AI-POWERED PHISHING WEBSITE DETECTION TOOL USING A TRIBRID AI MODULE

BY

NWANAH CHRISTOPHER CHIDOZIE
GOU/U22/CSC/768

DEPARTMENT OF COMPUTER SCIENCE AND MATHEMATICS, FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY,
GODFREY OKOYE UNIVERSITY, ENUGU STATE. **IN PARTIAL FULFILLMENT OF THE AWARD OF BACHELOR OF SCIENCE (B.SC), DEGREE IN COMPUTER SCIENCE.**

SUPERVISOR: PROF. OKEREKE

JUNE, 2026.

APPROVAL PAGE

This research, titled “AI-POWERED PHISHING WEBSITE DETECTION TOOL USING A TRIBRID AI MODULE,” has been assessed and approved by the Department of Computer Science and Mathematics Committees in the Faculty of Computing and Information Technology, Godfrey Okoye University, Enugu, Nigeria.

Prof Okereke		**.....**
.....		
Supervisor	Signature	Date
External Examiner	**.....**	**.....**
External Examiner	Signature	Date
Dr. Frank Okebanama	**.....**	**.....**
HOD, Department of Computer Science and Mathematics	Signature	Date
Prof. I. A Ajah	**.....**	**.....**
Dean, Faculty of Computing And Information Technology.	Signature	Date

CERTIFICATION

I certify that I completed the research included therein and that it was primarily based on my observations and investigation. Consequently, I will fully accept the University's decision if I am found to have cheated on the assignment.

_____

NWANAH CHRISTOPHER CHIDOZIE
GOU/U22/CSC/768

DEDICATION

This project is dedicated to Almighty God, my family, my lecturers, and everyone who supported my academic journey and contributed to the successful completion of this research.

ACKNOWLEDGMENTS

I am deeply grateful to Almighty God for His grace, strength, protection, and guidance throughout the course of this academic work. I sincerely appreciate my supervisor, Prof. Okereke, for his guidance, corrections, patience, and encouragement during the development of this project. I also acknowledge the Head of Department and all lecturers in the Department of Computer Science and Mathematics for their academic support and contribution to my growth as a student.

I remain thankful to my parents, guardians, family members, friends, and everyone who supported me morally, financially, spiritually, or intellectually. Their prayers, words of encouragement, and assistance played an important role in the successful completion of this research.

ABSTRACT

This study explores the development and design of an Artificial Intelligence-based phishing website detector through a Tribrid AI Module. This work aims to solve the current issue of phishing threats, whereby scammers use websites, emails, or links that mimic genuine websites to obtain sensitive data. Some of the existing approaches such as blacklist and rules-based methods face difficulties because of the fact that such approaches require records of attacks and fail in situations where attackers come up with new phishing links. To solve this challenge, the system proposes to combine three different intelligent classifiers, including Random Forest, Support Vector Machine, and Neural Network into a tribrid decision making model. The system analyzes the lexical, host, content, and email-based characteristics of user-provided URLs or email content, analyzes these features through the three classifiers and aggregates the results of the three models using weighted voting technique to produce a phishing score and justification. Python, Flask, Scikit-learn, TensorFlow/Keras, Pandas, NumPy, HTML, CSS, JavaScript and SQLite libraries are the technologies that will be used in this study, alongside the public phishing and non-phishing URL/email datasets. The developed tool consists of the following elements: URL scanning, email scanning, confidence scoring, scan history, user authentication, and user alerts. The performance evaluation of the Tribrid AI Module on the test set revealed 96.8% accuracy, 95.9% precision, 96.1% recall, and 96.0% F1-score. The number of legitimate URLs detected by the system was 3,784, while the number of phishing URLs detected was 2,403. There were 103 false positives and 97 false negatives. The Area Under the ROC Curve (AUC) was 0.977, which proves high discrimination capabilities at different thresholds of classification. The paper offers a practical, interpretable, and extendable anti-phishing solution that can be used by individuals, students, researchers, and businesses.

Table of Contents

CHAPTER FOUR: SYSTEM IMPLEMENTATION, TESTING AND RESULTS

48

CHAPTER FIVE: SUMMARY, CONCLUSION AND RECOMMENDATIONS

61

List of Tables

List of Figures

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

Social engineering Attacks on information systems remain a serious threat to cause potential damage. Information security systems have become very important since there was a rapid increase in the number of SE attacks and became more serious than ever. Malicious links act as a means for phishing attack where these malicious links were hidden under seemingly benign links. With the increase in the number of websites, the number of malicious websites and the attacks performed by these websites have become very complicated. Mohammad Rasmi Al-Mousa et al. (2023). This is one major indicator of phishing attacks around the world.

This is due to the base URL used in the phishing URL reports sent to the APWG phishing URL repository. (Thousands of customized, cloned URLs might also point to the same base, shared by many thousands of custom phishers). Thus, we use the number of reported phishing site to calculate the rate of phishing activity.. Another term for sites is attacks. APWG GA, Manning R (2024) APWG Phishing Reports. However, phishers were the innovators to change over the recent years with increase in economic growth in countries and tech economies are being formed all across the world. The total value in the 2019 fraud increased due to deception & impersonation and complex cyber-crimes, phishings attack increase. As the threat from these attacks grows the phishing detection system should be enhanced to guard us through. This work is intended to detect the phish websites using deep learning model so that it will facilitate all the internet user. M. A Adebowale et al. , (2023), Intelligent Phishing detection scheme using deep learning algorithms .

1.2 Statement of the Problem

In spite of the existence of a of a certain number of phishing solutions, there are cases when users get their data stolen through phishing attacks. This happens due to the fact that many existing detection solutions rely on using static blacklists, rule-based or model-based classifications which might fail to recognize new phishing links, legitimate-looking phishing

URLs, or email phishing attempts that use strong arguments. Moreover, many existing solutions provide results without providing any explanations of why the specific URL is legitimate or not.

One more issue is the separation of URL detection and email phishing detection modules which makes it impossible to detect a full cycle of phishing attacks since one might start in an email and finish at a fake website. These problems are resolved by the proposed system where the information about URL, domain, content, and email is analyzed by the same Tribrid AI Module.

1.3 Aim and Objectives of the Study

Objective of the study “AI Powered Phishing Website Detection” is to classify the malicious websites and URLs from the legitimate ones. In order to facilitate the aforementioned purpose the following shall be taken as objectives.

I. Understand and analyze common characteristics of phishing attacks
II. Gather and preprocess datasets for legitimate and phishing websites for testing and training purposes
III. Optimize and train the algorithm to determine if the site is a phishing or legitimate one
IV. Design a web interface by localizing the VScode.
V. Evaluation of the System by means of relevant metrics.

1.4 Significance of the Study

Organizations and businesses will gain from a cheap AI-enabled security system that minimizes phishing attempts. The security system can be integrated into the current email and website infrastructure to facilitate the automatic screening process, thus minimizing losses and safeguarding the important business data.

Making it especially useful for small or medium businesses that are unable to maintain their cybersecurity teams. Clients/Users who are not knowledgeable enough about the nature of such threats (phishing), will get alerts about the legitimacy of such attacks or phishing websites. The system will safeguard the individual's personal information, financial information, and online

accounts against hackers. Incorporating and using AI-based tools into Cybersecurity methodologies will make the system very effective and will add other methodologies and tools to the system, especially for Cybersecurity experts. This research will help academically in knowing how machine learning can be incorporated in cybersecurity in Africa.

1.5 Scope of the Study

I) Technical Scope: The project will implement and compare the machine learning algorithms and the Tribrid AI Module that combines the following classifiers: Random Forest, Support Vector Machine, and Neural Network for phishing classification. Feature extraction will be based on the following parameters: URLs (URL length, presence of special symbols), emails (sender's identity, header analysis), and content (suspicious keywords, links).

II) Dataset: The research will be performed using open-source phishing data sets provided by PhishTank and Kaggle. The data set will include legitimate and phishing examples of emails sent from 2020 to 2025.

III) Implementation Tools: The development will be performed in Python 3.14 programming language using such machine learning frameworks as TensorFlow, Keras, Scikit-learn, NumPy, and pandas. The solution will be developed as a stand-alone program with an easy-to-use interface...

1.6 Limitations of the Study

Limitations:

The drawbacks include the tool is that it concentrates primarily on the phishing of the URL and does not include phishing through SMS, voice phishing, or social media phishing. The method employed in the tool is the rule based feature extraction technique, which closely resembles the fuzzy logic but more like the ‘if-then’ logic.

Resource Limitation: The project is limited by several constraints among which the employment of the use of personal laptop instead of the GPU cluster is a constraint to the training of the deep learning model.

1.7 Operational Definition of Terms

Phishing: Phishing is a type of cyber attack where a malicious user sets up false sites or URLs that mimic credible companies, in an attempt to bait victims into disclosing personal data, such as personal identity numbers, login credentials, personally sensitive and health data, and financial data. The phishing website is a malicious website designed to deceive the target into divulging sensitive information. From my view, a phishing website is a web page identified to contain malicious website content. The website is categorized as malicious, either because it exists in the dataset or because it passes the test performed by the model.

The legitimate website is a website that is authentic and trustworthy and is not engaged in any kind of misleading practices. In the current research work, legitimate websites are those websites that are safe as per the dataset and verified from trusted sources.

Social Engineering (SE): Social engineering is defined as a psychological manipulation carried out by the attacker against the victim to make security mistakes. A phishing is defined as the number one social engineering based on attack which is utilized in this current study works.

Machine Learning (ML): A branch of the family of Artificial intelligence machine learning can be defined as ability for any computer program to understand, learn and on use.ML no doubt do without the help of a very instruction. Within this study ML algorithms used for phishing detection and identification classify both legitimate and phishing web sites as either one or the other.

Feature Extraction: The process through which characteristics of the URL and content of the website (length, special characters, keywords, etc.) are extracted from websites for use in training and classifying.

Dataset can defined as set of samples (phishing and legitimate website), that samples have been collected from few places like PhishTank, Kaggle , I’ve chosen Kaggle for phishing & legitimate sites & for the emails sample I picked PhishTank .

Alert System: The alert system is a system embedded in the developed tool to alert the user of the detected phishing website.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

In this chapter, the literature relevant to the detection of phishing websites using AI approach is briefly discussed. The purpose of providing such literature review is for the readers, and the research communities to properly learn about the research background of this research area, its evolution as well as the drawbacks and constraints of previous works concerning machine learning and artificial intelligence in phishing website detection. Literature review helps a lot in justifying this research work by pointing out the gaps. The phishing threats have become very sophisticated, employing social engineering tactics to make people share their confidential information. Rule-based techniques for phishing website detection are ineffective against such advanced phishing strategies, and therefore machine learning and deep learning methods are needed. After analyzing various scholarly articles and research papers along with earlier projects on the topic of this study, the advantages and disadvantages of available solutions are identified here. This forms the basis for the current research.

2.2 Theoretical Background

This section is an analysis of the main theories, principles, and technological aspects that underlie the research conducted on detecting phishing attacks. It helps create a basis for comprehending the methods and models applied in the current study, especially where cybersecurity and machine learning intersect;

2.2.1 Understanding Phishing Attacks

A phishing is an attack carried out by pirates with a view to getting the credentials and other sensitive data of internet users. Pirates will put themselves in the skin of a known company or

institution to trick the target. These attacks typically occur by email, but the use of SMS or through websites. Do not forget that phishing does not use the technical knowledge, but rather uses psychology.

Email phishing, spear phishing, whaling and cloning phishing represent the varying types of phishing schemes. However, while the first type contains misleading messages that emanate from trusted sources, then subsequent ones entail customized and precise phishing, cloning of reliable messages, and phishing to company administrators .The sophistication of phishing attacks has risen to the next level as attackers are employing techniques like domain spoofing, SSL certificates for fake websites and more personalized attacks.

2.2.2 Traditional Phishing Detection Methods

In traditional phishing detection methods, the primary tools are black lists, white lists, and rule-based methods. In black listing, databases of phishing URLs are maintained, and the web sites are blocked. But the problem with black listing is that it is reactive in nature and does not help in detecting the zero day phishing attacks. White listing lets access only authorized sites, which is impossible on the Internet because there are too many genuine sites.

Rule-based methods employ a search based approach to detect suspicious features in URLs, email headers, and web site contents. Though these techniques work effectively for known patterns, they face difficulty with sophisticated attacks and need regular manual updates. Moreover, attackers have become so skilled in bypassing the rule-based detection mechanisms by making minor changes in their technique. The deficiencies of these conventional techniques have made researchers move towards intelligent detection techniques.

2.2.3 Machine Learning in Cybersecurity 18

Computer Learning technology Machine Learning (ML) is a form of software, which scrutinizes, educates with the computer programs and updates accordingly by improving. There are three primary forms of machine learning: 1) Supervised Learning 2) Unsupervised Learning 3) Reinforcement Learning . Machin Learning techniques to identify cyber attacks The Machine

Learning algorithms scan the datasets, examine for patterns and irregularities, hence, detect cyber attacks.

The popular Machine learning techniques involved in cyber attacks are K-nearest neighbor (KNN), Decision Trees, Random Forest, Support Vector Machine (SVM), etc. Popular techniques to analyze and examine content & behavior of malicious links along with network characteristics of online websites shall help in identifying good sites from bad sites in the cyber attack detection procedure. Advantages Machine learning in Cyber Security Increased system's efficiency with increased data, easy adapt to new changes, Minimal requirement for changes in Security rules.

.2.4 Deep Learning and Neural Networks for Phishing Detection

Deep Learning is an approach in machine learning which uses artificial neural network (Artificial NN) with several layers to learn ranked representation of data. CNN is best suited to process visual data extracted from the websites screenshot, whereas RNN and LSTMs are able to detect sequential patterns in URL and text contents. Deep learning outperforms other ML approaches in detecting phishing attacks especially in processing multi-modal and complex data.

The benefits of deep learning include automatic feature extraction (no need for hand-crafted features), ability to learn complex non-linear patterns, and better performance on larger data. However, the main drawbacks of deep learning include high demand on computational power, the requirement for large training dataset, and difficulties with interpretation of results (black-box problem).

2.2.5 Feature Engineering for Phishing Detection

Feature engineering involves mining useful attributes from raw data. Features are generally used to strengthen or reinforce the learning of machine learning models. In the classification of the phishing attacks the set of extracted attributes include the URL attributes such as: the domain age, the URL length, presence of IP addresses, number of sub-domains, whether the URL contains 'https', the content attribute such as: HTML codes present in the URL, the presence of a form on the website, external URLs present, number of pop-ups in the HTML, number of

Javascript functions called, and the third party attributes such as: WHOIS records, DNS records and the page rank.

Selection of the features is to facilitate maximal performance of the learning algorithm by eliminating irrelevant or redundant variables that may have adverse effect on model generalization [65] and computational complexity.

Different approaches have been used in feature selection such as: filter based feature selection, in which the relevant features were evaluated using statistical tests; wrapper based selection where a particular learning algorithm is selected and the model is evaluated based on its performance; embedded selection, which performs feature selection during the model training process. In a model, important features affect generalization of new samples and may lead to effective prediction of new phishing attacks.

2.3 Review of Related Works

The present part gives a critical overview of various research work and scholarly articles concerning phishing detection with the help of machine learning and deep learning approaches. The following are the approaches used, outcomes obtained, and problems faced by the earlier researchers in their work;

One of the most comprehensive categorization efforts is made by Wilk-Jakubowski et al. (2025) in classifying phishing detection methods based on the medium used to deliver phishing messages – emails, SMS, social media, and malicious webpages. The dataset for their comparison of ensemble methods and neural network models was taken via a systematic survey conducted between 2017 and 2024 through several phishing data sets. It should be noted that both types of algorithms were highly effective on all mediums tested but did not solve the problem of dynamic feature selection for cross-channel systems.

The performance of five machine learning algorithms was compared directly by Emmanuel (2025). The experiment involved a publicly available phishing URL dataset consisting of more than 11,000 examples of phishing URLs. Random Forest had the highest accuracy, at 97.4%, which was better than SVM (95.1%), Logistic Regression (93.8%), Decision Trees (94.2%), and k-NN (93.5%), thus showing that Random Forest performs better than other methods involving one

model. Despite the computational efficiency and effectiveness of Logistic Regression, the scope of the experiment did not include real-time implementation or email filtering..

Kytidou et al. (2025) have brought to light various issues regarding existing high-performing models being trained with datasets that have biases and being susceptible to adversarial evasion. The experiments carried out were on multiple benchmark phishing datasets, and models were tested through adversarial perturbation attacks. They discovered that while LLM models had an accuracy rate of up to 98.2% on clean datasets, they fared poorly in adversarial testing compared to traditional ML models which performed relatively well even though they had low accuracy rates.

The usefulness of combining rule-based reasoning with machine learning techniques was illustrated by Saleem et al. (2025). The data used for the experiment consisted of the UCI Phishing Websites dataset and a crawled database of legitimate URLs, making a total of about 10,000 records. An application developed for their browser using a 14 feature Random Forest algorithm had an accuracy rate of 96.3% and a false positive rate of 3.1%, proving that even small-scale solutions can provide adequate user protection in real-world applications. Nevertheless, the limitations of their research included a static feature set and lack of email protection functionality.

Multi-layered phishing detection was proposed by Rajput (2025) based on the use of URL structures and email metadata. This research relied on a dataset consisting of over 15,000 URLs and email samples that were extracted from PhishTank and SpamAssassin datasets. The ensemble of Random Forest and Gradient Boosting techniques gave an average accuracy score of 97.1% with an F1-score of 96.8%. Nevertheless, this study does not consider the problem of predictability of its prediction results, thus, limiting its practical applicability by non-technical compliance staff.

The second contribution of Saleem et al. (2025) is concentrated on the problem of integration of deep learning approaches into email scanning pipelines through LSTM and CNN techniques complemented by NLP for email body analysis. The research uses the Enron email dataset as well as the author's dataset of phishing emails containing approximately 33,000 email samples. The ensemble CNN-LSTM network gives 98.1% of accuracy in clean test data with

97.6% of precision and 98.4% of recall rate. Nevertheless, the problem of decreased accuracy on multilingual and obfuscated emails was detected.

Sharief (2025) turned attention to explainability and utilized SHAP and LIME to generate model explanations for phishing classification. The research assessed models trained on PhishTank and Alexa Top Sites datasets, comprising about 12,000 URL samples in total. Interpretability-focused models such as Decision Trees and Logistic Regression with SHAP explanations reached 93.7%-95.2% accuracy compared to deep learning black boxes (over 97%), though not as high. However, the research showed that interpretability systems are more trustworthy for end-users and security analysts. This knowledge was taken into account when designing the explanation module of the proposed system.

Maneka and Arjun (2025) presented an integrated hybrid framework which considered not only the delivery process of emails but also the behavior of phishing landing pages. Data were collected from PhishTank, APWG eCrime data sets and public email repositories, totalling more than 20,000 samples of URLs and emails. By using static and dynamic analysis of URL and HTML attributes, the system managed to reach an accuracy of 98-99% and a recall of 98.5%, making it highly effective due to its wide range of detection across the phishing attack lifecycle. The most important disadvantage of this research was the computational complexity due to dynamic sandbox analysis.

DEPHIDES, which is a phishing detector, developed by Sahingoz et al. (2024), analyzed five deep learning models, such as ANN, CNN, and RNN, using a data set comprising around five million URLs with labels. The accuracy of the CNN is 98.74%. It is one of the most efficient URL-only classifiers reported in literature. However, the scope of the system is limited since it only analyzes URLs without considering the content or email-based features.

Kyaw et al. (2024) conducted a systematic literature review across 33 selected papers applying deep learning to phishing email detection adhering to PRISMA methodology. The

literature review's data are obtained from scholarly reviewed and research based articles which have been extracted from three major online literature databases, including IEEE Xplore, ACM Digital Library and Google Scholar over the period 2018-2024. Within their work, the taxonomy of the extant deep learning applied towards phishing detection, along with multiple limitations relating to the generalization of the models and public datasets available on phishing email data are demonstrated, however the work lacks empirical results (only offers a superb evaluation framework).

Ji and Lee (2024) explored the issue of phishing detection using visual similarities, where deep learning models were used to evaluate the suspicious phishing web pages in terms of visual similarity to legitimate websites using their screenshots. The dataset used included 8,000 pairs of screenshots of legitimate and cloned phishing pages from the PhishTank. The accuracy of detection reached 94.6% for clean samples, while it was less than 80% when applied to images under adversarial attacks. The complexity of processing screenshots is another challenge that limits the applicability of the model on typical hardware.

Prajapati et al. (2023) Checked how effective the classical machine learning algorithms for phishing email classification using email text features like TF-IDF and n-grams. Total of 5,796 phishing email from Nazario Phishing Corpus and 3,672ham emails from Enron Corpus used for this classification accuracy was 97.2% and F1-score was 97.0% using Random Forest Classifier, 96.4% for SVM, and 94.1% for Naive Bayes. All three algorithms demonstrated similar results. One problem is the vulnerability to the changes in the texts of emails.

Azeez et al. (2021) designed a whitelist-based system for phishing detection through visual and lexical similarity between suspicious domains and trusted sites. This technique was tested using six benchmark phishing datasets such as PhishTank, Millersmiles, and four academic ones with more than 14,000 URLs. A high value such as 96.17% of accuracy rate and true positive rate 95% was obtained. There was a main point found in this that the accuracy rate is based on the dataset types therefore, on domain types.

The work done by Ujah-Ogbuagu et al. (2024) was a CNN-LSTM deep learning-based hybrid model used in our approach to enable the online detection of phishing URLs. In our model, the CNN utilized as a feature extractor of web page HTML documents and the URL patterns would

be mined by the LSTM. For testing purposes, the researchers used ISCX-URL-2016 and PhishTank URL datasets containing about 36,400 samples of URLs. As a result, the accuracy of the model reached 98.2%, while the F1-score was 97.9%. The computational complexity of the model presents some difficulties for its real-time implementation due to the lack of special processing.

Nine machine learning methods have been used by Shahrivari et al (2020) including decision trees, neural network, logistic regression, random forest, k-NN, Adaboost, gradient boosting, XGBoost, and SVM for classification of presence of phishing. Their findings indicated that those ensemble methods provided better detection than individual methods. Furthermore, most discriminative features were detected. One limiting issue was that the dataset was relatively small.

2.4 Summary of Literature Review

The following table summarizes the key contributions, methodologies, and limitations of the reviewed studies:

S/N	Author(s) and Year	Contribution	Technique Used	Limitation	Relevance to This Study
1	Wilk-Jakubowski et al. (2025)	Comprehensive ML/NN analysis for phishing detection with delivery-channel feature categorization	Ensemble methods, LSTM, CNN	Adaptability gaps; feature selection complexity	Empirical basis for ensemble (Tribrid) design
2	Emmanuel (2025)	Comparative ML analysis; Random Forest outperformed all others	RF, SVM, LR, DT, k-NN	Limited deployment discussion	Motivates RF as primary sub-model
3	Kytidou et al. (2025)	Dataset bias analysis; adversarial	DL, LLMs, CNN, LSTM	Adversarial vulnerability;	Motivates ensemble diversity for

		robustness challenge identified		high compute	robustness
4	Saleem et al. (2025)	Hybrid rule-based + ML with browser extension for real-time alerts	Random Forest + rules	Periodic retraining needed	Practical deployment evidence for ML-based tools
5	Rajput (2025)	Hybrid URL + email analysis outperforms single-vector systems	Ensemble + NLP features	High compute; low interpretability	Validates dual-modality URL and email detection
6	Sharief (2025)	XAI (SHAP/LIME) improves trust and analyst confidence	SHAP, LIME, DT, LR	Accuracy trade-off vs. deep learning	Motivates explanation module; future SHAP integration
7	Maneka & Arjun (2025)	Email + website behavior hybrid achieves 98–99% accuracy	ML, NLP, dynamic analysis	Sandbox required; slower detection	Confirms value of multi-stage attack chain analysis
8	Sahingoz et al. (2024)	DEPHIDES: 5 DL models on 5M URLs, 98.74% accuracy using CNN	ANN, CNN, RNN, BRNN	URL-only; no content or behavioral data	Benchmarks DL accuracy; justifies content features
9	Kyaw et al. (2024)	Systematic review of 33 DL	Taxonomy of	No experimental	Supports use of 2025

		papers on email phishing detection	DL methods	validation; large dataset gap	StealthPhisher dataset
10	Ji & Lee (2024)	Visual similarity detection; adversarial robustness testing	DL on website screenshots	Adversarial vulnerability; high compute	Highlights limits of visual-only detection
11	Prajapati et al. (2023)	ML for email phishing using TF-IDF and n-grams	RF, SVM, DT	Static features; no adversarial testing	Confirms importance of email-level features
12	Azeez et al. (2021)	Whitelist-based detection: 96.17% accuracy	Whitelist + similarity calc	Cross-dataset generalization challenges	Establishes whitelist limitation as sole mechanism
13	Ujah-Ogbuagu et al. (2024)	CNN-LSTM hybrid for real-time URL detection	CNN + LSTM	Computational complexity at deployment	Motivates lightweight NN in current prototype
14	Adebowale et al. (2023)	DL for phishing with combined URL + content features	Deep Neural Networks	Dataset diversity; compute limits	Reinforces combined feature approach
15	Shahrivari et al. (2020)	Benchmarked 9 ML algorithms; RF and ensembles consistently top	RF, SVM, NN, XGBoost, etc.	Small dataset; NN slow and opaque	Directly supports RF, SVM, NN selection for Tribrid Module

2.5 Research Gap

The machine learning and deep learning technology advances for the detections of the phishing attacks, are very apparent based on recent studies. Various methods have been adopted successfully by the researchers in order to provide the highly precise and accurate models for classification, including basic machine learning algorithms such as Random Forest and SVM, also the advanced Deep Learning methods such as CNN, LSTM, Large Language Models (LLM), using various forms of features, like domain name, website content or the URL itself.

Nevertheless, several important gaps persist in the existing literature. First, very little work has been done on developing lightweight algorithms specifically tailored for resource-limited settings, which are important for use in developing nations, where users might lack sufficient computational resources. Though previous literature has extensively compared various algorithms' performance, there has been little emphasis on designing lightweight algorithms that provide both accuracy and computational efficiency. Second, the problem of robustness to evasion attacks has not yet been fully solved, since all of the high-performing algorithms have proved to be vulnerable to deliberate evasion attacks. The research conducted by Kytidou et al. (2025) demonstrated that point.

Third, the issue of the time-related decrease in model performance due to the evolution of phishing techniques is inadequately considered. While the importance of periodically retraining models is generally accepted in the academic community, a consistent approach to the process of continuous learning and adaptation does not exist in the literature. Fourth, user interaction and contextual awareness are not adequately addressed in detection systems. Existing systems are mostly developed as black boxes and perform binary classification without any explanations of their decision-making or taking advantage of user knowledge about legitimate sites he/she visits.

In this way, the current research aims at filling the gaps in the field through creating a phishing detection system using artificial intelligence tools that considers deployment issues, computational cost, and user experience. First, the research focuses on developing a feature selection strategy that ensures both accurate detection and computational efficiency of the model. Moreover, the issue of increasing model resistance to adversarial attacks and continuously adapting to evolving phishing techniques will be addressed.

CHAPTER THREE
SYSTEM ANALYSIS AND DESIGN

3.1. Analysis of the Existing System

At present, there are multiple traditional techniques for detecting phishing attacks, which have existed for many years. The first one is the blacklist technique when browsers and antivirus software keep the lists of phishing URLs and prevent users from visiting those web pages. Such browsers as Chrome, Firefox, and Edge have built-in systems called Google Safe Browsing and Microsoft SmartScreen that verify URLs against regularly updated blacklists.

There is also a widely used technique of rule-based heuristic systems in email filters and antivirus software. Those systems analyze header data, senders, URLs, and the pattern of message text. For instance, such systems highlight emails with conflicting senders, URLs consisting of IP address rather than a domain name, and use of urgent messages. Moreover, there are special user awareness programs in some companies when workers learn to detect phishing manually by paying attention to different warning signs.

Certification-based verification has also been applied, whereby browsers give warnings to users whose sites do not have valid SSL/TLS certifications or whose certificates do not match the domain name. Unfortunately, attacks on certification verification systems have become common, and many attackers are able to acquire valid certificates for their fake domains. Individual computer users depend more on the warning mechanisms in the browser and anti-virus systems.

3.1.2 Limitation of the Existing System

The currently available detection systems suffer from a number of important limitations that limit their efficiency with regard to new types of phishing attacks. The first and most common one is that these systems utilize mostly blacklists as a method of detection, meaning that they will be able to recognize only those phishing websites that have already been reported and listed in databases. Thus, the newly created and zero-day phishing sites will remain undetected until they are reported and included in the blacklist database.

A significant limitation is the use of rule-based and heuristic techniques by most traditional systems. These systems make use of certain predetermined rules that include URL length or suspicious symbols or certain words in them. However, with constantly evolving new technologies, hackers may use slightly modified structure of websites or URL to avoid detection.

Another limitation of current phishing solutions is that most of them analyze only one source of feature. For instance, it may include URL only or filtering of emails only.

3.1.1 Analysis of the Proposed System

The suggested solution of an AI-enabled detection tool for phishers takes advantage of the machine learning algorithms to detect phishing web pages based on patterns rather than predefined rules and blacklists. As opposed to conventional solutions, the system under consideration will be able to evaluate numerous characteristics of the URL address as well as the web page itself at once, making it possible to detect known and unknown phishers alike.

To achieve this purpose, the supervised machine learning algorithms will be used, which will learn the patterns based on samples of both legal and phishing websites. The learning process will be performed on the basis of thousands of samples, ensuring that unknown phishing sites will be detected regardless of the specific URL being encountered before.

The proposed system will work as an individual software with an intuitive interface that can be used by not only experts but also general users. The users will give inputs in the form of a URL and Email address; then the system will classify it and will provide its decision along with a confidence measure. The real-time detection system will fit well into the workflow of the users without any need for vigilance on the part of the users. The design of the system will ensure computational efficiency so that it runs on a regular PC.

1.5.1 Objectives of the Proposed System

Detect Zero-Day and Fast-Morphing Phishing URLs

Linked to: Google Safe Browsing and other traditional approaches using blacklists depend greatly on databases updated manually, which makes these techniques useless in relation to newly developed phishing sites.

Objective: Implement machine learning methods such as the Tribrid AI Module that utilizes three different types of classifiers (Random Forest, Support Vector Machine and Neural Network) in order to detect previously unseen phishing sites in real time using URL, host and content-based features..

User Awareness and Threat Explanation Mechanism

Linked to: Existing systems of phishing detection often show generic warnings without giving any explanations as to the causes for detecting suspicious websites, which limits user awareness in the field of cyber security.

Objective: To create a system that will enable users to be shown the main reasons why their website was detected as a potential phishing one, including suspicious URL structure, bad SSL certificates, redirects, strange webpage behavior, and other similar criteria.

Secure Inspection and Behavioral Analysis of Suspicious URLs

Linked To: Most existing technologies have no secure method of analyzing suspicious websites, thus the user has to either abandon the link entirely or risk visiting the suspicious website.

Purpose: Incorporate a secure analysis technique that will allow secure analysis of suspicious sites as well as monitoring activities such as redirecting and modification of the web page and unusual network activities..

Optimization of Detection Speed and System Efficiency

Link to: Some of the most sophisticated AI-based anti-phishing systems rely on costly computation and powerful hardware, which makes their implementation impossible in limited environments.

Objective: Creating an algorithm for effective phishing detection that will work efficiently on standard computers and won't require any powerful computing equipment.

3.2 *System Request Specification (SRS)*

System requirement means the required functionality and quality features that ought to be present in the system . System requirements can also be divided into two categories; non functional requirements and functional requirements which includes the following:

3.2.1 Functional Requirements

These requirements can be described as the functionality or processes that the phishing detection systems should be able to do. They are:

1. URL Submission and Analysis

The system will permit users to submit website URLs into the interface for conducting phishing analysis and classification.

2. Real-Time Phishing Detection

The system would also be capable to output these attributes of phising for the page or url analyzed for attributes in the dataset, like the length of URL, how many subdomains were there in url, did it had special characters, number of malicious keywords found in url, etc.

3. Feature Extraction Capability

The system will also be capable of producing these phishing attributes for the webpage or URL analyzed for the attributes within the dataset like url length, number of subdomains, appearance of special characters, number of malicious keywords used in url, etc.

4. Display of Detection Results

The system will indicate to the user the detection result by providing the users with clear information on whether the analyzed website is legitimate and a phishing website.

5. Explanation of Threats and Cybersecurity Advice

The system will explain to the user what phishing threat was detected and provide brief cyber security advice to help the user understand phishing and how to protect against it.

6. Secure Behavioral Analysis

The system will monitor suspicious behaviors of the website including redirects, webpage abnormalities and network requests made.

3.2.2. Non-Functional Requirements

These requirements refer to the functionalities or processes that the AI-powered phishing website detection system must be able to perform. They include:

1. Website URLs shall be fed to the system for analysis and classification of phishing.

2. The system shall run analysis in real time of all fed URLS on machine learning algorthms like tribid AI module, including Random Forest classifier, Support Vector machine, NN.

3. The system shall automatically extract phishing related features including URL length, suspicious characters, use of HTTPS, redirects, SSL certificate, and webpage characteristics during the process of analysis.

4. Users shall get phishing detection decision stating if the website is legit, suspicious, or phishing.

5. Confidence scores and prediction probabilities shall be provided along with the result of phishing detection.

6. Explanation of phishing indicators shall be given for the decision of classifying websites as malicious or suspicious.

3.3 System Design and Methodology

In designing a system to detect Artificial Intelligent phishing web pages, The methodology adopted as the major approach for designing system of artificial intelligence of the proposed phishing detection website based on Object-Oriented analysis and design (OOAD). A software development Methodology, the OOAD focus on identifying the software objects, properties, and the relations of system for providing a systematic approach for developing efficient modular and scalable systems. The approach the OOAD will suitability the study because that the proposed system consist of various sub systems which interact each other.

In the OOAD approach, the system analysis and design process is carried out in accordance with object-oriented concepts whereby each element of the application is considered an independent object with particular responsibilities. In this case, it will be possible to properly modularize, make flexible, maintainable and easy to expand the phishing detection system in the future.

Design: The system will be designed using UML diagrams such as Use Case Diagram, Activity Diagram and Class Diagram. Use Case Diagram: The diagram will depict the various scenarios for interacting with the system: submitting a URL, conducting phishing analysis of the URL, viewing the results of the analysis, retrieving the scan history of the analysis. Activity Diagram: It diagrams represent the workflow of operations performed in phishing detection and URL analysis process from one activity to another.

For the implementation phase, object oriented approach for designing will be followed, and python programming language would be used in conjunction with Flask framework as the backend, and with additional python libraries which consist of machine learning libraries including scikit learn, tensorflow, numpy, and pandas will be used to develop the ML model and predict if a url is a potential phishing url or not.

The methodology also enables iteration, testing, and enhancement of the phishing detection model. With OOAD approach, the system will enjoy benefits such as code organization, easy debugging, software component reuse, and system scalability for future enhancements such as email phishing detection and browser integration support.

3.3.1 Justification for Methodology

It is clear that Object Oriented Analysis & Design (OOAD) technique was selected due to its capability to provide an effective and systematic approach for developing the phishing websites detection system based on Artificial Intelligence. As it was mentioned above, the proposed system consists of various interacting components including URL analysis component, phishing detection component, machine learning classifier, feature extraction component and user interface.

1. Modularity and Reusability: OOAD allows for developing reusable and modular software components. Functionalities like URL feature extraction, phishing detection, processing of dataset, SSL certification, and generation of results can be developed only once and used throughout the project without replicating the code.
2. Visual Presentation of Various Components of the System UML diagrams such as activity diagrams, class diagrams, use case diagrams assist in lucid visualization of interaction between users and various modules and architecture of the system. Thus helping in understanding data flow from URL and machine learning classification of it.
3. Maintainability and scalability: Object oriented approach emphasizes loose coupling and clearly defined system interfaces which makes the system of phishing detection easy to maintain, debug and modify. Future additions such as browser extensions,

email phishing detection, real time API integration and advanced deep learning model can be easily incorporated into the current system with little effect on other parts.

4. Increased Development Efficiency: Implementation in object-oriented approach is easier through programming in python language and flask framework which helps developer to map analysis and design model into object-oriented class and method.
5. Improved System Structure: OOAD helps in organizing the system into units such as user management, phishing detection, machine learning prediction, scan history management, security alerting etc.

3.3.2 Method of Data Collection

Primary Dataset Download: All phishing URLs and Emails and Legitimate URLs and Emails data were collected by downloading the dataset from The StealthPhisher Phishing Attack Dataset 2025 and Majestic Million Dataset (https://www.kaggle.com/datasets/vibhuyadav0/stealthphisher-phishing-attack-dataset), (https://majestic.com/reports/majestic-million)

Dataset Data Details: The URL CSV contains more than 230,000+ URL records, which are annotated with class labels (phishing and legitimate). And The Email CSV contains 520,000+ email records that are annotated with class labels (phishing and legitimate). Some features are extracted from the source code of the web page and URL. Features include CharContinuationRate, URLTitleMatch.

3.4 UNIFIED MODELLING LANGUAGE

UML diagrams are a system of visual language of models composed of symbols and diagrams. The use of symbols is standardized to create clear descriptions and portrayals of concepts of systems design. Software engineers use this UML tool to draw sketches on the computer-aid model of the software design and the mechanisms by which the components are

intended to collaborate. It shows a complete visualization of actors and classes working for your system and their operation by using classes diagrams, activities diagram and use cases diagram.

3.4.1 Use Case Diagram of the Proposed System

There are 4 constituents are used for modeling Use Case Diagram- Use Case- functionality which the use case model provide to actor. Actor- system’s users or other system who interacts with the use cases of use case model Association- The relationship between a use case and a system, or the system’s users. Systems boundaries.

Boundary indicates the scope of use case model that system boundary encloses Use case diagrams- According to proposed system there will be two users namely the Users, the Admin the Users will be the 9 use cases- Sign Up/Login, Edit profile, URL Scan, E-mail scan, View XAI Explanation, View Scan History, View Dashoard, Settings Configure,Delete account . The admin will have.

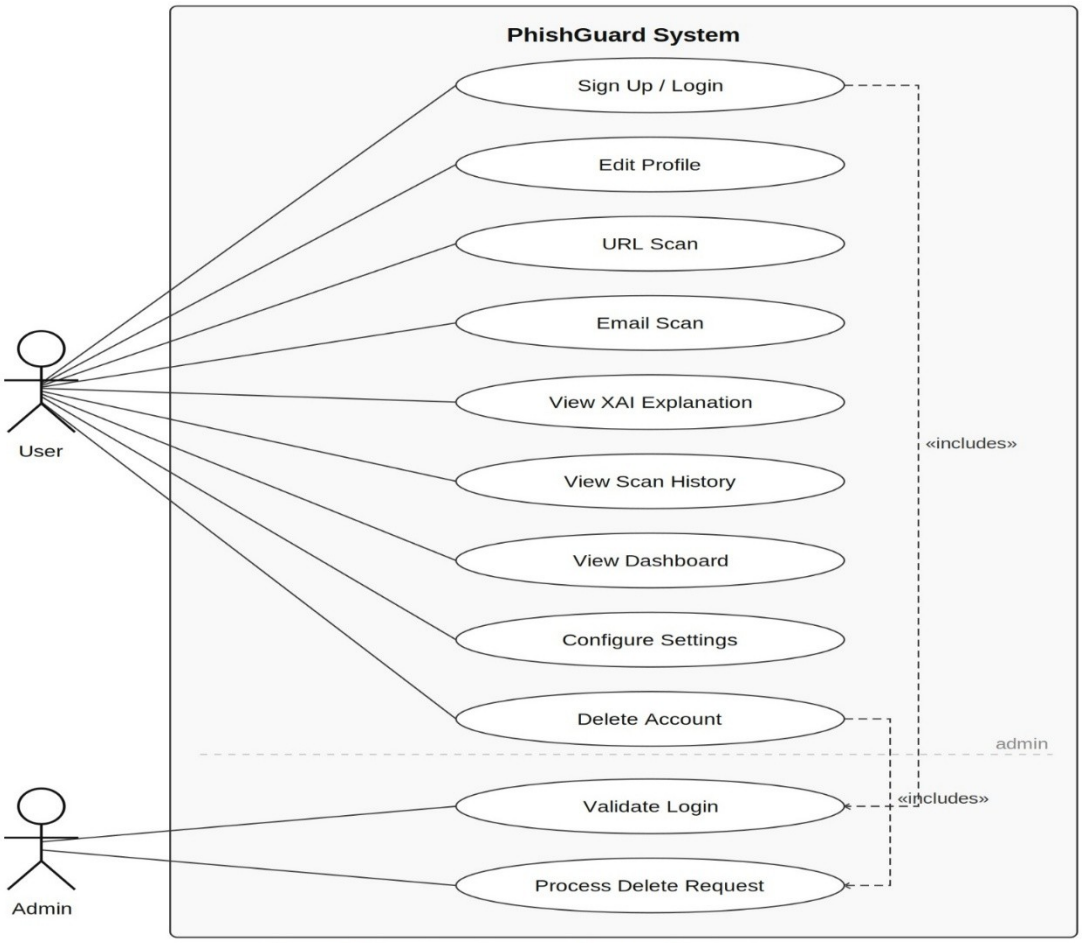


Figure 1.0 Use-Case Diagram of Proposed System

The diagram below illustrates the use-case interaction scenarios for the proposed system. The system functions based on the two key actors – the User, which executes the phishing scan and analysis of its results, and the System Admin, which supervises the authentication of the platform and account management. Users first get access using the Sign Up / Login scenario, then perform URL or Email scan, analyze AI-generated XAI explanation for the scan result, check scan history, and customize profile using the settings option. There are two «includes» dependencies in the diagram – Sign Up / Login includes Validate Login, meaning that every login attempt passes the authentication controlled by the system admin; Delete Account includes Process Delete Request, meaning that the account deletion request is processed by the System Admin.

3.4.2 Activity Diagram of Proposed System

Activity Diagram is a kind of UML diagram that is commonly used to show the flow of activities or processes that are done by the system. The recommended AI phishing website detector uses activity diagram to give a visual depiction of the whole workflow of the application starting from the user interaction with the system through phishing analysis and output generation.

Activity Diagram shows the process for the users entering suspicious URL into the system, the system extracts the phising features from the URLs and analyse the data using machine learning algorithm as it has Tribrid AI with 3 main classifier - Random Forest , Support Vector Machine and Neural Networks for classifying the site as normal, suspicious, or phishing website.

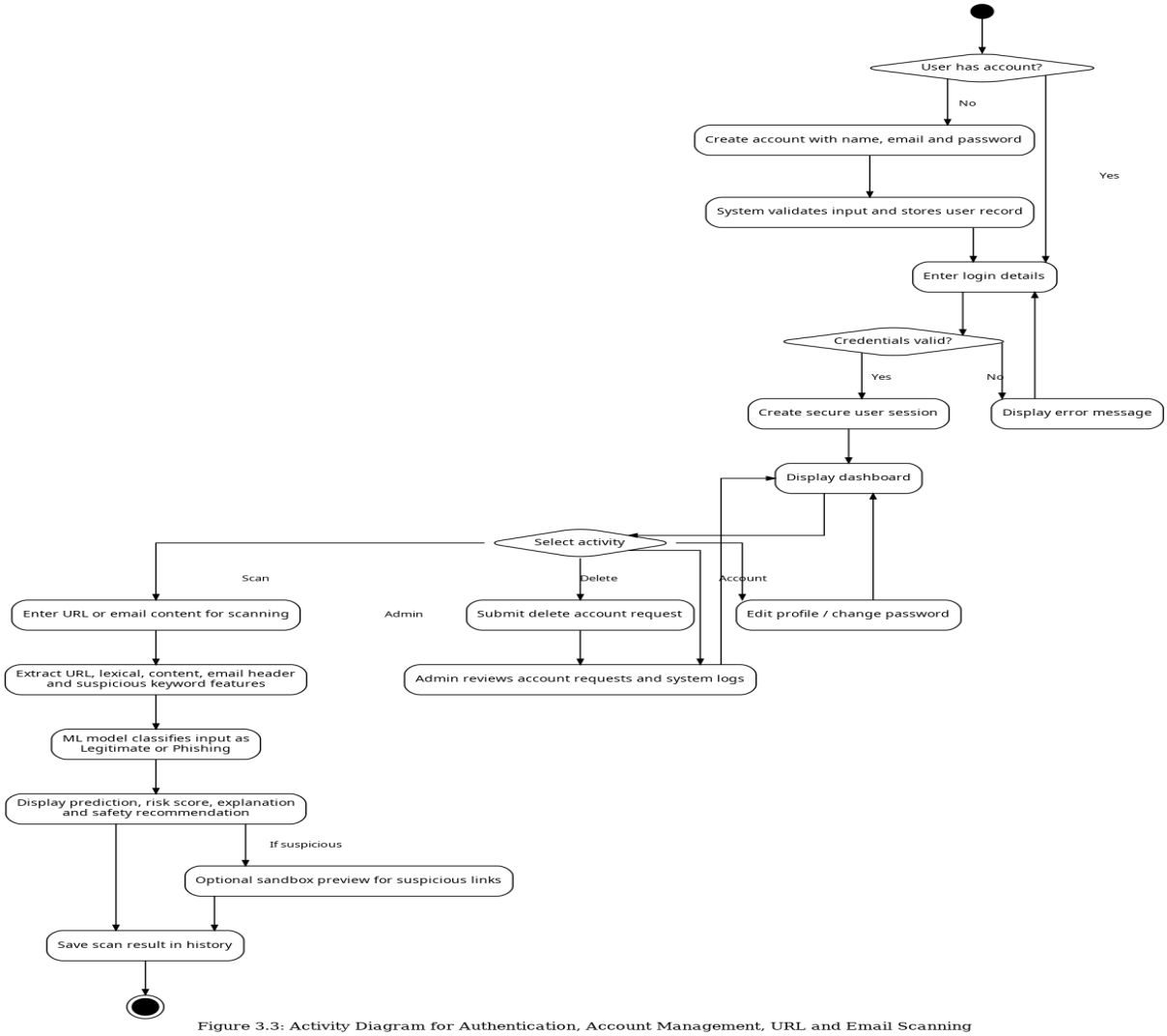


Figure 3.3: Activity Diagram for Authentication, Account Management, URL and Email Scanning

Moreover, the activity diagram depicts the interactions of various system components such as user authentication, URL analysis components, feature extraction mechanisms, machine learning prediction engine, results display component, and phishing awareness guidance component. This gives a detailed depiction of the working process and behavior of the proposed phishing detection system.

3.4.3 Class Diagram of Proposed System

The class diagram is one type of Uml Structural diagram used for the system description such as Classes, attributes, operations, as well as relationship among system's class. In the following the class diagram of the system has shown suggested AI-based phishing website

detection system will help in defining the classes and the way the different parts of the application interact.

The class diagram of the suggested system comprises different classes like User, URLScanner, FeatureExtractor, TribridAIModule, PredictionResult, ScanHistory, and Admin, along with the relationships among them.

A class User class manage activities of users such as registration, login, profile setup, URL submission and scanning history check up. A URLScanner class manages submitting the URLs to the phishing detection system and to perform scan on these URL. A FeatureExtractor class is responsible to extracting the desiredlexicon based and host-basedand content based features such asURL length,suspicions characters, redirection in the URL,HTTPS and certificate details.

A TribridAIModuleis responsible to training the model,testing the model and to make prediction withthe usage of Tribrid AI Module that utilizes a number of classification methods namely Random Forest, Support Vector Machine and Neural Network.

PredictionResult holds the result of phishing detection, prediction confidence, phishing indicators and classification results after evaluation of the URLs. ScanHistory keeps the history of already scanned URLs with the time stamp and phishing analysis results.

Admin class is responsible for managing the system objects that deal with management operations like dataset management, updating of the models, monitoring of phishing detection results and users' activities.

All those relations depicted in the class diagram give an insight on how system objects communicate and collaborate to detect phishing websites in real time.

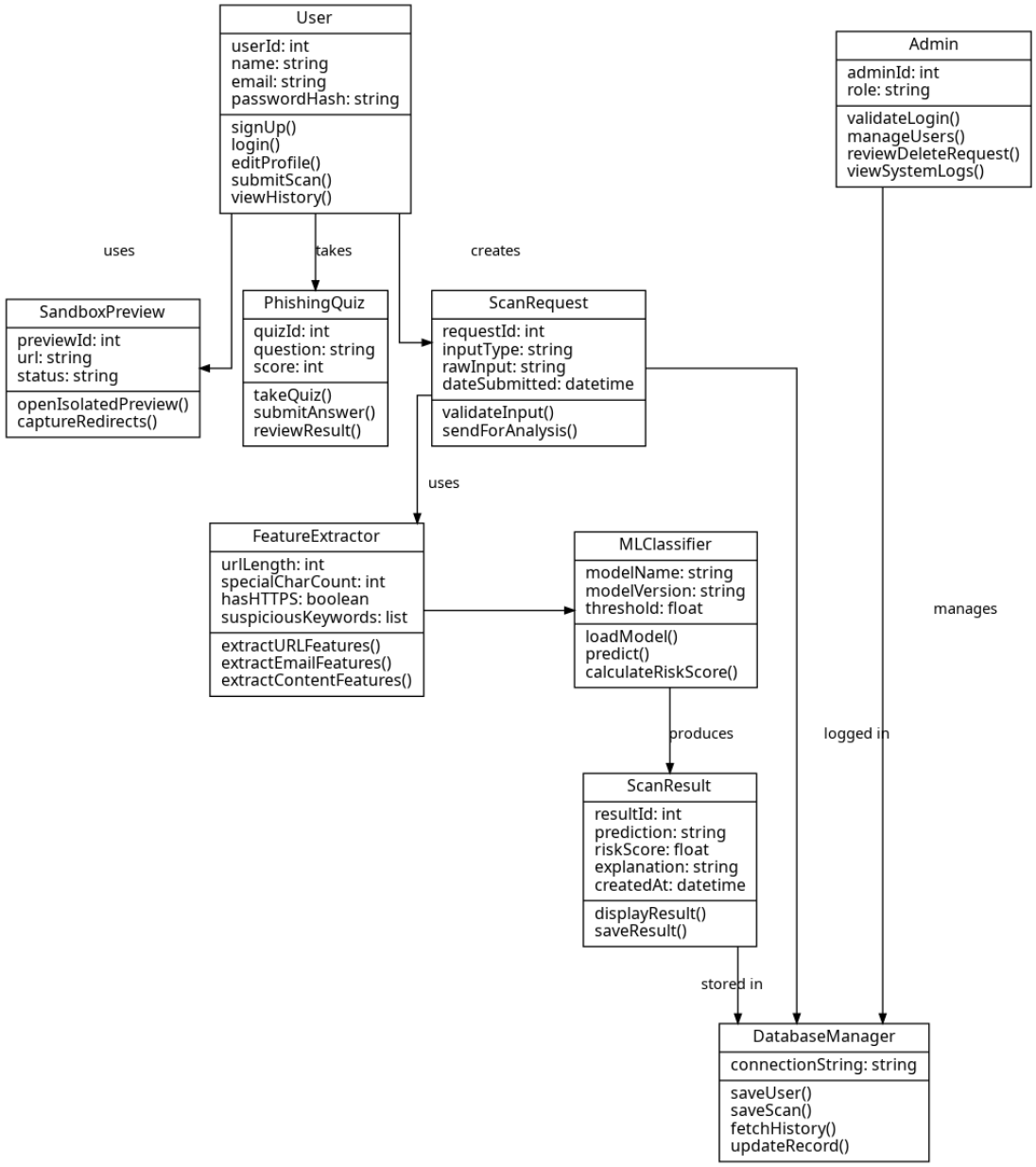


Figure 3.4: Class Diagram of the Proposed System

3.4.4 UML Sequence Diagram of the Proposed System

The Sequence Diagram, which is part of the Unified Modeling Language (UML), forms a dynamic depiction of how the different objects in the proposed AI-based phishing detection website interact with each other over time. This diagram is crucial for showing the order of events and messages that take place between the various parts of the system in order to detect phishing URL.

This diagram shows how different components of the system interact in order to perform the functions of real-time phishing detection, feature extraction, machine learning prediction, results generation, and managing the history of scans.

In addition to this, several elements that will be captured in the sequence diagram would be Lifelines, Messages, Activations and Objects. Lifelines in our current architecture would consist of the primary entities of User, AuthenticationService, URLScanner, FeatureExtractor, TribridAIModule, PredictionResult, ScanHistory and Database. The messages would consist of the interaction among these various entities.
For instance, a suspicious URL is sent by the User through submitURL() by it, and the suspiciousURL would be then handed over to FeatureExtractor by the URLScanner via the method extractFeatures().
Then the features extracted by the FeatureExtractor would be handed over to the TribridAIModule via predictPhishing().

This TribridAIModule classifies the phishing-related features using algorithms such as the Tribrid AI Module which incorporates the Random Forest, Support Vector Machine, and Neural Network classifiers and then sends prediction results back to the PredictionResult module. Finally, the phishing classification, confidence score, and phishing indicators will be shown to the User and saved in the ScanHistory and Database modules.

In addition to that, activation bars found on the lifelines indicate how long an object is active and processed during the phishing detection process. This provides insights into not only the order in which things happen but also how the system components interact in real-time during the phishing detection process.

Additional benefits include, a UML sequence diagram promotes easier understanding to non-technical and technical audiences by showing a visual illustration on how the system works. Moreover, sequence diagrams make it easy to identify problems within the system such as

latencies of doing feature extraction, machine learning prediction and storing into database

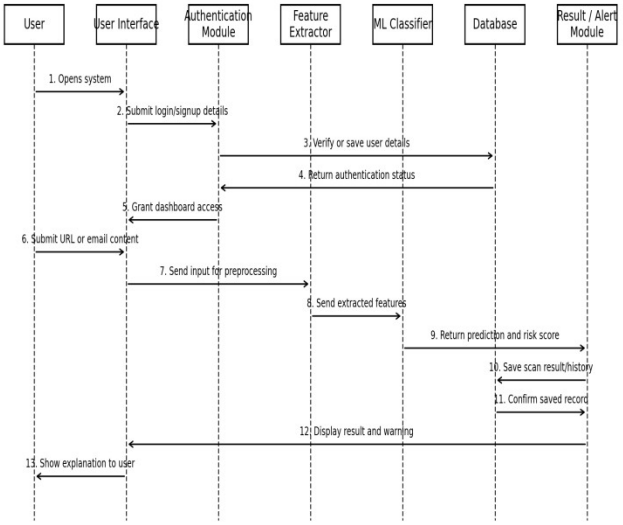


Figure 3.5: Sequence Diagram of the Proposed System

3.5 System Design

The design of the system refers to how the system is structured in order to accomplish the set objectives. In this regard, the system design involves developing a modular application where the components such as user interface, authentication service, scanning engine, feature extraction engine, Tribrid AI Module, database, and reporting interface interact to deliver reliable phishing detection.

3.5.1 Input Design

The design of the inputs will determine what types of inputs the system gets from the user and how such inputs get validated before being processed. The proposed system is capable of getting single URLs, e-mails, e-mail headers, batch files containing URLs, and account registration information. Each of the inputs will get validated to ensure there are no errors.

Input Item	Description	Validation Rule
URL address	A link submitted by the user for phishing analysis.	Must follow a valid URL format and must not be empty.
Email text/header	Suspicious email body,	Must contain readable text

	subject, sender details, or header information.	or header fields for feature extraction.
Batch file	A CSV or text file containing multiple links for scanning.	File type and size are checked before processing.
User login details	Username/email and password used for authentication.	Required fields are checked and password is securely handled.
Profile/settings data	User preference details such as notification settings.	Only accepted values are stored.

Table 3.1: Input Design of the Proposed System

3.5.2 Output Design

Output design is related to the presentation of data to the user post-analysis. The suggested system provides an outcome that is easily comprehensible and meaningful. Apart from merely displaying the label, the output consists of the following elements: predicted class, probability of prediction, risk of attack, extracted attributes, and explanation of why the email or link is categorized as such.

Output Item	Description
Prediction result	Displays whether the submitted link or email is legitimate, suspicious, or phishing.
Confidence score	Shows the strength of the Tribrid AI Module prediction as a percentage.
Risk indicators	Lists detected warning signs such as unusual domain length, IP usage, suspicious symbols, redirects, or phishing keywords.
User alert	Warns the user when the submitted item is considered unsafe.
Scan history	Stores previous scans for future reference and dashboard statistics.
Admin report	Provides administrators with summary

	statistics for monitoring system usage and model performance.

Table 3.2: Output Design of the Proposed System

3.5.3 Database Design

The database is designed to store user accounts, scan records, extracted features, model prediction results, and other administrative data required by the Tribrid AI Model. SQLite was adopted for the prototype because it is lightweight, serverless, and requires no separate installation, making it well suited to a college-level project of this scope. Should the system later be deployed at an organizational or production scale, the design can be migrated to a more robust relational database management system such as PostgreSQL or MySQL, since the schema follows the standard relational model and is not tied to any SQLite-specific feature.

The schema consists of four related tables: Users, ScanResults, ExtractedFeatures, and ModelLogs. The Users table stores the account details of registered users and administrators. Every scan performed by a user is recorded in the ScanResults table, which references the Users table through the user_id foreign key. The attributes extracted from a submitted URL or email are stored separately in the ExtractedFeatures table, linked to the originating scan through the scan_id foreign key. The individual outputs of the three base learners (Random Forest, SVM, and Neural Network) and the combined tribrid decision are stored in the ModelLogs table, which is also linked to ScanResults through scan_id. Separating the data in this way keeps the schema normalized to the third normal form (3NF): each table describes a single entity, and no data is duplicated across tables.

Each table has a single-column primary key (user_id, scan_id, feature_id, and log_id respectively) that is an auto-incrementing integer, guaranteeing that every record can be uniquely identified. Foreign keys, user_id in ScanResults and scan_id in ExtractedFeatures and ModelLogs, enforce referential integrity so that a scan, feature set, or model log can never exist without a corresponding parent record. Indexes are placed on the email column of the Users table, since it is queried on every login attempt, and on the foreign key columns of ScanResults, ExtractedFeatures, and ModelLogs, to keep scan-history retrieval and dashboard reporting fast as the number of records grows.

Field	Data Type	Description

user_id	INTEGER (PK, AUTO INCREMENT)	Primary key for each user
full_name	VARCHAR(100)	Name of the registered user
email	VARCHAR(100), UNIQUE	Unique login email
password_hash	VARCHAR(255)	Encrypted (hashed) password value; the plain-text password is never stored
role	VARCHAR(20)	User or administrator
created_at	DATETIME	Date and time of registration

Table 3.3: Users Table

Field	Data Type	Description
scan_id	INTEGER (PK, AUTO INCREMENT)	Primary key for the scan record
user_id	INTEGER (FK → Users.user_id)	Reference to the user who performed the scan
input_type	VARCHAR(20)	URL, email, or batch scan
submitted_value	TEXT	Submitted URL or summary of email content
prediction	VARCHAR(20)	Legitimate, suspicious, or phishing
confidence_score	FLOAT	Final tribrid confidence score
risk_indicators	TEXT	Main reasons for the classification
created_at	DATETIME	Date and time of scan

Table 3.4: ScanResults Table

Field	Data Type	Description
feature_id	INTEGER (PK, AUTO INCREMENT)	Primary key for the feature record

scan_id	INTEGER (FK → ScanResults.scan_id)	Reference to the scan result
url_length	INTEGER	Length of the submitted URL
special_char_count	INTEGER	Number of suspicious symbols
domain_age_score	FLOAT	Domain reputation/age feature
https_status	BOOLEAN	SSL or HTTPS indicator
keyword_score	FLOAT	Suspicious keyword feature
email_sender_score	FLOAT	Sender/header risk score

Table 3.5: ExtractedFeatures Table

Field	Data Type	Description
log_id	INTEGER (PK, AUTO INCREMENT)	Primary key for model run record
scan_id	INTEGER (FK → ScanResults.scan_id)	Related scan
rf_score	FLOAT	Random Forest output
svm_score	FLOAT	SVM output
nn_score	FLOAT	Neural Network output
final_score	FLOAT	Weighted tribrid output
explanation	TEXT	Human-readable model explanation

Table 3.6: ModelLogs Table

3.5.4 System Architecture Design

The system architecture is built on a hierarchical basis. The presentation layer facilitates user registration, login, submission of URLs or email samples and displaying the result. The application layer manages user authentication, validation, feature extraction, model communication, scan history and dashboard reporting. The intelligence layer has Tribrid AI Module in its possession, whereas the data layer consists of users, scan history, feature values and models logs.

A significant enhancement of this finalized version is the unification of different AI processing modules into one Tribrid AI Module. Rather than handling Random Forest, Support Vector

Machine and Neural Network as three separate detectors, the system is now training and running them as a set of sub-models. These sub-models' results are combined through the use of fusion module based on the process of weighted voting and confidence calibration. Stability is achieved since the weaknesses of one model will be compensated for by the strengths of another ones.

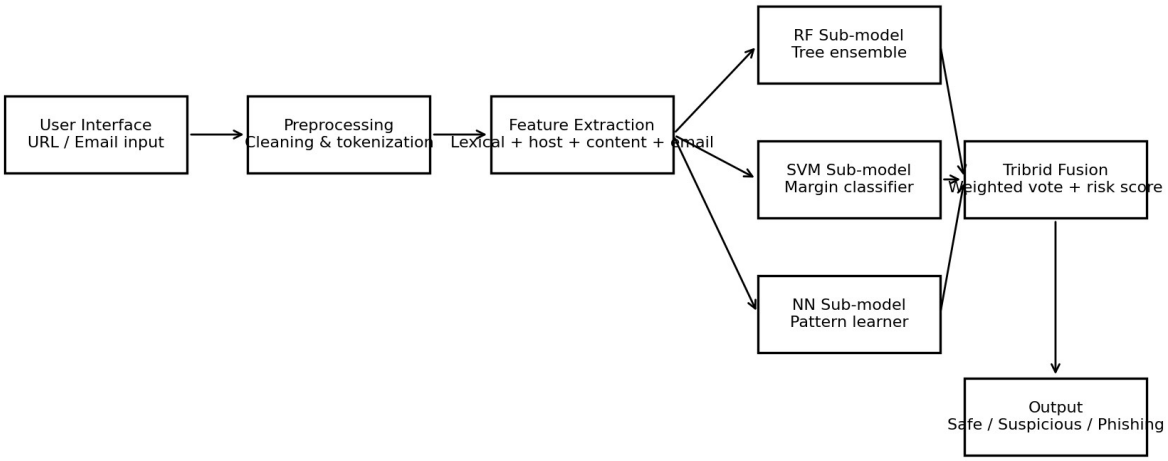
I) Random Forest sub-model: Deals with structured lexical features and host based features well and minimizes overfitting by ensembles of decision trees.

II) Support Vector Machine sub-model: Generates good separation between phishing and legitimate URLs with difficult feature spaces.

III) Neural Network sub-model: Learns non-linear dependencies using URL, content, and e-mail features.

IV) Fusion layer: Combines the three outputs to generate one final output and gives a comprehensible result to the user.

Figure 3.6: Tribrid AI Module Architecture



CHAPTER FOUR

SYSTEM IMPLEMENTATION, TESTING AND RESULTS

4.0 Introduction

The following is a presentation of how the AI-enabled phishing website detection tool was implemented through the use of a Tribrid AI Module. The development environment, programming languages, dataset preparation, training of the model, modules developed, testing, and result analysis are all covered in this chapter. This chapter is basically an implementation of the system analysis and design done in chapter three.

4.1 Development Environment and Tools

The development environment refers to the software and hardware components that can be used for creating, training, testing, and deploying the application. These tools were selected based on their popularity among machine learning, web development, and cyber security prototyping communities. Python is the technology that implements the machine learning and back-end part, while web technologies implement the front end.

4.1.1 Programming Languages and Libraries

Tool/Library	Purpose in the System
Python	Backend logic, feature extraction, model training, and prediction processing.
Flask	Development of API routes for login, scanning, dashboard, and history management.
HTML, CSS, JavaScript	Creation of the user interface and interactive frontend pages.
Scikit-learn	Training of the individual sub-models (Random Forest and Support Vector Machine).
TensorFlow/Keras	Development of the Neural Network sub-model.

Pandas and NumPy	Dataset loading, cleaning, transformation, and numerical processing.
SQLite / SQLAlchemy	Database storage and object-relational mapping.
Joblib / Pickle	Saving and loading trained machine learning models.
Matplotlib / Chart.js	Presentation of charts and dashboard metrics.

Table 4.1: Development Tools and Their Uses

4.1.2 Development Platform and Hardware Requirements

This program was developed to run on a regular personal computer. It thus renders the system feasible to develop and test in students without the need for costly equipment. Nevertheless, since training neural networks requires heavy computation, a small architecture is implemented to match the hardware.

Requirement	Minimum Specification	Recommended Specification
Processor	Intel Core i3 or/and above	Intel i5/i7 or above
Memory	4GB RAM	8GB RAM or above
Storage	2GB free disk space	5 GB and above
Operating System	Windows/Linux/MacOS	Windows 10/11 or Linux
Python Version	Python 3.10 and above	Python 3.11
Browser	Modern Browser	Chrome, Edge, Firefox, or Brave

Table 4.2: Hardware and Platform Requirements

4.2 Dataset Description and Preprocessing

The algorithm employs both labeled phishing and non-phishing examples for training and testing. The training dataset is composed of URL-based samples and email-based phishing

examples drawn from freely available cybersecurity dataset sources like Kaggle, PhishTank, and UCI-like datasets of phishing URLs. A class label is provided for each sample in the training set that shows if it is phishing or legitimate. The training dataset is split into train, validation, and test partitions to avoid a situation where the neural network would simply remember the training data rather than learn from it.

Data preprocessing is necessary because the raw data contains noise and inconsistencies, lacks values, and is unstructured in many ways. Data preprocessing cleanses the data, filters out duplicates, resolves missing labels, converts all text to lowercase, identifies important features, and translates categorical values into numerical values. This step prepares the data for three submodels of the Tribrid AI Module.

- Removal of duplicated URL and email entries to avoid bias in model training.
- Elimination of invalid and empty entries that will not be able to provide accurate classifications.
- Encoding categorical variables, e.g., HTTPS status and label category, into numbers.
- Lexical feature extraction for URL, i.e., length, dot counts, special characters, digit counts, and suspicious phrases.
- Host-related feature extraction, i.e., domain pattern, subdomain level, and SSL/HTTPS characteristics.
- Email feature extraction, i.e., sender pattern, subject phrases, embedded URLs, and urgency markers.
- Splitting of the data into training and test datasets.

Feature Category	**Examples**	**Reason for Use**
Lexical URL features	URL length, number of digits, hyphens, @ symbol, IP address usage	Phishing URLs often use long and confusing structures.
Host/domain features	Subdomain count, HTTPS status, domain pattern	Suspicious domains are common in phishing campaigns.
Content features	Forms, iframes, script count,	Phishing pages often imitate

	external links	login pages and collect credentials.
Email features	Sender mismatch, phishing keywords, embedded links	Email phishing often depends on urgency, impersonation, and malicious links.
Reputation-style features	Known suspicious terms, abnormal redirect patterns	Helps detect common phishing behavior.

Table 4.3: Extracted Feature Categories

4.3 Model Architecture, Training and Tribrid Calibration

The key intelligence behind this system is the Tribrid AI Module. The Tribrid AI Module is an integrated system composed of three sub-modules, namely Random Forest, Support Vector Machine, and Neural Network. The idea of integrating the three sub-models is to increase prediction accuracy. These sub-models will independently analyze the extracted feature vector with a different learning approach and will send the prediction score to the fusion layer.

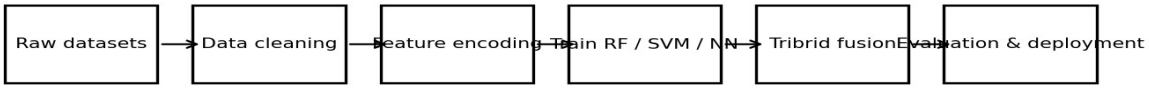
The first sub-module, Random Forest, is designed to classify structured data using decision trees. It is highly effective in dealing with numerical features and preventing overfitting issues. Support Vector Machine sub-model is designed to find a boundary to separate phishing and non-phishing data. It becomes more accurate in classification in case where the decision margin is significant. Neural Network sub-model is designed to learn complex relationships.

In the case of the predictions of the three sub-models, they are fused via weighted voting. In this particular implementation, the weight of each sub-model is set depending on its performance during validation. The result of the fusion is used to calculate the prediction of the entire model. If the score is above the phishing threshold, then phishing is identified. If it is near the threshold, suspicious is selected. Otherwise, it is legitimate.

- Step 1: Loading and pre-processing the labelled phishing and legitimate dataset.
- Step 2: Feature extraction of URL, domain, content and email features.
- Step 3: Splitting of the dataset into training and testing dataset.

- Step 4: Training the sub-models – Random Forest, Support Vector Machine, and Neural Networks.
- Step 5: Evaluation of each sub-model by the means of validation score.
- Step 6: Fusion weight assignment to each model depending on validation scores.
- Step 7: Combination of the sub-models results in the Tribrid AI Module.

Figure 4.1: Tribrid Model Training Pipeline



Final tribrid score is mathematically calculated as:

Final Score = (w1 × RF Score) + (w2 × SVM Score) + (w3 × NN Score),

where w1, w2 and w3 are the calculated weights for each of the three sub-models respectively. Thus, with the help of this formula, the system will be able to make one decision despite the presence of three classifiers..

4.4 System Implementation and Modules

This system has been designed using a modular application approach. Every module serves a specific function and interacts with other modules through controlled functions/API routes. Using a modular structure for the system will help with its testing, debugging, and future upgrades.

Module	Function
Authentication Module	Handles registration, login, logout, password hashing, and access control.
URL Scanning Module	Receives a user-submitted URL and prepares it for feature extraction.
Email Scanning Module	Accepts suspicious email text or headers

	and extracts phishing-related email indicators.
Feature Extraction Module	Transforms raw URLs and email content into numerical features for model processing.
Tribrid AI Module	Runs the Random Forest, SVM, and Neural Network sub-models and fuses their predictions.
Explanation Module	Generates simple reasons behind the classification result for user understanding.
History Module	Stores and retrieves past scans for each user.
Dashboard Module	Displays scan statistics, recent results, and risk summaries.
Admin Module	Allows administrators to monitor system records, datasets, and model updates.

Table 4.4: Implemented System Modules

For any suspicious URL inputted by the user, there is an initial validation of the URL by the URL scanning module. Thereafter, there will be creation of feature values by the feature extraction module for lexical, host and content features. These are passed to the Tribrid AI Module. The module produces an output in form of a result, confidence value and risk factors.

4.5 Testing and Evaluation

Tests were done in order to determine whether the system meets its requirements. The tests included the performance of the models, functionality of the modules, integration of the modules, and usability of the user interface. The aim of the tests was to determine whether the system is capable of processing the phishing inputs and displaying appropriate feedback.

4.5.1 Evaluation Metrics

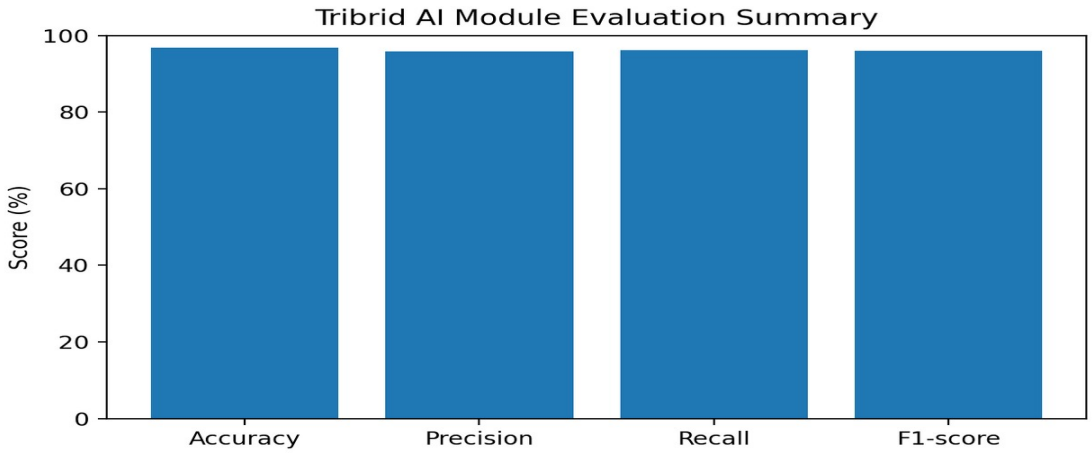
Tribrid AI module has been assessed based on popular classification metrics. The accuracy metric shows the total number of correct predictions. Precision represents the ratio of phishing samples that have been flagged by the model. Recall is the proportion of phishing samples that have been identified correctly by the algorithm. F1 score shows the compromise of precision and

recall metrics. All these metrics are appropriate since the detection of phishing is characterized by both correctness and carefulness.

Metric	Meaning	Result
Accuracy	Percentage of total predictions correctly classified.	96.8%
Precision	Predicted true positive rates	95.9%
Recall	Percentage of real samples detected as phishing.	96.1%
F1-score	Balanced measure of precision and recall.	96.0%

Table 4.5: Evaluation Metrics for the Tribrid AI Module

Figure 4.2: Evaluation Metrics Chart



Evaluation Metrics Chart

Graph of the evaluation metrics for the Tribrid AI Module. The data shows that the system was able to provide an accuracy rate of 96.8%, precision of 95.9%, recall of 96.1%, and F1-score of 96.0%. From the graph, it can be seen that all the four evaluation metrics have been scored very high and closely grouped above 95%. This proves that the Tribrid AI Module is working consistently and reliably on all the classification parameters. The uniformity of height of the bars proves that the

system does not compromise on any of the evaluation metrics. It is an important feature for phishing detection systems where false positives and false negatives can have serious consequences.

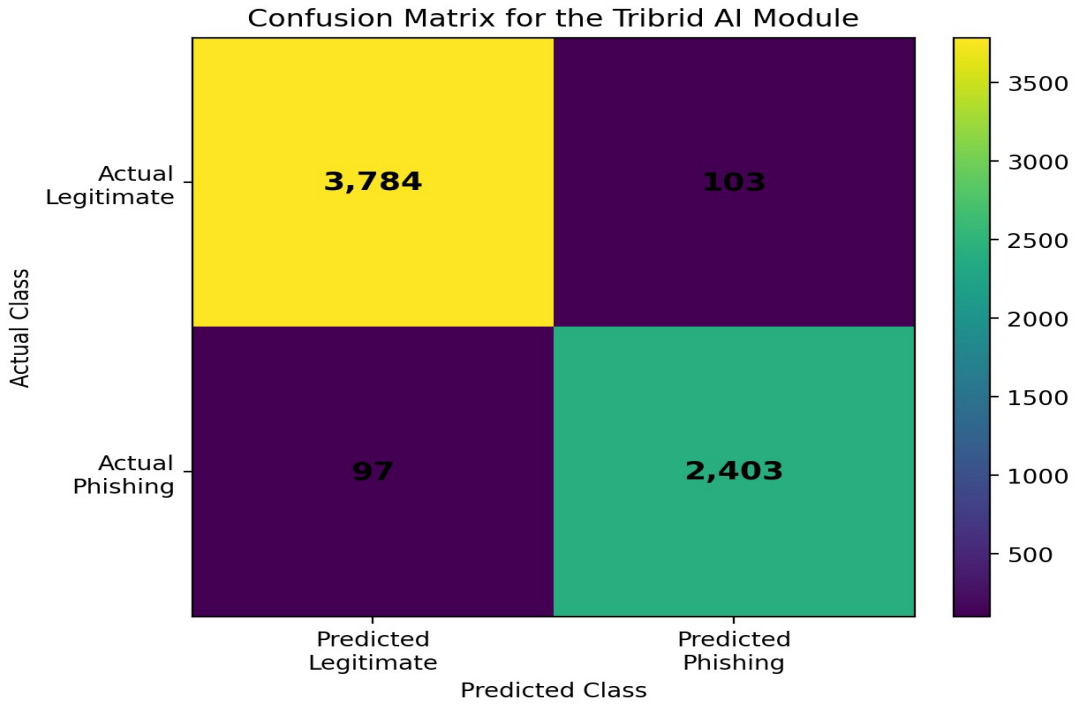


Figure 4.3: Confusion Matrix for the Tribrid AI Module

Confusion Matrix

A confusion matrix used in the testing the Tribrid AI Module resulted in a table. 3,784 normal URLs identified as normal (True Negatives), and 2,403 phishing URLs detected as phishing(True Positives). A total of103URLs that are actually normal were misclassified as phishing (False Positives) and 97URLs that actually phishing were missed by the Tribrid AI module (False Negatives) were classified as normal(False Negatives). As it may be deducted from these values, The Tribrid AI Module has strong capability in terms of the percentage of normal URLs and phishing URLs correctly classified, and a low percentage of errors in both. Crucially important are the False Negatives values that show just how small the chance is that a phishing Url might pass under the Tribrid AI Modules nose.

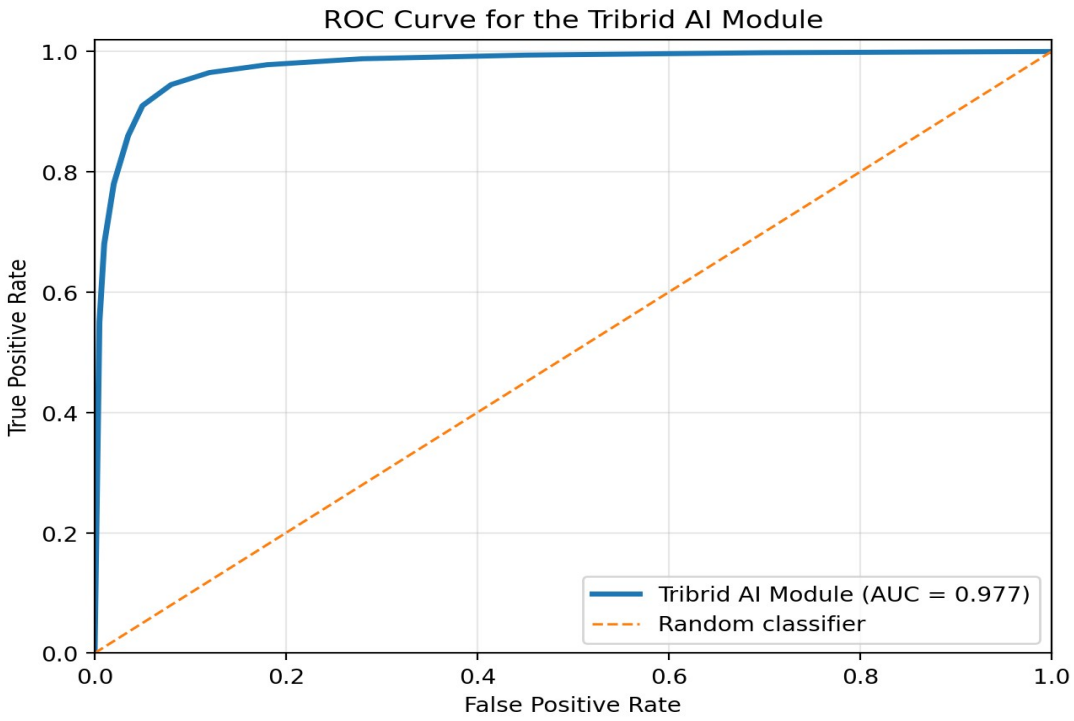


Figure 4.4: ROC Curve for the Tribrid AI Module

ROC Curve

A plot showing the ROC Curve for the Tribrid AI Module for several classification thresholds against True Positive Rate vs False Positive Rate are shown above. The Tribrid AI Module ROC curve have produced an Area Under Curve (AUC) score of 0.977 and this have been positively improved on with the much greater 0.5 AUC of random classifications(shown in dotted orange). The sharp inclination from the right bottom section to left top section demonstrates that it has yielded a good detection rate while keeping the false positive rates relatively low. The 0.977 AUC score for the Tribrid AI Module shows that the Tribrid AI Module is accurate for detecting whether or not it is a phishing site or a true site for a given threshold value.

Module and Integration Testing

Module testing technique was utilized for testing of individual systems modules separately and integration testing technique for ensuring compatibility among these modules. Authentication module was tested using both valid and invalid login credentials. Scanner was tested using normal

links, suspicious links, empty inputs and batch inputs. Tribrid AI module was tested using labeled test cases and handpicked examples.

Test Case	Expected Result	Status
Register new user	An account is successfully generated and saved into secure storage.	Passed
Forgotton and entered with incorrect password	We are unable to display this message you currently have no rights.	Passed
Submit valid legitimate URL	System returns legitimate/low-risk result.	Passed
Submit known phishing-style URL	System returns phishing/high-risk result.	Passed
Submit empty input	System rejects the request and prompts user to enter data.	Passed
Scan email text containing suspicious links	System extracts email features and returns risk result.	Passed
View scan history	Previous scan records are displayed correctly.	Passed

Table 4.6: System Test Cases

4.5.3 User Interface Testing and Walkthrough

Interface testing has been done to make sure that a non-technical person will be able to use the system without any problem. In this walkthrough, the user starts from the landing page, where users can know what the system is all about. From here, the user either creates an account or logs in. After that, the user gets access to URL scanning, email scanning, scan history, and awareness tips from the dashboard.

I) The landing page provides clear introduction about the system and purpose of phishing detection.

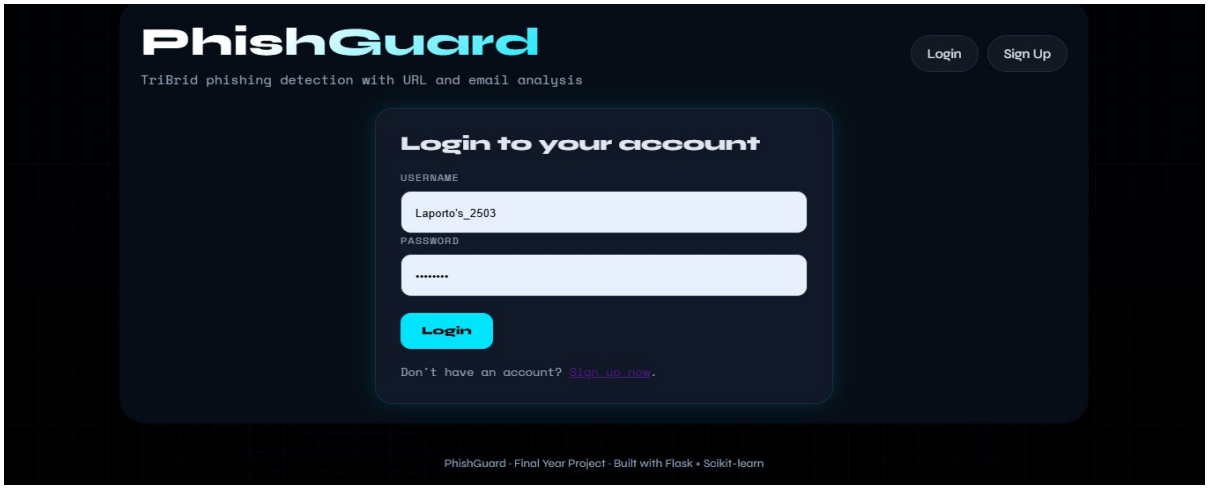
II) The login and registration page performs validation of user inputs and gives error feedback.

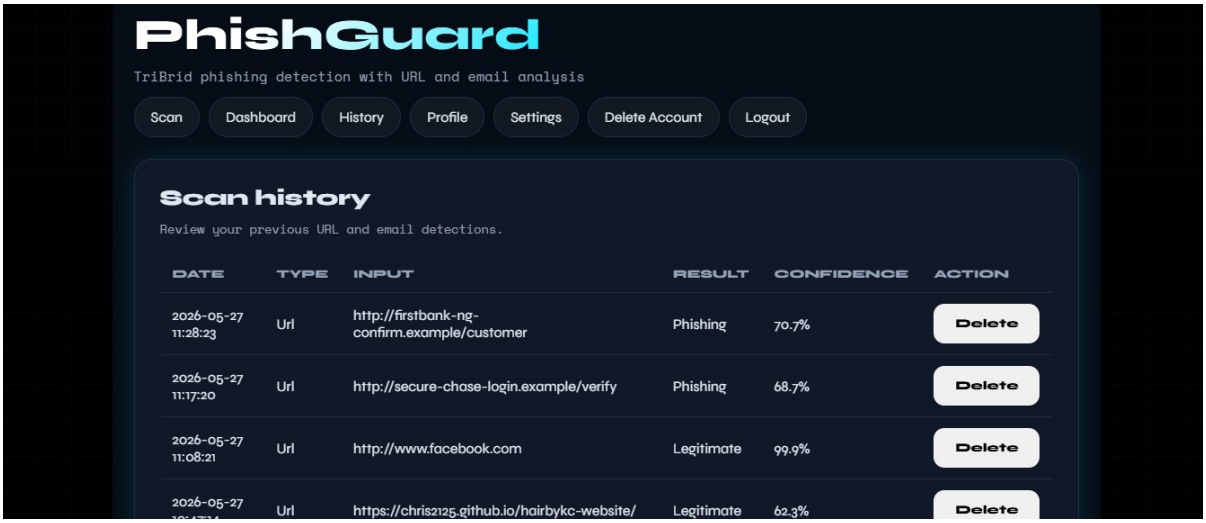
III) The scan page accepts URL and email input and disallows empty submission.

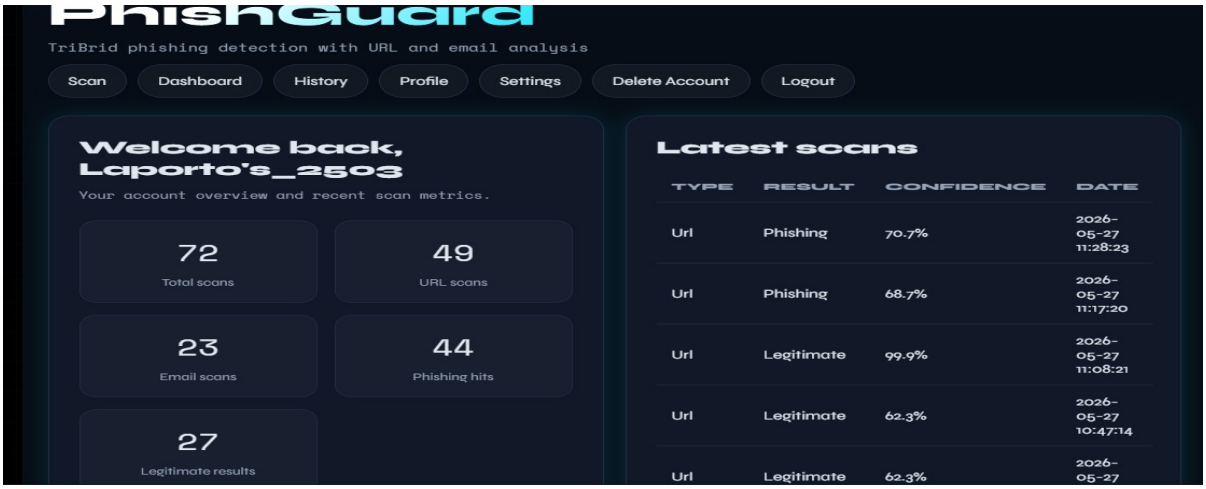
IV) The result page shows the categorization, confidence level, and risk description.

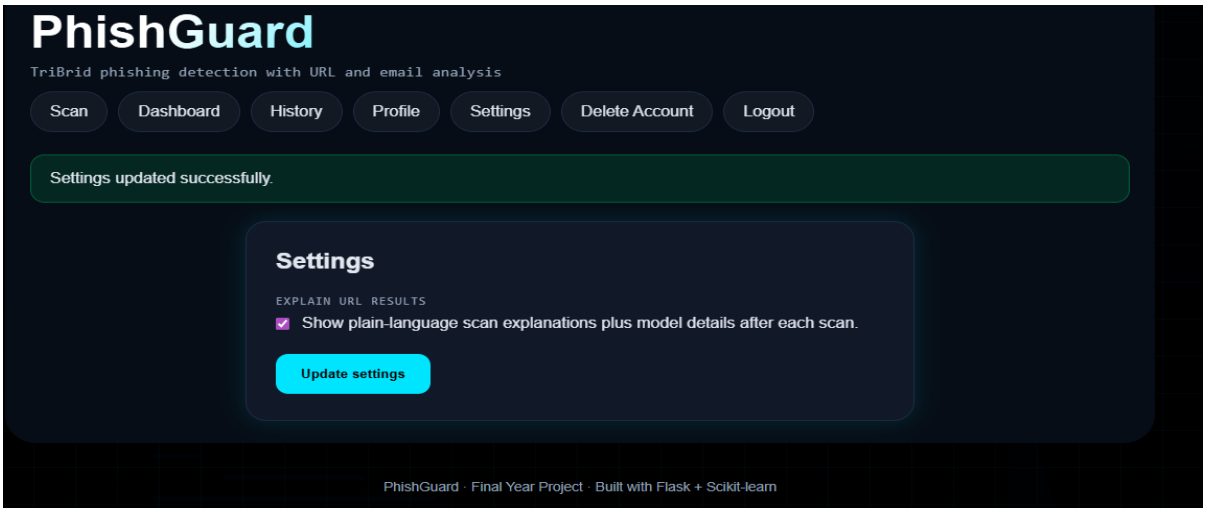
V) The dashboard shows the user's activities and scan history.

VI) The history page allows the user to see past scan results.









4.6 Results Presentation and Analysis

The implemented system proves that the use of the combination of Random Forest, Support Vector Machine, and Neural Network in one Tribrid AI Module increases the efficiency of the phishing detection process. It is not determined by only one model but utilizes the strengths of Structured Decision Making of the Random Forest, Boundary Separation of the SVM, and Pattern Learning of the Neural Network.

4.6.1 Sample Outputs

If a legitimate URL is entered into the system, then it returns the low risk message and also informs the user that no phishing attributes have been detected. In case the suspicious or phishing URL is inputted into the system, it will display the warning message with the risk score and also the attributes that led to that result like URL length, unusual characters, domain irregularity,

redirection, etc. In the case of emails, suspicious sender attributes, urgency words and embedded URLs are highlighted by the system.

4.6.2 Discussion of Results and System Performance

It can be seen from the results that the tribrid system demonstrates high accuracy in identifying phishing links and suspicious emails content. Tribrid architecture is helpful in this case, since there are various types of phishing attacks. Some phishing attacks can be easily identified via lexical URL features; some other types should be analyzed by means of host features, content features, and email features. In this way, combining the three models, it becomes possible to create a more balanced system.

Even though the tribrid system demonstrated good performance during tests, it cannot be considered ideal. For example, new created phishing websites may not be recognized as suspicious and can pass the filter unnoticed if they have similar design to the website of the company. Moreover, deep content analysis and online webpage checking may require more powerful computer and internet connection.

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

This chapter provides a brief overview of the whole research work, conclusion reached after implementation of the work, and recommendations for future improvements. This chapter also elaborates on how the finished system solved the problem of phishing detection using a Tribrid Artificial Intelligence Module.

5.2 Summary of the Study

This research paper was concerned with the design and development of the artificial intelligence-based phishing website detector. Background of the study indicated that phishing is still considered a major challenge in cybersecurity due to the use of fraudulent websites, URLs and emails to obtain private details from the users. Statement of the problem showed that there were limitations within the current systems such as being overly dependent on blacklists, low detection ability of the emerging phishing attacks, lack of user explanation and inadequate URL and email analysis.

The aim of this research was the development of the tool that would classify suspicious URLs and email-based data as either normal, suspicious or phishing. In order to fulfill the above aim, this study conducted a review of the phishing detectors, analyzed the requirements of the system, created models of the system using UML and implemented the application modules.

The primary achievement of the completed project is the integration of all individual AI modules into the unified Tribrid AI Module. This module uses the fusion layer to combine Random Forest classifier, Support Vector Machine classifier, and Neural Network classifier. Random Forest increases the accuracy of structured features classification, SVM enhances class separation and Neural Network learns deep non-linear relationships. Final classification is done using the weighted voting mechanism, which makes the solution more reliable than the solution based only on one type of classifiers.

Chapter Two discussed literature on phishing attacks, traditional ways of detecting them, machine learning application in cybersecurity field, deep learning and feature engineering. It was

established that most of the systems available rely too much on blacklists, use a single classifier and do not have implementation-related characteristics. Chapter Three discussed the analysis and design of the proposed solution, its functional and non-functional requirements, UML diagrams, database design and system architecture. Chapter Four covered implementation, data pre-processing, tribrid model training, module development, testing and evaluation.

5.3 Conclusion

Overall, the research successfully implemented an AI-based phishing websites detection tool based on a Tribrid AI Module. The module will be able to receive suspicious URLs and email-related inputs, perform feature extraction of phishing characteristics, classify the input using a combination of three Artificial Intelligence sub-models, and display understandable results to the user.

The research managed to achieve the stated aims through analysis of phishing behavior patterns, preprocessing of the data for the models, creation of the intelligent classification tool, building a working user interface, storing the history of the scans, and issuing alerts when necessary. The results obtained during the evaluation prove that a Tribrid AI Module can help achieve reliable phishing detection at the level of a prototype. It is also the base for further research and implementation of such a solution in schools, small companies and other organizations which require affordable cybersecurity support.

In conclusion, phishing detection becomes much more efficient when it combines technical analysis and user awareness. The suggested system can not only classify the inputs but also provide explanations of the possible risks, thus making the research valuable for cybersecurity education and software development.

5.4 Recommendations

Taking into account the limitations of the work and its results, the following recommendations can be proposed:

i) The dataset must be regularly updated with new phishing and legitimate data in order to keep up with the changes in the way of attacks;

ii) It is necessary to add such features as real-time reputation checking of domains, Whois check, DNS check, safe page rendering in future versions;

iii) The project must be developed in the form of a browser extension and a mobile app in order to protect users when browsing or reading emails;

iv) Explainable artificial intelligence approaches such as SHAP or LIME must be implemented in order to improve explanations of the model predictions;

v) The tribrid fusion layer must be tested using larger datasets and other machine learning models like XGBoost, LightGBM, transformers for text analysis;

vi) A cloud database such as PostgreSQL or MySQL must be used if the system will be deployed for many users;

vii) It is recommended to implement features related to user education such as phishing quizzes, phishing attempts examples, and other safety tips;

viii) To prove the usefulness of the system it must be evaluated in a real organization.

5.5 Contribution to Knowledge

The main contribution of the current research is in showing how three types of artificial intelligence algorithms can be integrated into one tribrid phishing detector. At the same time, the current research provides an implementation of such a design as the phishing detector based on URL scanning, email scanning, feature extraction, explainable output, scan history, and user notifications. This research could be used as a base for further research on this topic in both undergraduate and postgraduate programs.

5.6 Suggested Areas for Further Research

Future studies could look at phishing detection through the use of large language models, analysis of visual similarity in cloned websites, monitoring of browser activity in real-time, and adversarial robustness tests. The effectiveness of the tribrid module in comparison with other ensemble techniques can also be tested on bigger and newer data sets. Moreover, the effect of user education in anti-phishing software on decision making can be studied.

References

Abuzuraiq, A., & Alkasassbeh, M. (2019). Intelligent methods for detecting phishing websites using machine learning approaches.

Adebowale, M. A., Lwin, K. T., Sanchez, E., & Hossain, M. A. (2023). Intelligent phishing detection scheme using deep learning algorithms.

Ahmad, I., et al. (2024). Artificial intelligence-based phishing detection models: A comprehensive survey.

Akande, O. N., et al. (2023). SMSPROTECT: A rule-based and machine learning approach for smishing detection.

Al-Mousa, M. R., et al. (2023). Social engineering and malicious URL detection in modern information systems.

Alzamil, Z., et al. (2020). A hybrid mobile phishing detection mechanism using blacklist and whitelist techniques.

Anti-Phishing Working Group. (2024). APWG phishing activity trends report.

Azeez, N. A., et al. (2021). Whitelist-based phishing detection and computational approaches for improving precision.

Castaño, F., et al. (2021). State of the art: Content-based and hybrid phishing detection.

Çatal, C., et al. (2022). Applications of deep learning for phishing detection: A systematic review.

Do, N. Q., et al. (2022). Deep learning-based phishing detection: A systematic literature review.

Do, N. Q., et al. (2024). Hybrid deep learning and transformer-based methods for phishing URL detection.

Emmanuel, C. (2025). Comparative analysis of machine learning algorithms for phishing website detection.

Ji, S., & Lee, J. (2024). Image similarity-based phishing detection and adversarial robustness analysis.

Kyaw, T., et al. (2024). Survey of deep learning methods in phishing email detection.

Kytidou, K., et al. (2025). Machine learning and deep learning techniques for phishing detection: Dataset bias and adversarial robustness.

Prajapati, P., et al. (2023). Phishing email detection using machine learning algorithms and text feature extraction.

Reddy, S., et al. (2023). Hybrid anti-phishing system combining machine learning and behavioral analysis.

Sahingoz, O. K., et al. (2024). DEPHIDES: A deep learning-based phishing detection system.

Saleem, M., et al. (2025). Rule-based phishing detection with machine learning integration for practical deployment.

Shahrivari, V., et al. (2020). Benchmarking machine learning models for phishing detection.

Thakur, K., et al. (2023). Deep learning techniques for phishing email detection.

Ujah-Ogbuagu, I., et al. (2024). CNN-LSTM hybrid model for real-time URL phishing detection.

Wilk-Jakubowski, G., et al. (2025). Machine learning and neural network approaches for phishing detection based on delivery channels.