentiate between legitimate and phishing URLs (Büber et al., 2017). To address this multifaceted challenge, various approaches have emerged, including heuristic analysis, visual similarity assessment, list-based evaluations, and advanced techniques rooted in machine learning and deep learning. Our phishing detection tool strategically focuses on machine learning-based approaches, leveraging the power of algorithms to dynamically adapt to evolving phishing tactics and enhance its capacity to detect intricate patterns and anomalies (Safi & Singh, 2023).

In a literature review comparing machine learning algorithms for phishing detection, Random Forest demonstrated the highest accuracy when optimized with feature selection, outperforming other algorithms. This superior performance can be attributed to Random Forest's ability to handle large datasets and mitigate overfitting by aggregating multiple decision trees, making it the preferred algorithm for its robustness against evolving phishing techniques (Almseidin et al., n.d.).

Currently, there are no comprehensive phishing detection tools designed specifically for the context of Thailand. This research aims to address that gap by focusing on user-centric approaches and educational content. Unlike existing tools, this tool emphasizes the educational context, providing users with relevant content to raise security awareness. Additionally, the tool utilizes interpretable machine learning models to effectively identify and categorize phishing links. From an academic perspective, this research prioritizes both interpretability and accuracy in machine learning, achieved through experiments with varied feature extraction approaches. The ultimate goal is to create a comprehensive solution that not only detects phishing threats but also enhances the user experience by offering a deeper understanding of these threats through educational resources.



### 3. DESIGN REQUIREMENTS

There is currently a lack of tools focusing on the interpretability of phishing URLs, along with educational aspects. Furthermore, in the Thai context, people are currently facing this issue, yet there are no prevention tools available in Thailand. Therefore, our aim is to provide interpretable models with high performance, incorporating an educational context in web application.

#### 3.1 Model Design Requirements

The model's design for detecting phishing URLs must meet specific accuracy and quality benchmarks. The goal is to create a reliable model that minimizes errors; therefore, a certain criteria has been set to ensure that the detection process is reliable and effective.

**Table 3.1:** Model design requirement.

Requirement	Target Value
Accuracy	$\geq 90\%$
Precision	$\geq 90\%$
Recall	$\geq 80\%$
False Positive Rate	$< 5\%$

According to the (Safi & Singh, 2023) research, most of the top models achieve over 90 percent accuracy. This project, therefore, aims for an accuracy and precision rate of 90% or higher to ensure that phishing URLs are accurately identified while minimizing false positives. Additionally, maintaining a recall rate above 80% is crucial for capturing a significant proportion of phishing URLs. To maintain user confidence, the system must also keep the false positive rate below 5%, thus ensuring the quality of detection.

#### 3.2 Web application Requirements

To create a tool that can reliably facilitate the use of the trained machine learning models, the web application must have several functional and non-functional requirements.

### ***3.2.1 Functional Requirements***

Functionally, the application must allow users to submit URLs for analysis and determine whether they are phishing or not. To achieve this, the application will have to identify which features in the URL contribute to its classification as phishing and display results based on relevant factors. Furthermore, adding informational context can help users understand why a specific URL is classified as phishing. Additionally to offer values post-scan, providing users with a record of past submission by maintaining a history of analyzed URLs is also required.

**Table 3.2.1:** Functional Requirements.

<b>Requirements</b>	<b>Functions</b>
URL Submission	Users shall submit URLs for phishing analysis.
URL Analysis	The Application must be able to be used to determine if URLs are a phishing attempt or not. The tool shall be able to show a list of factors (features) in the submitted URL that contribute to its classification and display classification results based on relevant factors
URL Feature Explanation	The tool shall explain the meaning of features contributing to a phishing URL in an informational context.
URL History Analysis	The tool shall record the history of URL analyzes.

### ***3.2.2 Non-Functional Requirements***

The application must be able to respond to each request as fast as possible to ensure a responsive experience during the URL analysis process. The user interface should be simple, intuitive, and user-friendly, facilitating ease of use. Compatibility is also key, with the application designed to work seamlessly across various web browsers and devices. These requirements ensure that the application not only functions effectively but also provides a positive user experience.

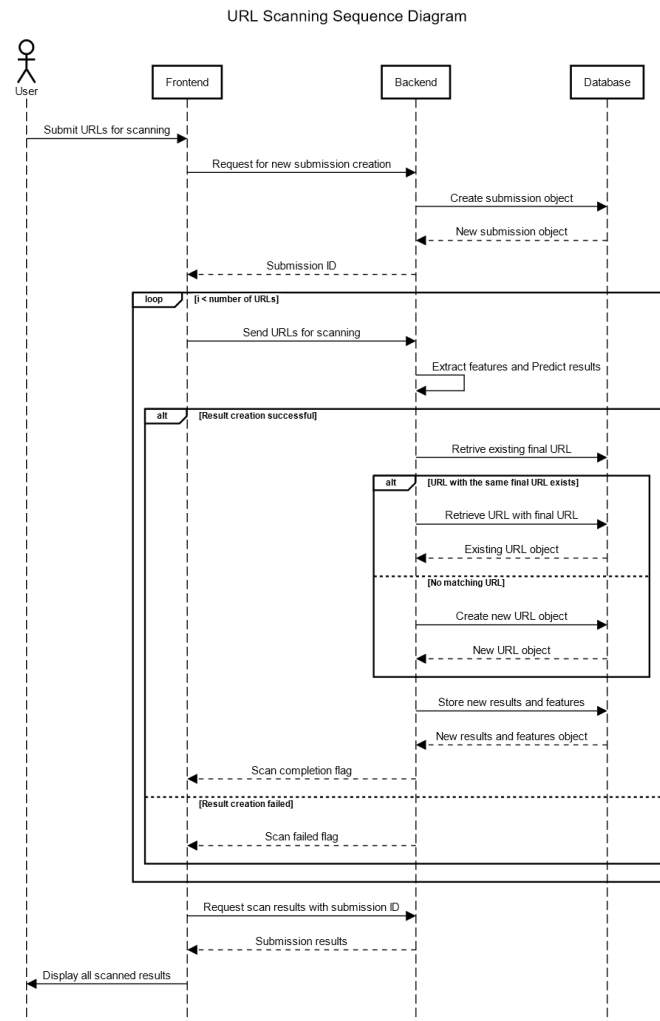
**Table 3.2.2:** Non-Functional Requirements.

<b>Requirements</b>	<b>Function</b>
Performance	The tool shall have quick response times for URL analysis.
Usability	The user interface shall be simple, intuitive and user-friendly.
Compatibility	The application shall ensure compatibility with various web browsers and devices

## 4. DESIGN DESCRIPTION

### 4.1 Overview

The design aims to provide an effective phishing detection solution through two main components: Model Development and Web Application. The Model Development involves training a machine learning model to distinguish between legitimate and phishing URLs, while the Web Application serves as the user interface for accessing the detection functionality.



**Figure 4.1:** Sequence diagram shows how URLs are processed from user input.

To utilize the whole system, the user can use common web browsers such as Mozilla Firefox, Google Chrome, Safari, or Microsoft Edge to access the web application. Then, the user can submit URLs for analysis. When the backend application receives URLs from the frontend application, the trained model is utilized to create prediction results and store them along with other useful information about the URL in the database. Finally, the results can be retrieved by the frontend application to be displayed to the user.

## 4.2 Detailed Description

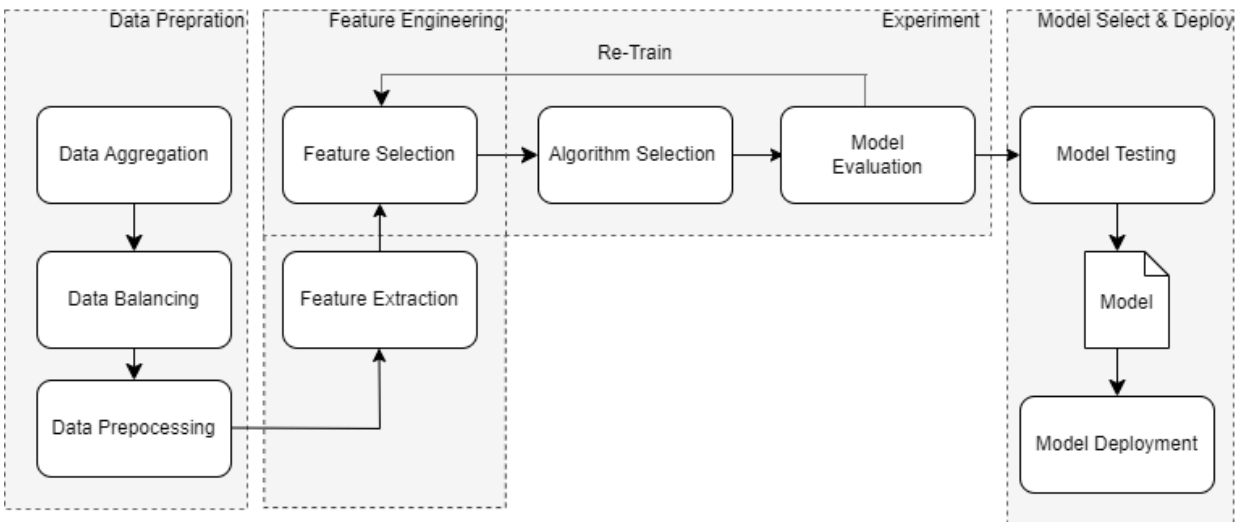
### 4.2.1 Model Development

In the Model Development phase, the goal is to develop a model that can meet the predefined requirements set initially. This process involves data preparation, feature engineering and experimentation, and model selection and deployment.

**Data Preparation.** The process begins with data preparation, which involves data aggregation, balancing and preprocessing. Then, feature extraction is performed in a Python environment using Jupyter Notebooks hosted on AWS Sagemaker. AWS SageMaker is chosen because of its high computing power, allowing for comprehensive URL extraction, as well as its robust security features.

**Feature Engineering and Experimentation.** The extracted features undergo experiments for feature selection and model training. The best model among all candidates is then selected for deployment in the web application. The chosen model uses the Random Forest algorithm and Particle Swarm Optimization for feature selection. The process of conducting experiments and the criteria for choosing the best model will be discussed later in Section 5.

**Model Selection and Deployment.** The selected model is serialized into a pickle format for later use in the backend application, which will serve as the interface between the frontend application and the database.



**Figure 4.2.1:** MLOps process for selecting the model for web application.

### 4.2.2 Web Application

The goal of developing the web application is to provide a way to be able to conveniently utilize the machine learning models. To accomplish this, a frontend application and backend application program are developed. The frontend application will allow users to interact with the input URLs for analysis, where the server-side component integrates the trained model and performs the URL classification.



**Figure 4.2.2:** Overview of the application structure and technologies used in each part.

**Frontend Application.** The frontend application needs to fulfill several key functionalities to provide a seamless user experience and facilitate interaction with the backend application. In this application, there are 3 main goals to achieve overall: create a tool that users can input URLs to be scanned, display those results, and allow users to explore other results that were previously created.

In addition to the main functionality, it is also important to create an optimal baseline for long term usage and development, taking into account what technology to use and the cost to deploy the application.

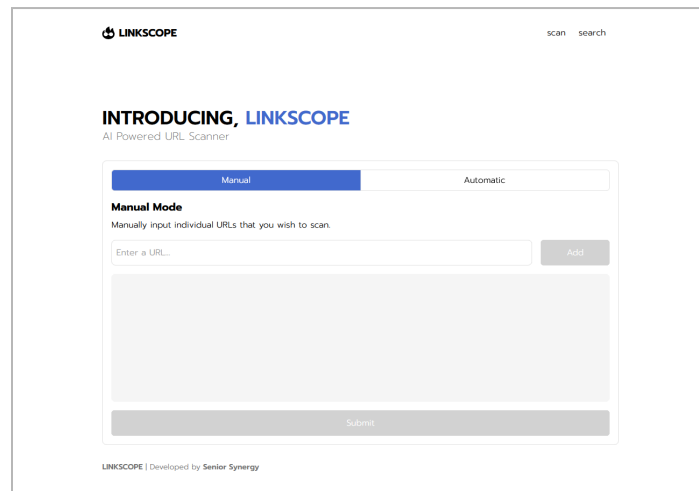
**Framework.** The frontend application is developed on NextJS framework to take advantage of its server-side rendering, routing, and other features it provides out of the box. Utilizing Typescript, static typing can further help improve code quality during the development process. To facilitate rapid development, Tailwind CSS is implemented to eliminate the need to write CSS classes for each individual UI component.

**Deployment.** The frontend application is deployed on Vercel, a platform that provides cloud services for deploying the frontend application. The service is chosen due to the ease of deployment and other useful features such as automatic scaling, continuous deployment, and built-in analytics.

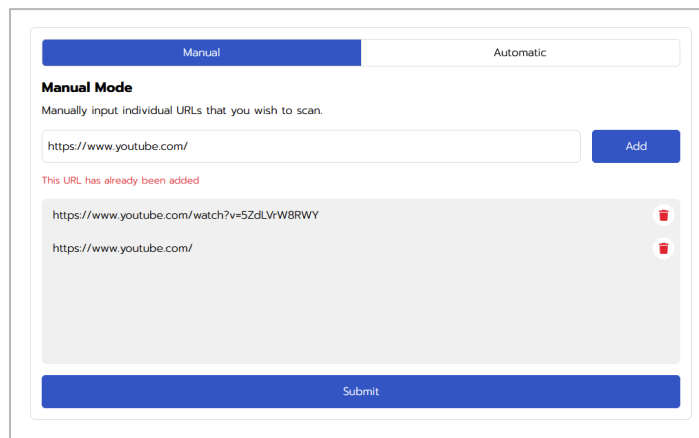
**User Interface Design.** The design of the application needs to follow concepts such as simplicity, clarity, and consistency. Furthermore, to improve accessibility, having a responsive design is crucial. The application must be able to adapt to different screen sizes and devices to provide a seamless user experience; therefore, considerations such as how to display elements in various screen sizes, both narrow and wide, organizing data in large amounts in pages, and more.

(Note: Screenshots of the application shown are from early version of the application, and they are subject to changes)

**URL Scan Page.** Concepts such as Miller’s Law and Gestalt Principles are also utilized to help create a more cohesive UI by reducing distance between elements and giving them appropriate sizing with appropriate grouping to improve ease of use.










**Figure 4.2.3:** A screenshot of the scan page of the application.



**Figure 4.2.4:** Different states the scanner interface can be in.

**Search Page.** In the application, simplicity and clarity is emphasized. The focus is to simplify the action that the user needs to take. This complies with Hick's Law, which states that lowering the number of choices that the user has to make can reduce the time it takes to complete the task. This page also shows a strong use of Jakob's Law, which focuses on the use of UI design with established conventions. In this case, search and filter UI follows the common structure that would be commonly found on other established websites or applications.

SEARCH		
Explore our database of previously scanned URLs using our Search feature.		
Enter search terms...		Apply
Sort By: <input type="radio"/> Older <input type="radio"/> Newer <input type="radio"/> A-Z <input type="radio"/> Z-A		
URL Submission	Date	Status
https://next789n.com/?token=dPmfE7FSYG0Dqg0e https://next789n.com/?token=dPmfE7FSYG0Dqg0e	 4/24/2024 10:24:16 AM	suspicious
www.google.com https://www.google.com/?gws_rd=ssl	 4/24/2024 10:13:53 AM	safe
www.google.com https://www.google.com/?gws_rd=ssl	 4/24/2024 10:13:27 AM	safe
https://www.facebook.com/wannayetmuay https://www.facebook.com/login/?next=https%3A%2F%2Fwww.facebook.com%2Fwannayetmuay	 4/24/2024 9:25:30 AM	safe
https://www.facebook.com/wannayetmuay https://www.facebook.com/login/?next=https%3A%2F%2Fwww.facebook.com%2Fwannayetmuay	 4/24/2024 9:16:12 AM	safe
https://linkscope-front.vercel.app/scan https://linkscope-front.vercel.app/scan	 4/24/2024 7:47:47 AM	suspicious
https://linkscope-front.vercel.app/scan https://linkscope-front.vercel.app/scan	 4/23/2024 6:24:48 PM	suspicious

**Figure 4.2.5:** A screenshot of the search page of the application.

**Result Page.** Other than design considerations applied in other parts of the application, this page also takes Miller's Law into consideration. Since there is a need to display a lot of information in this page, information of different groups are broken into different parts so that the user can focus on each part of the result without being overwhelmed.

LINKSCOPE

scan search

### https://www.facebook.com/wannayetmuay

Redirected to: https://www.facebook.com/login/?next=https%3A%2F%2Fwww.facebook.com%2Fwannayetmuay

Safe

Google Safe Browsing

Wed Apr 24 2024

The website's domain name is managed by RegistrarSafe, LLC. The main IP address is 157.240.217.35, located in United States. The website's SSL certificate was issued on 3/29/1997, 5:00:00 AM and is valid until 3/30/2032, 4:00:00 AM.

Our model has classified this URL as "safe," with a trust score of 3.33 out of 5. Additionally, Google Safe Browsing has classified the URL as safe.

Model Result

URL Scanned

https://www.facebook.com/login/?next=https%3A%2F%2Fwww.facebook.com%2Fwa...  
157.240.217.35

More Details >

Probability

33.41%

Verdict

safe

Trust Score

3.33

Date & Time Created

Wed Apr 24 2024 09:25:30 GMT+0000 (Coordinated Universal Time)

Extracted Features

Domain Length

Address Bar

Count the characters in the hostname string.

16

WWW

Address Bar

If the URL has 'www' as the subdomain, then return 0; otherwise, return 1.

false

**Figure 4.2.6:** A screenshot of the result page of the application.

**Backend Application.** The backend application facilitates all the requests from the frontend application for submitting URLs for phishing detection, and retrieving results. It also provides an environment for the model to be deployed and utilized, and bridges the connection between the frontend application and the database.

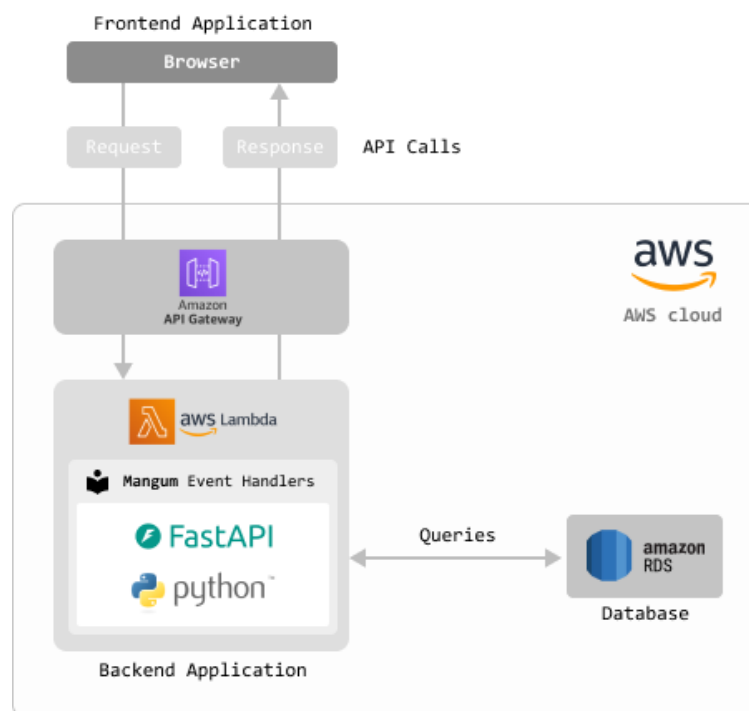
To utilize the model pickle file format is used to store the trained model to be used later on by the backend. When it is needed, features extracted act as an input to the model, and the model will produce the prediction result to be further processed.



**Figure 4.2.6:** Model deployment workflow.

**FastAPI and Model Integration.** The backend application utilizes FastAPI to create RESTful APIs for the frontend application. Since the model was developed in a python environment, using FastAPI unifies the need for a python environment in the backend application.

**Deployment.** The backend application is developed to be deployed on AWS Lambda. Due to the serverless nature of the service, there are various benefits such as cost efficiency, automatic scaling, and reduced operational overhead.



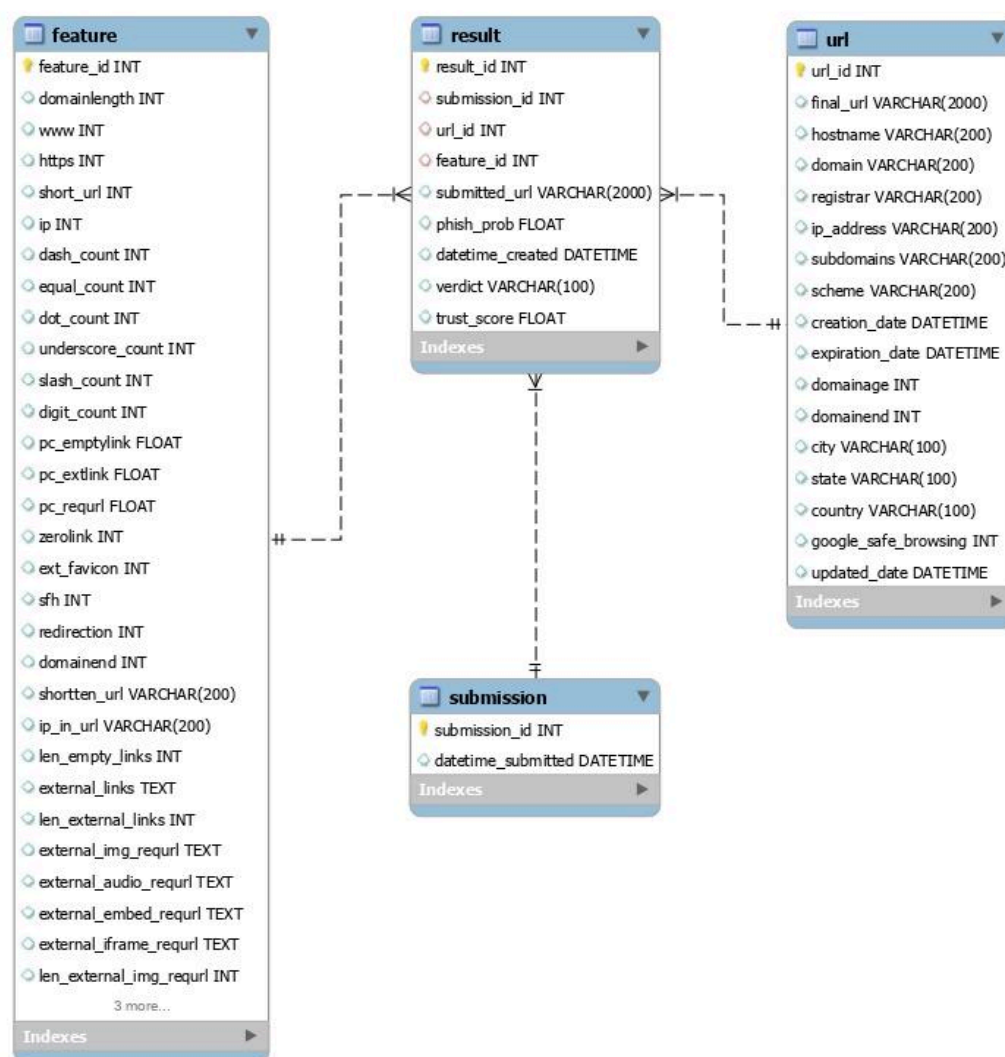
**Figure 4.2.7:** A diagram of the overall structure of the backend application and how it interacts with other parts of the application.



**Database.** To keep records of the results provided by the machine learning model, creating a database is required. This ensures that historical data of each unique URLs can be recorded and reviewed on demand. Information such as extracted features, model results and extra URL details are stored.

**MySQL.** To create the database, MySQL is chosen as the relational database management system. Due to its reliability, performance, and extensive features for managing structured data. It offers robust support for a wide range of applications.

**Deployment.** Amazon RDS is the first choice when it comes to the deployment of the database due to its ease of access and scalability without the overhead of traditional database administration tasks such as hardware provisioning, software installation, patching, and backups.



**Figure 4.2.8:** Entity relationship diagram.

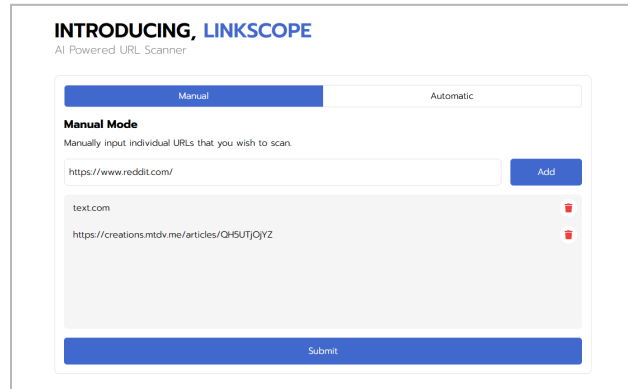
## 4.3 Use

The application provides a range of features, from scanning URLs and viewing the results, to exploring previous entries. To gain access to the application, the user can use common web browsers such as Mozilla Firefox, Google Chrome, Safari, or Microsoft Edge to access the web application.

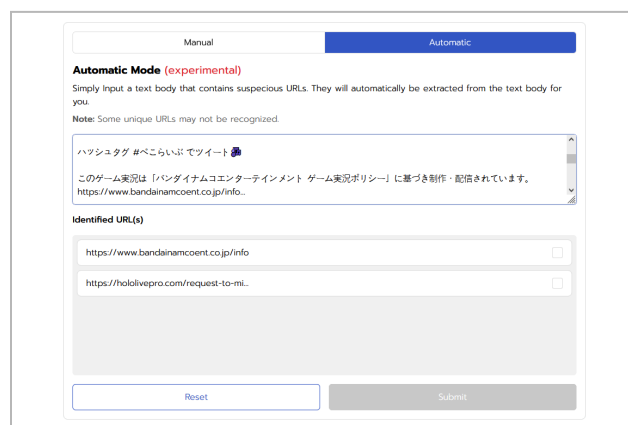
(Note: Screenshots of the application shown are from early version of the application, and are subject to changes)

### 4.3.1 Scanning URLs

After entering the application, the user will be greeted with the URL input form. This form allows the user to input up to 5 urls to be scanned at once. Additionally, the user can optionally try an “automatic detector” which will automatically detect URLs from a text body. This can help eliminate the need to individually extract the URL from a complicated text body.

The screenshot shows the 'INTRODUCING, LINKSCOPE' header with the subtitle 'AI Powered URL Scanner'. Below this are two tabs: 'Manual' (selected) and 'Automatic'. The 'Manual Mode' section instructs the user to 'Manually input individual URLs that you wish to scan'. It features a text input field containing 'https://www.reddit.com/' with an 'Add' button to its right. Below the input field is a list of URLs: 'text.com' and 'https://creations.mtdv.me/articles/QH5UTQjYz', each with a red square icon to its right. At the bottom of the form is a large blue 'Submit' button.

**Figure 4.3.1:** The URLs are entered into the list and are ready to be scanned.

The screenshot shows the 'Automatic Mode (experimental)' section. It instructs the user to 'Simply input a text body that contains suspicious URLs. They will automatically be extracted from the text body for you.' A note states: 'Note: Some unique URLs may not be recognized.' The text input field contains a tweet in Japanese: 'ハッシュタグ #ぺこらいぶでツイート このゲーム実況は「バンダイナムコエンターテインメント ゲーム実況ポリシー」に基づき制作・配信されています。 https://www.bandainamcoent.co.jp/info...'. Below the input field is a section titled 'Identified URL(s)' which lists two URLs: 'https://www.bandainamcoent.co.jp/info' and 'https://hololivepro.com/request-to-mi...'. At the bottom are 'Reset' and 'Submit' buttons.

**Figure 4.3.2:** An example usage of the automatic URL detector.

Once the user is ready, the user can initiate the scanning process by clicking the “submit” button, and the user will be redirected to the “result page.” If there are multiple URLs submitted in the same submission, the application will redirect the user to the “submission results” page

first to show the summary of all the URLs that the user has submitted; otherwise, the user will be redirected directly to the “result page” of that URL.

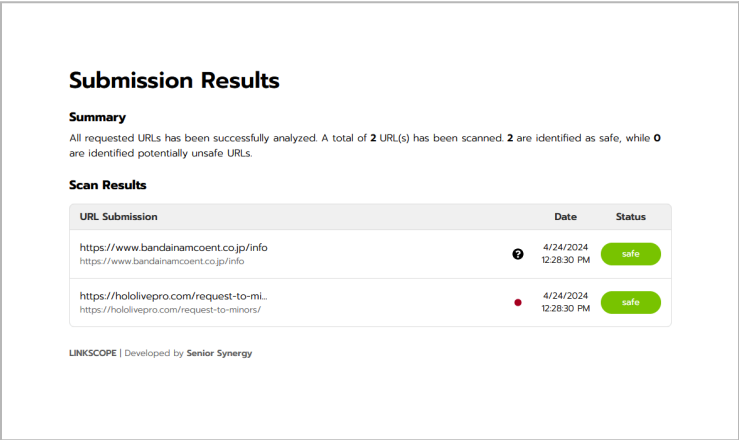


Figure 4.3.3: Submission results page for summarizing all the results submitted

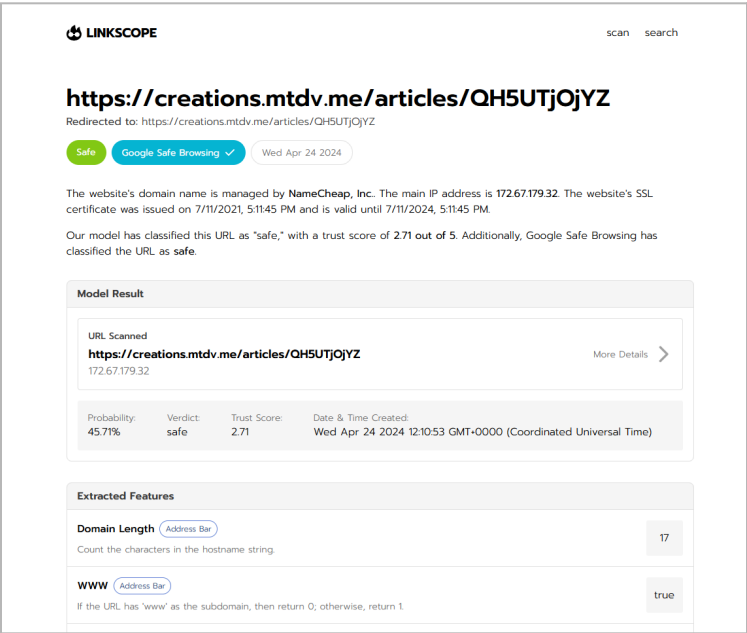
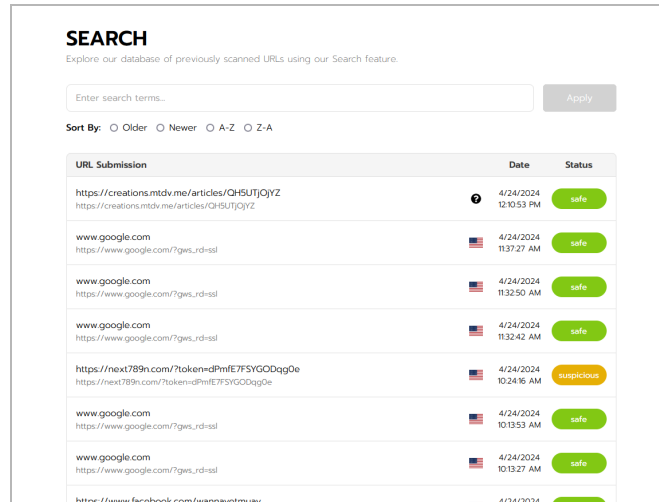


Figure 4.3.4: Result page for displaying information about the URL and the scan results.

In this page, the user can also explore different aspects of the URL such as the classification of the URL, basic information, the model results, and features that are extracted from the URL that were used to predict the results if the URL is considered safe or not.

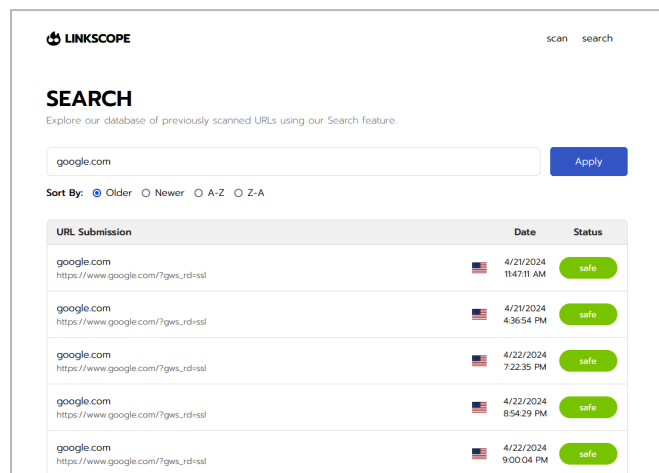
### 4.3.2 Exploring Previously Scanned URLs

The application allows users to explore previously scanned URLs in the application’s database. To explore the database, the users can navigate to the “scan” page. Then, a list of URLs will be displayed for the user to see along with the search bar with sort options.



**Figure 4.3.5:** Search page of the application.

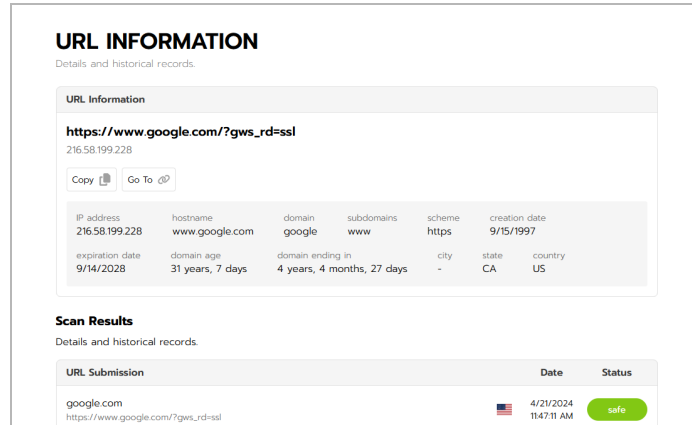
Here, the user can explore the list of results in a set of 10 results per page. The pagination of results is crucial as it helps reduce the number of results that have to be queried from the database and sent through HTTP request.



**Figure 4.3.6:** Search page displaying results with “google.com” in its URL. The results are sorted in a descending order according to its creation date.

### 4.3.3 Exploring URL Information and Its Associated Results

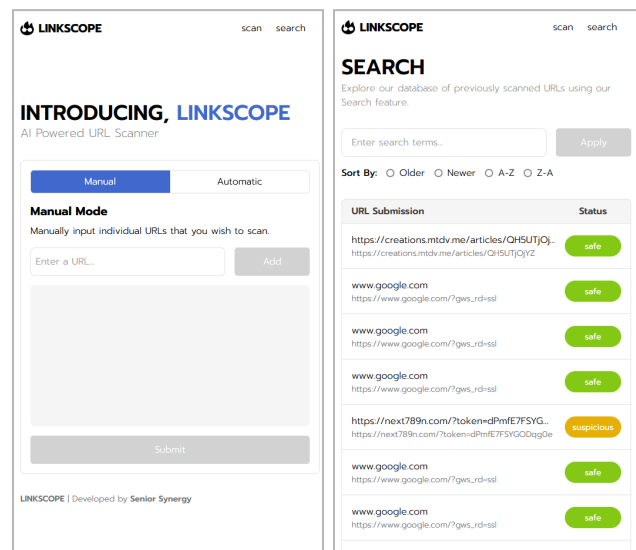
The application allows users to view a set of more detailed information of the unique URLs scanned such as its IP address, creation date, expiration date, county, and more. This page also shows a list of results that are associated with this URL.



**Figure 4.3.7:** URL information page shows the URL’s information and previous scans.

### 4.3.4 User Interface Responsiveness

In order to facilitate the use of the application across all types of devices, UI responsiveness is taken into consideration in the application design and development process. This ensures that the application will have a functioning UI whether the screen is small or big, narrow or wide.



**Figure 4.3.8:** Example of how the UI changes responsively to a narrower screen.

## 5. EXPERIMENTAL SETUP AND METHODS

### 5.1 Experimental Setup

The experimental setup consists of a systematic and comprehensive approach to develop an effective phishing detection model and gather insights from usability testing to understand users' needs. This section is divided into two parts: experimental setup and methods for both the model experiment and usability testing.

#### 5.1.1 Model Experimental Setup

For the Model Experiment Setup, four components must be defined before conducting the experiment to select the best model: dataset, features explored, feature selection approaches, and algorithms used.

**Dataset.** The dataset consists of a total of 70,000 URLs, with 35,000 phishing URLs collected from PhishTank and PhishScore, and 35,000 safe URLs derived from top SEO domain lists. Before proceeding, all duplicate URLs were removed to ensure data integrity. The data was then split into a training set (70 percent) and a test set (30 percent), making sure there was no correlation between the two sets to maintain the validity of the model evaluation. This resulted in 49,000 URLs in the training set and 21,000 URLs in the test set.

**Features Explored.** After obtaining all the URLs, feature extraction is performed. In this study, 29 features are selected for analysis, categorized into four types:

1. Address Bar-based: This category examines the characteristics of the URL address.

**Table 5.1.1** : Address bar based features explored.

Feature Name	Explanation
domainlength	count the characters in hostname string
www	If url has 'www' as subdomain then return 0, else 1
subdomain	If url has more than 1 subdomain then return 1, else 0
https	If url contains 'https' then return 0, else 1
http	If url contains 'http' then return 0, else 1
short_url	If url is a short url returns 1,else 0
ip	If url contains 'ip address' returns 1,else 0
@	Count the '@' characters in url
-	Count the '-' characters in url
=	Count the '=' characters in url

.	Count the '.' characters in url hostname
_	Count the '_' characters in url
/	Count the '/' characters in url
digit	Count the digits (0-9) characters in url
log	If url contains 'log' word in url then return 0, else 1
pay	If url contains 'pay' word in url then return 0, else 1
web	If url contains 'web' word in url then return 0, else 1
cmd	If url contains 'cmd' word in url then return 0, else 1
account	If url contains 'account' word in url then return 0, else 1

2. HTML/DOM Structure-based: This category involves analyzing the HTML or DOM structure of the web page.

**Table 5.1.2 :** HTML/DOM Structure based features explored.

Feature Name	Explanation
pemptylinks	Percentage of empty links  <b>Note :</b> An empty link is a hyperlink that does not lead to a different web page. When clicked, it typically results in staying on the current page or displaying a blank page. This can occur when the link's URL is missing or invalid.
pextlinks	Percentage of external links that direct you to another site with a different domain from the submitted URL.  <b>Note:</b> An external link is a hyperlink that directs users from one website to a different website with a different domain. When clicked, the link takes the user away from the current site and navigates them to a new site.
prequrl	Percentage of external resource URLs:  <b>Note:</b> An external resource URL refers to the web address of a resource, such as images, audio files, or embedded content, that is hosted on a different domain from the main or submitted URL.
zerolink	If the URL page has no links in the HTML body, return 1; otherwise, return 0.
extfavicon	If the favicon URL is from a different domain than the submitted URL, return 1; otherwise, return 0.

submit2email	If the html page contains "\b(mail\( \) mailto:?)\b" then return 1, else 0
sfh	SFHs that contain an empty string or “about:blank” or lead to different domain sites from submitted url, like form['action'] == "" or form['action'] == "about:blank" then return 1, else return 0

3. Abnormal-based: This category assesses the abnormal behavior of the submitted URL.

**Table 5.1.3 :** Abnormality based features explored.

Feature Name	Explanation
redirection	<p>If clicking the submitted URL results in a redirection to another URL, return 1; otherwise, return 0.</p> <p>For example, clicking www.eabc1255.com and being redirected to www.eabc5255.com.</p>

4. Domain-based: This category considers factors related to domain registration and expiration dates.

**Table 5.1.4 :** List of domain based features explored.

Feature Name	Explanation
domainage	The difference between expiration time and creation time (Domain_Age), if the domain age is less than 6 month then return 1, else return 0
domainend	Calculate the difference in days between the current date and the expiration date (registration length). If the difference is less than or equal to one year, return 1; otherwise, return 0.



**Feature Selection Approaches.** The purpose of feature selection is to select a subset of relevant features from a large number of available features to achieve similar or even better classification performance than using all features.

### *1. Recursive Feature Elimination with Cross-Validation (RFECV)*

Recursive Feature Elimination with Cross-Validation (RFECV) is a robust technique for feature selection that combines recursive feature elimination with the power of cross-validation. The process begins by building a model with all available features and then recursively removing the least significant features, one by one. At each step, the model is evaluated using cross-validation to determine its performance. This approach helps to identify the optimal set of features that contributes to the best model performance while avoiding overfitting. RFECV is particularly useful in high-dimensional datasets, allowing researchers to focus on the most impactful features for training their models.

### *2. Particle Swarm Optimization (PSO).*

Particle Swarm Optimization (PSO) is a population based technique to address feature selection problems in this project due to better representation, capability of searching large spaces, and being less expensive computationally. In Particle Swarm Optimization (PSO), a group of candidate solutions, known as a swarm, is represented by particles within a defined search space. The algorithm begins by randomly initializing the position of each particle. The particles then traverse the search space, adjusting their positions based on their own individual experiences, aiming to find the optimal solution.

### *3. Random Forest Feature Selection.*

Random Forest Feature Selection is an embedded method that uses the importance scores from a Random Forest model to select the most relevant features. As the model is trained, each feature's contribution to prediction is assessed by its frequency in splitting decision tree nodes. Features with higher scores are deemed more significant. This approach automatically identifies key features, reducing the need for manual selection, and is especially effective in high-dimensional data scenarios.

**Algorithms.** Algorithm selection involves choosing the most suitable algorithms for a specific task based on their performance characteristics. The following are the selected algorithms in three different approaches

### *1. Random Forest (RF) Algorithm*

Random Forest is recognized for offering the highest accuracy among machine learning algorithms and is frequently employed in phishing URL detection. Random forests address the problem of overfitting in decision trees by creating multiple trees, each trained on random subsets of features. Data for prediction is passed through all the trees, and their outputs are combined through a simple voting mechanism. This method reduces overfitting, though it requires more computation time than a single decision tree.

## *2. LightGBM*

Identified as the second-highest performer in the research. It is considered to be one of the best choices in various machine learning applications, including phishing URL detection. LightGBM is a fast and efficient tool for building machine learning models. Created by Microsoft, it's designed to work well with large datasets and lots of features. LightGBM uses a special way of picking which data points to focus on and groups similar features together to speed up training. Because of its speed and accuracy, it's popular among data scientists and often used in competitions and industry projects.

## *3. Support Vector Classification (SVC)*

Support Vector Classification (SVC) is a machine learning method used to classify data into different categories. It's a part of the larger family of Support Vector Machines (SVMs). SVC works by finding the best dividing line (or hyperplane) that separates data into its respective classes. It does this by maximizing the margin, which is the distance between the line and the nearest data points from each class. This approach helps ensure that the classification is as robust as possible, even with complex datasets.

### ***5.1.2 Web Application Usability Testing Setup***

In conducting usability testing for our phishing detection application, our primary objective is to assess its effectiveness, efficiency, and user satisfaction. To achieve this, the test operator will enlist the participants, including friends or family members, to ensure a diverse range of perspectives. The testing environment can be either online, using platforms such as Zoom and Discord, or in-person for face-to-face interaction. Participants will begin with an introductory session where they provide demographic information and details about their prior experience with similar applications.

During the testing session, participants will be asked to perform specific tasks such as scanning a URL for phishing, submitting a URL, and interpreting the results provided by the application. Additionally, they will be tasked with searching within the application. We will meticulously record task completion times, identify any errors encountered, and evaluate the effectiveness and efficiency of the application in fulfilling its intended purpose. Following the tasks, we will conduct post-interviews to gather feedback on usability, user satisfaction, and areas for improvement, ensuring that the application meets the needs and expectations of its users.

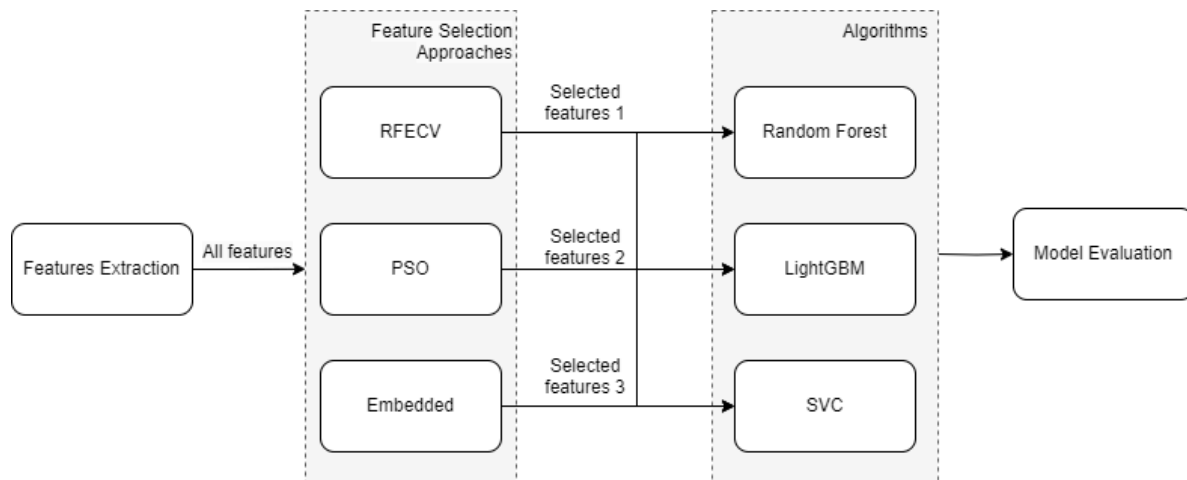
## 5.2 Experimental Methods

### 5.2.1 Model Methodology

According to Figure 4.2.1, the experimental process for selecting the best model begins after data preparation and feature extraction. The experiment encompasses three main processes: feature selection, algorithm selection, and model evaluation.

For feature selection, as stated in Section 5.1.1, RFECV, PSO, and Embedded methods are used to select optimal features from the total of 29 features derived from the feature extraction process. This approach yields three optimal subsets of features, one from each selection method. Next, three algorithms—Random Forests, LightGBM, and SVC—are used to train models on each of these subsets.

Finally, model evaluation is conducted to select the best model and determine which subset of features and which algorithm produces the best performance. This assessment uses a range of performance metrics, including accuracy, precision, recall, and false positive rate. This comprehensive evaluation process allows for an accurate and reliable assessment of the model's effectiveness in detecting phishing attempts. The criteria for choosing the best model primarily focus on accuracy and training time.



**Figure 5.2.1:** Machine Learning Experiment Process.

### 5.2.2 Web Application Usability Testing

The web application usability testing aimed to assess the effectiveness, efficiency, and user satisfaction of the phishing detection application. Recruitment for the test involved inviting participants, primarily friends and family, who were briefed on the test's purpose and invited to participate online or in person. During the test execution, participants were asked to complete two tasks: using a scan function and a search function.

In the scan function, the test operator provided a URL, submitted it to the phishing detection application, and interpreted the results. The search function required participants to find information about the phishing web on the application's database. Task completion time and number of errors were recorded, and participants were evaluated based on their performance.

Following functionality testing, participants filled out a questionnaire form using a Likert scale to assess various aspects of usability. They rated their self-efficiency, ease of use, user-friendliness, behavioral intention, and security awareness.

**Self Efficiency.** The goal is to determine if users can use the application skillfully, and whether they feel the need for a specialist to help using it. This will help evaluate the level of confidence when the user is using the application.

**Ease of Use.** The goal is to determine if users can think that the overall design of the application is easy and straightforward enough to navigate during the real usage.

**User-friendliness.** The goal is to determine if the design of the application is well-organized and easy to find different elements, as well as how visually appealing the user interface is to the average user.

**Behavior Intention.** The goal is to see if users are likely going to be using the application in the future.

**Security Awareness.** The goal is to evaluate if the application itself has provided the value that it aimed to provide to the users regarding phishing URL detection and the informational content that the application provides.

Additionally, post-interviews were conducted to gather qualitative feedback. Participants rated their overall experience, identified useful features or redundancies, commented on task completion times, described any encountered issues, expressed confidence in using the application in the future, and provided suggestions for improvement.

Data from the questionnaire forms, task performance, and post-interviews were collected and analyzed to evaluate the application's usability. This involved analyzing Likert scale responses, identifying patterns in task completion times and errors, and extracting insights from participants' feedback to inform future improvements to the application.

## 6. RESULTS AND DISCUSSION

### 6.1 Results

This section presents the outcomes from the model experiment along with the results from website usability testing. User feedback and performance metrics were analyzed to gauge the overall effectiveness of the design. The insights from this analysis will be used to inform future improvements and guide the next steps for the project.

#### 6.1.1 Model Results

The results obtained from applying the machine learning algorithms with different feature selection approaches are outlined and discussed here. The analysis reveals how each algorithm performs and evaluates the role that different feature selection methods play in model accuracy. By comparing these results, we can identify the most effective approaches, providing guidance for future experiments and supporting ongoing refinement of our models.

In the first result, all features extracted are included to be used to predict the classification of the provided URL. As a result, the following results are obtained:

**Table 6.1.1** : Classification Results of All Extracted Features.

Algorithm	Accuracy	Precision	Recall	F1-score	Processing time (s)
Random Forest	96.49 %	96.81 %	96.14 %	96.49 %	11.75
LightGBM	95.61 %	96.07 %	95.11 %	95.61 %	0.41
SVC	94.41 %	95.67 %	93.03 %	94.41 %	40.22

The table 6.1.1 provides classification results using all extracted features with Random Forest achieving the highest accuracy at 96.49%, closely followed by LightGBM at 95.61%, and SVC at 94.41%. Processing time varies significantly, with LightGBM completing tasks in just 0.41 seconds, compared to 11.75 seconds for Random Forest and 40.22 seconds for SVC. Overall, Random Forest appears to be the most accurate, while LightGBM is the most efficient in terms of processing time

Modifying the number of selected features to 25, including, 'domainlength', 'www', 'subdomain', 'https', 'short\_url', '-', '=', '!', '\_', '/', 'digit', 'log', 'pay', 'web', 'account', 'pemptylinks', 'pcextlinks', 'pcrequrl', 'zerolink', 'extfavicon', 'submit2email', 'sfh', 'redirection', 'domainage', and, 'domainend,' the following results are obtained:

**Table 6.1.2:** Classification Results for Feature Selection by RFECV Method.

Algorithm	Accuracy	Precision	Recall	F1-score	Processing time (s)
Random Forest	96.43%	96.77%	96.08%	96.43%	12.29
LightGBM	95.60%	96.05%	95.11%	95.60%	0.29
SVC	94.44%	95.68%	93.08%	94.43%	31.46

The table 6.1.2 shows classification results after using the Recursive Feature Elimination with Cross-Validation (RFECV) method to select the top 25 features, eliminating features such as 'http', 'ip', '@', and 'cmd'. Despite reducing the feature set, Random Forest maintained high accuracy at 96.43%, while LightGBM saw a slight decrease in accuracy but significantly faster processing time of 0.29 seconds. SVC also saw a minor improvement in accuracy to 94.44%, with a reduced processing time of 31.46 seconds. This comparison indicates that Random Forest remains the most accurate algorithm, while LightGBM is the most efficient in terms of processing time, suggesting a trade-off between accuracy and speed when applying feature selection methods.

Modifying the number of selected features to 19, including, 'domainlength', 'www', 'https', 'short\_url', 'ip', '-', '=', '!', '/', '\_', 'digit', 'pemptylinks', 'pcextlinks', 'pcrequrl', 'zerolink', 'extfavicon', 'sfh', 'redirection' and 'domainend', the following results are obtained:

**Table 6.1.3:** Classification Results for Feature Selection by *Particle Swarm Optimization (PSO)*.

Algorithm	Accuracy	Precision	Recall	F1-score	Processing time (s)
Random Forest	96.13%	96.27%	95.97%	96.13%	11.49
LightGBM	95.31%	95.60%	94.99%	95.31%	0.27
SVC	94.17%	95.32%	92.91%	94.17%	29.73

The table 6.1.3 summarizes the results of classification after applying feature selection using the Particle Swarm Optimization (PSO) method, which reduced the number of features to 19. In this scenario, Random Forest again shows the highest accuracy, with a slight decrease to 96.13%. LightGBM follows closely with an accuracy of 95.31% but achieves the shortest processing time at just 0.27 seconds. SVC retains a relatively high accuracy of 94.17% with a processing time of 29.73 seconds. These results suggest that PSO-based feature selection can

lead to a faster overall processing time while maintaining strong accuracy across the algorithms, with LightGBM offering the best balance between accuracy and speed.

Modifying the number of selected features to 11, including, 'domainlength', 'www', 'https', '-', '/', 'digit', 'pemptylinks', 'pcextlinks', 'pcrequrl', 'zerolink', 'redirection,' the following results are obtained:

**Table 6.1.4:** Classification Results for Feature Selection by *Embed Method (Random Forest Feature Selection)*.

Algorithm	Accuracy	Precision	Recall	F1-score	Processing time (s)
Random Forest	95.13%	95.09%	95.17%	95.13%	10.38
LightGBM	94.58%	94.83%	94.31%	94.58%	0.22
SVC	92.55%	93.12%	91.88%	92.55%	24.29

The table 6.1.4 outlines the classification results after feature selection using the embed method (Random Forest Feature Selection), with the number of selected features reduced to 11. The Random Forest algorithm maintains an accuracy of 95.13%, with a processing time of 10.38 seconds. LightGBM closely follows with an accuracy of 94.58% and the shortest processing time of 0.22 seconds. SVC, while having the lowest accuracy at 92.55%, demonstrates an F1-score of 92.55%, with a processing time of 24.29 seconds.

### ***6.1.2 Web Application Usability Testing Result.***

The results from Usability testing average duration for completing each task ranged from 30 seconds to 1 minute, providing valuable insights into the clarity and user-friendliness of the application interface. Nonetheless, no errors were encountered during task completion, indicating that even novice users were able to navigate and utilize the application effectively.

**Table 6.1.5:** Result from the questionnaire.

Category	Question	Average Score	Category Average Score
Self efficiency	I can use the application skillfully	3.88	3.67
	I don't need specialist to help using the application	3.50	
	I feel confident using the application	3.63	
Ease of use	Using the application is easy and straightforward	3.75	4.04
	I find it easy to navigate the application	4.25	
	The steps to use the application are clear and simple	4.13	
User friendliness	The application has a user friendly interface	4.25	4.13
	The design of the application is visually appealing	4.00	
	The features in the application are well-organized and easy to find.	4.13	
Behavior Intention	I intend to use the application myself in the future	4.25	3.96
	I would recommend the application to my friends	4.13	
	I would likely choose the application over other similar tools in the future.	3.50	
Security Awareness	Using the application raises me awareness in security aspects	4.25	4.38
	I understand more about phishing websites after using the application	4.38	
	The application helps me recognize phishing website characteristics	4.50	

**Post-test Interview.** From the interview after the usability test, several suggestions for improvement were provided by the users.

**Terms and technicality.** The extracted features were found to be difficult to understand for novice users, suggesting the need for clearer explanations or user guidance. There was confusion regarding the feature extraction descriptions, particularly the statement "if..., then return 0; otherwise, return 1," with testers unsure whether the result on the right side indicated true or false. In addition to that, the users also expressed confusion over the term "URL" and suggested using a more user-friendly term like "link" in the scan page.

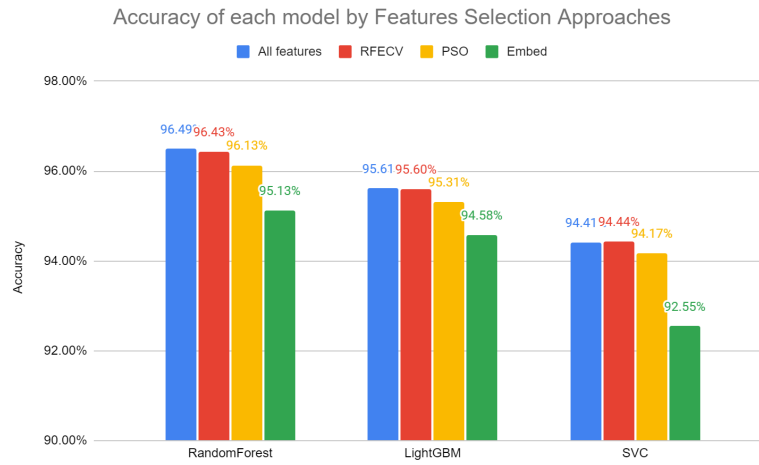
**Highlights on Important Information.** Some testers found that the trust score section was not prominent enough, recommending that it be made more noticeable within the application interface.

**Accessibility.** Testers recommended creating a Thai version of the application to improve user experience for local users.

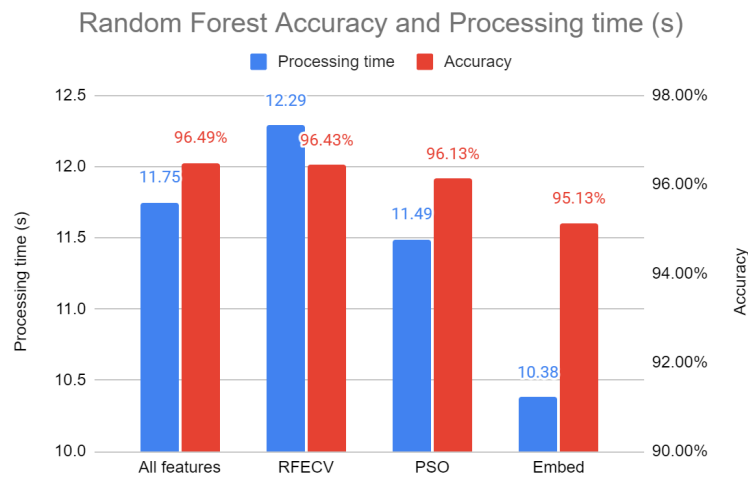


## 6.2 Discussion

### 6.2.1 Model Discussion



**Figure 6.2.1:** Feature importance of selected model.



**Figure 6.2.2:** Feature importance of selected model.

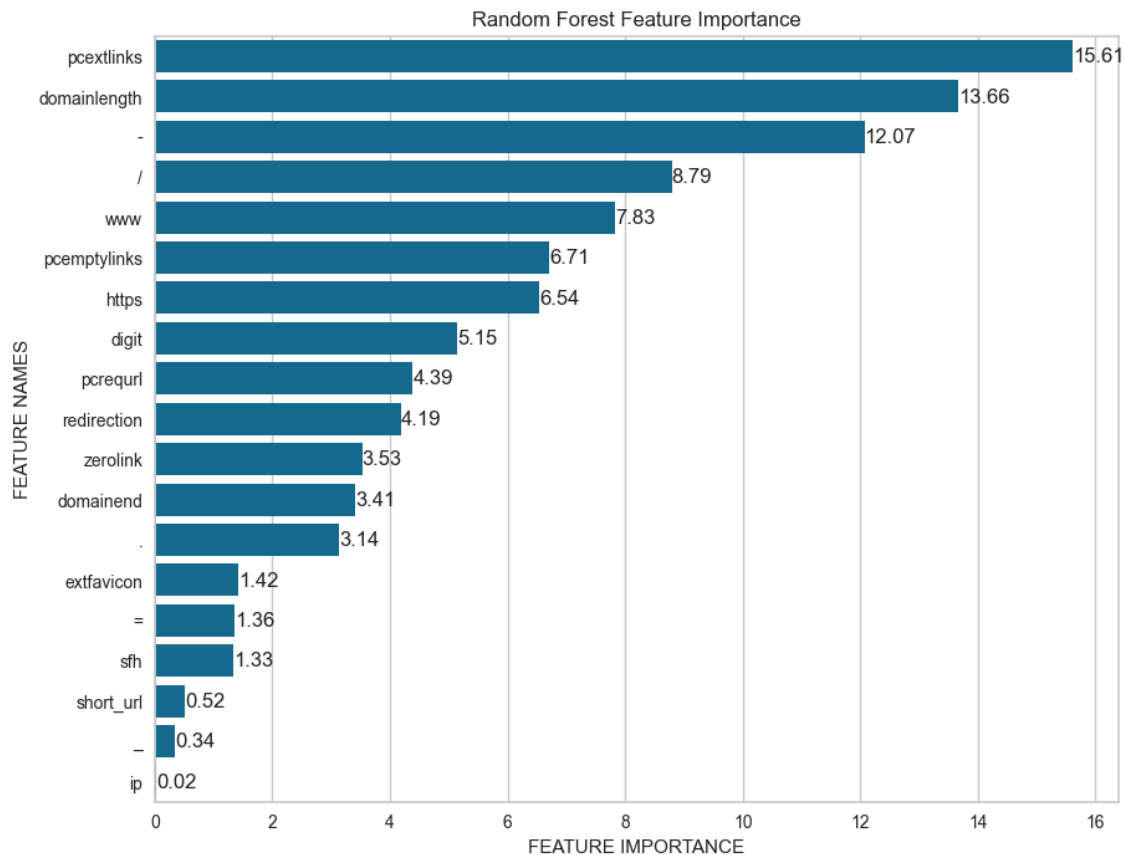
In order to select the best model for the web application, Random Forest is chosen for its high accuracy, leading among three algorithms. However, in the context of a web application, processing time is also a critical factor. When users submit URLs for analysis, the feature extraction process takes time, adding to the overall latency. This consideration makes it important to balance accuracy with processing time. While Random Forest with all features and with RFECV provide the highest accuracies at 96.49% and 96.43%, respectively, their longer processing times of 11.75 and 12.29 seconds are a drawback for web applications, where user experience depends on quick responses.

Given this context, Particle Swarm Optimization (PSO) and the Embed Method are left as potential feature selection approaches for Random Forest. Although the Embed Method offers a

shorter processing time of 10.38 seconds, its lower accuracy of 95.13% might not meet the requirements of a high-stakes web application. The PSO approach, on the other hand, achieves a balance, with an accuracy of 96.13% and a processing time of 11.49 seconds.

Although PSO with Random Forest doesn't have the highest accuracy or the shortest processing time, it provides a satisfactory trade-off. The moderate processing time still allows for a responsive user experience, while the accuracy remains quite high. This makes Random Forest with PSO an attractive choice for web applications where users submit URLs for feature extraction, ensuring a balance between accuracy and efficiency.

Therefore, Random Forest with PSO emerges as the best compromise, achieving 96.13% accuracy with a reasonable processing time of 11.49 seconds. This combination provides a balanced trade-off between accuracy and speed, making it an optimal choice for reliable performance without excessive computational demands.



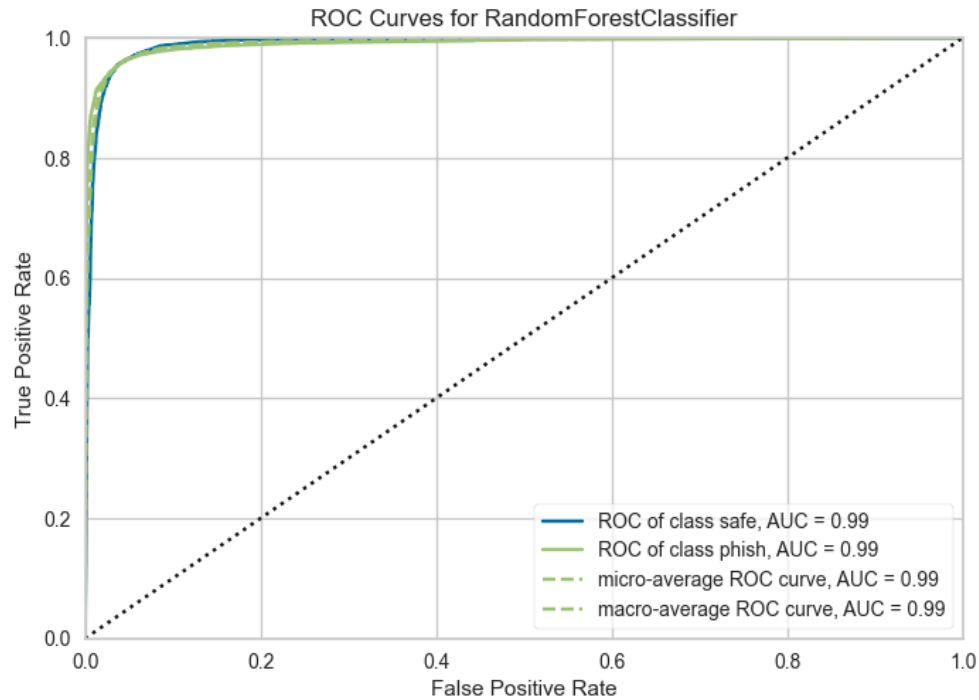
**Figure 6.2.3:** Feature importance of selected model.

Further analysis of the features selected from PSO using Random Forest shows that the feature importance analysis reveals 'pcextlinks' (15.61%) as the most significant feature, indicating that the number of external links plays a crucial role in the model's predictions. 'Domainlength' (13.66%) closely follows, suggesting that the length of the domain is also a strong predictor in the classification task. The third most important feature is '-' (12.07%), highlighting the influence of hyphens in URLs. Other significant features include '/', 'www',

'pemptylinks', 'https', 'digit', 'pcreql', 'redirection', and 'zerolink', with importance ranging from 8.79% to 3.53%. This indicates a broad range of factors related to URL structure, security, and content that influence the model.

Features with lower importance, such as 'domainend', '.', 'extfavicon', '=', 'sfh', 'short\_url', '\_', and 'ip', show that while they contribute less to the overall outcome, they still add some value. This breakdown helps in understanding which features drive the model's performance, guiding further optimization and potential simplification by focusing on the most impactful ones.

Additionally, the ROC AUC score of 0.99 indicates the model's excellent ability to distinguish between phishing and safe URLs. This high score demonstrates the model's robustness in accurately identifying potential threats, reinforcing the importance of feature selection and analysis in achieving optimal performance. Moreover, the false positive rate stands at just 0.02%, indicating a minimal chance of incorrectly classifying safe URLs as phishing.



**Figure 6.2.4:** ROC curves of selected model.

**Table 6.2.5:** Comparisons of Results With Other Work.

Paper	Technique	Accuracy
[4]	Fuzzy rough set feature selection	~ 95.00 %
[5]	Fuzzy logic using mostly DOM and URL features	91.46 %
[6]	Word focussed feature set, as well as search engine results	94.91 %
This Paper	Random forest with PSO	<b>96.13 %</b>

### ***6.2.2 Web Application Usability Test Discussion***

The usability testing results indicate that the majority of participants were able to use the core features of the application, with new users taking approximately one minute to become familiar with the interface. However, many users expressed confusion regarding the extracted features detail and, importantly, the participants requested a Thai version of the application. The findings about flaws in the design of the application provide valuable insights for refining the application's usability and enhancing the user experience.

**Future Improvements.** With the usability test done, various opinions that are formulated by real users can significantly help improve the overall design and operation of the application. As pointed out by the users, points of improvement can be summarized as such:

***Extracted Feature Presentation.*** Considering users found the technical features hard to understand, de-emphasizing more technical parts of the results that is not the main focus for most users can significantly help reduce the load on the users.

***Trust Score Visibility.*** Explore ways to make it more prominent on the page for better visibility.

***Terminology Clarification.*** Testers suggested replacing "URL" with "Link" during manual mode to improve clarity. Implementing this change could enhance user understanding.

***Thai Version Requirement.*** With a strong recommendation from testers, prioritizing the development of a Thai version to better serve that user base may be vital to the accessibility of the application in the long run.

***Extract Feature Description Confusion.*** Users were confused by the exact feature description, indicating a need for clearer language or a redesign to better communicate its functionality.

The findings about flaws in the design of the application provide valuable insights for refining the application's usability and enhancing the user experience.

## **7. PROJECT GLOBAL IMPACT**

With a painstakingly engineered machine learning model for detecting phishing URLs and an easily accessible web application,, it is possible to help improve people's quality of life on the Internet long term. The application can help people recognize the importance of preventive measurements against phishing threats.

The application's user-friendly design will be accessible to individuals from diverse backgrounds and skill levels. Additionally, being able to incorporate a multi-language version ensures that users from various linguistic backgrounds will be able to effectively utilize the application effectively when there is an increase in global outreach.

With these factors in mind, personal and financial information can be protected from phishing attacks, and this will contribute to the preservation of sensitive personal and financial information that may be lost due to the lack of capable protective measures, thus mitigating potential risks and ensuring the safety and well-being of Internet users around the world.

## **8. CONCLUSIONS**

The significance of this paper lies in its contribution to the protection of internet users from the pervasive threat of phishing attacks. By bridging the gap between users and phishing detection applications, our research addresses a critical need in cybersecurity. The project's evaluation yielded insightful results, highlighting both strengths and weaknesses that inform future development and refinement.

### **8.1 Assessment**

Our design project effectively tackled the identified gap in existing phishing link detection tools by prioritizing user-centric features and interpretability. Through comprehensive users, we ensured that the tool addresses the specific needs and challenges faced by internet users. The strengths of our design include the integration of machine learning algorithms, a user-friendly interface design, and the provision of comprehensive educational content.

Despite its strengths, our design faces challenges such as limitations in model generalization and the difficulty in quantifying the impact of educational content. Additionally, the emergence of new potential threats underscores the need for continuous monitoring and updates.

### **8.2 Next Steps**

Moving forward, further evaluation and refinement of the machine learning model are essential to enhance the tool's effectiveness and adaptability. Continuous monitoring and updates to the tool and its database are necessary to stay ahead of evolving threats. Collaboration with cybersecurity experts and industry stakeholders will ensure alignment with global standards and best practices. Moreover, research and exploration of additional features, functionalities, and potential integrations with existing cybersecurity frameworks will contribute to the overall effectiveness of the tool in combating phishing attacks.

## 9. REFERENCES

- [1] Almseidin, M., Zuraiq, A. A., Alkasassbeh, M., & Alnidami, N. (n.d.). Phishing Detection Based on Machine Learning and Feature Selection Methods. *International Journal of Interactive Mobile Technologies*, 13(12), 171. <https://doi.org/10.3991/ijim.v13i12.11411>
- [2] Büber, E., Demir, Ö., & Şahingöz, Ö. K. (2017). Feature selections for the machine learning based detection of phishing websites. *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*. <https://doi.org/10.1109/idap.2017.8090317>
- [3] Safi, A., & Singh, S. (2023). A systematic literature review on phishing website detection techniques. *Journal of King Saud University - Computer and Information Sciences*, 35(2), 590–611. <https://doi.org/10.1016/j.jksuci.2023.01.004>
- [4] Y. Zhang, J. Hong, & L. Cranor, “Cantina: A Content-Based Approach to Detecting Phishing Web Sites,” *Proceedings of the 16th international conference on World Wide Web - WWW '07*, 2007, pp. 639-648.
- [5] H. Chapla, R. Kotak and M. Joiser, "A Machine Learning Approach for URL Based Web Phishing Using Fuzzy Logic as Classifier," *2019 International Conference on Communication and Electronics Systems (ICCES)*, Coimbatore, India, 2019, pp. 383-388, doi: 10.1109/ICCES45898.2019.9002145.
- [6] S. Marchal, J. François, R. State and T. Engel, "PhishStorm: Detecting Phishing With Streaming Analytics," in *IEEE Transactions on Network and Service Management*, vol. 11, no. 4, pp. 458-471, Dec. 2014, doi: 10.1109/TNSM.2014.2377295.