IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, November 2026
Impact Factor: 8.2

SecureBank: An Intelligent Banking Security System for Detecting Fraudulent Transactions, Phishing URLs, Malicious APK Files and DDoS Attacks

Anurag N P[1], Adhwayth Raj B[2], Anzal S[3], Harikrishna P P[4], Dency David E[5], Anjitha P[6]
Student, Department of Computer Science and Engineering[1,2,3,4]
Assistant Professor, Department of Computer Science and Engineering[5,6]
Thejus Engineering College, Thrissur, Kerala, India

Abstract: *Online banking has made financial services convenient, but it has also exposed users and banks to fraudulent transactions, phishing attacks, malicious mobile applications and Distributed Denial of Service (DDoS) attacks. Existing banking systems usually rely on predefined rules and on separate security tools that do not work together. This paper presents the design of SecureBank, an intelligent banking security system that brings banking operations and automated threat detection into one application. The banking application is built with Django and MySQL and has separate Admin, Bank and User modules for account management, fund transactions and transaction history, with rule-based fraud detection on transaction amount, frequency and other predefined conditions. Spring Cloud microservices, reached through an API Gateway, provide phishing URL classification using a machine-learning model trained on a labelled dataset, static analysis of uploaded Android application package (APK) files, and a sandbox-based network monitoring service that identifies DDoS-like activity and blocks users when configured request thresholds are exceeded. Detection results and security events are logged, and SMTP email alerts notify users and administrators. A survey of twenty studies shows that existing work treats each threat separately. The paper presents the proposed architecture and data flow diagrams; the system is an academic prototype and does not process real monetary transactions.*

Keywords: Banking security, fraud detection, phishing URL detection, APK malware analysis, DDoS detection, Django, Spring Cloud, microservices

I. INTRODUCTION

Digital banking, mobile banking applications and online payment services let users carry out financial transactions without visiting a branch. The same dependence on internet-based platforms exposes users and financial institutions to fraudulent transactions, phishing attacks, malicious applications and DDoS attacks, which can cause financial loss, unauthorised access to sensitive information and disruption of banking services. Identifying and preventing suspicious activity is therefore a basic requirement of secure digital banking.

Banking applications have traditionally relied on authentication, predefined security rules and transaction verification. These give basic protection but may not identify different kinds of cyber threats efficiently. Manual monitoring is slow, separate security tools make threat management more complex, and delayed identification of malicious activity reduces the availability and reliability of banking services.

Recent technology offers practical answers to each of these threats. Machine-learning models can learn URL characteristics associated with phishing websites; static analysis can examine APK files for suspicious permissions and

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 2, November 2026 Impact Factor: 8.2

malicious characteristics; and network monitoring can analyse request patterns to spot abnormal traffic linked to DDoS attacks. What is still missing is a single banking application that combines essential banking operations with all of these mechanisms, keeps security logs and notifies users promptly.

This work presents SecureBank, which integrates a Django-based banking application with Spring Cloud security microservices and a MySQL database. The objectives are to (i) develop a secure, user-friendly web application for managing banking operations and user accounts, (ii) detect suspicious transactions using predefined fraud rules, (iii) identify phishing URLs using machine learning, (iv) analyse APK files for potentially malicious applications using static analysis, (v) monitor network traffic and detect possible DDoS attacks using a sandbox-based service, (vi) provide automated security responses, threat logs and email notifications, and (vii) integrate banking operations and security services through Django and Spring Cloud microservices.

The rest of the paper is organised as follows. Section II reviews the literature, Section III presents the system design, including the existing and proposed systems, architecture, data flow diagrams and requirements, and Section IV concludes the paper and outlines future work.

II. LITERATURE SURVEY

Twenty research papers published between 2023 and 2025 on phishing detection, Android malware detection, DDoS detection and banking fraud detection were studied. They are grouped below by the threat they address; Table I summarises the research gaps.

A. Phishing URL Detection

Zieni et al. [1] survey list-based, similarity-based and machine-learning-based detection of phishing websites and discuss the challenge of evolving attacks. Kulkarni et al. [2] categorise detection approaches into URL-based, webpage-content-based, third-party-based and visual techniques. Roy et al. [3] extract URL features and train a machine-learning model, reporting 99% accuracy, while Ferdaws and Majd [4] compare engineered-feature models with CNN and LSTM deep-learning models, the best of which performed better. Choutapally and Das [5] add embedded-link analysis and WHOIS-based domain validation to URL classification, and Duarte et al. [6] use a LightGBM classifier on newly registered and parked domains of a financial institution, reporting 97.28% accuracy on a balanced dataset. These works depend on the training dataset and on recognising changing attack patterns, and none of them is part of a complete banking application with automated threat handling.

B. Android Malware Detection

Kaithal and Sharma [7] review machine-learning techniques for Android malware, and Raghuraman et al. [8] combine CNN and GRU models to detect several malware categories, including banking malware, but note the computational resources and dataset that deep learning requires. Kurniawan et al. [9] use static analysis, feature extraction and feature selection, where Random Forest reached 97% accuracy; static analysis may miss obfuscated or dynamically executed behaviour. Yunmar et al. [10] compare static, dynamic and hybrid analysis, noting that dynamic analysis needs additional resources and a controlled environment. Adriansyah et al. [11] combine ensemble models with SHAP to explain feature influence, with XGBoost reaching about 94.88% accuracy on CICMalDroid2020, although performance depends on how well the dataset represents new malware.

C. DDoS Detection

Owusu et al. [12] survey online DoS/DDoS detection using traffic sampling, change-point detection and machine learning, and the balance between accuracy and computational efficiency. Chauhan et al. [13] classify traffic on the CSE-CIC-IDS2018 dataset with 99.98% reported accuracy, which does not by itself establish effectiveness in every live environment. Minhas et al. [14] detect known and zero-day DDoS attacks at the fog level by combining machine learning, deep learning, few-shot learning and resource-usage analysis. Saiyed et al. [15] use an attention-based CNN–LSTM ensemble with pruning, quantisation and SHAP for consumer IoT networks, and Li et al. [16] reduce dimensionality with a truncated Lanczos-TensorSVD algorithm before XGBoost classification, reporting 99.15%

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 2, November 2026 Impact Factor: 8.2

accuracy. These studies target IoT or large-scale networks rather than online banking, so they must be adapted to a banking application's own traffic.

D. Banking and Credit-Card Fraud Detection

Sarker et al. [17] and Peneti et al. [19] apply machine learning to transaction data to classify transactions as legitimate or fraudulent, noting that class imbalance and data quality affect performance and that false alarms inconvenience genuine customers. Kiran et al. [18] evaluate Naive Bayes, KNN, Random Forest, Logistic Regression, SVM and Decision Tree, with Random Forest reaching 94.98%. Yadavalli et al. [20] review AI-based banking-fraud techniques and the limitations of rule-based methods, namely poor adaptability to evolving fraud and high false-positive rates, and point to explainable AI, federated learning, graph neural networks and blockchain as future directions.

TABLE I: Summary of Research Gaps

Focus area	Refs.	Research gap
Phishing URL detection	[1]–[6]	Depends on training data and changing attacks; not part of a banking application
Android malware	[7]–[11]	Static analysis may miss obfuscated or runtime behaviour; deep models need resources
DDoS detection	[12]–[16]	High dataset accuracy may not hold in live traffic; IoT focus rather than banking
Banking fraud	[17]–[20]	Class imbalance, false alarms and poor adaptability of rule-based methods

The survey shows that machine learning and related techniques are already used successfully against each threat, but every work addresses only one of them and none provides the banking operations, threat logging and alerting around the detector. Phishing studies do not analyse malware or network traffic, DDoS studies do not examine transactions, and the reviews do not describe an operational system. This gap between separate detectors and one integrated banking application motivates the proposed system.

III. DESIGN

A. Existing System and Problem Statement

Existing banking security uses traditional rule-based techniques and separate tools: fraud detection based on predefined rules and transaction patterns, phishing detection based on blacklists and URL filtering, Android malware tools that examine permissions, signatures and behaviours, and network monitoring for abnormal traffic. These tools operate independently and lack a unified mechanism for monitoring several threats within one banking application. Rule-based systems may fail on new and evolving attack patterns, phishing URLs change frequently to avoid detection, malware using obfuscation may escape detection, and high traffic volumes make DDoS detection difficult.

The problem addressed here is to design a unified banking security system that combines banking operations with automated detection of fraudulent transactions, phishing URLs, malicious APK files and DDoS-like network activity, maintains security records, and notifies users and administrators when suspicious activity is detected.

B. Proposed System and Workflow

SecureBank is a web-based banking application developed with Django and MySQL, with separate Admin, Bank and User modules. Banks manage user accounts, and users perform fund transactions and view their transaction histories. Suspicious transactions are identified by rule-based fraud detection that considers transaction amounts, frequency and other predefined conditions.

Spring Cloud microservices provide the security analysis. A machine-learning model trained on a labelled phishing URL dataset (Scikit-learn, Pandas, NumPy) classifies submitted URLs as phishing or safe. Uploaded APK files are examined by static analysis to identify suspicious permissions, file hashes and potentially malicious patterns. A sandbox-based network monitoring service analyses request patterns to identify DDoS-like activity and blocks a user

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, November 2026
Impact Factor: 8.2

when configured request thresholds are exceeded. Detection results and security events are stored in the database, and SMTP email notifications alert users and administrators.

The workflow is: (1) users access the application through the web interface; (2) the back end authenticates the request and processes the banking operation; (3) the relevant security module analyses the transaction details, URL, APK file or network traffic; (4) the module identifies suspicious activity using its configured model or rules; (5) the results and security events are stored in the database; and (6) if configured, email alerts are sent and predefined responses are initiated.

The main advantages are a single platform for banking operations and several detection mechanisms, centralised security logs and email alerts, and a modular architecture in which each service can be maintained and enhanced independently. SecureBank is an academic prototype; it does not process real monetary transactions or integrate with a live payment gateway.

C. System Architecture

The architecture (Fig. 1) is built around the Django banking application, the Spring Cloud API Gateway, three security services and the storage and notification components. Administrators, bank users and customers access the Django application, which provides the user interface, authentication, account management and transaction processing. Each transaction is sent to the fraud detection module, and the fraud check result is passed to the Spring Cloud API Gateway.

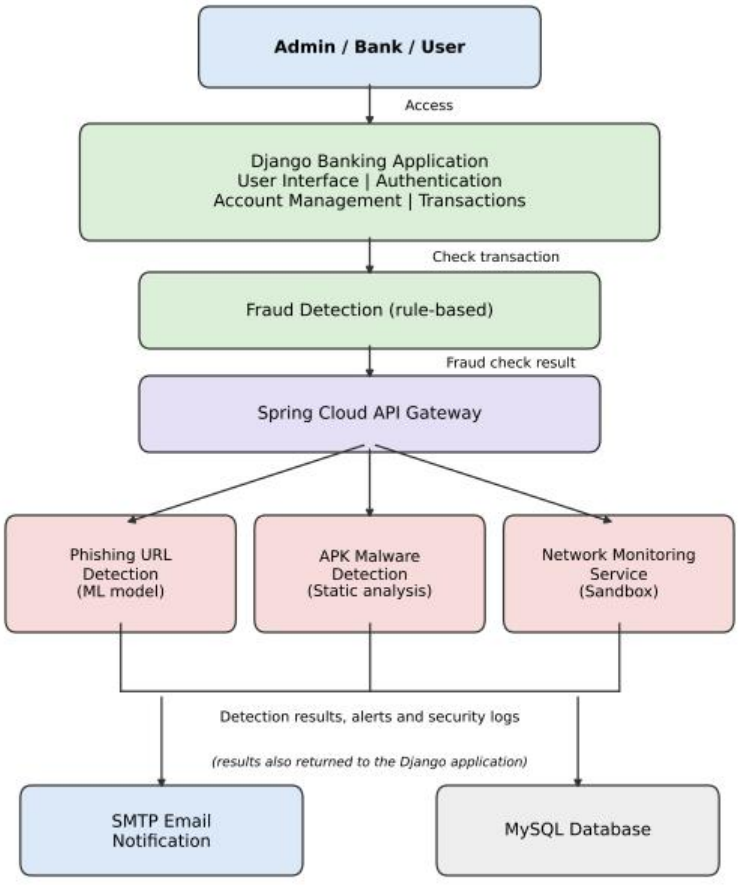


Fig. 1. System architecture of SecureBank.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 2, November 2026 Impact Factor: 8.2

The API Gateway receives URL, APK and traffic requests and routes them to the phishing URL detection service (machine-learning model), the APK malware detection service (static analysis) and the network monitoring service, which works together with a sandbox environment. Detection results are returned to the Django application. Fraud, phishing, malware and DDoS alerts are sent through the SMTP email notification service, and users, banks, accounts, transactions and security logs are stored in the MySQL database. Separating the banking application from the security services allows any service to be improved or replaced without changing the others.

D. Data Flow Diagrams

The Level 0 DFD (Fig. 2) shows SecureBank as a single process with external entities User, Bank and Administrator, a MySQL database and an SMTP email server. Users submit login details, banking requests, transaction details, URLs and APK files, and receive responses, alerts and notifications. The Bank supplies banking updates and approvals, and the Administrator manages users, configures alerts and receives reports, logs and alerts.

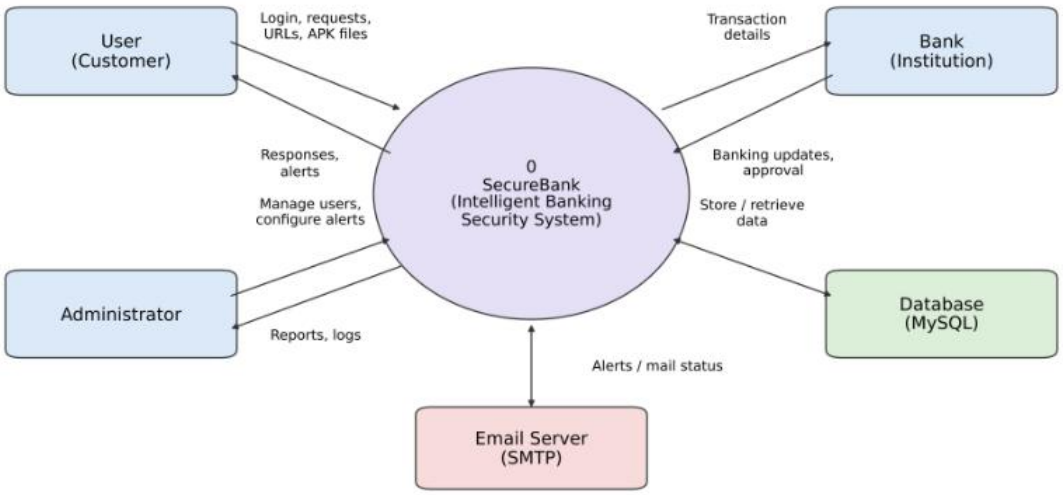


Fig. 2. Level 0 data flow diagram (context diagram).

The Level 1 DFD (Fig. 3) expands this into six processes and six data stores. Process 1.0 handles user and banking management using D1 (User Database). Process 2.0 detects fraudulent transactions and writes to D2 (Transaction Logs); process 3.0 detects phishing URLs and writes to D3 (Phishing Logs); process 4.0 analyses APK files and writes to D4 (Malware Logs); and process 5.0 performs network monitoring and DDoS detection and writes to D5 (Network Logs). Process 6.0, Security Logging and Alerts, collects the detection results, stores them in D6 (Security Logs), sends email alerts through SMTP when configured, and provides logs and reports to the Administrator.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, November 2026
Impact Factor: 8.2

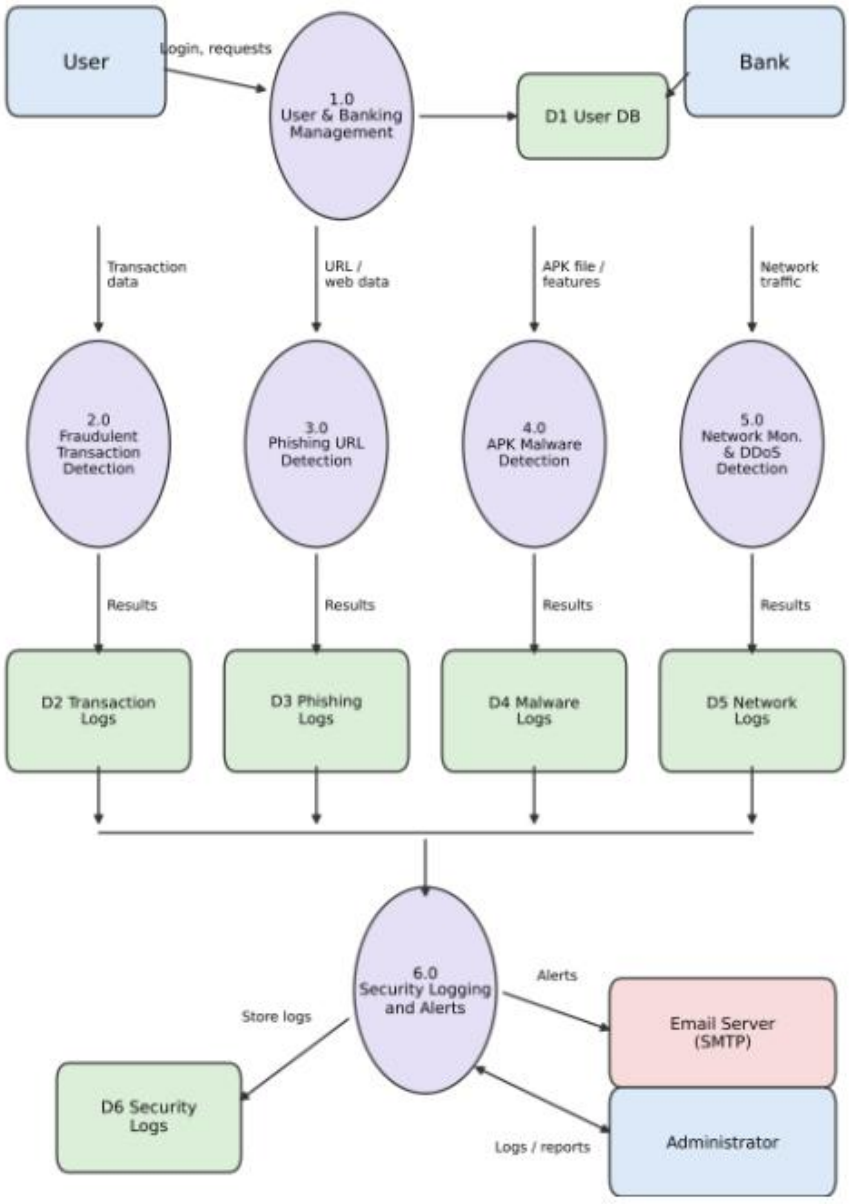


Fig. 3. Level 1 data flow diagram.

E. Hardware and Software Requirements

The system runs on modest hardware and needs no specialised equipment. Table II lists the requirements for developing and running SecureBank.

TABLE II: Hardware and Software Requirements

Component	Specification
Processor	Intel Core i3 or equivalent
RAM / Storage	8 GB RAM; 256 GB HDD/SSD

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, November 2026
Impact Factor: 8.2

Component	Specification
Display / Input	1366 × 768 monitor; keyboard and mouse; internet connection
Operating system	Windows 10/11 or Linux
Languages	Python, Java
Front end	HTML, CSS, JavaScript, Bootstrap
Back end	Django Framework, Spring Boot
Database	MySQL
Microservices	Spring Cloud, API Gateway
Machine learning	Scikit-learn, Pandas, NumPy
Network monitoring	Wireshark / Python-based traffic monitoring
Malware analysis	Python libraries for APK static analysis
Email service	SMTP
Tools	Visual Studio Code, IntelliJ IDEA; Chrome or Firefox; Git (optional)

IV. CONCLUSION AND FUTURE WORK

This paper presented the study and design of SecureBank, an intelligent banking security system. Online banking increases exposure to fraudulent transactions, phishing, malicious applications and DDoS attacks, and a review of the literature showed that each threat is well studied separately but that an integrated banking application is missing. The proposed system supports user and account management and fund transactions, and integrates rule-based fraud detection, machine-learning-based phishing URL detection, static analysis of APK files and sandbox-based network monitoring for DDoS-like activity, with security logging and email alerts. The requirements, architecture and data flow diagrams have been prepared, and the modules will be implemented, integrated and evaluated in the next phase. SecureBank is an academic prototype and does not process real monetary transactions.

Future work includes:

real-time threat detection with minimal delay;

deep learning and other advanced models to improve accuracy and reduce false positives [4], [8];

automated threat prevention such as blocking suspicious requests, restricting compromised accounts and isolating malicious files;

sandbox-based dynamic malware analysis [10];

a centralised security dashboard for threats, transaction alerts and reports;

multi-factor authentication;

continuous model training on new, verified data;

integration of trusted threat-intelligence sources;

cloud deployment for scalability; and

a mobile application delivering security notifications.

REFERENCES

1. R. Zieni, L. Massari and M. C. Calzarossa, “Phishing or not phishing? A survey on the detection of phishing websites,” 2023.
2. Kulkarni, V. Balachandran and T. Das, “Phishing webpage detection: Unveiling the threat landscape and investigating detection techniques,” IEEE Commun. Surveys Tuts.
3. P. K. Roy, A. Kumar and A. Singh, “Advanced learning for phishing URLs detection to secure consumer-centric applications,” 2024.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 2, November 2026
Impact Factor: 8.2

4. R. Ferdaws and N. Ebrahimi Majd, "Phishing URL detection using machine learning and deep learning," 2024.
5. R. Choutapally and T. Das, "Web-based machine learning framework for phishing URL detection and analysis," 2025.
6. J. D. Duarte et al., "Machine learning for early detection of phishing URLs in parked domains: An approach applied to a financial institution," 2025.
7. P. K. Kaithal and V. Sharma, "A systematic review and future perspective of Android malware detection based machine learning techniques," in Proc. IEEE ICTBIG, 2023.
8. D. Raghuraman, S. Soniya and S. Shivani, "Automatic Android malware detection using CNN and GRU," 2024.
9. F. Kurniawan, D. Stiawan, D. Antoni, M. Y. Idris and R. Budiarto, "A robust and efficient machine learning framework for enhancing early detection of Android malware," 2025.
10. R. A. Yunmar, S. S. Kusumawardani, Widyawan and F. Mohsen, "Hybrid Android malware detection: A review of heuristic-based approach," 2024.
11. R. Adriansyah, P. Sukarno and A. A. Wardana, "Android malware detection using ensemble learning and feature selection with insights from SHAP explainable AI," in Proc. 11th IEEE ISCMI, 2024.
12. E. Owusu et al., "Online network DoS/DDoS detection: Sampling, change point detection, and machine learning methods," IEEE Commun. Surveys Tuts.
13. S. Chauhan, P. Byahatti and S. K. Patel, "Securing networks: Leveraging machine learning for enhanced DDoS detection," 2024.
14. M. R. Minhas et al., "F-OSFA: A fog level generalizable solution for zero-day DDoS attacks detection," 2025.
15. M. F. Saiyed, I. S. Al Anbagi and M. S. Hossain, "Interactive and explainable optimized learning for DDoS detection in consumer IoT networks," 2025.
16. S. Li, J. Xu, P. Liu, X. Li et al., "Truncated Lanczos-TSVD: An effective dimensionality reduction algorithm for detecting DDoS attacks in large-scale networks," 2024.
17. Sarker, M. A. Yasmin, M. A. Rahman and B. R. Roy, "Credit card fraud detection using machine learning techniques."
18. V. S. Kiran et al., "Detecting credit card fraud using machine learning models," in Proc. 2nd IEEE ICMACC, 2024.
19. S. Peneti, S. Rama Krishna, A. Kiran and H. Kumar, "Credit card fraud detection using machine learning," IEEE conference paper, 2024.
20. R. Yadavalli, R. Polisetti and R. R. Kurada, "Analysis on AI-based techniques for detection of banking frauds: Recent trends, challenges, and future directions," 2025