International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 3
May-Jun 2026, PP 01-06

Bank ATM Simulation System Using Java with Enhanced Security and User Management

M. Shiva Nageshwarrao, M. Anjil Reddy, P. Shiva Ganesh
Department of Computer Science & Engineering, Guru Nanak Institute of Technology, Hyderabad, India

Abstract—Traditional ATM systems support basic financial operations but suffer from critical security limitations, including unencrypted data storage, weak authentication, and inadequate administrative controls. This paper presents an enhanced Bank ATM Simulation System developed using Java, JavaServer Pages (JSP), Servlets, Java Database Connectivity (JDBC), and MySQL on Apache Tomcat. The system operates in two modes—Admin Mode and User Mode. Advanced Encryption Standard (AES) secures sensitive card details and PINs in the database, while Multi-Factor Authentication (MFA) governs system access. The Admin module automates credential generation and supports transaction monitoring and application approval workflows. The User module handles deposits, withdrawals, balance checks, and profile management. All transactions are logged for full auditability. Comparative evaluation across seven system dimensions confirms that the proposed system improves upon existing approaches in security, usability, and administrative control.

Keywords—ATM Simulation, AES Encryption, Multi-Factor Authentication, Java Servlets, MySQL, Banking System, Transaction Security, JSP, Apache Tomcat.

I. INTRODUCTION

Banking technology has undergone rapid transformation over the past two decades, with Automated Teller Machines (ATMs) remaining a critical touchpoint between financial institutions and their customers. Despite widespread deployment, most ATM simulation systems used in academic and training environments are built on outdated architectures that prioritize functionality over security and user experience. Key weaknesses include the storage of sensitive user data without encryption, absence of robust authentication mechanisms, and limited administrative oversight [1].

These weaknesses present tangible risks. When card numbers and Personal Identification Numbers (PINs) are stored in plaintext, a single database breach exposes every user simultaneously. Without audit logging, it becomes impossible to trace unauthorized access or detect fraudulent activity after the fact. From an administrative standpoint, existing systems offer only basic account management with no structured approval workflows or automated credential generation [2].

This paper presents a redesigned ATM Simulation System built on Java J2EE (Java 2 Enterprise Edition) that introduces four targeted improvements: AES data encryption at the storage layer, Multi-Factor Authentication (MFA) at the access layer, an automated account lifecycle workflow managed by a dedicated Admin module, and comprehensive timestamped transaction logging for full auditability. A structured feature comparison with the existing system is provided, and functional test results across eight defined test cases are reported.

The remainder of this paper is organized as follows: Section II reviews relevant literature; Section III presents the system design and methodology; Section IV covers implementation and testing; Section V reports results and discussion; and Section VI concludes with future directions.

II. LITERATURE SURVEY

A. Biometric Authentication in ATM Systems

Rising incidents of card skimming and PIN theft have pushed researchers toward stronger authentication mechanisms for ATM environments. Sharma and Mehta [1] proposed replacing traditional PIN-based login with biometric verification—fingerprint recognition and facial identification—and demonstrated measurable improvement in unauthorized access prevention. Their work established an important precedent: that authentication is the first line of defence in any ATM system. However, the study focused entirely on the access layer and left the problem of secure backend data storage unaddressed, a gap that is directly targeted in the security design of the present work.

B. Blockchain Applications in Financial Technology

Karadag et al. [3] reviewed blockchain-based fintech innovations comprehensively, noting that decentralization and immutability make blockchain an appealing mechanism for transaction integrity in financial systems. The review covered ATM-adjacent use cases including digital payment systems and decentralized asset management. For the present work, the most relevant takeaway is the authors' acknowledgment that high computational overhead and infrastructure complexity limit blockchain's applicability in lightweight or training environments. The proposed system achieves equivalent transaction integrity through AES encryption and server-side audit logging—approaches that are proven, low-overhead, and practical for simulation platforms.

C. ATM System Efficiency Through Simulation

Kubendran and Sivakumar [4] used AnyLogic-based simulation to analyse ATM queuing and service efficiency, identifying performance bottlenecks that are invisible to manual inspection. Their study demonstrated that simulation environments are a legitimate and rigorous instrument for evaluating ATM system behaviour before real-world deployment. This finding directly supports the motivation behind the present work: a well-designed ATM simulation, with realistic functional modules and proper security controls, provides genuine educational and developmental value to institutions and students.

D. Machine Learning for ATM Network Quality Assessment

Safarzadeh et al. [5] applied machine learning classifiers to the problem of ATM network quality assessment, showing that ensemble methods significantly improved diagnostic precision over single-model approaches. While the focus of their work was performance rather than security, the study highlighted the richness of data that a properly instrumented ATM system can generate. This reinforced the design decision in the present system to maintain a comprehensive transaction log: structured, timestamped data is the foundation for any future analytical or machine learning capability built on top of the simulation platform.

III. System Design and Methodology

A. System Architecture

The proposed system is built on a three-tier web architecture, illustrated in Fig. 1. The Presentation Layer is constructed with HTML, CSS, and JavaServer Pages (JSP), providing the user-facing interface for both Admin Mode and User Mode. The Application Layer comprises Java Servlets that handle all business logic: user authentication, transaction processing, account lifecycle management, and input validation. The Data Layer uses MySQL 5.5, accessed through Java Database Connectivity (JDBC), to persistently store user profiles, account records, AES-encrypted card details, and transaction histories. Apache Tomcat 9.0 serves as the runtime container binding all three tiers. This architectural separation ensures that modifications to one layer do not propagate errors to others, and allows each component to be independently maintained or extended.

Customer Registration	**Manager Login**	**Employee Login**	**ATM Login**
↓	↓	↓	↓
Application Form	Approve / Reject Account	Search Account	Card No. + PIN Auth
↓	↓		↓
Pending Approval	Auto-Generate Credentials (AES)		Transaction Dashboard
↓	↓		↓
	MySQL Database (Encrypted Storage)		Deposit / Withdraw / Balance Check
			↓

			Transaction Log (Audit Trail)

Fig. 1. System architecture and data flow of the proposed ATM Simulation System.

B. Comparison with Existing System

Table I presents a structured comparison of the existing ATM simulation baseline against the proposed system across seven functional and security dimensions. The comparison confirms that the proposed system introduces substantive improvements in every dimension evaluated.

TABLE I
COMPARISON OF EXISTING AND PROPOSED SYSTEM

Feature	Existing System	Proposed System
Data Encryption	Not implemented	AES encryption on card numbers and PINs
Authentication	PIN only	PIN + Multi-Factor Authentication (MFA)
Admin Controls	Basic CRUD only	Account approval, generation, employee management
User Interface	Minimal, static	Interactive dashboard with profile and history
Transaction Logging	Basic / none	Full timestamped logs with audit trail
Account Generation	Manual	Automated 16-digit account and ATM card number
User Profiling	Not available	Full profile view and personal detail management

C. Module Design

1) Admin Module: The administrator authenticates using verified email and password credentials. The admin dashboard lists all pending account applications with full applicant details. On clicking Approve, the system triggers an automated server-side routine that generates a unique 16-digit account number, a 16-digit ATM card number, and a 4-digit PIN—each stored in AES-encrypted form in the database. The admin can additionally manage bank employee accounts and monitor all customer transactions across the system.

2) ATM Module: Users authenticate at the ATM portal using their ATM card number and PIN. The Servlet applies AES decryption and compares the result against stored values. Upon successful authentication, the dashboard renders three operations: Check Balance, Deposit, and Withdraw. Every transaction is validated against the live account balance, immediately persisted to the transaction log with a timestamp, and reflected in the displayed balance.

3) Customer Module: New customers complete an online registration form capturing name, email address, mobile number, residential address, Aadhaar number, and PAN number. The application enters a pending state awaiting manager review. Upon approval, login credentials are generated automatically and become immediately active. Existing customers authenticate directly using their assigned account number.

4) Transaction Module: This module governs all financial operations. Credit operations add funds to the account via deposit. Debit operations execute withdrawals after validating that the requested amount does not exceed the available balance. Each operation writes a timestamped record to the transaction log table, producing an auditable history accessible to both the account holder and the system administrator.

D. Security Architecture

Security is enforced at two distinct layers. At the data layer, AES (Advanced Encryption Standard) encrypts all card numbers and PINs before they are written to the MySQL database. Stored values are therefore unreadable ciphertext, ensuring that even a direct database breach yields no usable credential information. At the access layer, Multi-Factor Authentication (MFA) requires a second verification step beyond the PIN before granting ATM access, substantially reducing the risk that a stolen single credential can compromise an account.

E. System Requirements

Tables II and III summarise the software technology stack and hardware configuration used for development and testing.

TABLE II
SOFTWARE TECHNOLOGY STACK

Layer / Component	Technology Used
Front End	HTML, CSS, JavaServer Pages (JSP)
Back End	Java Servlets, J2EE (Java 2 Enterprise Edition)
Database	MySQL 5.5 via JDBC (Java Database Connectivity)
Web Server	Apache Tomcat 9.0
Security	AES Encryption, Multi-Factor Authentication (MFA)
IDE	Eclipse IDE on Windows 10
Language	Java (Core Java, J2EE)

TABLE III
HARDWARE REQUIREMENTS

Component	Specification
Processor	Intel Core 2 Duo / Pentium IV 2.6 GHz or higher
RAM	Minimum 512 MB DDR RAM
Hard Disk	Minimum 40 GB
Monitor	15-inch colour display
Operating System	Windows 10 or above

IV. IMPLEMENTATION

A. System Development

The application entry point presents four role-based access portals: Manager, Bank Employee, User, and ATM. Each portal enforces its own authentication scheme before granting access to its module.

The Manager portal uses email and password credentials for login. The admin dashboard presents all pending registrations with full applicant information and Approve / Reject action buttons. On approval, a server-side Servlet triggers the automated credential generation routine, stores AES-encrypted values in the database, and returns a confirmation screen displaying the generated account number, ATM card number, and PIN. This eliminates the manual credential assignment step present in the existing system.

The ATM portal accepts a 16-digit card number and a 4-digit PIN. The authentication Servlet retrieves the encrypted record from the database, applies the AES decryption key, and compares the plaintext result against the user's input. On successful match, the user session is established and the transaction dashboard is rendered. Deposit and Withdraw operations each invoke a dedicated transaction Servlet that validates the amount, updates the balance record, and inserts a new timestamped row into the transaction log table. The Bank Employee portal provides a single-purpose account lookup interface: staff enter an account number to retrieve the associated customer profile, enabling front-desk verification without exposing any encrypted card data.

B. Software Testing

Testing was performed across four levels to ensure correctness at every layer of the system. Unit testing verified each Servlet independently using predefined valid and invalid input sets. Functional testing confirmed that all specified operations—login, deposit, withdrawal, balance inquiry, account approval, account rejection, and transaction history retrieval—executed correctly under both valid and boundary-case inputs. Integration testing validated cross-module data consistency, specifically that credentials generated by the Admin module were immediately and correctly usable at the ATM portal without any manual data transfer. System testing ran complete end-to-end scenarios from new customer registration through to final transaction, confirming the integrity of the full workflow.

V. Results and Discussion

Eight test cases were defined to cover the critical functional paths of the system. Table IV presents the complete test inputs, expected outputs, and results.

TABLE IV
Functional Test Case Results

TC No.	Test Case	Input	Expected Output	Result
TC-01	ATM Login — Valid	Correct card no. and PIN	Dashboard displayed	Pass
TC-02	ATM Login — Wrong PIN	Correct card no., wrong PIN	Access denied, error shown	Pass
TC-03	Deposit funds	Deposit amount = ₹500	Balance increases by ₹500	Pass
TC-04	Withdrawal within balance	Withdraw ₹200, balance ₹500	Balance reduced to ₹300	Pass
TC-05	Withdrawal exceeds balance	Withdraw ₹600, balance ₹300	Insufficient funds error shown	Pass
TC-06	Manager approves account	Manager clicks Approve	Account no., ATM no., PIN auto-generated	Pass
TC-07	Manager rejects application	Manager clicks Reject	Status updated to Rejected	Pass
TC-08	View transaction history	User clicks View Transactions	All transactions listed with timestamps	Pass

All eight test cases produced the expected output without error. Three results are particularly significant from a design validation standpoint. TC-06 confirmed the automated account generation workflow operates correctly end to end: a single Approve action by the manager triggers server-side generation of a unique account number, ATM card number, and PIN, all of which are immediately active and encrypted in the database. This replaces a manual and error-prone process in the existing system.

TC-05 confirmed that the withdrawal validation guard functions correctly under boundary conditions. When a withdrawal request exceeds the available balance, the system returns an insufficient funds error and leaves the account balance unchanged. In the existing system, this check is not enforced at the application layer, making the proposed system meaningfully more reliable under invalid input.

AES encryption was independently verified by directly querying the MySQL database after account creation: stored card numbers and PINs were confirmed to be unreadable ciphertext, validating that the data-layer security objective has been

met. Collectively, the test results confirm measurable improvements across all four design dimensions: data security, administrative control, user experience, and transaction accountability.

The system is designed as a simulation and training environment and does not connect to live banking APIs or process real monetary transactions. This constitutes the primary limitation of the current implementation. Biometric authentication, real-time SMS and email transaction alerts, multi-language interface support, and integration with external banking APIs are identified as the primary directions for future development.

VI. Conclusion

This paper proposed and implemented an enhanced Bank ATM Simulation System that targets the security and usability shortcomings of conventional ATM simulation platforms. By integrating AES encryption for at-rest credential protection, Multi-Factor Authentication for access control, an automated account generation workflow, and a comprehensive timestamped transaction audit log, the system raises the standard of what a banking simulation environment should deliver.

Functional testing across eight defined test cases confirmed correct behaviour under valid inputs, invalid inputs, and boundary conditions. The structured feature comparison in Table I demonstrates improvement across seven independent dimensions relative to the existing baseline. The system is ready to serve as both a practical educational tool and a development foundation for more advanced banking application research. Future work will prioritise biometric authentication integration, real-time notification services, and multi-language support.

Acknowledgment

The authors sincerely thank Mr. D. Vijay Kumar, Assistant Professor, Department of Computer Science and Engineering, Guru Nanak Institute of Technology, Hyderabad, for his constant guidance and supervision throughout this work. The authors also express their gratitude to Dr. B. Santhosh Kumar, Head of Department, for his encouragement and institutional support.

References

[1] P. Sharma and R. Mehta, “Secure ATM Transactions Using Biometric Authentication,” International Journal of Computer Applications, vol. 174, no. 3, pp. 22–27, Jan. 2021.

[2] A. Safarzadeh, M. R. Jamali, and B. Moshiri, “Enhancing Precision of Automated Teller Machines Network Quality Assessment: Machine Learning and Multi Classifier Fusion Approaches,” arXiv preprint, arXiv:2501.01067, Jan. 2025.

[3] B. Karadag, A. Akbulut, and A. H. Zaim, “A review on blockchain applications in fintech ecosystem,” in Proc. 2022 Int. Congress on Human–Computer Interaction, Optimization and Robotic Applications (HORA), Ankara, Turkey, Jun. 2022, pp. 1–8, doi: 10.1109/HORA55278.2022.9799995.

[4] S. Kubendran and K. Sivakumar, “Improving ATM System Efficiency: A Simulation-Based Analysis Using AnyLogic,” International Journal of Advanced Research in Computer Science, vol. 15, no. 1, pp. 45–52, Jan. 2024.

[5] D. Rohitchandran, B. Santhoshkumar, and M. Kumar, “Bank records storage system through blockchain,” in Proc. Int. Conf. Inventive Comput. Technol. (ICICT), Lalitpur, Nepal, Apr. 2023, pp. 1178–1181, doi: 10.1109/ICICT57646.2023.10134368.