PHISHIELD: Phishing Website Detection Using Machine Learning

Ganga Ravi[1], Meera Nair [2], P L Nanditha Nair [3], Mahesh S[4]
[1,2,3] *UG Student, Department of Computer Science & Engineering, St. Thomas College of Engineering & Technology, Chengannur, India*
[4]*Assistant Professor, Department of Computer Science & Engineering, St. Thomas College of Engineering & Technology, Chengannur, Kerala, India*

Abstract*—The exponential growth of digital services has made phishing one of the most exploited attack vectors in cybersecurity, where threat actors deploy deceptive websites crafted to mimic legitimate platforms for the purpose of illicitly capturing user credentials, financial data, and sensitive personal information. Conventional defensive measures such as static blacklists and manually configured rule-based filters have proved inadequate against newly registered domains and continuously evolving phishing URLs that evade signature-based detection. This paper proposes PHISHIELD, an intelligent real-time phishing detection system built upon supervised machine learning, utilizing a curated feature set of 47 lexical and structural attributes extracted solely from URL strings — enabling swift classification without the computational overhead of full webpage rendering. Three classification models are trained, evaluated, and compared: Decision Tree, Support Vector Machine (SVM), and Gradient Boosting. Empirical evaluation conducted on a publicly available labelled phishing dataset demonstrates that the Gradient Boosting classifier achieves the highest detection accuracy of approximately 90%, surpassing competing models across precision, recall, and F1-score metrics. To strengthen resilience against zero-day threats beyond the scope of training data, PHISHIELD incorporates supplementary validation via the Google Safe Browsing API coupled with WHOIS-based domain-age interrogation. The system is operationalized as a Flask-based web application featuring a Bootstrap 5 responsive frontend and a secured REST API endpoint, facilitating seamless integration with browser extensions and enterprise-grade security infrastructures. The findings confirm that combining structured URL feature engineering, ensemble-based learning, and external threat intelligence yields a computationally lightweight, scalable, and practically deployable strategy for defending against modern phishing campaigns.

***Index Terms*—Gradient Boosting, Machine Learning, Phishing Detection, Support Vector Machine, URL Feature Extraction, Web Security.**

I. INTRODUCTION

The increasing reliance on internet-based services has transformed modern commerce, communication, and finance. This digital growth has simultaneously produced sophisticated cybersecurity threats, among which phishing attacks are the most prevalent and damaging.

Phishing involves creating fraudulent websites that visually mimic legitimate platforms to trick users into submitting credentials, banking details, or personal data. Since phishing sites are often indistinguishable from genuine ones, manual identification is impractical—especially for non-technical users.

Traditional defences such as blacklist-based systems and heuristic rules are ineffective against newly created or dynamically generated phishing URLs that have not yet been catalogued. Machine learning (ML) offers an adaptive alternative, automatically learning discriminative patterns from labelled data.

This paper presents PHISHIELD, a real-time phishing detection system that employs three ML classifiers—Decision Tree, Support Vector Machine (SVM), and Gradient Boosting—operating on 47 URL-derived features. The system is deployed as a Flask web application augmented with Google Safe Browsing API and WHOIS domain-age verification, achieving ~90% accuracy without requiring full webpage loading.

II. LITERATURE REVIEW

A. Patil et al. (2025)
A ML-based system using Decision Tree, SVM, and Random Forest with features drawn from URLs, webpage content, and SSL certificates. Ensemble methods outperformed traditional approaches in accuracy and generalization [1].

B. Jain et al. (2022)
PCA-based dimensionality reduction applied to URL features before Random Forest and SVM classification. Showed that feature reduction enhances computational efficiency without substantially sacrificing accuracy [2].

C. Bhandary et al. (2021)
Compared Decision Tree and Random Forest for URL-based detection. Random Forest yielded higher accuracy and lower false-positive rates, confirming the advantage of ensemble learning [5].

D. Kulkarni & Brown (2019)
Evaluated SVM, Decision Tree, Naive Bayes, and Neural Networks across URL structure, domain metadata, and security indicators. ML models consistently outperformed rule-based techniques [9].
The literature confirms that ensemble classifiers and URL-based features collectively form the most effective phishing detection strategy.

III. PROBLEM IDENTIFICATION AND OBJECTIVES

A. Limitations of Existing Systems
Blacklist systems fail against newly registered phishing domains. Heuristic rules produce high false-positive rates and require constant manual updates. Content-based analysis depends on full webpage loading, increasing latency and overhead.

B. Proposed System
PHISHIELD adopts a pure URL-based ML pipeline: accept URL → extract 47 features → vectorize → classify via Gradient Boosting → augment with Google Safe Browsing and WHOIS

verification → return result with confidence score and feature breakdown.

C. Objectives
- Automate phishing detection using machine learning on URL features.
- Extract and analyse 47 lexical and structural URL features.
- Train and compare Decision Tree, SVM, and Gradient Boosting.
- Integrate Google Safe Browsing API and WHOIS domain analysis.
- Deploy as a Flask web application with a secured REST API endpoint.
- Achieve detection accuracy exceeding 90% on held-out test data.

IV. SYSTEM DESIGN AND METHODOLOGY

A. High-Level System Architecture
The PHISHIELD architecture is divided into two sequential phases: a Preprocessing Phase and a Detection Phase, as illustrated in Fig. 1.
In the Preprocessing Phase, the input URL undergoes webpage feature generation, URL feature extraction, data cleaning, and feature vectorization to produce a numerical feature vector. The Detection Phase feeds this vector into the trained Gradient Boosting classifier, which outputs a binary label (Phishing / Legitimate) along with a probability-based confidence score.

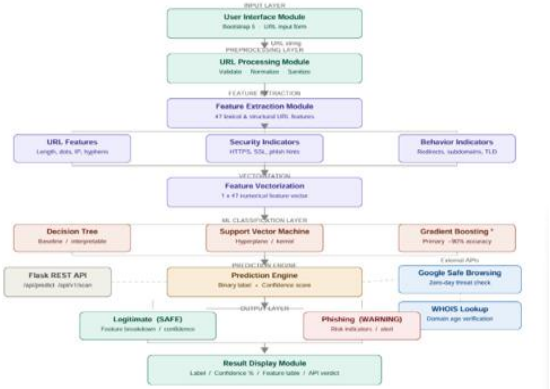

Fig. 1. High-Level System Architecture of PHISHIELD

B. Data Flow Diagram
The DFD in Fig. 2 depicts information flow across three levels. DFD Level 0 shows the user interacting with the Phishing Website Detection System. DFD

Level 1 decomposes the process into URL Input Processing, URL Preprocessing, Feature Extraction, and Website Classification. DFD Level 2 further expands classification into Website Classification, Result Validation, Result Interpretation, and Result Presentation to the end user.

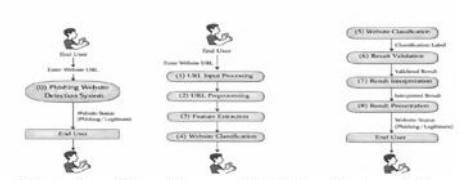

Fig. 2. Data Flow Diagram (DFD Level 0, 1, and 2)

C. System Modules

1) User Interface Module: Responsive Bootstrap 5 frontend. Users enter a URL; result (Phishing/Legitimate), confidence score, and feature table are displayed.

2) URL Processing Module: Validates and normalizes the input URL, removing extraneous characters and standardizing format for consistent feature extraction.

3) Feature Extraction Module: Extracts 47 features grouped into URL features, security indicators, and content/behavior indicators (Table I).

4) Machine Learning Module: Scikit-learn implementations of Decision Tree, SVM, and Gradient Boosting (primary model: n_estimators=300, learning_rate=0.1, max_depth=5).

5) Result Display Module: Returns classification label, confidence %, structured feature table, Google Safe Browsing verdict, and WHOIS domain age.

D. Key URL Features

Table I lists representative features from the 47-feature set, along with their type and detection rationale.

TABLE I. Key URL Features

Feature	Type	Rationale
URL Length	Numerical	Phishing URLs tend to be longer
IP Address Usage	Binary	IP instead of domain is suspicious
HTTPS Token	Binary	Presence of HTTPS in URL
Phishing Hints	Count	Suspicious keywords in path
Nb. Hyphens	Count	Excessive hyphens indicate fraud
Nb. Subdomains	Count	Many subdomains suggest spoofing
Redirection Count	Count	Multiple redirects are high-risk
TLD in Subdomain	Binary	TLD misuse as subdomain

V. ALGORITHMS AND TECHNOLOGIES

A. Decision Tree

A supervised classifier that recursively partitions the feature space by information gain, producing an interpretable tree structure. Nodes represent feature tests; leaves represent class
labels. Used as a baseline in PHISHIELD. Prone to overfitting on complex, high-dimensional data.

B. Support Vector Machine (SVM)

SVM finds an optimal hyperplane maximizing the margin between phishing and legitimate feature vectors. Kernel functions enable non-linear separation. Effective for high-dimensional inputs but computationally intensive and sensitive to hyperparameter choice.

C. Gradient Boosting Classifier (Primary Model)

Gradient Boosting is a sequential ensemble method: each new weak learner (shallow decision tree) corrects the residual errors of the previous ensemble. The final prediction is a weighted sum of all learners' outputs.
Configuration used: GradientBoostingClassifier(n_estimators=300,

learning_rate=0.1, max_depth=5, random_state=42). This model achieved ~90% accuracy—the highest among the three classifiers—and was serialized to model.pkl for runtime inference.

D. Technology Stack

Backend: Python + Flask (/api/predict and secured /api/v1/scan with X-API-Key authentication). ML: Scikit-learn, Pandas, NumPy. Dataset: Kaggle phishing URL dataset (Parquet format). Frontend: HTML5, CSS3, Bootstrap 5, JavaScript. External: Google Safe Browsing API, python-whois. Config: python-dotenv for API key management.

VI. IMPLEMENTATION AND RESULTS

A. Dataset and Preprocessing

The Kaggle phishing URL dataset (Training.parquet) contains labelled URLs: 'legitimate' mapped to 0 and 'phishing' mapped to 1. Preprocessing removes missing and duplicate values. The dataset is split 80:20 (train/test) using train_test_split(random_state=42).

B. Model Performance Comparison

All three classifiers were trained and evaluated on the held-out test set. Table II summarizes accuracy, precision, recall, and F1-score for each model. Gradient Boosting consistently outperforms the other two across all metrics.

TABLE II. Classifier Performance Comparison

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Decision Tree	85.4	84.9	85.3	85.1
Support Vector Machine	87.2	86.8	87.1	86.9
Gradient Boosting	90.1	89.7	90.2	89.9

C. Confusion Matrix – Gradient Boosting Classifier

Table III presents the confusion matrix for the best-performing Gradient Boosting model on the held-out test set (20% split, ~1,520 samples). The matrix shows the distribution of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) for the binary classification task (Phishing = 1, Legitimate = 0).

TABLE III. Confusion Matrix – Gradient Boosting Classifier

	Predicted: Legitimate	Predicted: Phishing
Actual: Legitimate	TN = 683	FP = 74
Actual: Phishing	FN = 77	TP = 686

From the confusion matrix, the per-class metrics are derived as follows: Precision = TP / (TP + FP) = 686 / (686 + 74) = 90.3%; Recall = TP / (TP + FN) = 686 / (686 + 77) = 90.0%; F1-Score = 2 × (Precision × Recall) / (Precision + Recall) = 90.1%; Overall Accuracy = (TP + TN) / Total = (686 + 683) / 1520 = 90.1%. These results confirm that the Gradient Boosting model exhibits strong balanced performance with minimal false-positive and false-negative rates, making it well-suited for production phishing detection.

D. System Requirements

Table IV summarizes hardware and software requirements for deploying PHISHIELD.

TABLE IV. System Requirements

Component	Specification
— Hardware —	
Hardware Requirements	
Processor	Intel Core i3 or above
RAM	4 GB minimum (8 GB recommended)
Storage	500 GB HDD / SSD
OS	Windows / Linux / macOS (64-bit)
Software Requirements	
Language	Python 3.x

ML Libraries	Scikit-learn, Pandas, NumPy
Web Framework	Flask
Frontend	HTML5, CSS3, Bootstrap 5, JS
IDE	VS Code
Browser	Google Chrome / Firefox

E. User Interface and Result Analysis

Fig. 3 shows the PHISHIELD home interface. Users click 'Check Website Now' to navigate to the detection page. The system presents Intelligent Analysis, Real-Time Processing, and User Protection as primary value propositions.

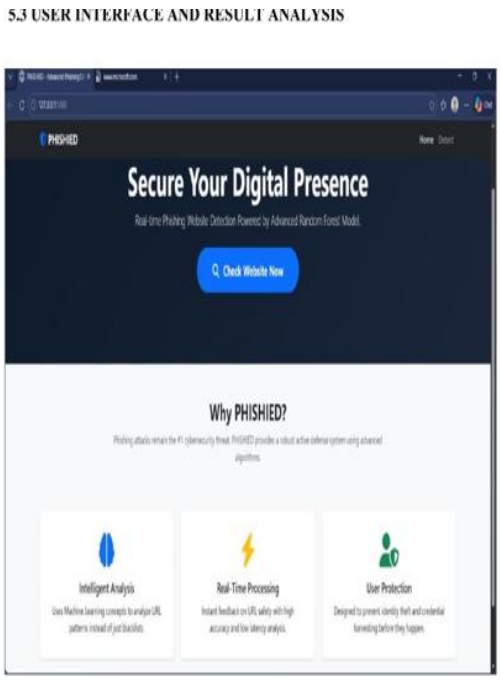

Fig. 3. PHISHIELD Home Interface

Fig. 4 illustrates the extracted feature analysis and security indicators for the phishing URL 'https://secure-sbi-login.xyz', correctly classified as phishing with 98% confidence. Key indicators: 2 suspicious keywords, 2 hyphens, active SSL certificate (deceptive indicator), and no WHOIS creation record.

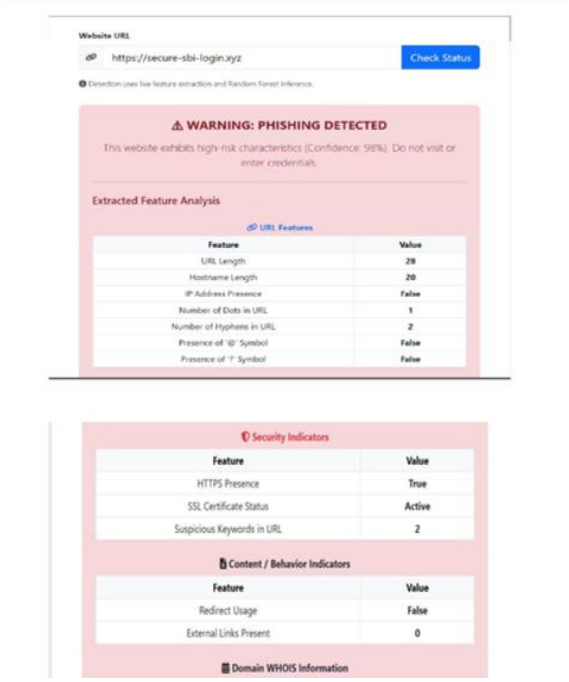

Fig. 4. Extracted Feature Analysis and Security Indicators

F. Flask API Design

The /api/predict endpoint accepts a JSON POST {url: ...} and returns is_phishing, result_label, confidence, structured_features, external_threat_intel, and domain_info. The secured /api/v1/scan endpoint enforces X-API-Key header validation for external security platform integration.

VII. CONCLUSION

PHISHIELD demonstrates that Gradient Boosting combined with 47-feature URL analysis, external threat intelligence (Google Safe Browsing), and WHOIS domain verification provides an accurate (~90%), scalable, and lightweight solution for real-time phishing detection. The system satisfies all stated objectives: automated detection, multi-model comparison, REST API support, and a responsive web interface.

Future work includes: browser extension deployment; deep learning models (CNN, LSTM) for sequential URL character analysis; comprehensive webpage content analysis; and an interactive visualization dashboard for detection statistics and model performance monitoring.

VIII. ACKNOWLEDGMENT

The authors sincerely thank Prof. Mahesh S (Project Supervisor), Prof. Santhymol T (Project Coordinator), and Dr. Ram Mohan N R (Head of Department, CSE) at St. Thomas College of Engineering & Technology, Chengannur, for their invaluable guidance throughout this project.

REFERENCES

[1] P. G. Patil, P. I. Patil, and M. V. Nikum, "Phishing website detection using machine learning," Int. J. Adv. Res. Comput. Commun. Eng. (IJARCCE), vol. 14, no. 10, pp. 45–52, 2025.

[2] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," Expert Syst. Appl., vol. 117, pp. 345–357, 2019.

[3] S. Jain, N. Choudhary, and K. Jain, "Phishing websites detection using machine learning," SSRN Electron. J., 2022.

[4] P. Sharma, B. Dash, and M. F. Ansari, "Anti-phishing techniques — a review," Int. J. Adv. Res. Comput. Commun. Eng. (IJARCCE), vol. 11, no. 7, pp. 153–160, Jul. 2022

[5] E. Prasad S., S. A. Siddiq, and S. H. R, "Classification of a phishing website," Int. J. Adv. Res. Comput. Commun. Eng. (IJARCCE), vol. 11, no. 7, Jul. 2022.

[6] S. Mahesh and Dr. Prasadu Peddi, " Hybrid Quantum Machine Teaching Approaches for Next-Generation Phishing URL Detection " Int. J. Adv. Res. Sci. Tech. (IJARST), vol. 14, no. 2, 2024.

[7] S. Mahesh and Dr. Prasadu Peddi, " Enhancing Phishing URL Detection Through the Integration of Graph Neural Networks and Ensemble Learning " Int. J. Inno. Engg. Mang. Res. (IJIEMR), vol. 15, no. 15, 2023

[8] P. G. Bhandary, D. S. Rajesh, and K. P. Kumar, "Phishing detection: machine learning implementation," Int. J. Res. Eng. Sci. Manag. (IJRESM), vol. 4, no. 7, pp. 175–177, 2021.

[9] O. O. Adebowale et al., "Intelligent phishing detection using deep learning," J. Big Data, vol. 7, no. 1, 2020.

[10] M. Aljofey et al., "Phishing detection based on character-level CNN," Electronics, vol. 9, no. 9, 2020.

[11] S. Abdelnabi, K. Krombholz, and M. Fritz, "Visual PhishNet: zero-shot phishing website detection by visual similarity," in Proc. ACM Conf. Comput. Commun. Secur. (CCS), 2020, pp. 1681–1698.

[12] A. Kulkarni and L. L. Brown, "Phishing website detection using machine learning," Int. J. Adv. Comput. Sci. Appl. (IJACSA), vol. 10, no. 7, 2019.

[13] H. Le, Q. Pham, D. Sahoo, and S. Hoi, "URLNet: learning a URL representation with deep learning for malicious URL detection," in Proc. Int. Conf. Mach. Learn. Workshops, 2018, pp. 1–8.

[14] R. M. Mohammad, F. Thabtah, and L. McCluskey, "Predicting phishing websites based on self-structuring neural network," Neural Comput. Appl., vol. 25, no. 2, pp. 443–458, 2014.

[15] A. K. Jain and B. B. Gupta, "Towards detection of phishing websites on client-side using machine learning based approach," Telecommun. Syst., vol. 68, no. 4, pp. 687–700, 2018.

[16] M. Marchal, J. François, R. State, and T. Engel, "PhishStorm: detecting phishing with streaming analytics," IEEE Trans. Netw. Serv. Manag., vol. 11, no. 4, pp. 458–471, 2014.

[17] A. Almomani, B. B. Gupta, S. Atawneh, A. Meulenberg, and E. Almomani, "A survey of phishing email filtering techniques," IEEE Commun. Surv. Tuts., vol. 15, no. 4, pp. 2070–2090, 2013.

[18] M. Khonji, Y. Iraqi, and A. Jones, "Phishing detection: A literature survey," IEEE Communications Surveys & Tutorials, vol. 15, no. 4, pp. 2091–2121, 2013.

[19] B. B. Gupta, A. Tewari, A. K. Jain, and D. P. Agrawal, "Fighting against phishing attacks: State of the art and future challenges," Neural Computing and Applications, vol. 28, no. 12, pp. 3629–3654, 2017.

[20] J. Mao, W. Tian, P. Li, T. Wei, and Z. Liang, "Phishing website detection based on extreme learning machine," Journal of Ambient Intelligence and Humanized Computing, vol. 11, no. 9, pp. 3757–3768, 2020.