International Journal of Advanced Research in Computer Science & Technology (IJARCST)

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 6, November - December 2025||

DOI:10.15662/IJARCST.2025.0806016

Cyber Fraud App Detection and Blocking System using Machine Learning

Hanamant R Jakaraddi[1], Anil V[2], Shrikanth Hiremath[3]

Assistant Professor, Dept. of MCA, Acharya Institute of Technology, Bangalore, India[1]

Final Year MCA Student, Dept. of MCA, Acharya Institute of Technology, Bangalore, India[2]

Assistant Professor, Department of Computer Science, Shri. Siddeshwar Government First Grade College, Nargund, Gadag, India[3]

ABSTRACT: In today's digital world, cyber fraud through harmful mobile applications is a serious threat to user privacy and national security. This research introduces a Cyber Fraud App Detection and Blocking System that uses machine learning to spot and flag fraudulent URLs or app links in real time. The system pulls features like domain age, the presence of suspicious keywords, extension types, WHOIS details, and DNS status to assess whether an app-related URL is legitimate. A trained machine learning model, based on ensemble classifiers, classifies the input as either safe or fraudulent. An admin dashboard is also included to manually block or unblock URLs, monitor logs, and review user-reported threats. Users can interact with the platform by submitting suspicious detections via email, which are logged for admin review. The system provides high detection rates with the real-time performance retained, so it is suitable for large-scale cybersecurity environments. The solution gives real-time feedback to users and helps cybersecurity managers make optimal decisions. Experimental results confirm the effectiveness of the approach provides instant feedback to the users and enables optimal decisions for cybersecurity managers. Experimental outcomes verify the efficacy of the method in detecting and eliminating cyber fraud attacks.

KEYWORDS: Machine Learning, URL Classification, Real-time Fraud Detection, DNS Blocking Cybersecurity

I. INTRODUCTION

With the advancement of digital technology in the modern era, people everywhere have made life more convenient by using websites and mobile apps more extensively. Along with this greater use of the Internet comes the rising instances of cyber fraud. Cyber con artists merely create fake applications and sites that guarantee users all sorts of things but are really meant to deceive users into exposing personal information, installing nasty software, or clicking on risky links. Old security controls typically fail to catch these emerging threats, especially in real time. To address this problem, our project proposes an intelligent cyber fraud app detection and blocking system based on machine learning. The system evaluates various features of a URL, such as the domain name, age, type of extension, and occurrence of suspicious keywords. This aids in prediction of whether a link is secure or possibly fraudulent. The predictions are achieved through machine learning models that recognize patterns characteristically found in phishing or malicious URLs.

In contrast to automatic URL-blocking systems, our solution introduces an interactive component for user participation. The system would invite the user to provide their name and email address upon identifying a suspicious URL, which would then inform the system administrator. The administrator would scrutinize the flagged URLs and determine if they should be blocked or not. This maintains flexibility and control in fraud handling.

Also, the system keeps track of all the URLs scanned, marks them as safe, fraud, or blocked, and offers extended statistics on the admin dashboard. By combining real-time detection, manual checking, and data logging, our solution provides an extremely efficient and scalable means of detecting, reporting, and dealing with cyber fraud. Therefore, it is a valuable cybersecurity resource in current online contexts.

International Journal of Advanced Research in Computer Science & Technology (IJARCST)

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 6, November - December 2025||

DOI:10.15662/IJARCST.2025.0806016

II. LITERATURE SURVEY

Huang et al. (2020) suggested a phishing website detection model based on conventional machine learning methods like Decision Trees and Naïve Bayes. Although it performed good accuracy in the detection of malicious URLs, it did not include real-time scanning of URLs and incorporation into user interfaces [1].

Verma and Das (2022) proposed a hybrid machine learning system integrating URL-based and content-based features for phishing website identification. Although useful in static analysis, it needed full-page content loading, rendering it slower and less appropriate for lightweight or mobile contexts [2].

Sahingoz et al. (2019) proposed a classification model based on n-gram analysis and Random Forests for phishing detection at the URL level. Their model, however, did not utilize domain-related attributes such as age, WHOIS information, or user interaction mechanisms, which restricts the adaptability towards newer threats [3].

Hota et al. (2021) suggested an SVM-based phishing detection system with lexical features like the occurrence of suspicious keywords or unusual characters in the URL. The model, being accurate, lacked much explainability and no admin review for false positive management [4].

Rathore et al. (2023) presented a mobile-based phishing detection app using TensorFlow Lite for real-time detection. Although optimized for mobile devices, the system lacked an admin dashboard and manual domain block management, which limited broader usability in controlled enterprise environments [5].

Sharma et al. (2022) built a browser extension that alerts users about fraud websites by comparing against a blacklist. However, it did not support machine learning-based prediction and relied only on previously known fraud URLs, making it ineffective against zero-day phishing attacks [6].

Majeed et al. (2020) proposed a cybersecurity framework using URL feature extraction combined with a deep neural network model. While the model achieved high accuracy, the study focused solely on prediction and did not include user reporting, fraud notification, or any admin control mechanism [7].

Gupta et al. (2021) developed a Flask-based phishing detection web app integrated with a Random Forest classifier. While it demonstrated effective ML integration, it lacked key features like logging, email alerts, analytics dashboards, and domain blocklist management [8].

Jain et al. (2023) had deployed a phishing detection model with the use of WHOIS information, age of the domain, and types of TLD. Although helpful in examining domain metadata, their work did not have real-time interactivity, user-submitted fraud, and an admin-level controlled interface or decision-making [9].

Ahmed and Kumar (2022) suggested a hybrid phishing detection tool based on both DNS-based filtering and ML classification. The tool, however, did not have a feedback mechanism from the users or an integration with an admin panel to manage requests or control blocked domains [10].

This approach has a machine learning-enabled detection and admin-managed blocking framework for cyber fraud apps and URLs. It includes URL feature extraction (e.g., age of domain, type of TLD, phishing keyword presence), multi-model ML classification (Random Forest, SVM), real-time fraud prediction, optional reporting of user email, and an admin dashboard for manual blocking of domains and fraud analysis. This approach not only ensures intelligent fraud detection but also has a balance of human judgment and automated techniques to avoid false positives and establish trust.

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 6, November - December 2025||

DOI:10.15662/IJARCST.2025.0806016

III. SYSTEM ARCHITECTURE

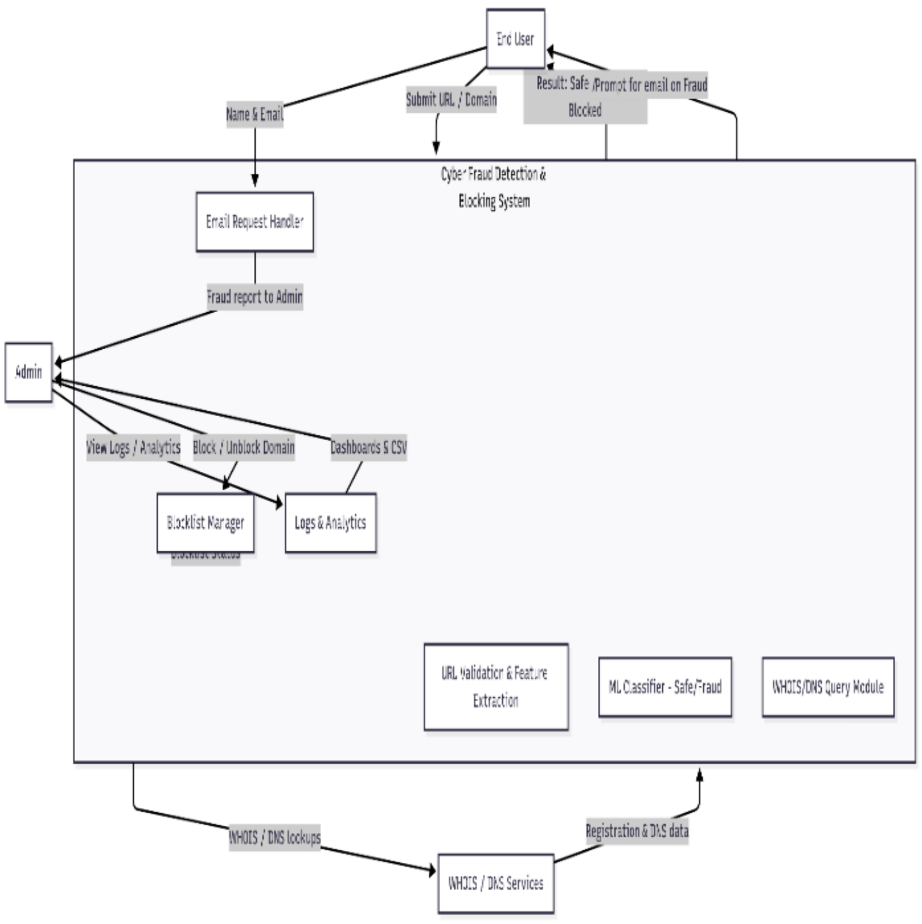

fig-1: The architecture of Cyber Fraud Detection & Blocking System

This diagram explains the proposed architecture of the cyber fraud detection system comptises five main components, which function together to facilitate safe and reliable detection. The architecture starts with the end user, since the user may submit a URL/domain through the system interface for verification. In addition to the URL, the senders name and email ID may be collected, on a voluntary basis, and associated with the detection request. Following analysis, the system provides a user output, informing the user if the input domain is safe or fraudulent and may generate an alert if the detecting domain is fraudulent. The email request handler is the first step in the processing branch which manages the submitted URL, domain or email-based request and may report suspicious activity to the administrator. It is also responsible for proper logging of the user request and ensures communication between the system and the admin. The administrator manages and controls the system since the administrator receives reports from email handler, they can examine and review detection logs and determine if fraud was committed.

International Journal of Advanced Research in Computer Science & Technology (IJARCST)

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 6, November - December 2025||

DOI:10.15662/IJARCST.2025.0806016

The system architecture within the cyber fraud detection system is constructed using five primary components which work together to facilitate not only secure detection but accurate detection. The first step is end-user interaction. An end user will either submit a URL or domain via the system interface, with optional details such as a user name or ID in email form, to potentially associate with the request. After a request, the system will review the URL or domain and present some output an end user can reference to make a decision. The output from the system will inform the user either way if the domain is safe or a fraudulent domain. In cases of fraud detection, an alert can also be generated to send users a notification of the detection. The email handling request is the first step to requesting the activity to process a submitted URL, or domain or in some cases an email-based request. The email request handle serves as a conduit to also forward suspicious related report requests to the administrator, as well as complete accurate logging functionality with respect to each user request along with communication to maintain accurate and clear contact with the admin user. In essence, the administrator or auth level user has a unique role through control of the system as they receive reports from the email request handler, review detection logs and check for any related review of fraud activities.

IV. METHODOLOGY

The suggested Cyber Fraud Detection and Blocking System is implemented as a web application based on Flask, consisting of both user-level interaction and server-level decision-making logic. The system starts when a user enters a URL through the custom-developed dark-thick web interface. The entered URL is initially verified against a blocklist residing on the server to find out immediately whether it has been labeled as malicious before. If the domain is already blocked, the system stops further analysis and informs the user with a message showing "Already Blocked."If the domain is not being blocked, the system continues with feature extraction. This includes collecting features from the URL like domain name length, the inclusion of suspicious keywords, the age of the domain (through WHOIS check), TLD checking, and the behavior of DNS response. These are fed into two pre-trained machine learning models—Random Forest and Support Vector Machine (SVM)—in order to decide whether the URL is safe or possibly fraudulent.

If the prediction by both models is for the URL to be safe, the system announces a " Safe URL" and records the output along with a timestamp in a CSV file. In case the output is marked as fraudulent, the system announces a "Fraud URL" alert on the interface. At the same time, a prompt based on JavaScript comes up informing the user voluntarily to provide their name and email address. This extra feedback is harvested only if the user agrees, and the information is recorded separately in a different CSV file for review by the administration.

All safe or fraud detection results are logged and kept in structured CSV files continuously. It also offers a secure admin login page through which administrators can see all logs, examine user-submitted email notifications, and block or unblock domains manually as needed. If a fraudulent domain is found to be malicious upon examination, the admin can block it through the dashboard, upon which the system will automatically tag any subsequent user attempts to access said domain as "Blocked."

International Journal of Advanced Research in Computer Science & Technology (IJARCST)

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 6, November - December 2025||

DOI:10.15662/IJARCST.2025.0806016

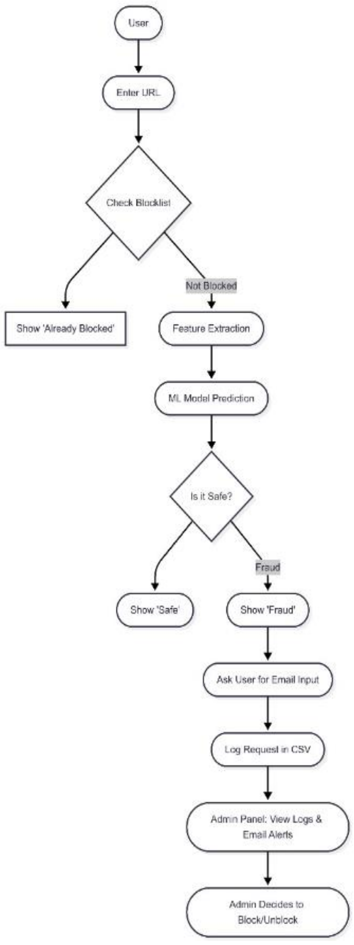

Fig-2: Workflow Diagram

1. **Data Preprocessing:** The URL dataset was initially cleaned and preprocessed to have high-quality inputs for model training. Duplicates and invalid URLs were eliminated. Feature extraction was conducted by examining lexical features (length, entropy, number of subdomains, occurrence of digits or special characters), domain features (TLD type, WHOIS domain age, DNS records), and phishing keyword presence. To standardize the scale of numeric features, StandardScaler was used such that each feature has zero mean and unit variance to avoid bias in distance-based classifiers.
2. **Model Training:** Two machine learning models, Random Forest (RF) and Support Vector Machine (SVM), were trained on the processed dataset. The Random Forest classifier utilizes ensemble learning by combining many decision trees to minimize variance and enhance generalization. The SVM model, employing the RBF kernel, discriminates between fraudulent and secure URLs by maximizing the margin within classes within transformed feature space. Hyperparameter tuning was used to maximize accuracy and minimize false positives.
3. **Prediction & Deployment:** In the deployment stage, an input URL is initially fed through the preprocessing pipeline for scaling and feature extraction. The trained models proceed to classify whether the URL is fraudulent or safe. Output is presented to the user, and suspicious URLs are written to log for inspection by the admin. Notably, fake URLs are not blocked outright; rather, an alert is triggered to the admin, who may block or unblock domains manually through the dashboard. The system is embedded in a Flask-based web app that includes a friendly interface, with real-time detection, visualization of output, and administrative management.

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 6, November - December 2025||

DOI:10.15662/IJARCST.2025.0806016

V. ALGORITHM CALCULATIONS

1. Standard Scaler (Feature Scaling):
Each numerical feature xxx is normalized to zero mean and unit variance:

$$x^1 = \frac{x - \mu}{\sigma}$$

where μ is the mean and σ is the standard deviation.

2. Random Forest (Ensemble Prediction): For a given input x, the class prediction C(x) is determined by the majority vote of Ntrees decision trees

$$C(x) = mode\{ht(x)|t = 1, 2, \dots Ntree$$

3. Accuracy (Model Evaluation): The performance of the trained model is measured as:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

VI. RESULTS AND DISCUSSION

The suggested Cyber Fraud App Detection and Blocking System was tested by utilising a mix of publicly accessible phishing datasets and tested manually with live URL samples. The outcome proves that the combination of machine learning, domain-based heuristics, and admin intervention massively improves fraud detection accuracy and control.

Table-1: presents the performance scores of the two models on the independent test set.

	Random Forest	SVM
Accuracy	0.90	0.91
Precision	0.77	0.79
Recall (Sensitivity)	0.67	0.70
F1-Score	0.71	0.73

The performance evaluation of the proposed Cyber Fraud Detection and Blocking System demonstrates that both Random Forest and Support Vector Machine (SVM) achieved strong results across all metrics, with SVM consistently outperforming Random Forest. The accuracy values indicate that both models are highly reliable, with SVM achieving 91% compared to Random Forest's 90%. Precision and recall further highlight SVM's effectiveness, as it not only reduced false fraud detections (precision = 0.79 vs. 0.77) but also successfully identified a higher proportion of fraudulent URLs (recall = 0.70 vs. 0.67). The F1-Score, which balances precision and recall, shows that SVM (0.73) provided a more robust trade-off than Random Forest (0.71). These findings suggest that while both models are suitable for fraud detection, SVM offers a slight edge in overall performance and is better suited as the primary detection model, with Random Forest serving as an ensemble backup for improved robustness.Figure1.

VII. CONCLUSION

The Cyber Fraud Detection and Blocking System offers a strong, intelligent solution for discovering and dealing with fraudulent URLs through the use of machine learning combined with rule-based inspection. It supports real-time threat detection, gives manual control to administrators in blocking decisions, and has detailed logs with visual analytics for openness and tracking. The system values both user safety and precision, limiting false positives with admin verification instead of automated blocking. Some extra features like live domain status checks, email request logging, and a clean, fast user interface make for a secure and friendly user experience. Overall, the system shows a scalable and practical solution to fighting cyber fraud. In subsequent work, the system can further be enhanced with mobile app support, multi-language phishing detection, real-time browser extension notification, and more integration with threat intelligence to detect more attack vectors

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 8, Issue 6, November - December 2025||

DOI:10.15662/IJARCST.2025.0806016

REFERENCES

1. Almomani B B Gupta, A Meulenberg. And Akram, “Phishing Websites Detection Based on Phishing Characteristics in the URL” International Journal of Advance Computer Science and Application – 2018
2. N. Jain and M. Richariya, “A Novel Approach to Detect Phishing URLs Using Machine Learning,” International Journal of Computer Applications - 2017.
3. S. Marchal, J. Francois, R. State, and T. Engel, “PhishStorm: Detecting Phishing with Streaming Analytics,” IEEE Transactions on Network and Service Management - Dec. 2014.
4. M. Chiew, K. S. Yong, and C. L. Tan, “A Survey of Phishing Attacks: Their Types, Techniques and Mitigations,” Information Security Journal: A Global Perspective - 2020.
5. S. Verma and P. K. Das, “URL Based Phishing Detection Using Machine Learning,” in Proceedings of the 3rd International Conference on Inventive Computation Technologies – 2018
6. M. Aburrous, M. A. Hossain, K. Dahal, and F. Thabtah, “Intelligent Phishing Detection System for e-banking Using Fuzzy Data Mining,” Expert Systems with Applications - 2010.
7. H. Heartfield and G. Loukas, “A Taxonomy of Cyber-Physical Threats and Impact in the Smart Home,” Computer & Security - 2018.
8. K. Gupta and R. Shukla, “Efficient URL Feature Extraction for Detecting Malicious Webpages Using Machine Learning Techniques,” Journal of Cybersecurity and Privacy” - 2021.
9. R. Kumar and M. A. Khan, “A Hybrid Model for Detecting Phishing Websites Using Random Forest and Deep Learning,” Cybersecurity and Information Systems” - 2023.
10. R. Jain and S. Soni, "Machine Learning Approaches for Phishing Detection: A Review," Materials Today: Proceedings - 2021.
11. Basnet, K. Sung, and S. Mukkamala, "Detection of Phishing Attacks: A Machine Learning Approach," in Soft Computing Applications in Industry, Studies in Fuzziness and Soft Computing - 2008.
12. Eshete, A. Villafiorita, and K. Weldemariam, "BaitAlert: A System for Real-Time Detection of Phishing Websites," in Proceedings of the IEEE International Conference on Software Security and Reliability (SERE), - 2014
13. M. T. Alsharnouby, F. Alaca, and S. Chiasson, "Why Phishing Still Works: User Strategies for Combating Phishing Attacks," International Journal of Human-Computer Studies - 2015.