Journal of Informatics and Web Engineering

Vol. 5 No. 2 (June 2026) eISSN: 2821-370X

AI-Powered Threat Hunting for Email Phishing Attack Detection Using Natural Language Processing (NLP)

Mohd Azriy Akmalhazim Bin Mohd Nazariee[1], S Prabha Kumaresan[2, 3*], Alaa Haddad[4], Mohamed Uvaze Ahamed Ayoobkhan[5]**

[1,2,3,4]Faculty of Computing and Informatics, Multimedia University, Persiaran Multimedia, 63100 Cyberjaya, Malaysia
[3]Centre of Natural Language Processing (NLP), CoE for Artificial Intelligence, Multimedia University, Persiaran Multimedia, 63100 Cyberjaya, Malaysia
[5]School of Digital Technologies, American University of Technology, Beshyogoch St 1, 100060, Tashkent, Uzbekistan
**corresponding author: (prabha.kumaresan@mmu.edu.my; ORCiD: 0000-0002-0969-7428)*
***Corresponding author: (mayoobkhan@aut-edu.uz; ORCiD: 0000-0001-9120-4516)*

Abstract - Phishing attacks remain as a significant cybersecurity threat, aiming to steal sensitive information by exploiting human vulnerability. Traditional phishing email detection often struggles to keep up with the latest attack strategies developed by the attackers which results in high false positive rates and the limited contextual understanding on the email contents. Therefore, to address these challenges, this research proposes a solution via an AI-powered threat-hunting model integrating Natural Language Processing (NLP) techniques for phishing email detection in English through the development of PhishGuard AI application. The application is developed as a web-based software solution designed to be accessible to both users with and without technical expertise. The model leverages Word2Vec with TF-IDF weighting for feature extraction and uses an XGBoost classifier. A comprehensive testing process using various metrics will evaluate the computational efficiency and effectiveness of the model. The model's robustness and generalisability were rigorously tested using two distinct datasets which are CEAS_08.csv for in-distribution training and SpamAssasin.csv for out-of-distribution evaluation. The primary value of this model lies in its proactive threat-hunting capability, which distinguishes it from reactive systems that rely on known threat examples. The findings derived from the study aim to enhance to the domain of phishing email detection and contributing to the development of a more robust cybersecurity solution that can help in safeguarding both the individuals and organisations safety in our country.

Keywords—Phishing, Natural Language Processing, Machine Learning, Artificial Intelligence, Application

Received: 30 June 2025; Accepted: 1 September 2025; Published: 16 June 2026

Journal of Informatics and Web Engineering
https://doi.org/10.33093/jiwe.2026.5.2.10

Published by MMU Press. URL: journals.mmupress.com/jiwe

1. INTRODUCTION

In today's interconnected digital landscape, phishing attacks are one of the most pressing cybersecurity challenges. Delivered through fraudulent emails, these deceptive attacks exploit human vulnerabilities and enable the attackers to steal the targeted victim's sensitive information such as login credentials, financial data, or personal details [1], [2]. Given that email remains a primary communication tool for individuals and organizations, cybercriminals continue to use it as a cost-effective and widespread vector for phishing attacks [3]. The consequences of such attacks are far-reaching and impacting society on so many levels, starting from the individuals' privacy, organizations' financial stability, and even national security. In fact, phishing attacks are the top fraud incidents reported by Malaysian internet users to CyberSecurity Malaysia [4] with 901 and 733 cases reported in the first and second quarter of 2024 respectively.

There are three main objectives that are wanted to be achieved through the execution of this research, which are as highlighted below.

1. To develop an AI-powered threat-hunting application which integrates Natural Language Processing (NLP) techniques to effectively identify and classify phishing .
2. To conduct comprehensive testing and evaluation of the model developed using various metrics.
3. To evaluate and compare the computational efficiency and effectiveness of the model developed for email phishing attacks.

2. LITERATURE REVIEW

Conventional email phishing detection systems, which are primarily rule-based or blacklist-driven, often fall short in identifying modern, sophisticated threats. These models are not context-aware and tend to struggle with the subtle linguistic patterns and deceptive language commonly found in phishing messages. The current phishing detection systems are characterised by low context-awareness and therefore perform poorly when they are confronted by subtle linguistic indicators, and misleading rhetoric commonly contained in phish emails. One of the constant results of this weakness is an excessive false-positivity and the impossibility to quickly respond to the changes in the methods of attack [5], [6]. Since adversarial threat actors have been improving their craft over time by writing messages that are more like legitimate messages, the effectiveness of historical solutions that rely on pattern matching with a fixed pattern or on lists of known malicious URLs become much less effective [7], [8]. Furthermore, most simplistic systems have a low level of generalisation to previously unseen phishing variants, most likely as they do not have a well-developed capability to linguistically understand.

The new developments in Artificial Intelligence, especially in NLP, opened more promising pathways of phishing detection. By adopting the principles of NLP, machines gain the ability to analyse textual information in a way users do, comprehending to ascertain with a level of proximity the extraction of sentiment, syntactic, and semantic patterns. These abilities enable AI-powered applications to recognise suggestive content or features of an email like urgencies, lies, and manipulation of an email [9], [10], [11]. Approaches like part-of-speech tagging, named entity recognition, semantic embedding (such as Word2Vec and BERT) or tokenization have been effectively used in many studies to enhance the accuracy of detection [12], [13].

A recent empirical study has shown how linguistic features which are often ignored by rule-based systems can serve to greatly enhance the robustness of anti-phishing frameworks. In this research work, an AI-based threat-hunting solution is created, named PhishGuard AI, which combines natural-language processing approaches and machine learning models to differentiate between phishing emails and real correspondences written in the English language. To do so, the Word2Vec feature extraction technique is utilised based on vectors, while XGBoost represents the classifier engine. The experience of past research involving the analysis of the efficiency of Word2Vec used together with XGBoost [14], [15], [16] and a high proportion of legitimate emails compared to the number of phishing emails [17], [18] have guided the design of this architecture. In addition, the interpretability and computational efficiency of Logistic Regression are beneficial to its real-time application and improve transparency, which can suit the modern needs of cybersecurity.

The main value of the proposed model is its proactive nature. Unlike reactive protection services operating on known threat examples, PhishGuard AI is designed to proactively scan incoming email messages in search of an emerging

phishing pattern with the use of textual context, semantic, and other indicators to flag suspected abuse. This proactive approach is aligned with the existing cybersecurity paradigms since they emphasize automated threat-hunting capabilities [19], [20], [21]. Comprehensive tests will be performed to assess the model in terms of standard metrics and the model will be compared to existing methods in terms of efficiency and effectiveness.

3. RESEARCH METHODOLOGY

PhishGuard AI is a system that aims at identifying phishing emails in English using an AI-driven threat-hunting architecture. It incorporates the use of NLP and machine learning, which provides web-based software application that can be easily applied to both individuals with technical know-how as well as individuals who have not the technical know-how. The goal is to provide responsive, interpretable and correct detection experience. The system design diagrams were graphically and practically depicted by employing a total of five system design-based diagrams, which were rich picture, use case, activity, sequence, and class diagrams.

3.1 Rich Picture Diagram

The rich picture diagram offers a holistic and informal visualization of the system environment, illustrating the interplay between core actors: users, the admin, and attackers. It reflects how users interact with the system by submitting emails for analysis and providing feedback, while the admin manages system improvements, and the attacker influences system inputs indirectly through phishing attempts. Figure 1 presents the rich picture diagram developed for this paper.

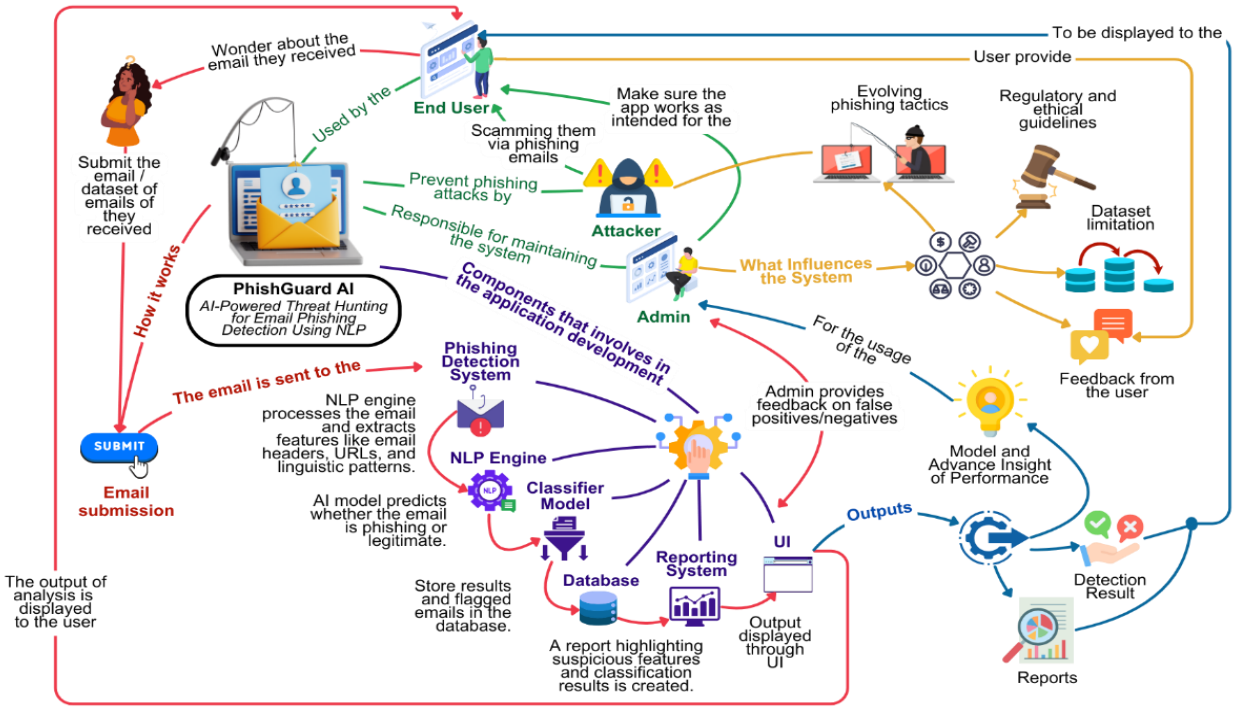

Figure 1. The Rich Picture Diagram for Phishguard AI Application

3.2 Use Case Diagram

The use case diagram captures functional interactions with two main actors: users and admins. Users can sign up, log in, submit emails, view phishing analysis, download reports, and access analysis history. Admins, in contrast, can monitor logs, manage user accounts, provide feedback, and maintain the system. These interactions were clearly modelled to guide development and align stakeholder expectations. Figure 2 presents the use case diagram developed for this project.

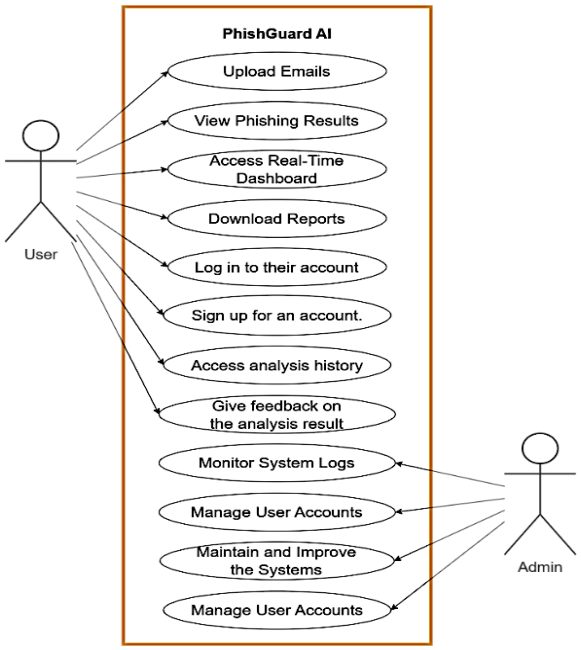

Figure 2. The Use Case Diagram for Phishguard AI Application

3.3 Activity Diagram

The activity diagram outlines the core process flow within the system. It begins with the user submitting an email via the interface. The system then extracts relevant features using the NLP engine, such as URLs, headers, and language cues—and classifies the email through a machine learning model. The result is stored in the database and shared with both the user and admin for feedback. This feedback loop supports continuous system learning and refinement. Figure 3 illustrates the activity diagram developed for this project.

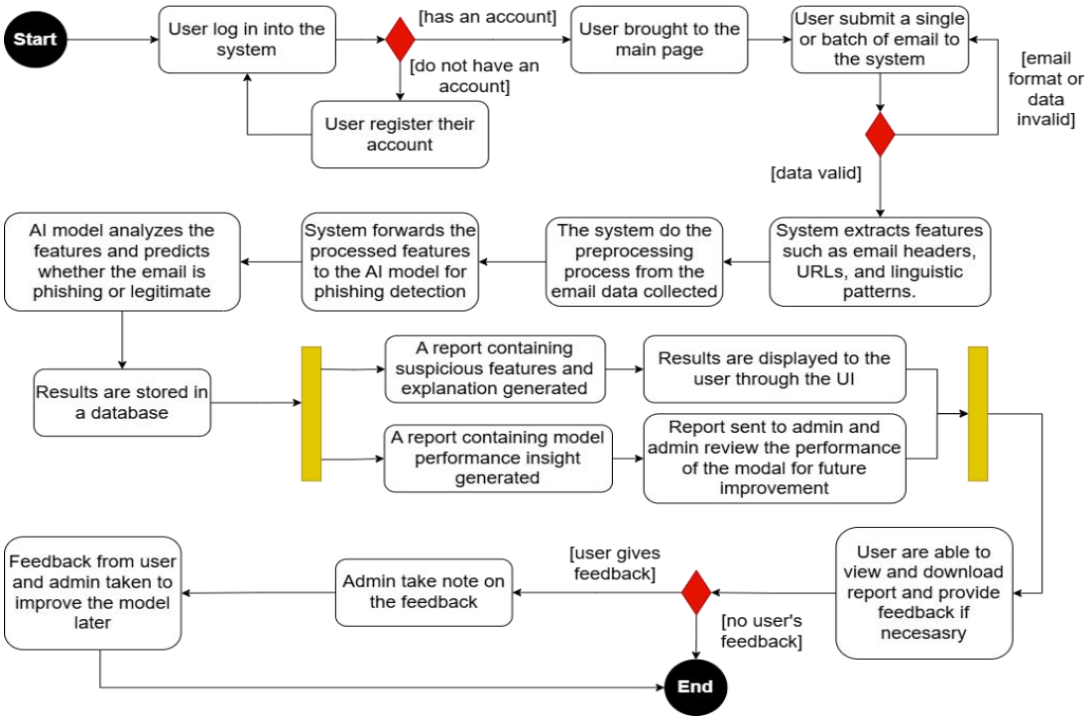

Figure 3. The Activity Diagram for Phishguard AI Application

3.4 Sequence Diagram

The sequence diagram provides a chronological step-by-step depiction of how the system handles an email submission. After user input, the system routes the email through the NLP engine for preprocessing and feature extraction, followed by classification using Logistic Regression. Outputs are stored in a database, and two parallel actions occur: the UI presents results to the user, and the admin reviews flagged outputs, offering feedback that helps retrain and improve model accuracy over time. The sequence diagram developed is presented in Figure 4.

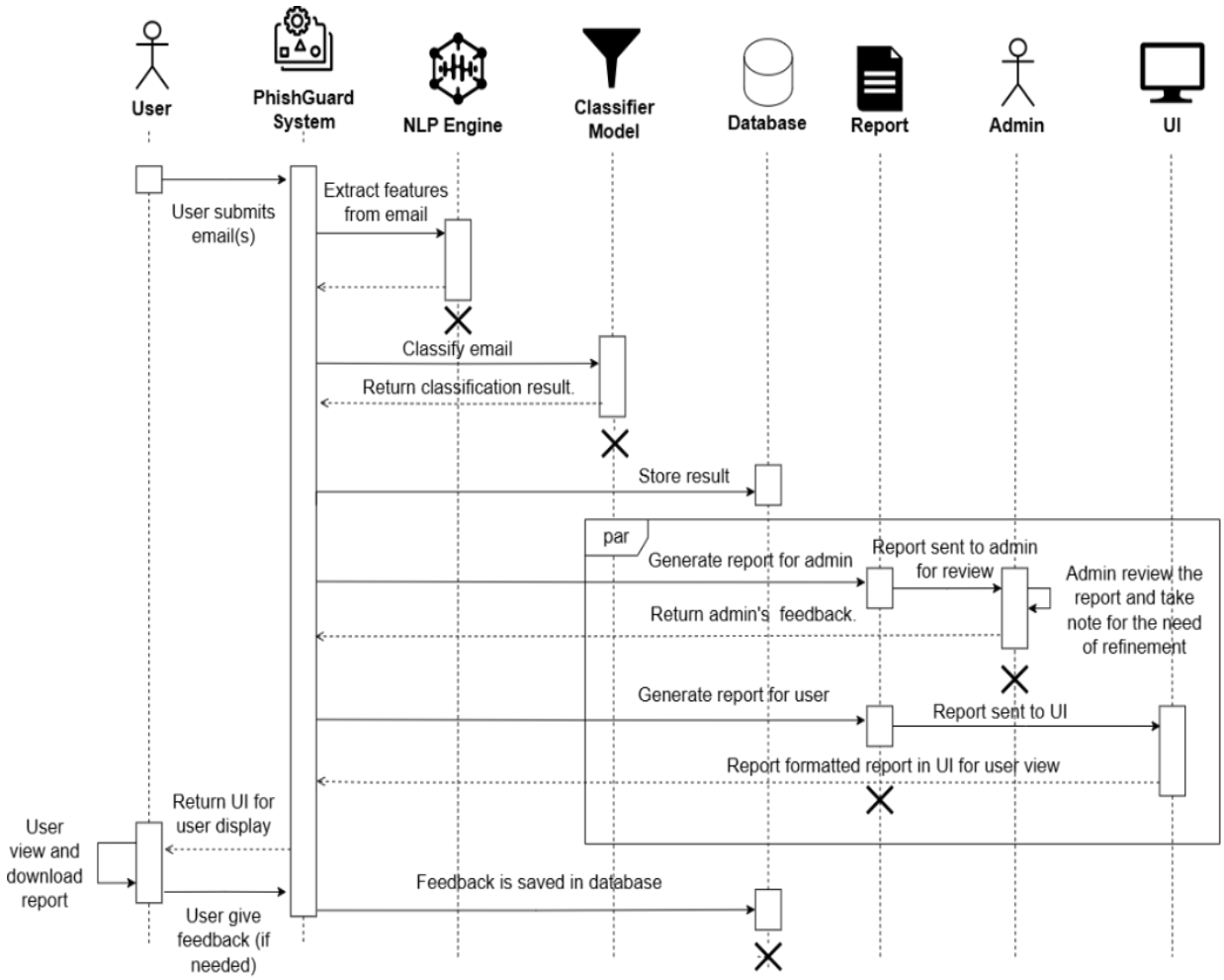

Figure 4. The Sequence Diagram for Phishguard AI Application

3.5 Class Diagram

The class diagram models the software components in object-oriented fashion, as shown in Figure 5. Major classes include:

- User, which handles submission and result viewing.
- PhishGuardSystem, which orchestrates email analysis.
- NLP Engine, for feature extraction.
- Classifier, for prediction.
- Database, for data storage and retrieval.
- Report, which formats output for display.
- Admin, who performs review and feedback.
- UI, the interface through which users interact with the system

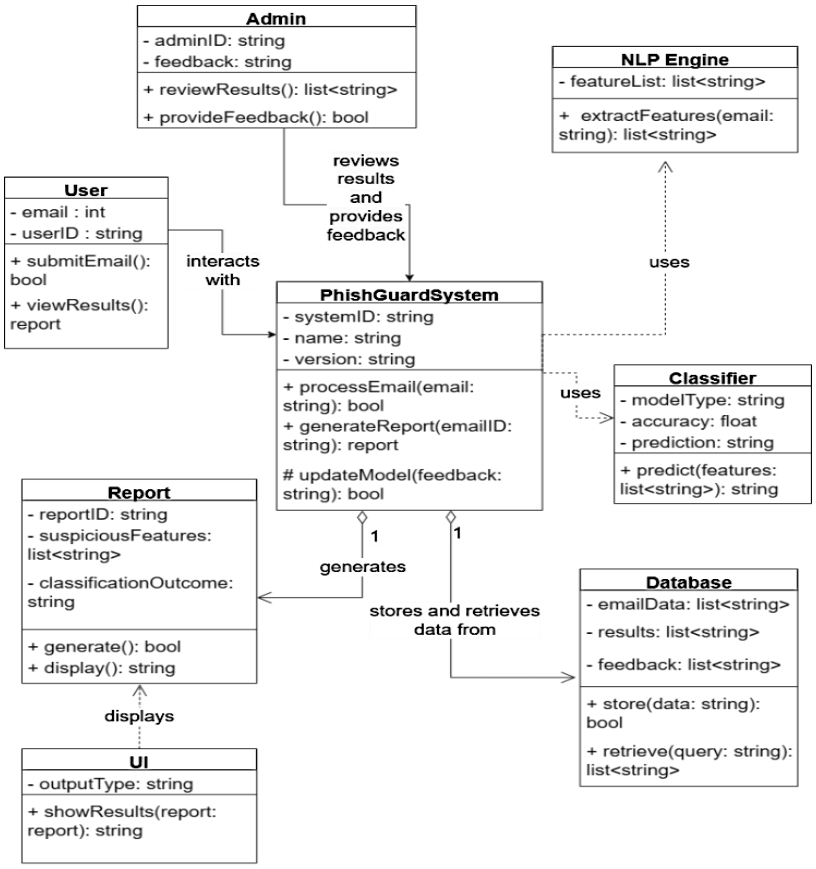

Figure 5. The Class Diagram for Phishguard AI Application

3.6 Interface Design

The User Interface (UI) design prioritizes clarity and responsiveness. PhishGuard AI, an anti-phishing defence system, is deployed as a React-based application with Vite framework support, and TailwindCSS acts as a constant visual style, even with different screen resolutions. The main user interaction mechanisms include core interface elements, such as dashboard, the analysis form, and the history page where the reports could be downloaded. Since the entire functionality of the system is available through this front-end interface, deployment not only offers smooth user experience, but also minimized phishing attack vulnerability. The architecture of the system is modular and provides vertical scalability, which enables it to adapt to new phishing methodologies. Comprehensive schematics that are created throughout the design process make the development process visible, expressible and in resonance with the identified goals of these systems. The interface design of the application for the sign-up page can be viewed in Figure 6. Similarly, the interface design for the log in page can be viewed in Figure 7.

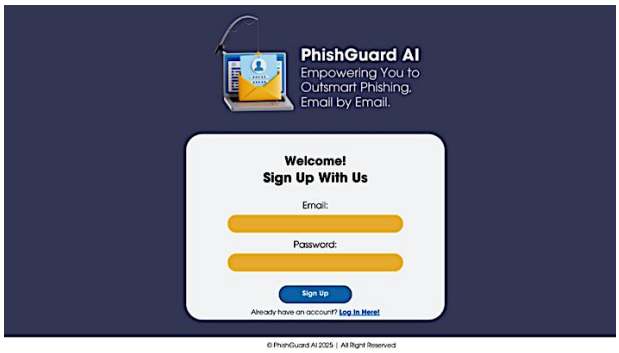

Figure 6. UI for The Sign-Up Page

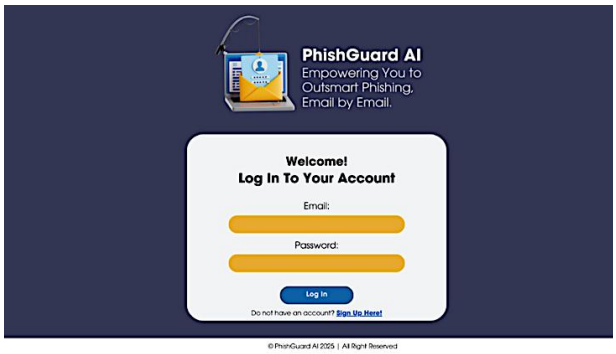

Figure 7. UI For the Log In Page

The interface design for the main page can be viewed in Figure 8.

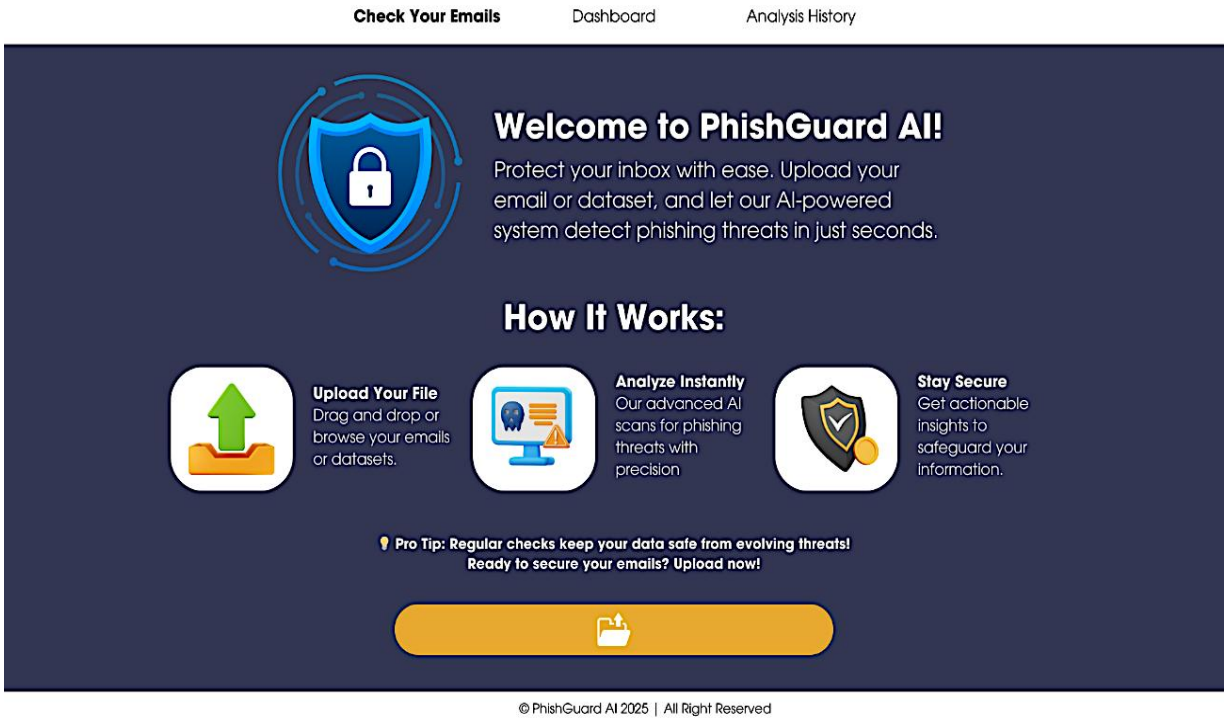

Figure 8. UI For the Main Page

Following that, the interface design for the dashboard page can be viewed in Figure 9.

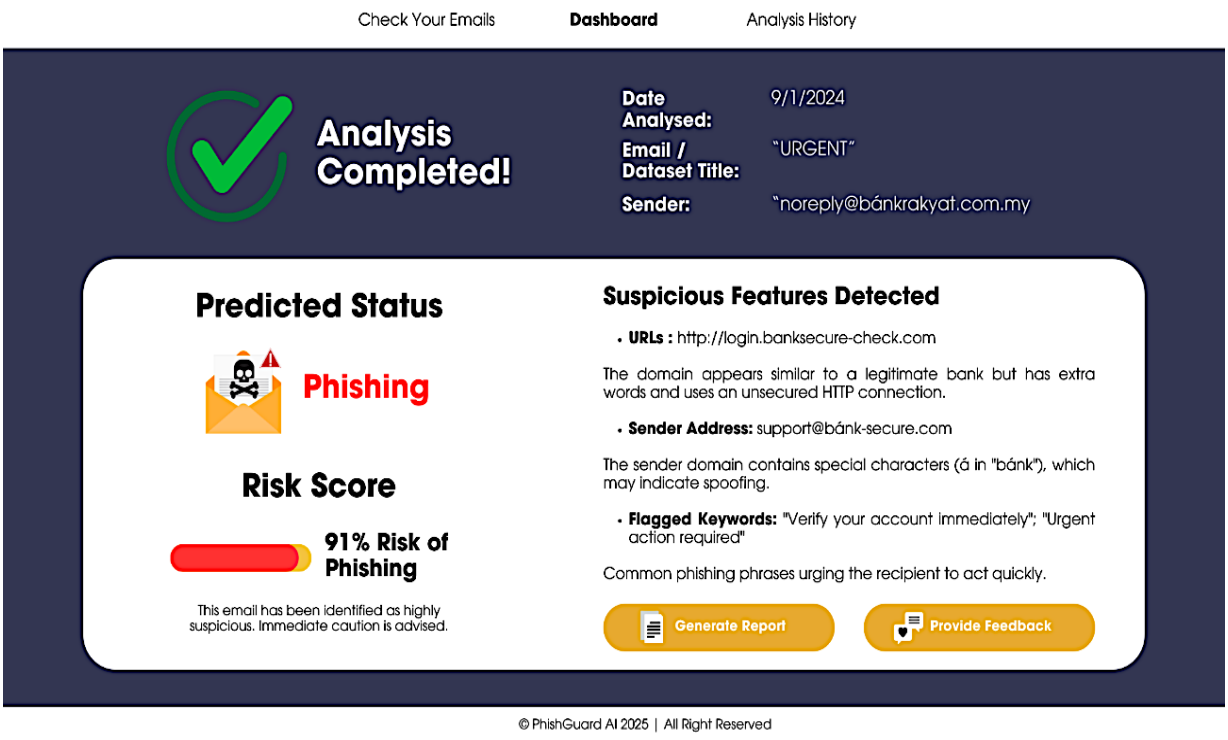

Figure 9. UI for the Dashboard Page

Lastly, the interface design for the analysis history page can be viewed in Figure 10.

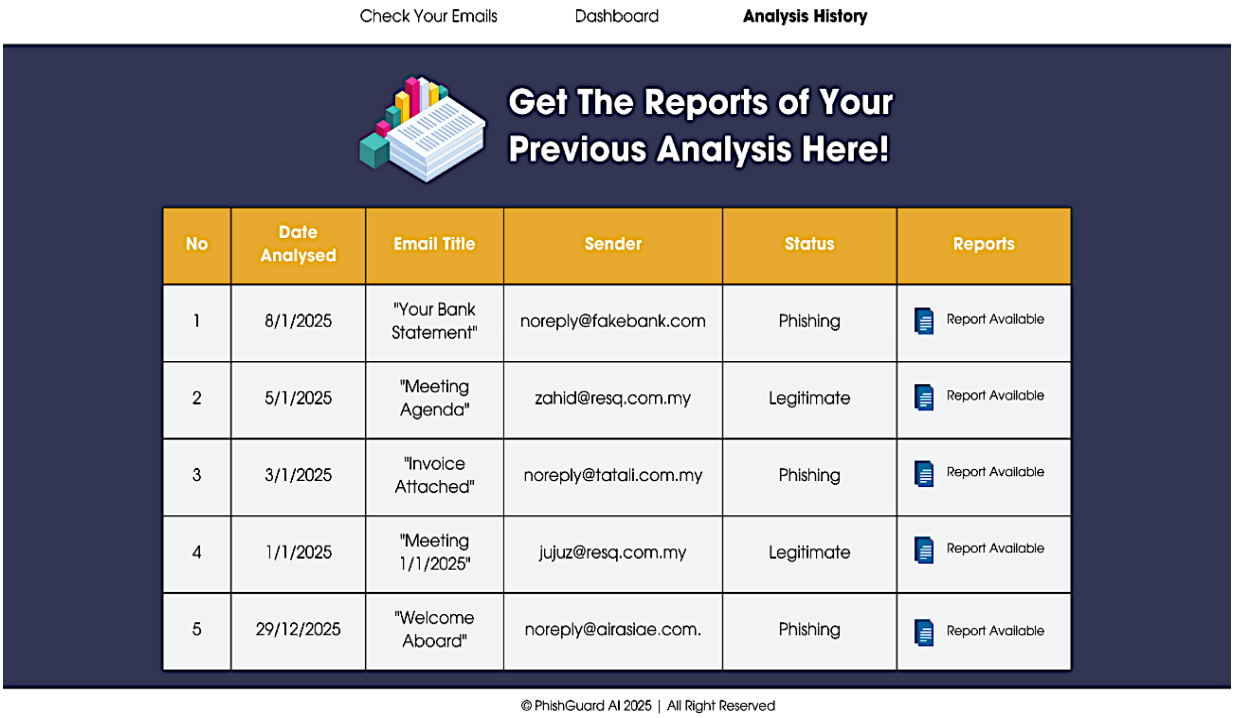

Figure 10. UI for the Analysis History Page

3.7 Analysis on Datasets Used for Model Training

There are two dataset that is utilised in developing the application, namely CEAS_08.csv and SpamAssasin.csv that is collected and curated by Naser [8]. Exploratory Data Analysis (EDA) processes had been done on both datasets chosen. Through this technique, the characteristics and patterns within the dataset were identified, and any anomalies or issues could be detected and addressed before the feature engineering and model training phases commenced. This implementation ensures that subsequent training processes will proceed smoothly. Several analyses were conducted and the details of each processes done will be discussed in the underlying subsections.

3.7.1 Dataset for In-Distribution (ID) Test Data (CEAS_08.csv)

The CEAS_08.csv was selected to be the main dataset for training phase of the model developed due to the significant number of data present in the dataset which facilitate in making the prediction more accurate and robust. The details of the analysis done on this dataset are described in the following subsections.

Shape and Overview of the Datasets

Initial inspection done on the CEAS.08.csv dataset was conducted to understand its basic structure and the data qualities issues that may present in the dataset. Firstly, the dimension of the dataset is assessed and it revealed that this dataset comprises 39,154 email entries and seven columns. Of these, five of them are object types (Sender, Receiver, Date, Subject, Body, Label, URLs) and two of them are int64 (Label and URLs).

The inspection also highlights the presence of missing entries in the dataset. Upon inspection, there are 462 missing entries in the receiver column while the subject column had 28 missing entries. All columns are complete without any missing values. Finally, the few rows from the dataset is observed and it confirmed that the presences of diverse email content and meta data information as shown in Table 1.

Table 1. A Few First Data from CEAS_08.csv

Sender	Receiver	Date	Subject	Body	Label	URLs
Young Esposito <Young@iworld.de>	user4@gvc.ceas-challenge.cc	Tue, 05 Aug 2008 16:31:02 -0700	Never agree to be a loser	Buck up, your troubles caused by small dimensi...	1	1
Mok <ipline's1983@icable.ph>	user2.2@gvc.ceas-challenge.cc	Tue, 05 Aug 2008 18:31:03 -0500	Befriend Jenna Jameson	\nUpgrade your sex and pleasures with these te...	1	1
Daily Top 10 <Karmandeep-opengevl@universalnet...>	user2.9@gvc.ceas-challenge.cc	Tue, 05 Aug 2008 20:28:00 -1200	CNN.com Daily Top 10	>+=+...	1	1
Michael Parker <ivqrnai@poboox.com>	SpamAssassin Dev <xrh@spamassassin.apache.org>	Tue, 05 Aug 2008 17:31:20 -0600	Re: svn commit: r619753 - in /spamassassin/tru...	Would anyone object to removing .so from this ...	0	1
Gretchen Suggs <externalsep1@loanofficertool.com>	user2.2@gvc.ceas-challenge.cc	Tue, 05 Aug 2008 19:31:21 -0400	SpecialPricesPharm Moreinfo	\nWelcomeFastShippingCustomer Support\nhttp://7...	1	1

Distribution of Phishing vs. Non-Phishing

An analysis was performed on the label column to gain the understand on the distribution of phishing emails within the dataset compared to the non-phishing emails it has. This step is crucial for assessing the class balance which is particularly important for choosing the suitable data preparation strategies during the model training process. The label column which represents the email types is divided into two distinct categories which are 0 for legitimate emails and 1 for phishing emails.
The result of the analysis revealed that the dataset is slight imbalance in the dataset as per following:

- Phishing emails (Label 1): 21,842 entries, accounting for 55.79% of the dataset.
- Legitimate emails (Label 0): 17,312 entries, representing 44.21% of the dataset.

The result is visualised in the following Figure 11.

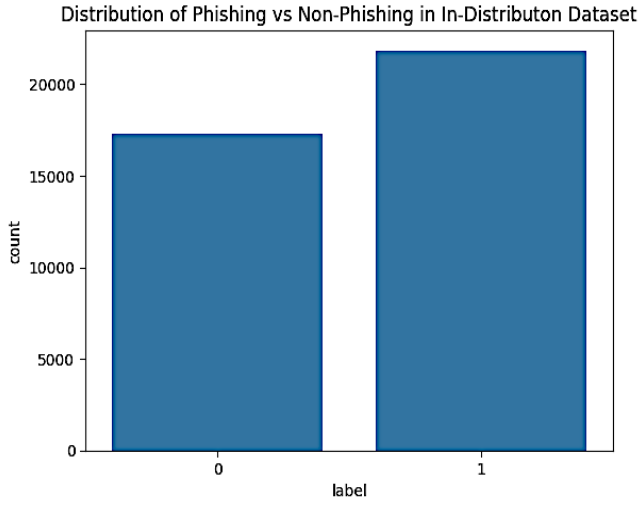

Figure 11. Distribution Of Phishing Vs Non-Phishing In CEAS_08.csv

Analysis on the Top 20 Common Words

To understand linguistic differences between email types, word frequencies were analysed for both phishing and legitimate emails after preprocessing (lowercasing, punctuation/HTML/URL/number removal, tokenization, stop-word removal, and lemmatization).

Firstly, the analysis is done to determine the top 20 most common words in the phishing emails. From the observation, it can be seen that common words in phishing emails include terms like 'top', 'news', 'cnncom', and 'cnn'. For this list of vocabulary, it can be seen that significant portion of phishing attempts in this dataset might be related to news, media, or popular events, where it potentially leveraging current affairs or well-known brands to entice the recipients of the emails.

The presence of 'alert' keyword within the list also reinforces the typical fearmongering strategy used by these attackers. They planned their attack with inducing the sense of urgency or triggering the recipience with warning tone that are often used in phishing attack. The list of top 10 most common words in phishing email can be seen via Figure 12.

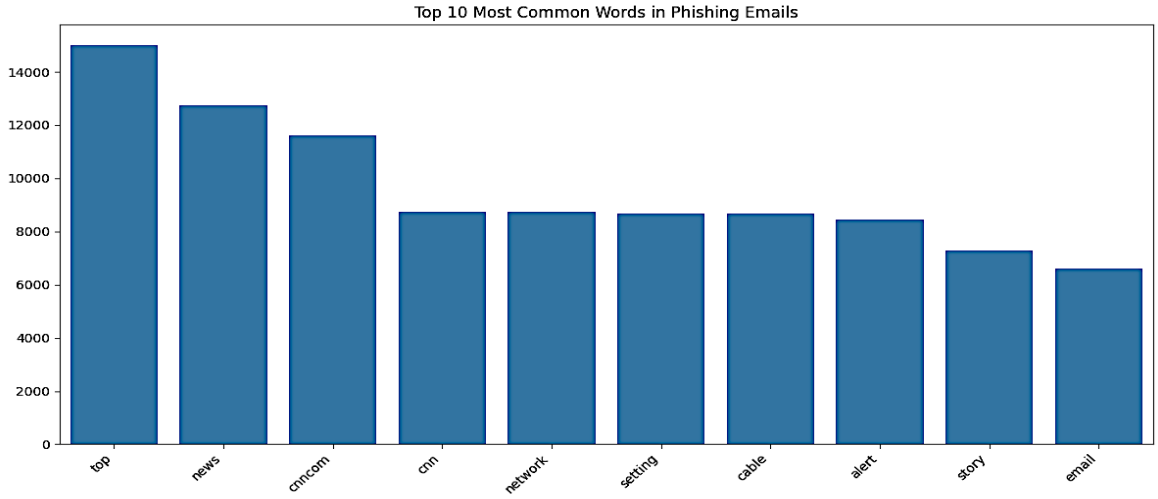

Figure 12. The Top 10 Most Common Words in Phishing Email In CEAS_08.csv in Descending Order

Other than that, the analysis is also done to determine the top 20 most common words in the legitimate emails. The analysis revealed that the legitimate emails from the dataset frequently is it contain terms such as 'submissionid', 'submission', 'note', and 'added'. This vocabulary strongly suggests that these emails originate from automated systems, academic contexts (e.g., 'university'), or technical communications (e.g., 'system', 'file', 'virus' in a non-malicious context, like virus alerts or reports). This is reflecting a typical operational or administrative correspondence.
The presence of these distinct terms helps can help the model to establish a "normal" linguistic profile for benign emails within the dataset. This understanding is truly crucial for a phishing detection model, as it allows the model to differentiate authentic communications from malicious ones, by recognizing deviations from this expected legitimate vocabulary. The list of top 10 most common words in legitimate email is visualised through Figure 13.

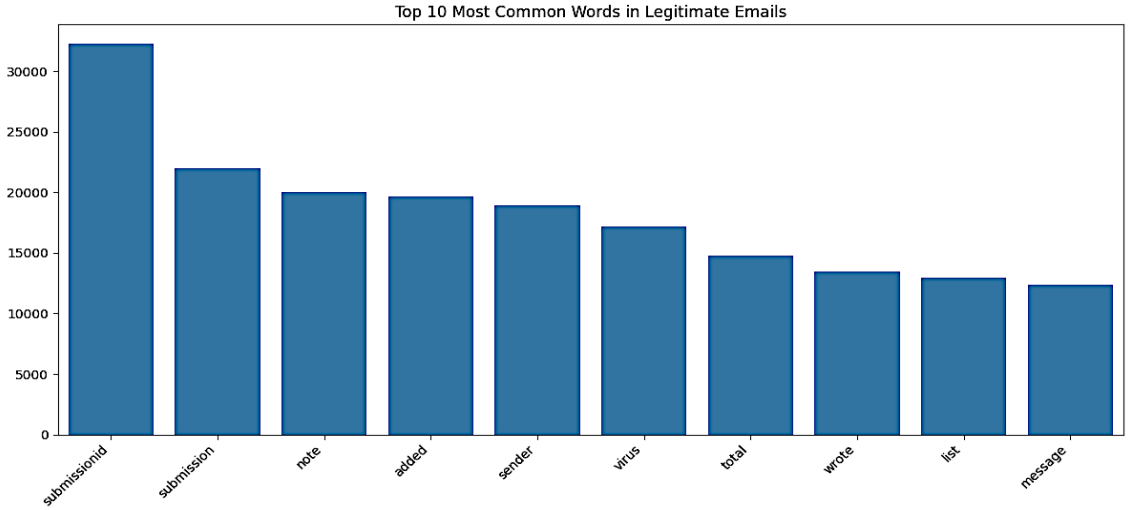

Figure 13. The Top 10 Most Common Words in Legitimate Email in CEAS_08.csv in Descending Order

3.7.2 Dataset for Out-of-Distribution (OOD) Test Data (SpamAsssasin.csv)

In addition, SpamAssasin.csv was selected as the OOD dataset that serves as validation data and was implemented in the testing process to test how well the model developed can generalises to new, unseen data that doesn't fit the patterns it was trained on previously. The details of the analysis done on this dataset are described in the following subsections.

Shape and Overview of the Datasets

The inspection that was done on the SpamAsssasin.csv dataset revealed that the dimension of the of this dataset comprises of 5809 email entries and the same seven columns where five of them are object types (Sender, Receiver, Date, Subject, Body, Label, URLs) and two of them are int64 (Label and URLs).

The inspection also highlights the presence of missing entries in the dataset. Upon inspection, there 210 missing entries in the receiver column while the subject column had 16 missing entries. In addition, this dataset has 1 missing entry in body label. All other columns are complete without any missing values The few first rows from the dataset are observed and it confirmed that the presences of diverse email content and meta data information as shown in Table 2.

Table 2. A Few First Data from SpamAssasins.csv

Sender	Receiver	Date	Subject	Body	Label	URLs
Robert Elz <kre@munnari.OZ.AU>	Chris Garrigues <cwg-dated-1030377287.06fa6d@D...	Thu, 22 Aug 2002 18:26:25 +0700	Re: New Sequences Window	Date: Wed, 21 Aug 2002 10:54:46 -0500 ...	0	1
Steve Burt <Steve_Burt@cursor-system.com>	'"zzzzteana@yahoogroups.com"' <zzzzteana@yahoo...	Thu, 22 Aug 2002 12:46:18 +0100	[zzzzteana] RE: Alexander	Martin A posted:\nTassos Papadopoulos, the Gre...	0	1
"Tim Chapman" <timc@2ubh.com>	zzzzteana <zzzzteana@yahoogroups.com>	Thu, 22 Aug 2002 13:52:38 +0100	[zzzzteana] Moscow bomber	Man Threatens Explosion In Moscow \n\nThursday...	0	1
Monty Solomon <monty@roscom.com>	undisclosed-recipient: ;	Thu, 22 Aug 2002 09:15:25 -0400	[IRR] Klez: The Virus That Won't Die	Klez: The Virus That Won't Die\n \nAlready the...	0	1
Stewart Smith <Stewart.Smith@ee.ed.ac.uk>	zzzzteana@yahoogroups.com	Thu, 22 Aug 2002 14:38:22 +0100	Re: [zzzzteana] Nothing like mama used to make	> in adding cream to spaghetti carbonara, whi...	0	1

Distribution of Phishing vs. Non-Phishing

For the OOD dataset. the result of the analysis revealed that the dataset is imbalance in the dataset as per following:

- Phishing emails (Label 1): 4,091 entries, accounting for 70.43% of the dataset.
- Legitimate emails (Label 0): 1,718 entries, representing 29.57% of the dataset.

The result is visualised in the following Figure 14.

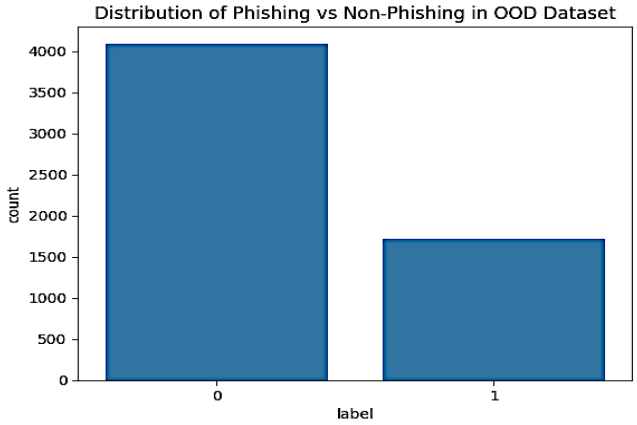

Figure 14. Distribution Of Phishing Vs Non-Phishing in Spamassasin.csv

Analysis on the Top 20 Words

The same analysis done in SpamAssasin.csv determined the top 20 most common words in phishing emails. Compared to the results found in the ID dataset, the top words in phishing emails in this dataset are heavily skewed toward action-oriented and financially motivated language. The words such as "email" (4,162), "free" (2,392), "click" (1,764), and "money" (1,478) are frequent which therefore are aligned with the typical goal of phishing attacks to manipulate recipients into clicking links or providing personal information.

This word usage highlights a different strategy from the ID dataset's news-related approach. Instead, it focuses on direct financial scams and enticing offers. The presence of these keywords reinforces the direct-response nature of these attacks. The list of the top 10 most common words in phishing emails is visualized in Figure 15.

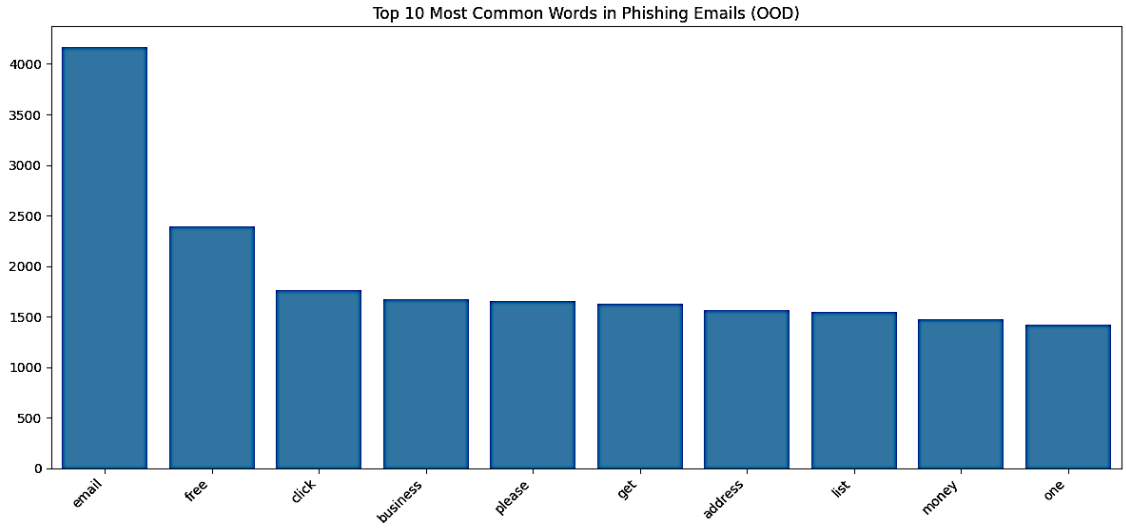

Figure 15. The Top 10 Most Common Words in Phishing Email in Spamassasin.csv in Descending Order

Furthermore, as done for ID dataset, the same analysis was conducted to determine the top 20 most common words in legitimate emails. From the observation, it can be seen that a distinct pattern, with the most common words being more general and less urgent. Terms such as "list" (3,082), "one" (2,797), and "get" (2,500) appear most often. The presence of technical terms like "Linux" (1,626) and specific URLs like "httpwwwcnetcombgif" (1,921) suggests these emails are part of specialized discussions, further distinguishing them from the content of phishing emails. The list of the top 10 most common words in legitimate emails is visualised in Figure 16.

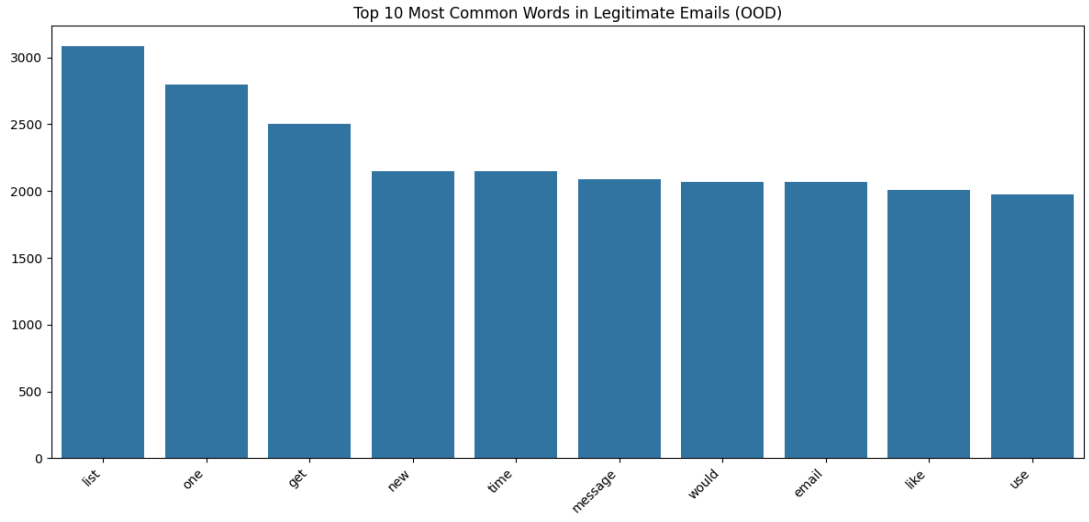

Figure 16. The Top 10 Most Common Words in Legitimate Email in Spamassasin.csv in Descending Order

3.7.3 The Differences and Relevancy of the Dataset Chosen

The selection and use of both the CEAS_08.csv (ID dataset) and SpamAssasin.csv (OOD dataset) were crucial for developing a robust and generalizable phishing detection model. The two-dataset approach allows for a comprehensive evaluation of the model's performance not only on data similar to its training set but also on new, unseen data with different characteristics. This signifies the real-world performance on phishing attacks received by the public that are constantly changing, especially in terms of strategy. The key differences between the two datasets are highlighted in the way the classes are imbalanced and the linguistic and semantic differences present across the data.

Firstly, the class imbalances in the distribution of the phishing and non-phishing emails in these two datasets. Both datasets exhibit a class imbalance, but the degree of imbalance and the dominant class differ, which is important for model validation. The CEAS_08.csv (ID dataset) has a slight imbalance, with phishing emails accounting for 55.79% (21,842 entries) and legitimate emails at 44.21% (17,312 entries). This minor imbalance is a realistic representation of many email environments and necessitates careful handling during model training to avoid bias.

On the other hand, for the SpamAssasin.csv (OOD dataset), a more significant imbalance was observed, with phishing emails comprising 70.43% (4,091 entries) and legitimate emails making up only 29.57% (1,718 entries). The higher proportion of phishing emails in the OOD set provides a strong test of the model's ability to correctly identify the minority class (legitimate emails) under more challenging conditions. The implementation of this dataset for testing purposes therefore is critical in assessing its true generalization capability.

The comparison of the distribution of email status in the two datasets can be seen clearly in Figure 17.

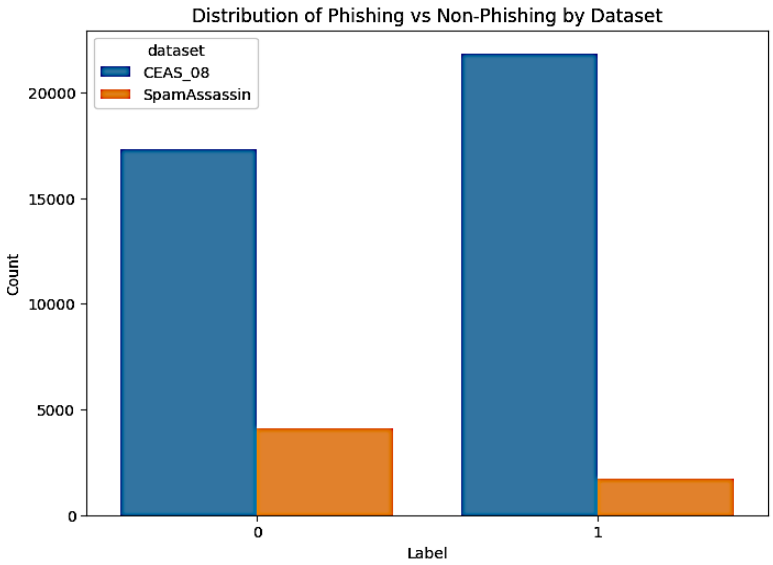

Figure 17. The Differences in the Distribution of the Email Status in CEAS_08.csv and Spamassasin.csv

Another crucial element is the linguistic and semantic differences that is presented by the datasets. ID Linguistic Profile: The top words in the CEAS_08.csv phishing emails are heavily related to news and media, featuring terms like 'news', 'cnncom', and 'alert'. This suggests that the attacks in this dataset are predominantly based on fear-mongering and leveraging well-known brands to entice recipients. The SpamAssasin.csv phishing emails, in contrast, are characterized by action-oriented and financially motivated language, with high-frequency words such as 'email', 'free', 'click', and 'money'. This demonstrates a different phishing strategy focused on direct financial scams and urgent calls to action.

The linguistic differences between these two datasets are critical. By training a classification model only on the CEAS_08.csv news-related scams corpus, there is a risk of overfitting to happened on context-specific lexical features. Testing the model on an OOD dataset, one that includes a different distribution of phishing attacks, gives a demanding indicator of generalization. The high accuracy in the OOD environment indicates that the model has learned the essential features of phishing communication (e.g., urgency, manipulation) instead of memorizing the text of training

samples. The exercise will become a crucial antecedent for developing a flexible and practice-based phishing detection system that can respond to emerging threats.

3.7.4 Pre-processing Steps for the Datasets

The data cleaning process was a crucial step in the model training phases. With proper data cleaning, the quality and consistency of the dataset used can be ensured, therefore contributing to a good detection result. Several steps that had been taken to conduct this process to both dataset which include:

1. Removal of Duplicate Rows: The duplicated rows that may exist in the dataset are removed to prevent redundancy that may lower the detection result.
2. Handling The Missing Values: The potential of having a non-complete value in the dataset is eliminated by dropping the missing values, especially in the crucial columns like body and label. This ensures that the training process of the model is only utilising the complete and relevant data from the dataset.
3. Exclusion of Empty Text in The Dataset: Rows where the body column contained only whitespace or was empty after the stripping process is done are removed as it is no longer relevant to the classifier training.

3.7.5 Features Engineering and Transformation

Feature engineering and transformation is a process that involves converting the raw email data in the dataset to a structured numerical format that is suitable for the machine learning models training. It helps by catching the necessary and relevant features that is crucial for identifying the phishing emails. Several steps are done to execute this and this include:

1. Text Feature Extraction
   During this stage, the raw body and subject column from the email is preprocesses where include tokenisation, sop-word removal and lemmatisation. This standardised the text of the content, which therefore helps in reducing the noise present in the dataset and prepare it to meaningful feature extraction process.
   Then, these processed texts are joined into a cohesive string which are joined_content and joined_subject respectively to prepare them for the vectorisation process that is done using the Word2Vec technique. The Word2Vec embeddings were generated for both email body and subject. Each word was represented by a dense vector, capturing its semantic meaning and crucial to capture the semantic context and relationship that is connected between the words.
   Finally, the TF-IDF weighting is applied to the embedding generated to assigns importance to the words that frequently present in a specific email but rare across the whole dataset used. It assists in helping the model to focus on unique or informative words that are crucial in separating the phishing email from the non-phishing ones.

2. Metadata Feature Engineering
   Even though email metadata is not a part of the main content of the email, it can offer a valuable contextual clue to the model regarding an email's legitimacy. For this reason, there are two main features that were engineered which are the domain extraction from email addresses and temporal features from the date column. The domain extraction involves generating the sender_domain and receiver_domain that are extracted from the respective email addresses. These features offer insight for identifying the legitimacy of the email origins and common attack targets. Beyond this, the temporal features such as hour and day_of_week were also extracted from date column. These features can reveal pattern in phishing attacks for example, emails being sent outside of business hours or on specific days that differs from a legitimate one.

3. Handcrafted Numerical Features
   A few custom features were engineered so that the structural characteristics of the emails can be captured better. This includes:
   i. body_length

This captured the character length of the email body that the emails have. An unusually short or extremely long email body may indicate suspiciousness as attackers could be using brevity to confuse the receiver or providing overwhelming details to distract them.

ii. subject_length
This identified the character length of the email's subject. The same rationale applies as the previously mentioned feature, which is too short or long emails may be an indication to a red flag.

iii. num_urls
URLs also lays a crucial importance in the detection. This feature counts the number of URLs present in the email data as a present of links is a common red flag in suspicious emails.

iv. num_special_chars_body
This counts the number of unusual symbols or punctuation that is present in the email body for example !, @. #. $, % or ^. Attacker may use these punctuations to invoke the sense of urgency or grabbing the receiver's attention.

v. num_special_chars_subject
This counts the number of unusual symbols or punctuation that is present in the email subject with the same reason applied as like in body, for example, attacker may use phrases with this punctuation as in "URGENT!".

4. Categorical Feature Transformation
The features that are categorical such as sender_domain, receiver_domain, hour and day_of_week were transformed using OneHotEncoder into a binary numerical format. The transformation assists in preventing the model from assuming ordinal relationship between the values. One-hot encoding here is important as it enables the features to be understood and used effectively by the model, thus eliminating the risk of artificial ordering that might be inferred by the model.

5. Feature Combination
At the end of the features engineering, we obtained a lot of features gathered which include the metadata features and the handcrafted features. Before the feature extraction process ended, all of the engineered and extracted features discussed earlier were combined horizontally into a single feature matrix that serves as the input for the ML model. This comprehensive approach provides a holistic representation of the email and enables the model to capture a wide range of the indicators needed to determine the email's status.

4. RESULTS AND DISCUSSIONS

4.1 Evaluation Metrics

A set of strict metrics was used to evaluate thoroughly the phishing detection models. These metrics provided an understanding of various aspects of predictive performance especially due to the imbalance between phishing and genuine emails. The most important metrics that are being computed on the test set will include:

1. Accuracy
Accuracy, which is defined as the proportion of correctly classified instances (including both true positives and true negatives) out of the total instances. While accuracy serves as a general indicator, it can sometimes be misleading in datasets with significant class imbalance.

2. Precision
Precision, representing the proportion of correctly identified phishing emails among all emails predicted as phishing. High precision indicates a low false positive rate, meaning fewer legitimate emails are incorrectly flagged. The formula for Precision is given as shown in Equation (1).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (1)$$

3. Recall
Recall (Sensitivity), which is the proportion of actual phishing emails that were correctly identified. High recall signifies a low false negative rate, meaning fewer phishing emails are missed. This metric is

particularly critical in phishing detection, where failing to detect a malicious email can have severe consequences. The formula for Recall is given as shown in Equation (2).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (2)$$

4. F1-score
   The F1-score, which is the harmonic mean of Precision and Recall. This metric provides a single, balanced measure that is particularly useful in cases of uneven class distribution, as it penalizes models that perform poorly in either precision or recall. The formula for F1-score is as given in Equation (3).

$$\text{F1} - \text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

5. Receiver Operating Characteristic Area Under the Curve (ROC AUC) Score
   The ROC AUC Score, which measures the model's ability to distinguish between positive and negative classes across various classification thresholds. An AUC closer to 1.0 indicates a better overall discriminatory power of the model.

4.2 Initial Performance on ID Test Data (CEAS_08.csv Test Set)

Figures 18 and 19 show overall (across models) ROC and precision-recall curve pairings, respectively, thus facilitating side-by-side comparison of the diagnostic accuracies of all models with ID. The illustration of this can be found in the following figures.

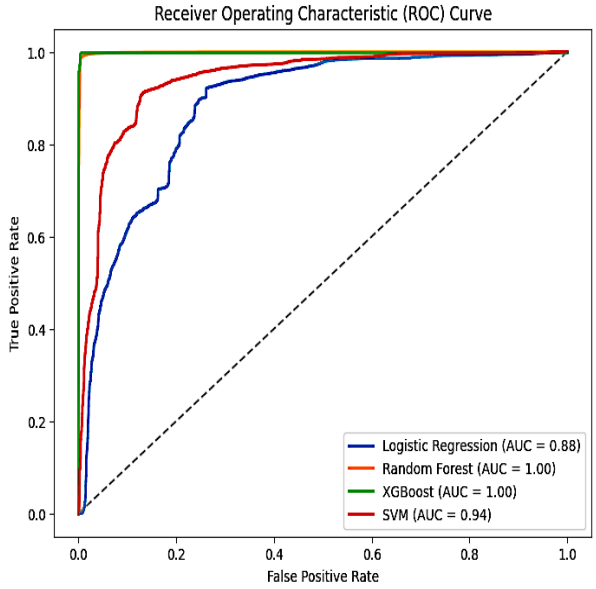

Figure 18. Receiver Operating Characteristic (ROC) Curves for All Models on The CEAS_08 ID Test Set

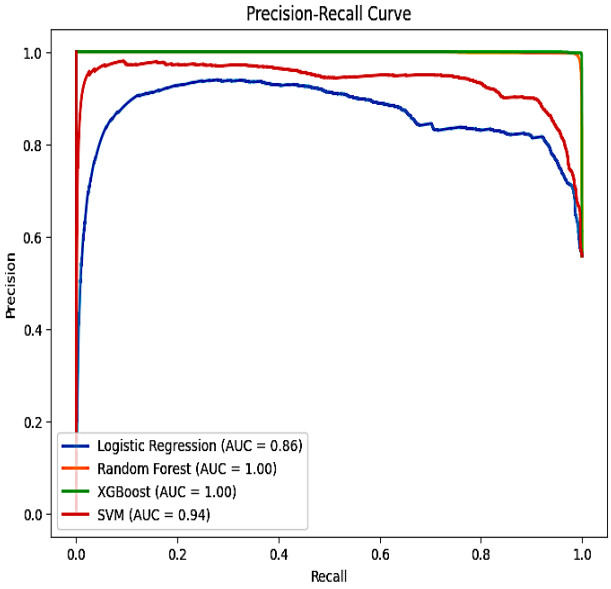

Figure 19. Precision-Recall Curves for All Models on the CEAS_08 ID Test Set

Table 3 on the other hand offers a tabular view of evaluation measures by the model on the ID test set. The Logistic Regression and Support Vector Machine (SVM) perform rather similarly, with Random Forest and XGBoost providing almost perfect accuracy results, meaning their capacity to represent the complexity of the training distribution is higher than average.

Table 3. Model Performance on the ID Test Data

Model	Accuracy	Precision (Phishing)	Recall (Phishing)	F1-score (Phishing)	ROC AUC
Logistic Regression	84.55%	82.43%	91.90%	86.90%	0.8908
SVM	89.25%	89.66%	91.26%	90.45%	0.9393
Random Forest	99.58%	99.79%	99.45%	99.62%	0.9998
XGBoost	99.78%	99.86%	99.75%	99.81%	0.9998

The differences that occur across the training performance are presented in Figure 20.

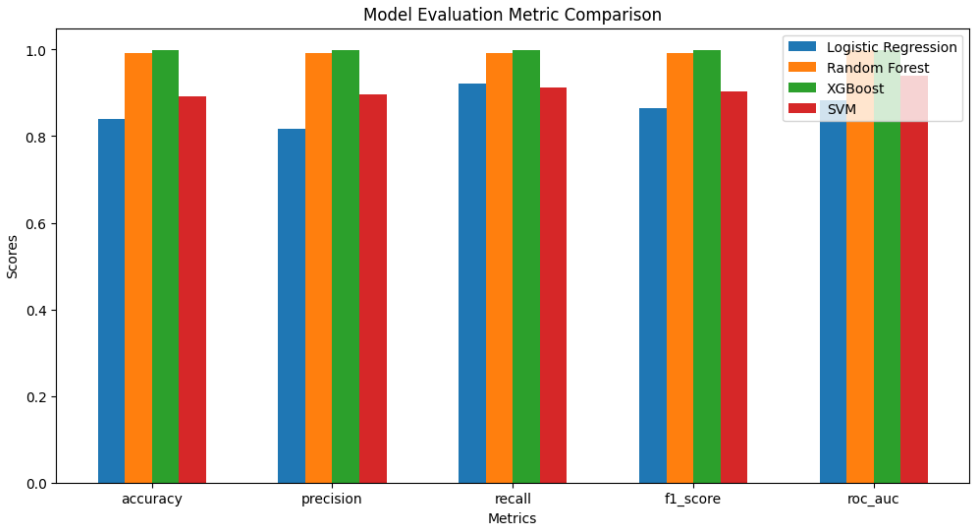

Figure 20. The Modals Evaluations on the ID Test Data

4.2 Performance on OOD Dataset and Hyperparameter Tuning

To present a critical measure of how well models generalise, the evaluation framework was structured to test the models on a completely novel, OOD dataset. This test represented an essential test of generalization on a setting in which the OOD data was not predictive of the training distribution. The method of comprehensive hyperparameter tuning like GridSearchCV could not be implemented due to computational resources.

As a result, a more iterative, manual tuning process was put into use, with the main goal being to increase recall and F1-score on the OOD dataset. This process was firstly done by defining a base model for each of the machine learning models utilized in the experiment. Then, these base models were evaluated for their performance, and their hyperparameters were manually and iteratively adjusted to achieve better performance on the OOD data.
The result was a final set of tuned models, each with a specific configuration of hyperparameters optimized for the task, as follows:

1. Logistic Regression:
    - class_weight: 'balanced'
    - solver: 'saga'
    - max_iter: 3000
    - random_state: 42
2. Random Forest:
    - n_estimators: 20
    - max_depth: 10
    - min_samples_split: 5

- min_samples_leaf: 2
- max_features: 'sqrt'
- class_weight: 'balanced_subsample'
- random_state: 42

3. SVM
   - C: 1
   - kernel: 'rbf'
   - gamma: 'scale'
   - probability: True
   - class_weight: 'balanced
   - random_state: 42
4. XGBoost
   - n_estimators: 300
   - max_depth: 5
   - learning_rate: 0.1
   - subsample: 1.0
   - colsample_bytree: 1.0
   - eval_metric: 'logloss'
   - use_label_encoder: False
   - scale_pos_weight: The class weight was calculated using the Formula 5 to handle class imbalance.

$$\text{scale\_pos\_weight} = \frac{\text{len}(y_{\text{train}}) - \sum y_{\text{train}}}{\sum y_{\text{train}}} \quad (5)$$

   - random_state: 42

The result from the test conducted are as follows in Table 4 and Figure 21.

Table 4. The Model Performance on OOD Test Data

Model	Accuracy	Precision (Phishing)	Recall (Phishing)	F1-score (Phishing)	ROC AUC
Logistic Regression	29.6%	0.30	1.0	0.46	0.406
SVM	29.6%	0.30	1.0	0.46	0.458
Random Forest	67.6%	0.47	0.91	0.62	0.779
XGBoost	77.8%	0.59	0.85	0.69	0.876

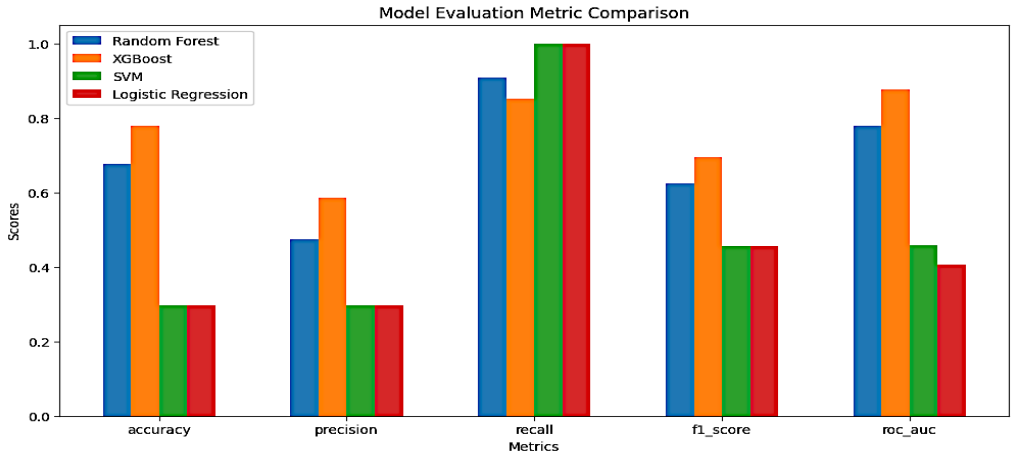

Figure 21. Bar Graph Comparing Accuracy, Precision, Recall, F1-score, and ROC AUC For All Models on the OOD Test Data

This outcome shows that XGBoost is the best choice for the backend integration since it has a better trade-off between recall and F1- score on an out-of-distribution (OOD) dataset. When it comes to detecting phishing emails this is important because the high recall, that is, detection of the phishing email is vital in reducing false negatives, which are undetected phishing emails that may involve serious security implications.

4.3 Computational Efficiency Analysis

Other than only evaluating the ability of models to detect the legitimacy of the emails, the computational efficiency was also conducted. It was assessed in the term of training time, the peak memory consumption during the training phase done, the model file size and the inference latency as it is critical to ensure that the PhishGuard AI application can scale effectively and perform well when it is deployed. The result from the analysis is tabulated in Table 5.

Table 5. The Computational Efficiency Measurements Recorded During the Model Training and Inference Processes

Model	Accuracy	Precision (Phishing)	Recall (Phishing)	F1-score (Phishing)	ROC AUC
Logistic Regression	29.6%	0.30	1.0	0.46	0.406
SVM	29.6%	0.30	1.0	0.46	0.458
Random Forest	67.6%	0.47	0.91	0.62	0.779
XGBoost	77.8%	0.59	0.85	0.69	0.876

From the observation, XGBoost model is the most favourable compared to the rest. This is because it can complete the training with the minimal time, memory have a compact size and can have instantaneous inference. Conversely. SVM Model required a significant amount of resource consumption while taking the longest time, highest memory usage, largest model size and a non-negligible inference time needed per sample. Therefore, because of these reasons, XGBoost is chosen to be integrated into the system.

4.4 Application Implementation

PhishGuard AI system employs a multi-layered software stack that is design with the intention to deliver a scalable and maintainable phishing detection system. The software stack comprises the development pipeline starting from the frontend that serve the user to the OS utilised in building and help in deploying the complete application. The breakdown of each layered as follows:

1. Operating System (OS)

   The OS forms the foundational layer of the software stack, managing hardware resources and providing essential services for application execution. For this application, there are 2 main OS that is leveraged here which are Windows and Linux. Windows is employed in the local development to provide the environment to support the coding processes where the components code are run in the system via IDE chosen which is VSCode. Conversely. Linux is utilised especially during the deployment of the classifier model as Firebase Cloud Functions as Google Cloud's infrastructure utilised Linux-based environment.

2. Runtime Environment

   The runtime environment is crucial to the system as it supply the needed context for the execution of the application code completed. Here, the high-level programming language is translated into operations that can be understood and run by the OS. PhishGuard AI utilises Node.js especially for the backend of the application and Python to facilitates the ML classifier execution.

3. Data Layer

   For the database, PhishGuard AI utilised Firebase Firestore which is a NoSQL cloud database. This platform is serving the purpose to store the result from the detection of the email done and the feedback given by the user on the detection result.

4. Backend Logic

   This layer handles various core application logic that is required for the application to work. This includes processing the incoming request made by the frontend layer, managing the data flow, interacting with the database and communication across the platforms. In the context of PhishGuard AI, two kinds of backends deployed which are through Node.js with Express.js for the application backend itself and Python with FastAPI for the detection process. In the system backend, the Node.js and Express pair serve as the backbone of the primary backend API where is responsible for defining the routes, handling the file uploads, parsing the email content submitted by the user, managing the user authentication and authorisation and interacting with Firebase Firestore for the information needed to be saved and displayed to the user.
   The pair also initiates the calls to the ML prediction service that utilised Python with FastAPI that receives the data and process it accordingly and provides the backend with the result of the detections and metrics relevant to be informed to the user via the frontend.

5. Application Programming Interfaces (APIs)

   There are two main types of APIs that is implemented in the design of the application which are Frontend-to-Backend API and Backend-to-ML API. These are integrated to facilitate the communication between the different components of the system therefore enable the seamless data exchange and functionality to occur.
   For the Front-end-to-Backend API, Express.js's RESTful APIs is utilised, providing the access for the React frontend to communicate with the Node.js backend developed. Similarly in the Backend-to-ML API, the same functionalities are integrated to allow the communication between the Node.js/Express backend and the Python/FastAPI ML model prediction in Cloud Run.

6. Frontend

   The front-facing UI of the application is built using React which is a popular JavaScript library and built using Vite which serve as a built tool and development server. The navigation between the pages is handled by the React Router DOM while the API request from this layer is managed by Axios.
   In term of styling of the PhishGuard AI's UI, Tailwind CSS is leverage as it offers an intuitive responsive prefix which provide a good, adaptable design layout across different device screen.

The full software stack can be summarised and visualised through Figure 22 that illustrates the stack used in the production.

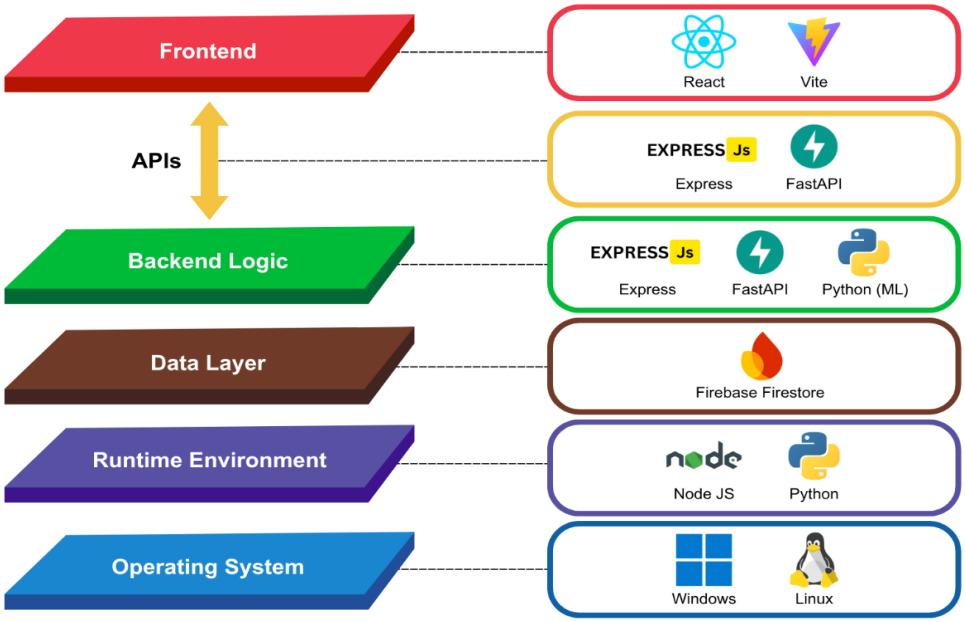

Figure 22. Layered Software Stack Architecture of PhishGuard AI

Comprehensive testing such as unit testing, integration testing, usability testing and acceptance testing had been done to check the performance of the application built. The result from the testing validated that the application is working as intended.

4.5 Comparative Analysis to Other Phishing Email Detection Systems

PhishGuard AI can be described as a significant extension of the traditional anti-phishing platform since it uses a proactive and AI-driven approach. Unlike rule-based or blacklist-driven systems, which are necessarily reactive and based on pre-defined lists of known threats, the system employs the most advanced NLP algorithms, allowing analysis of linguistic and semantic structures within email contents. This would allow PhishGuard AI to identify new, unseen, or zero-day phishing attacks, providing a more comprehensive and flexible defence against threats as they change. In that sense, the system differs significantly with a number of traditional and scholarly anti phishing approaches.

Another aspect that sets PhishGuard AI apart is its hybrid machine-learning architecture. The model utilizes a compounded Word2Vec feature-extraction scheme combined with the TF-IDF weighting mechanism that allows to capture the contextual meaning along with the importance of words. These attributes are subsequently input into an XGBoost classifier which is an algorithm known for its high accuracy and performance and can negotiate fine grain linguistic features that define the methodology of modern phishing crimes with much success often outdoing models that depend on simpler algorithms.

The rigorous evaluation scheme employed in the study lends credence to the practical effectiveness of the model. In turn, this was verified on two separate sets: an ID dataset and an OOD dataset. The former was used to train the network and the latter to estimate generalization. This dual-dataset setup was necessary in proving whether the model could work effectively on unseen data and therefore reduce the danger of overfitting. The evaluation shows the model has the capacity to respond to the dynamic, unpredictable threat of real-world phishing risks through demonstrating robustness across multiple data sources.

5. CONCLUSION

With the increasing rate of the email phishing attacks, the study aimed at creating AI-based threat-hunting model (PhishGuard AI) that could discriminate and classifying phishing emails effectively in the English language by using the NLP methods. The first stage of the research required complex data gathering and defining major requirements of the system to help with further work. Extensive literature review and background analysis were also carried upon to clarify such basics as threat hunting, the features of phishing emails, Artificial Intelligence (AI), NLP, and the technical and methodological issues of the current systems. It is based on this critical analysis that the research agenda was developed and subsequently used in designing the proposed model. After that, a thorough analysis of requirements was performed followed by comparative analytical analyses of two important works that enabled finding functional, non-functional, and user requirements. All these needs were closely incorporated in the main system design insertion. These requirements were carefully incorporated into the overall system design framework.

Through the project there are several key challenges that were encountered. The main problem that is faced with the development of the application is the performance of the model developed against real-world data. Models developed are not able to generalise well with the data outside the training data given – OOD data. This poses a main problem in term of its effectiveness in the implementation. Other than that, the computational resources constraint is inhibiting the effort to further enhance the model.

Therefore, to further enhance the quality of PhishGuard AI and NLP-based email phishing detection system in general, future work could focus on increasing the quality of detection of the classifier model in term of OOD to make sure that the detection made on the email are better, more accurate and robust. In addition, the implementation of AIGE key suggestion, for example the usage of eXplainable AI (XAI) can also be made to convey the understanding on the detection analysis to the application user. In conclusion, PhishGuard AI represents a significant step towards the development of a more robust and adaptive cybersecurity solutions for email phishing detection. It is hope that this effort could open the opportunity for more exploration in the strategies to safeguard individuals and organisations, for a better and safer digitalised world.

ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for the suggestions to improve the paper.

FUNDING STATEMENT

The authors received no funding from any party for the research and publication of this article.

AUTHOR CONTRIBUTIONS

Mohd Azriy Akmalhazim Bin Mohd Nazariee: Conceptualization, Data Curation, Methodology, Validation, Writing – Original Draft Preparation;
S Prabha Kumaresan: Conceptualization, Project Administration, Investigation, Supervision, Validation, Writing – Review & Editing;
Alaa Haddad: Validation, Data Curation;
Mohamed Uvaze Ahamed Ayoobkhan: Validation, Data Curation.

CONFLICT OF INTERESTS

No conflict of interest was disclosed.

ETHICS STATEMENTS

This research did not involve animal experiments, or social media data. Our study follows The Committee of Publication Ethics (COPE). https://publicationethics.org/

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] P. H. Kyaw, J. Gutierrez, and A. Ghobakhlou, "A systematic review of deep learning techniques for phishing email detection." *Electronics*, vol. 13, no. 19, pp. 3823, Sept. 2024, doi: 10.3390/electronics13193823.

[2] V. Malik, V. Rattan, J. Singh, R. Mittal, and U. Tandon, "Performance comparison of data mining classifiers on web log data." *Journal of Computational and Theoretical Nanoscience*, vol. 17, no. 11, pp. 5113–16, Nov. 2020, doi:10.1166/jctn.2020.9349.

[3] M. G. Ames, "Hackers, computers, and cooperation: a critical history of logo and constructionist learning." *Proceedings of the ACM on Human-Computer Interaction*, vol. 2, no. CSCW, pp. 1–19, Nov. 2018, doi:10.1145/3274287.

[4] H. R. Arabnia, Ed., in *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 1999)*, Las Vegas, NV, USA: CSREA Press, 1999.

[5] O. A. Lamina, W. A. Ayuba, O. E. Adebiyi, G. E. Michael, O. O. D. Samuel, and K. O. Samuel, "AI-powered phishing detection and prevention." *Path of Science*, vol. 10, no. 12, pp. 4001–10, Dec. 2024, doi:10.22178/pos.112-7.

[6] M. Cannice, "Q3 2023 Silicon Valley Venture Capitalist Confidence Index™ Quarterly Research Report." *SSRN Electronic Journal*, 2024, doi:10.2139/ssrn.4686324.

[7] M. Alanezi, "Phishing detection methods: A review." *Technium: Romanian Journal of Applied Sciences and Technology*, vol. 3, no. 9, pp. 19–35, Nov. 2021, doi:10.47577/technium.v3i9.4973.

[8] P. Kumar, D. Javeed, A. N. Islam, and X. R. Luo, "DeepSecure: A computational design science approach for interpretable threat hunting in cybersecurity decision making." *Decision Support Systems*, vol. 188, pp. 114351, Jan. 2025, doi:10.1016/j.dss.2024.114351.

[9] R. Rahman, and F. F. Abdulloh, "Performance of various naïve bayes using gridsearch approach in phishing email dataset." *Sinkron*, vol. 8, no. 4, pp. 2336–44, Oct. 2023, doi:10.33395/sinkron.v8i4.12958.

[10] B. M. Olukoya, G. O. Ogunleye, P. O. Olabisi, and A. S. Adegoke, "Heterogeneous ensemble feature selection: An enhancement approach to machine learning for phishing detection." *International Journal of Software Engineering and Computer Systems*, vol. 10, no. 1, pp. 60–74, Oct. 2024, doi:10.15282/ijsecs.10.1.2024.6.0124.

[11] A. Bezerra, I. Pereira, M. A. Rebelo, D. Coelho, D. A. D. Oliveira, J. F. P. Costa, and R. P. Cruz, "A case study on phishing detection with a machine learning net." *International Journal of Data Science and Analytics*, vol. 20, no. 3, pp. 2001-20, Sept. 2025, doi:10.1007/s41060-024-00579-w.

[12] D. Narciandi-Rodriguez, J. Aveleira-Mata, M. T. Garcia-Ordas, J. Alfonso-Cendon, C. Benavides, and H. Alaiz-Moreton, "A cybersecurity review in IoT 5G Networks." *Internet of Things*, vol. 30, pp. 101478, Mar. 2025, doi:10.1016/j.iot.2024.101478.

[13] D. O. Otieno, A. S. Namin, and K. S. Jones, "The Application of the BERT transformer model for phishing email classification." in *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)* [Torino, Italy], pp. 1303–10, 2023, doi:10.1109/COMPSAC57700.2023.00198.

[14] O. Abdelaziz, S. Deb, R. Hodhod, and L. Ray, "A novel phishing email detection algorithm based on multinomial Naive Bayes classifier and natural language processing:" *Proceedings of the 1st International Conference on Computing and Emerging Sciences* [Erbil, Iraq], pp. 69–73, 2020, doi:10.5220/0010412600690073.

[15] A. Chien, and P. Khethavath, "Email feature classification and analysis of phishing email detection using machine learning techniques." *2023 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)* [Nadi, Fiji], pp. 1–8, 2023, doi:10.1109/CSDE59766.2023.10487729.

[16] A. Yasin, and A. Abuhasan, "An intelligent classification model for phishing email detection." *International Journal of Network Security & Its Applications*, vol. 8, no. 4, pp. 55–72, July 2016, doi:10.5121/ijnsa.2016.8405.

[17] A. Abraham, G. Gressel, and K. Achuthan, "Temporal resilience of phishing detection models in machine learning." *SSRN Electronic Journal*, 2019, doi:10.2139/ssrn.3511056.

[18] T. A. Khan, R. Sadiq, Z. Shahid, M. M. Alam, and M. Mohd Su'ud, "Sentiment analysis using support vector machine and random forest." *Journal of Informatics and Web Engineering*, vol. 3, no. 1, pp. 67–75, Feb. 2024, doi:10.33093/jiwe.2024.3.1.5.

[19] A. Khan, K. Khan, W. Khan, S. N. Khan, and R. Haq, "Knowledge-based word tokenization system for Urdu." *Journal of Informatics and Web Engineering*, vol. 3, no. 2, pp. 86–97, June 2024, doi:10.33093/jiwe.2024.3.2.6.

[20] T. S. Tajamul, and K. Aman, "Unveiling the efficacy of AI-based algorithms in phishing attack detection." *Journal of Informatics and Web Engineering*, vol. 3, no. 2, pp. 116–33, June 2024, doi:10.33093/jiwe.2024.3.2.9.

[21] M. A. Daniel, S. C. Chong, L. Y. Chong, and K. K. Wee, "Optimising phishing detection: A comparative analysis of machine learning methods with feature selection." *Journal of Informatics and Web Engineering*, vol. 4, no. 1, pp. 200–12, Feb. 2025, doi:10.33093/jiwe.2025.4.1.15.

BIOGRAPHIES OF AUTHORS

	**Mohd Azriy Akmalhazim Bin Mohd Nazariee** is a final year Bachelor of Computer Science (Hons) (Cybersecurity) student from Multimedia University, Cyberjaya. His research focuses on the integration of NLP-integrated AI-powered Threat-Hunting application for email phishing detection. The research conducted is done by the creation of PhishGuard AI application where NLP and ML models are integrated into an application dedicated for analysis of the status of an email. Analysis of the model used was also conducted, contributing to input on the effectiveness of the ML model used as classifier. He can be contacted via two email addresses: 1211104288@student.mmu.edu.my or azriynazariee@gmail.com.
	**S Prabha Kumaresan** is an Assistant Professor at Multimedia University. She received her PhD in Engineering from the Multimedia University, Cyberjaya, Malaysia. Her research interests include machine learning, artificial intelligence, and data analytics, with applications across education and industry. In addition to her academic contributions, she has been actively involved in collaborative projects and initiatives that promote digital talent development. She is also committed to enhancing student engagement and mentoring, ensuring that learners are well-prepared to excel in future technologies and innovation-driven environments. She can be contacted at email: prabha.kumaresan@mmu.edu.my.
	**Alaa Haddad** is an Assistant Professor at the Faculty of Computing and Informatics, Multimedia University, Cyberjaya. Her research focuses on blockchain technology, cybersecurity, distributed systems, and secure application development. She supervises undergraduate and postgraduate students in secure and decentralized digital solutions. She received her B.Sc. in Software Engineering from Aleppo University, Syria, and her M.Sc. and Ph.D. degrees in Computer Information Engineering from the International Islamic University Malaysia (IIUM). She can be contacted at email: alaa.haddad@mmu.edu.my.
	**Mohamed Uvaze Ahamed**, is an accomplished academic and researcher with over 13 years of teaching and research experience across five countries. Currently serving as Assistant Professor in the School of Digital Technologies at the American University of Technology, Tashkent, Uzbekistan. He specializes in Computer Vision, Machine Learning, Medical Imaging, and AI. A senior IEEE member and Fellow of Advance HE (FHEA), he holds multiple international IPRs and patents, has authored books, and actively mentors students in Scopus-indexed research publications. He can be contacted at email: mayoobkhan@aut-edu.uz.