International Journal of Scientific Research in Engineering & Technology
Volume6, Issue6 (November- December 2026), PP: 50-55.
https://www.doi.org/10.59256/ijsreat.20260606008
www.ijsreat.com
ISSN No: 2583-1240

AI-Based Cyber Threat Detection System: A Hybrid Machine Learning and AI Reasoning Approach for Real-Time Malicious URL Detection

Dr. Bhuvaneshwari K V[1], Aditi V Shastry[2], Gazala Anjum B K[3], Jyoti S Hanagandi[4], Srushti Patil BN[5]

[1]*Associate Professor, Department of Information Science and Engineering, Bapuji Institute of Engineering and Technology Davangere, Karnataka, India.*
[2, 3, 4, 5]*Department of Information Science and Engineering, Bapuji Institute of Engineering and Technology Davangere, Karnataka, India.*

To Cite this Article: *Dr. Bhuvaneshwari K V[1], Aditi V Shastry[2], Gazala Anjum B K[3], Jyoti S Hanagandi[4], Srushti Patil BN[5],“ AI-Based Cyber Threat Detection System: A Hybrid Machine Learning and AI Reasoning Approach for Real-Time Malicious URL Detection”, International Journal of Scientific Research in Engineering & Technology, Volume 06, Issue 06, November-December 2026, PP:50-55.*

Copyright: ©2026 This is an open access journal, and articles are distributed under the terms of the Creative Commons Attribution License; Which Permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abstract: *Phishing and malicious URLs remain a common cybersecurity problem because attackers can create new links faster than static blocklists can be updated. This paper presents CyberShield, a real-time URL detection system that combines a calibrated machine-learning ensemble with a separate AI reasoning layer. Random Forest and XGBoost use 11 structural URL features to produce a confidence score. For URLs that are not covered by the trusted-domain whitelist, Google Gemini provides an additional assessment and a plain-English explanation. The system is integrated with a Chrome extension, a FastAPI backend, a React dashboard, email alerts, and a human-reviewed learning process. The evaluated dataset contains 641,119 URLs and is divided into training, calibration, and test subsets. On the held-out test set, the ML ensemble achieved 94.32% accuracy, 91.45% precision, 91.47% recall, a 91.46% F1-score, and a 98.46% AUC-ROC. These metrics describe the ML ensemble, not a separate quantitative evaluation of the complete Gemini-assisted application. Functional tests also checked URL scanning, warning display, and fallback to ML results when the AI service was unavailable. The results show the potential of combining URL-based machine learning with AI explanations, although broader end-to-end testing is needed.*

Keywords: *Phishing detection, malicious URL, Random Forest, XGBoost, confidence calibration, Gemini AI, Chrome extension, continuous learning.*

I. INTRODUCTION

Phishing and malicious URLs are common ways attackers trick users into sharing passwords, financial details, or other sensitive information. Traditional security systems often depend on blocklists and known threat databases, which may not include newly created malicious links [1]. Rule-based filters can also flag unusual but legitimate websites. Machine-learning models can recognize patterns in URLs, but their predictions may be less reliable when they encounter patterns that were not represented in the training data.

This work presents CyberShield, a system that combines machine learning with an additional AI reasoning step to assess

URLs in real time. Its main contributions are:

- A Random Forest and XGBoost ensemble trained on 11 structural URL features and evaluated using a separate test set.
- Probability calibration using a separate calibration subset to make confidence estimates more meaningful.
- A Google Gemini analysis layer that provides a second assessment and a plain-English explanation.
- A Chrome extension, React dashboard, FastAPI backend, and email alerts for malicious detections.
- A human-reviewed learning process in which candidate URLs are checked before being added to future training data.

The system analyzes URLs only. Full webpage content, downloaded files, and network-level traffic are outside the scope of this work.

II. RELATED WORK

Several studies have explored machine-learning and deep-learning methods for phishing detection. Sahingoz et al. [2] studied phishing detection using URL-based machine-learning methods. Ghalechyan et al. [3] compared deterministic and probabilistic neural networks for phishing URL detection and reported strong validation results in their experimental setting. Almujahid et al. [4] compared several machine-learning and deep-learning algorithms across two datasets and found that a convolutional neural network performed strongly in their experiments. Gopali et al. [5] evaluated sequential models, including LSTM, BiLSTM, temporal convolutional networks, and attention-based methods, for URL-based phishing

detection. Their results indicate that sequential approaches can perform well compared with traditional feature-based methods. These studies use different datasets and experimental settings, so their reported results are not directly comparable with CyberShield. CyberShield instead combines a compact tree-based ensemble with a separate AI reasoning component and browser-based workflow.

Existing research shows that URL features and deep-learning methods can help identify phishing patterns. However, model results depend on the dataset and evaluation method, and a classifier may still produce false positives or miss unfamiliar threats. CyberShield explores a practical combination of calibrated ML predictions, an additional AI assessment, user-facing explanations, browser protection, and human review.

III. PROPOSED SYSTEM

System Architecture

CyberShield has a training pipeline and a URL-scanning pipeline, as shown in Fig. 1. During training, each labelled URL is converted into the 11 structural features used by the Random Forest and XGBoost models. The models are calibrated and saved for later use. During scanning, the Chrome extension sends a URL to the FastAPI backend. The backend first checks the trusted-domain whitelist. For other URLs, the ML ensemble and Gemini analyze the URL independently when the AI service is available. The application then returns a verdict and the available explanation. Malicious verdicts can trigger email alerts, while URLs on which the two systems agree are recorded as candidates for human review.

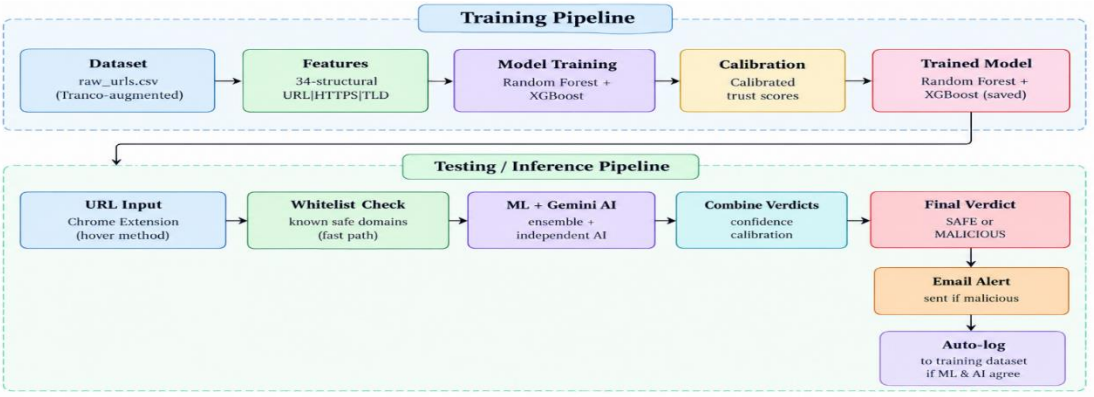

Fig. 1: Architecture of the Cyber Shield system.

Dataset

The evaluated dataset was taken from the project's processed features.csv file and contains 641,119 URLs. Label 0 represents legitimate URLs and label 1 represents malicious URLs. The current experiment does not include the additional 10,000 Tranco URLs mentioned in an earlier draft, so that augmentation is not claimed here. The dataset split used for the reported experiment is summarized in Table I.

Stage	Legitimate	Malicious	Total
Dataset used	428,080	213,039	641,119
Split summary	Training / calibration / test	—	410,316 / 102,579 / 128,224

Table I. Dataset Construction

Feature Extraction

Each URL is represented by 11 numerical features: URL length, hostname length, counts of dots, hyphens, at-signs, question marks, equals signs, digits, percent symbols, and slashes, plus an indicator for HTTPS usage. These features describe the visible structure of the URL without downloading or inspecting the full webpage. The models learn patterns across these features rather than treating any single characteristic as proof that a URL is malicious.

ML Ensemble and Calibration

Two tree-based models were trained: a Random Forest [6] with 100 trees and a maximum depth of 20, and XGBoost [7] with 100 estimators, a maximum depth of 6, and a learning rate of 0.1. The models were implemented using scikit-learn [8] and XGBoost. Class weights were used to account for the imbalance between legitimate and malicious examples. Probability calibration was performed on a separate calibration subset [9], without using the held-out test set for model fitting or calibration. The calibrated probabilities are combined using the following weighted ensemble:

$$P(\text{ensemble}) = 0.55\ P(\text{Random Forest}) + 0.45\ P(\text{XGBoost}) \quad (1)$$

A URL is classified as malicious when the ensemble probability is at least 0.5. Otherwise, it is classified as legitimate. Calibration can make probability estimates more useful, but it does not guarantee that every individual prediction is correct.

Gemini AI Reasoning Layer

For URLs that are not covered by the whitelist, the backend can send the URL and security-analysis instructions to Google Gemini [10]. Gemini returns an additional verdict, confidence estimate, and plain-English explanation. It is not used to train the ML models; it operates as a separate analysis step during detection. If Gemini is unavailable or its request quota is exhausted, the application falls back to the ML result. In that case, scanning remains available, but the additional AI assessment is not provided.

Chrome Extension, Dashboard and Email Alerts

The Chrome extension, built with Manifest V3 [11], checks links on hover or click and sends the URL to the FastAPI [12] backend through its /classify endpoint. For malicious verdicts, the extension displays a warning with the URL, confidence information, and detection reasons. The React dashboard supports manual URL scanning and displays monitoring information such as threat statistics, detection history, and model-performance visualizations. When the final application verdict is malicious, an email alert can be sent with the available URL and detection details.

Human-Verified Continuous Learning

URLs classified as malicious by both the ML ensemble and Gemini can be recorded as candidates for future training. The candidates are checked for duplicates and manually reviewed before they are added to the training dataset. The model can then be retrained and evaluated again. This review step is important because agreement between two systems is useful evidence, but it does not prove that a URL is malicious.

IV. RESULTS AND DISCUSSION

Model Performance

The ML ensemble was evaluated on a held-out test set that was kept separate from training and calibration. Table II reports the measured accuracy, precision, recall, F1-score, and AUC-ROC. These metrics describe the ML ensemble only; they should not be treated as measured end-to-end accuracy for Gemini or the full application.

Metric	Result
Accuracy	94.32%
Precision	91.45%
Recall	91.47%
F1-score	91.46%
AUC-ROC	98.46%

Table II. Ensemble Performance on the Test Set

Functional and Real-World URL Testing

The application was tested with legitimate URLs, phishing-style URLs, and several challenging patterns, as shown in Table III. In TC01 and TC02, the AI-assisted decision process returned SAFE for the tested legitimate domains. TC03 and TC04 were handled by the whitelist. The private-IP and learning-management-system examples in TC05 and TC06 were also returned as SAFE in these functional tests. The suspicious patterns in TC07–TC10 were returned as MALICIOUS. These examples confirm that the application workflow operated for the selected cases, but they are not a statistically representative benchmark of the complete system.

ID	Test input	Final result	Status
TC01	biet.ac.in	SAFE (AI corrected ML)	Pass
TC02	huggingface.co	SAFE (AI corrected ML)	Pass
TC03	quora.com	SAFE (whitelist)	Pass
TC04	et.internshala.com	SAFE (whitelist)	Pass
TC05	192.168.1.1/login	SAFE (private IP)	Pass
TC06	myuniversity-portal.instructure.com/login	SAFE (Canvas LMS)	Pass
TC07	paypal-secure-login-verify.tk	MALICIOUS	Pass

ID	Test input	Final result	Status
TC08	amaz0n-security-alert.xyz/confirm	MALICIOUS	Pass
TC09	192.168.45.99/account/verify.php	MALICIOUS	Pass
TC10	mystore-deals123.shop/product?id=45	MALICIOUS (~82%)	Pass

Table III. Representative Test Cases

System Implementation and Functional Testing

The following figures illustrate the implemented dashboard, model-analytics interface, browser-extension warning, and email-alert workflow.

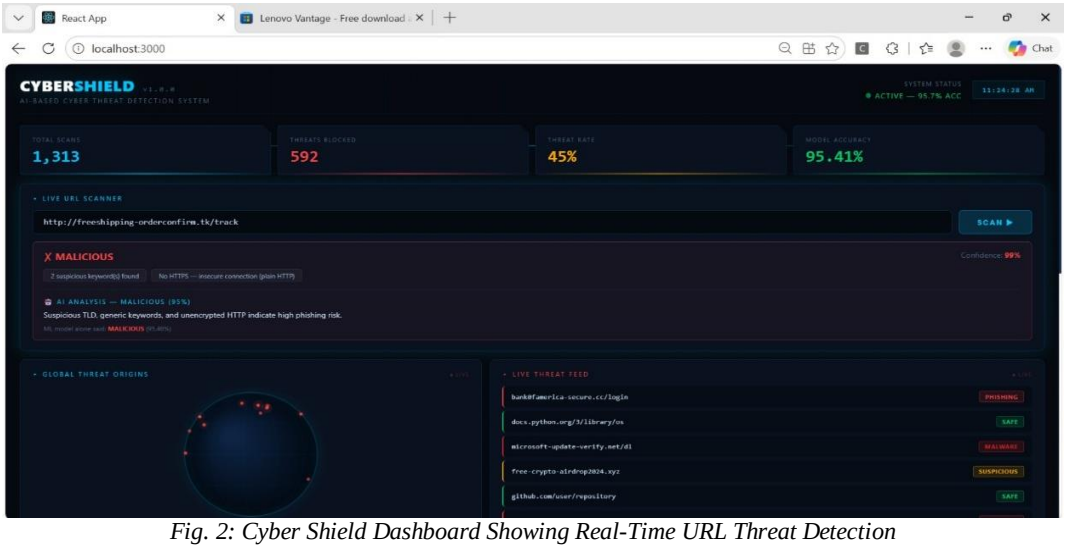

Fig. 2: Cyber Shield Dashboard Showing Real-Time URL Threat Detection

The CyberShield dashboard provides a centralized interface for monitoring and detecting malicious URLs in real time. It displays important statistics, including total scans, threats blocked, threat rate, and model accuracy. The live URL scanner allows users to enter suspicious URLs and obtain immediate classification results. It also displays the detection verdict, confidence score, suspicious URL characteristics, and AI-generated explanations. This helps users understand potential threats and make informed security decisions.

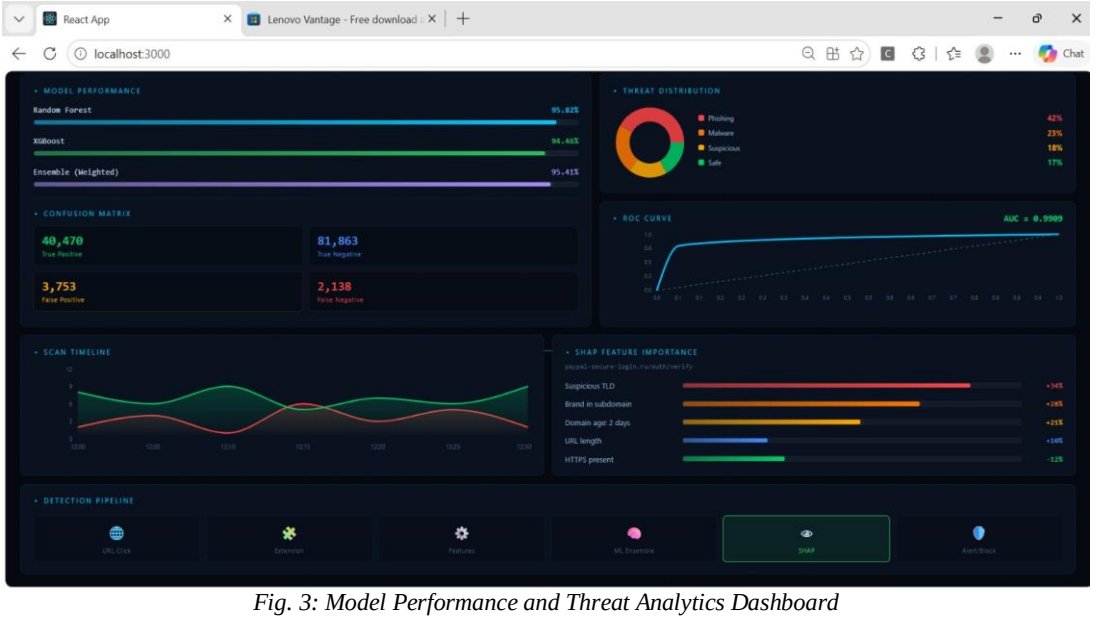

Fig. 3: Model Performance and Threat Analytics Dashboard

This figure illustrates the analytical capabilities of the CyberShield system through various model-performance visualizations. It displays the performance indicators of Random Forest, XGBoost, and the weighted ensemble model. The confusion matrix presents true positive, true negative, false positive, and false negative predictions. The ROC curve and threat distribution chart provide additional insights into classification performance and detected threat categories.

Furthermore, the scan timeline and SHAP feature importance visualization help analyze detection patterns and the contribution of individual URL features.

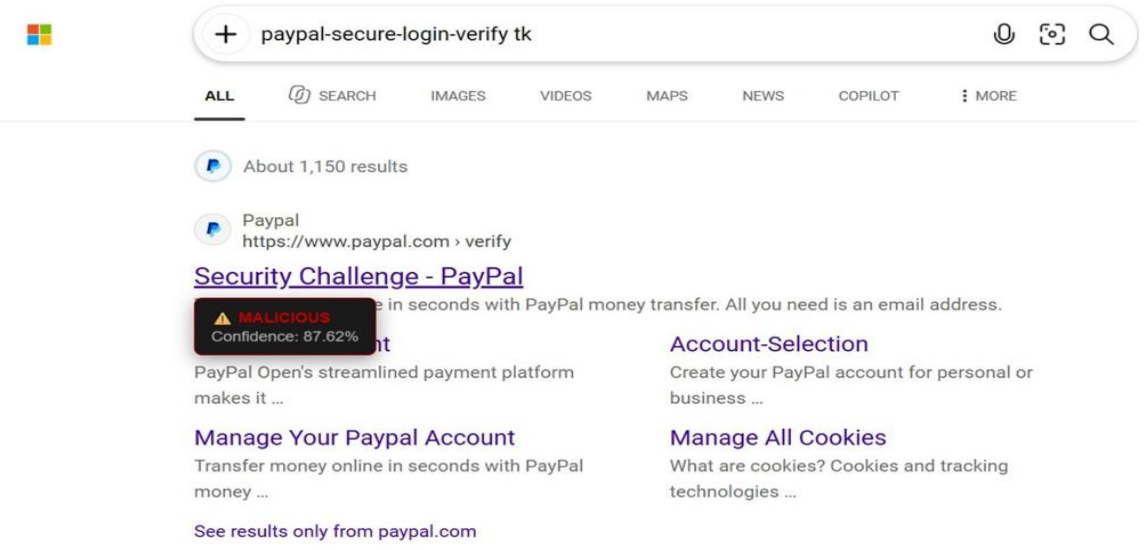

Fig. 4: Chrome Extension Displaying a Malicious URL Warning

The CyberShield Chrome extension provides real-time browser protection by identifying potentially malicious URLs. It displays a warning notification along with the confidence score when a suspicious URL is detected. In the illustrated example, a PayPal-related phishing URL is classified as malicious with 87.62% confidence, helping users avoid unsafe links.

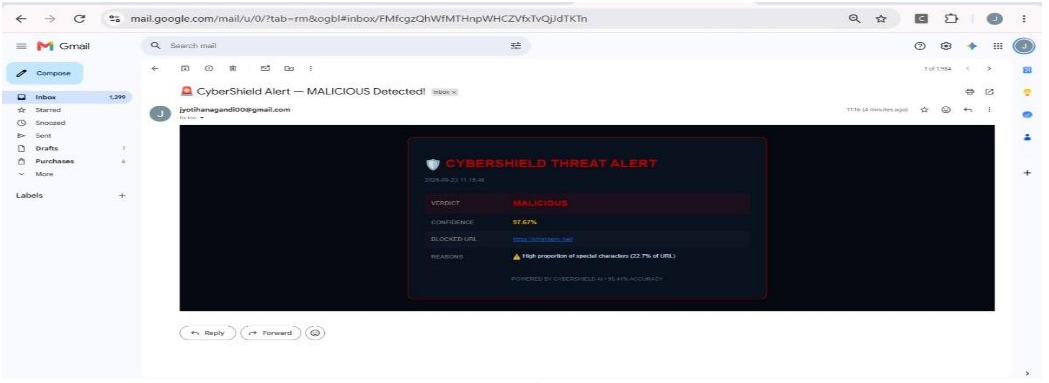

Fig. 5: Email Alert Generated for Malicious URL Detection

CyberShield incorporates an automated email notification mechanism to inform users about detected malicious URLs. Whenever a URL receives a malicious verdict, the system generates an alert containing relevant detection information. The notification includes the threat classification, confidence score, blocked URL, and reason for identifying the URL as suspicious. This allows users to remain informed about potential cybersecurity threats even when they are not actively monitoring the dashboard. The email alert mechanism adds an additional layer of security awareness and threat communication.

Confidence Calibration

Calibration was used because uncalibrated models can produce confidence values that appear more certain than the evidence supports. In one observed example, mystore-deals123.shop/product?id=45 received a malicious confidence of approximately 82%. This is an illustrative case, not a full calibration analysis. A reliability diagram or expected calibration error would provide a more systematic measurement of probability calibration.

Continuous Learning

The learning workflow was checked using a phishing URL for which both the ML model and Gemini returned a malicious verdict. The URL was recorded as a candidate, manually reviewed, and then promoted into the training dataset before retraining. This workflow keeps human verification between a system prediction and its use as training data. A separate evaluation is still needed to establish whether retraining improves generalization.

Advantages and Limitations

CyberShield brings together URL-based ML classification, an AI reasoning layer, calibrated confidence estimates,

browser warnings, a dashboard, and email alerts. It can provide users with an explanation rather than only a binary label, and its human-reviewed learning process reduces the risk of treating unverified predictions as ground truth. Its limitations include dependence on internet access and Gemini quota availability, possible false positives and false negatives, and the fact that it does not inspect webpage content, downloads, or network traffic. When Gemini is unavailable, the application uses the ML result alone, so some unfamiliar legitimate URLs may still be flagged.

V. CONCLUSION

This paper presented CyberShield, a malicious URL detection system that combines a calibrated Random Forest and XGBoost ensemble using 11 structural URL features with a separate Gemini reasoning layer. The system is available through a Chrome extension and React dashboard, uses a FastAPI backend, can send email alerts, and supports a human-reviewed learning workflow. On the held-out test set, the ML ensemble achieved 94.32% accuracy, 91.45% precision, 91.47% recall, a 91.46% F1-score, and a 98.46% AUC-ROC. Functional tests checked the main URL-scanning workflow and the ML-only fallback when the AI service was unavailable. Future work should evaluate the complete system on a larger, time-separated URL collection, measure the contribution of the Gemini layer quantitatively, and investigate webpage-content analysis and more formal calibration measures.

REFERENCES

1. Anti-Phishing Working Group (APWG), "Phishing activity trends report." [Online]. Available: https://apwg.org/trendsreports/
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. H. Ghalechyan, E. Israyelyan, A. Arakelyan, G. Hovhannisyan, and A. Davtyan, “Phishing URL detection with neural networks: An empirical study,” Scientific Reports, vol. 14, Art.no. 25134, 2024, doi: 10.1038/s41598-024-74725-6.
4. N. F. Almujahid, M. A. Haq, and M. Alshehri, “Comparative evaluation of machine learning algorithms for phishing site detection,” PeerJ Computer Science, vol. 10, Art. no. e2131, 2024, doi: 10.7717/peerj-cs.2131.
5. S. Gopali, A. S. Namin, F. Abri, and K. S. Jones, “The performance of sequential deep learning models in detecting phishing websites using contextual features of URLs,” in Proc. 39th ACM/SIGAPP Symposium on Applied Computing (SAC ’24), 2024, pp. 1064–1066, doi: 10.1145/3605098.3636164.
6. L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.
7. T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD '16), San Francisco, CA, USA, 2016, pp. 785–794.
8. F. Pedregosa et al., "Scikit-learn: Machine learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.
9. A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in Proc. 22nd Int. Conf. Mach. Learn. (ICML '05), Bonn, Germany, 2005, pp. 625–632.
10. Google AI for Developers, "Gemini API documentation." [Online]. Available: https://ai.google.dev/gemini-api/docs
11. FastAPI, "FastAPI documentation." [Online]. Available: