RES MILITARIS
Social Science Journal

Flask Server-based ATM Model with Fingerprint Access: A Desktop Application

K. Sangeetha[1], Nishitha Maramreddy[2], Ravi Shekar Potharlanka[2], Jayanth Reddy Addela[2]

[1]Assistant Professor, [2]UG Scholar, [1,2]Department of Computer Science and Engineering
[1,2]Kommuri Pratap Reddy Institute of Technology, Ghatkesar, Hyderabad, Telangana.

ABSTRACT

The use of biometric authentication methods, especially fingerprint recognition, has become increasingly popular in various applications, including secure access systems like ATMs. Traditional ATMs use a combination of physical cards and PINs for user authentication. Users insert their ATM cards and enter a personal identification number (PIN) to access their accounts, make transactions, and withdraw cash. However, this method has security limitations, such as the risk of PIN theft or card skimming. On the other hand, these traditional systems were vulnerable to security breaches. Thus, implementing fingerprint access enhances security significantly, ensuring that only authorized users can access their accounts and integrating this technology into desktop-based ATMs requires a robust and scalable framework, making Flask an ideal choice due to its versatility and reliability. Flask, a high-level Python web framework, provides a robust foundation for developing complex applications efficiently. The framework is open-source and has a large, active community that continually contributes to its growth and improvement. The need for a Fingerprint Access based ATM system arises from the increasing demand for enhanced security measures in financial transactions. Biometric authentication, particularly fingerprint recognition, provides a highly secure and convenient way for users to access their accounts. By implementing this technology in ATMs, financial institutions can significantly reduce the risk of unauthorized access and fraudulent activities.

Keywords: Web framework, HTML coding, Biometric system, Fingerprint recognition.

1. INTRODUCTION

Rapid development of banking technology has changed the way banking activities are dealt with. One banking technology that has impacted positively and negatively to banking activities and transactions is the advent of automated teller machine (ATM). With an ATM, a customer is able to conduct several banking activities such as cash withdrawal, money transfer, paying phone and electricity bills beyond official hours and physical interaction with bank staff. In a nutshell, ATM provides customers a quick and convenient way to access their bank accounts and to conduct financial transactions. Personal identification number (PIN) or password is one important aspect in ATM security system. PIN or password is commonly used to secure and protect financial information of customers from unauthorized access [1]. An ATM (known by other names such as automated banking machine, cashpoint, cash machine or a hole in the wall) is a mechanical system that has its roots embedded in the accounts and records of a banking institution [1]-[2]. It is a computerized machine designed to dispense cash to bank customers without need of human interaction; it can transfer money between bank accounts and provide other basic financial services such as balance enquiries, mini statement, withdrawal and fast cash among others [3]. The integration of biometric technology, specifically fingerprint recognition, into automated teller machines (ATMs) represents a significant leap forward in the realm of secure and efficient financial transactions. Traditional ATM systems, reliant on PIN codes and magnetic stripe cards, are facing increasing challenges related to security breaches, theft, and unauthorized access. In response to

these issues, this project aims to develop a cutting-edge desktop application for a fingerprint access-based ATM, leveraging the Django web framework for robust backend development.

Fingerprint Technology and Security

Fingerprint recognition, a form of biometric authentication, has gained prominence due to its unparalleled accuracy and difficulty to forge. Each person possesses a unique fingerprint pattern, making it an ideal candidate for secure identity verification. By integrating this technology into ATMs, we aim to enhance the security of financial transactions and mitigate the risks associated with traditional authentication methods.

The Role of Django Framework

Django, a high-level Python web framework, provides an excellent foundation for building secure and scalable web applications. Leveraging Django's features, such as its powerful ORM system, templating engine, and built-in security measures, we can create a robust backend for our ATM application. This integration not only ensures the security of user data but also facilitates the seamless development of user interfaces and business logic.

2. LITERATURE SURVEY

Crime at ATMs has become a nationwide issue that faces not only customers, but also bank operators and this financial crime case rises repeatedly in recent years [4]. A lot of criminals' tampers with the ATM terminal and steal customers' card details by illegal means. Once users' bank card is lost and the password is stolen, the users' account is vulnerable to attack. Traditional ATM systems authenticate generally by using a card (credit, debit, or smart) and a password or PIN which no doubt has some defects [5]. The prevailing techniques of user authentication, which involves the use of either passwords and user IDs (identifiers), or identification cards and PINs (personal identification numbers), suffer from several limitations [6]. Passwords and PINs can be illicitly acquired by direct covert observation. When credit and ATM cards are lost or stolen, an unauthorized user can often come up with the correct personal codes.

Despite warning, many people continue to choose easily guessed PIN's and passwords - birthdays, phone numbers and social security numbers. Recent cases of identity theft have heightened the need for methods to prove that someone is truly who he/she claims to be. Biometric authentication technology may solve this problem since a person's biometric data is undeniably connected to its owner, is nontransferable and unique for every individual. The system can compare scans to records stored in a central or local database or even on a smart card.

Biometrics can be defined as a measurable physiological and behavioral characteristic that can be captured and subsequently compared with another instance at the time of verification. It is automated methods of recognizing a person based on a physiological or behavioral characteristic [7]. It is a measure of an individual's unique physical or behavioral characteristics to recognize or authenticate its identity [8]. Common physical biometrics characteristics include fingerprint, hand or palm geometry, retina, iris and face while popular behavioral characteristics are signature and voice. Biometrics technologies are a secure means of authentication because biometrics data are unique, cannot be shared, cannot be copied and cannot be lost.

3. EXISTING SYSTEM

Automated Teller Machines (ATMs) have been pivotal in providing convenient access to financial services. Traditionally, ATM authentication relied on a combination of Personal Identification Numbers

(PINs) and magnetic stripe cards. While these methods have been widely adopted, they come with inherent limitations that compromise their effectiveness in the face of evolving security threats.

3.1 Personal Identification Numbers (PINs)

It is a numeric code typically consisting of four to six digits. It serves as a secret password that users input to authenticate themselves during ATM transactions.

Limitations:

- Vulnerability to Theft: PINs are susceptible to theft, either through shoulder surfing (someone observing the user entering the PIN) or by criminals installing hidden cameras or skimming devices on the ATM.
- Limited Complexity: Traditional PINs are often short and lack complexity, making them more susceptible to brute-force attacks where attackers systematically attempt all possible combinations.
- Difficulty of Memorization: Users may choose simple and easily memorable PINs, which increases the risk of unauthorized access. At the same time, complex PINs may be challenging for some users to remember.

3.2 Magnetic Stripe Cards

Magnetic stripe cards store account information in a magnetic stripe on the back of the card. Users insert the card into the ATM, and the machine reads the encoded data to identify the account and authorize transactions.

Limitations:

- Skimming and Cloning: Magnetic stripe cards are susceptible to skimming, where criminals install devices on ATMs to capture the card's magnetic stripe information. This information can then be used to create clone cards.
- Static Information: The data on the magnetic stripe is static and unchanging. Once captured, it remains the same, making it easier for attackers to reuse the information for fraudulent transactions.
- Global Standardization: The global reliance on magnetic stripe cards as a standard introduces uniform vulnerabilities, and any compromise in one part of the world can impact users globally.

3.3 Combined Limitations

Two-Factor Authentication Gap: Traditional ATM systems often rely solely on the combination of a card and a PIN, which falls short of modern two-factor authentication standards. Two-factor authentication combines something the user knows (PIN) with something the user has (card or device).

Dependency on Physical Elements: Both PINs and magnetic stripe cards depend on physical elements that can be lost, stolen, or damaged. This dependency poses a risk to users and introduces inconveniences such as card replacement processes.

4. PROPOSED METHODOLOGY

The limitations of traditional ATM authentication techniques highlight the need for more advanced and secure methods. The rise of biometric authentication, particularly fingerprint recognition, represents a paradigm shift towards more robust and user-friendly security measures. By addressing the vulnerabilities associated with PINs and magnetic stripe cards, the proposed project seeks to enhance

the security and overall user experience in the realm of ATM transactions. Therefore, this project implements a Flask web application for a simple ATM system. The application incorporates features such as user login, signup, deposit, withdrawal, and viewing account balance. It utilizes a MySQL database to store user information and transaction details. Additionally, the application includes functionality to handle user authentication using a combination of username, password, and fingerprint data.

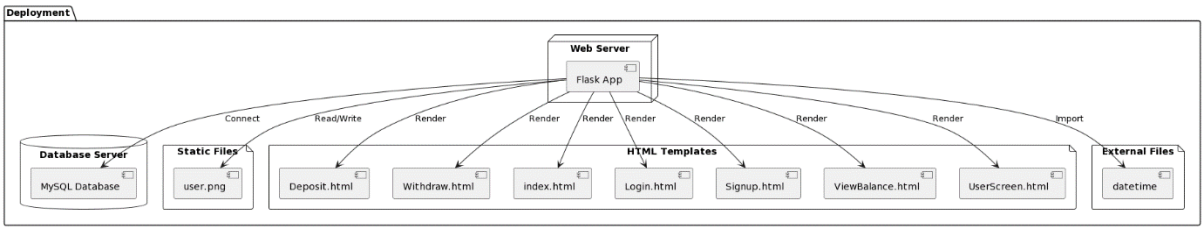


Figure 4.1: Overall design of proposed fingerprint-based ATM system.

4.1 Overview

This project represents the backend logic of a simple ATM web application built using the Flask framework. This web application includes features such as user registration, login, deposit, withdrawal, and the ability to view account balances. Notably, it integrates biometric authentication in the form of fingerprint data for user login. Below are the key components and an overview of the project:

1. Web Framework: The project utilizes the Flask web framework, a lightweight and modular framework for building web applications in Python.

2. Biometric Authentication: The application incorporates fingerprint data for user authentication, enhancing security beyond traditional username and password methods.

3. Database Interaction: The application interacts with a MySQL database to store user information, including usernames, passwords, contact details, and transaction history.

4. Functionalities:

- User Login and Authentication: The Login and LoginAction routes handle user login functionality. The user provides a username, password, and fingerprint data for authentication. Successful login sets a global variable uname to store the current user's username.
- User Registration: The Signup and SignupAction routes handle user registration. Users provide details such as username, password, contact number, email, address, gender, and fingerprint data during registration.
- Deposit and Withdrawal: The Deposit and DepositAction routes manage deposit functionality. Users can deposit funds into their accounts. The Withdraw and WithdrawAction routes handle user withdrawals. These routes involve updating the transaction history and the total balance in the database.
- View Account Balance: The ViewBalance route retrieves and displays the transaction history and account balance for the currently logged-in user.
- Logout: The Logout route logs the user out of the system.

5. Database Structure: The MySQL database contains tables such as users to store user information and transaction to record transaction history, including details like username, transaction amount, transaction type, transaction date, and total balance.

6. File Management: The application reads and writes fingerprint data from/to files in the static/users directory.

8. Global Variables: The use of global variables like uname to store the current user's username is employed for tracking user sessions.

9. Flask App Initialization and Run:

- The Flask application is initialized with a secret key for session management.
- The application runs when the script is executed.

4.2 Flask: A Micro Web Framework

Flask stands out as a lightweight and versatile web framework designed for Python developers seeking simplicity, ease of use, and extensibility in their web application projects. Developed by Armin Ronacher, Flask follows the micro-framework philosophy, providing the essentials required for building web applications without imposing a rigid structure or unnecessary dependencies. This approach allows developers the freedom to make decisions about project architecture, database choices, and other components, promoting flexibility in the development process. One of Flask's key features is its intuitive routing system, where URL patterns are easily mapped to Python functions using decorators. This simplicity makes it effortless to handle different HTTP methods and define routes, contributing to a clean and concise codebase. Flask incorporates the Jinja2 template engine, enabling the separation of logic and presentation in HTML templates. This empowers developers to render dynamic content seamlessly, enhancing the maintainability and readability of their code. Handling HTTP requests is straightforward with Flask, providing easy access to incoming request data and form submissions. Request and session objects simplify the management of client data, while decorators allow the creation of middleware functions to preprocess or manipulate requests. Flask's support for building RESTful APIs further extends its utility, offering features like URL parameters and the ability to return JSON responses. This flexibility makes Flask suitable for developing both traditional web applications and modern web services.

4. RESULTS AND DISCUSSION

To implement this project, we have designed following modules

- Signup: using this module user can sign up with the application by using username, password and finger print image. All signup details will be saved in MYSQL database
- Login: using this module user can login to application by entering username, password and finger print image given at signup time to authenticate himself
- Deposit: after successful authentication user can deposit amount and it will add to his account
- Withdraw: using this user can withdraw amount if sufficient balance available
- View Balance: using this module user can view available balance

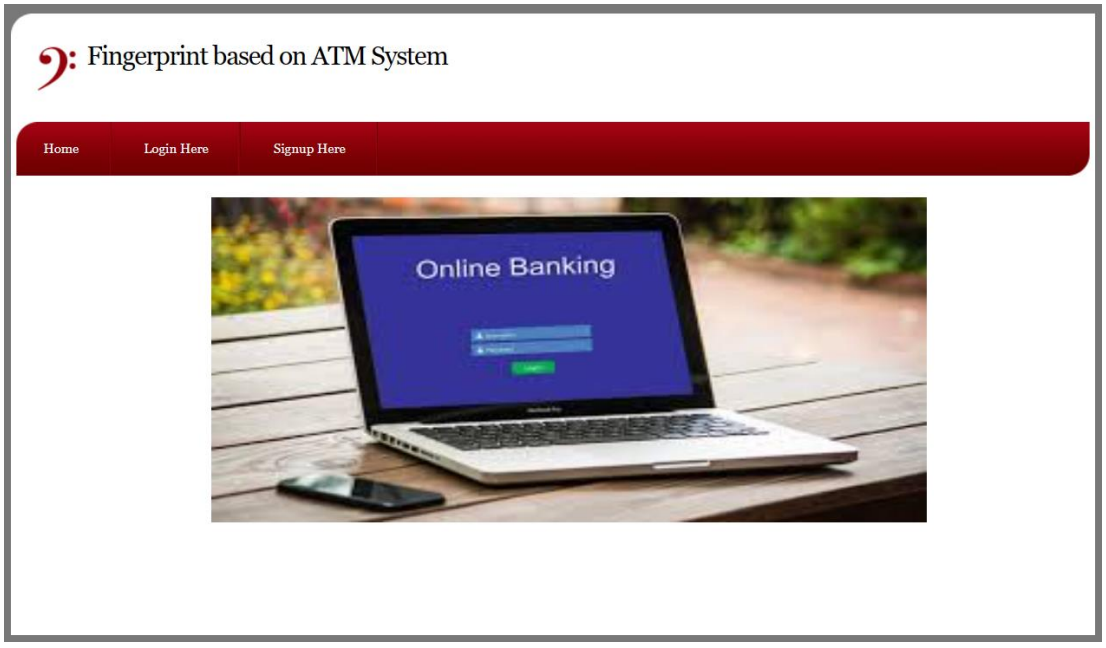


Figure 2: Web application of proposed fingerprint-based ATM system.

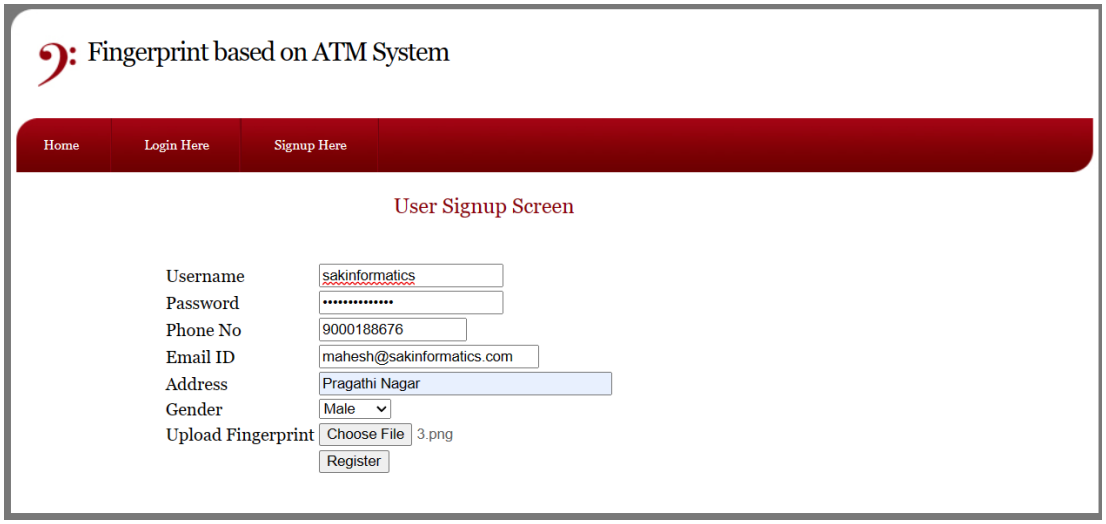


Figure 3: User signup screen.

Figure 2 represents a screenshot of the web application's main interface. It includes elements such as navigation menus, user account information, and options for performing various actions within the fingerprint-based ATM system. Figure 3 displays the user signup screen of the web application. It typically includes form fields for the user to enter information such as username, password, contact details, email, and address. The signup screen allows new users to register for the ATM system. Figure 4 indicates the successful completion of the user signup process. It displays a confirmation message or direct the user to a new screen, acknowledging that the registration was successful. This step confirms that the user has been added to the system.

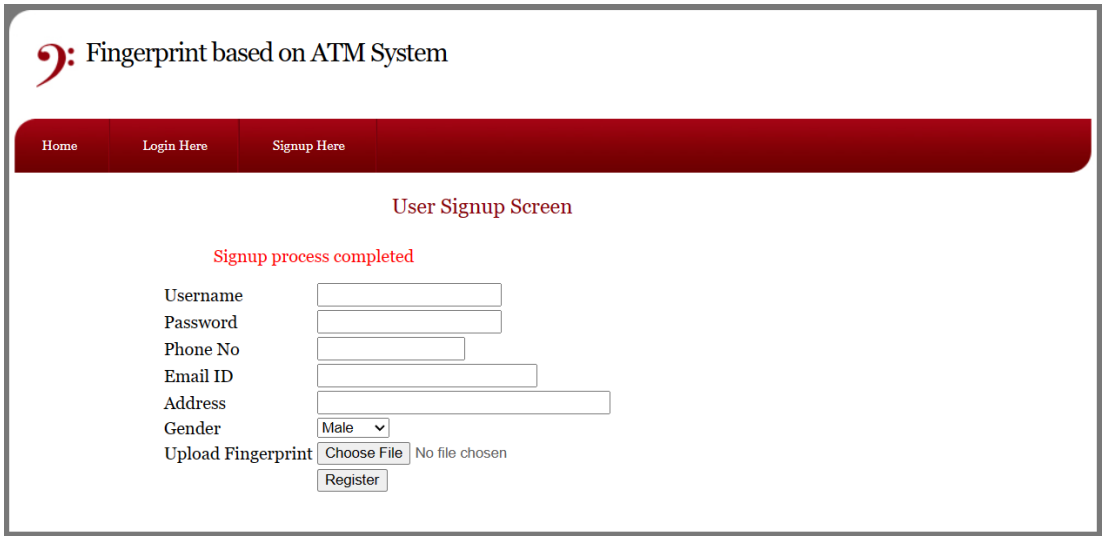


Figure 4: Sign up process completed.

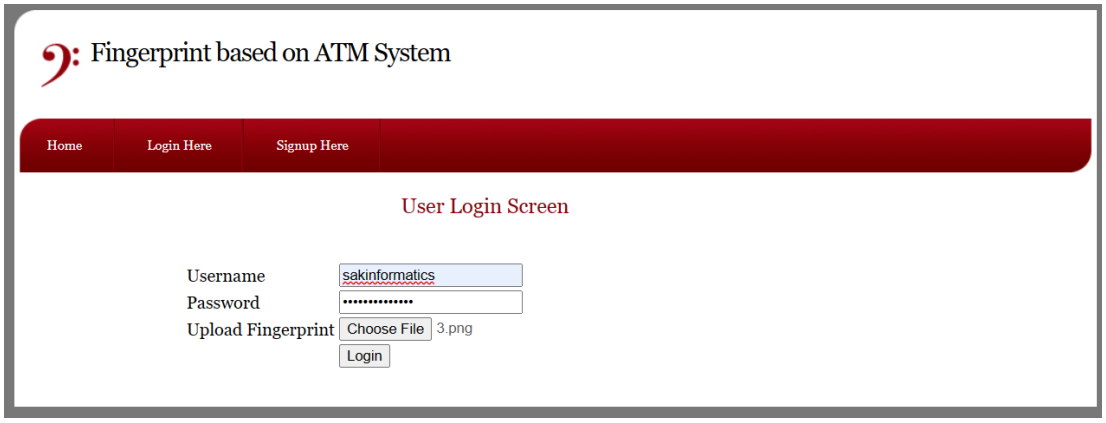


Figure 5: User login screen with biometric registration.

Figure 5 is the user login screen, as depicted in this figure, it includes fields for the user to enter their login credentials, such as username and password. Additionally, it has another filed i.e., upload fingerprint option for users to authenticate using fingerprint data as part of the login process. Figure 6 represents a page that appears after a successful user login. It displays options for various transactions, including deposit, withdrawal, view balance, and logout. Users can navigate to these options to perform different actions within their accounts. Figure 7 depicts screens or pages for specific operations within the ATM system. For example, there are separate screens for depositing an amount, viewing the account balance, withdrawing funds, and again viewing the updated balance. Each screen is designed to guide the user through specific financial transactions.

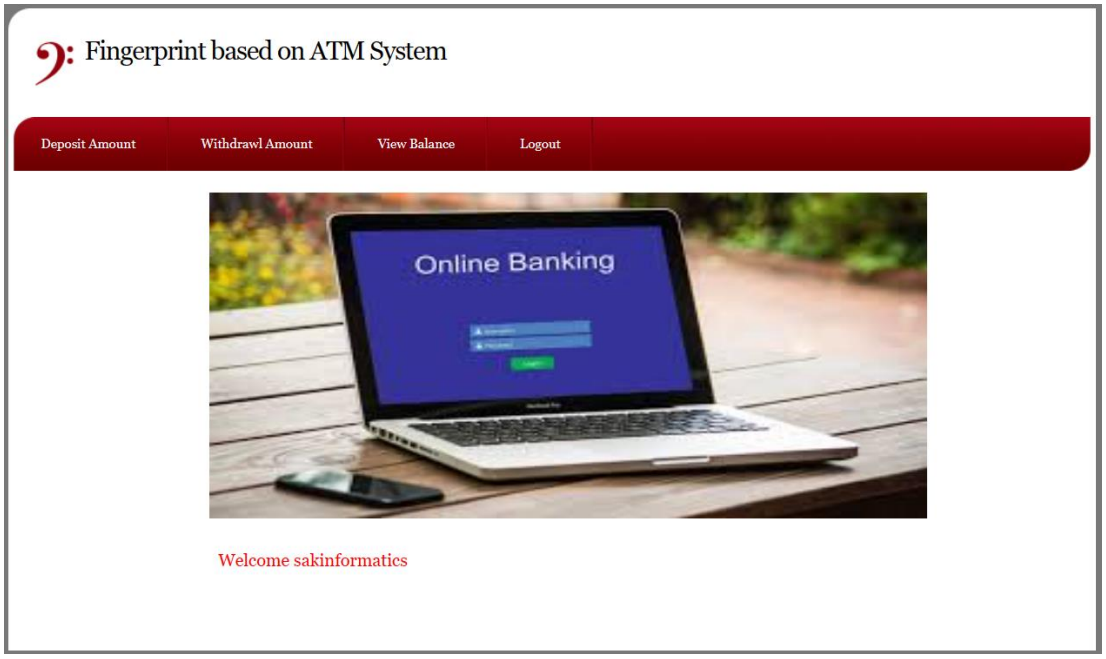
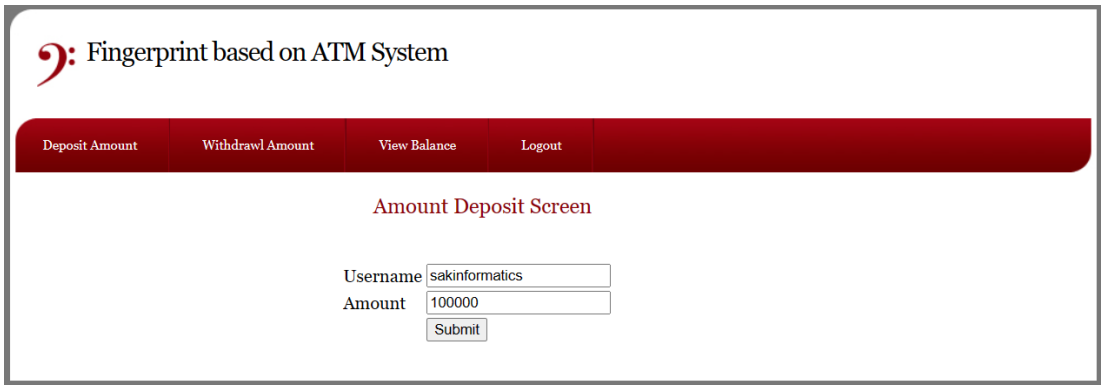
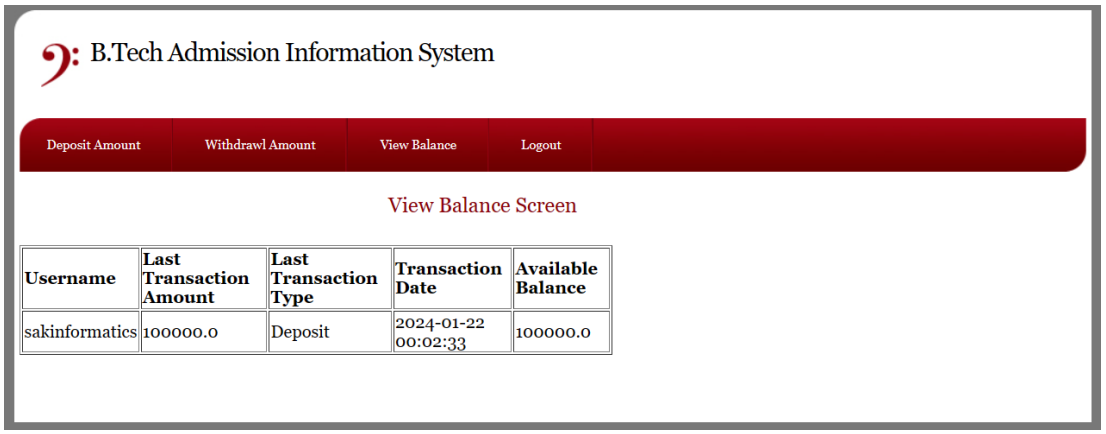


Figure 6: User login successful page showing deposit, with drawl, view balance and logout options.





Username	Last Transaction Amount	Last Transaction Type	Transaction Date	Available Balance
sakinformatics	100000.0	Deposit	2024-01-22 00:02:33	100000.0

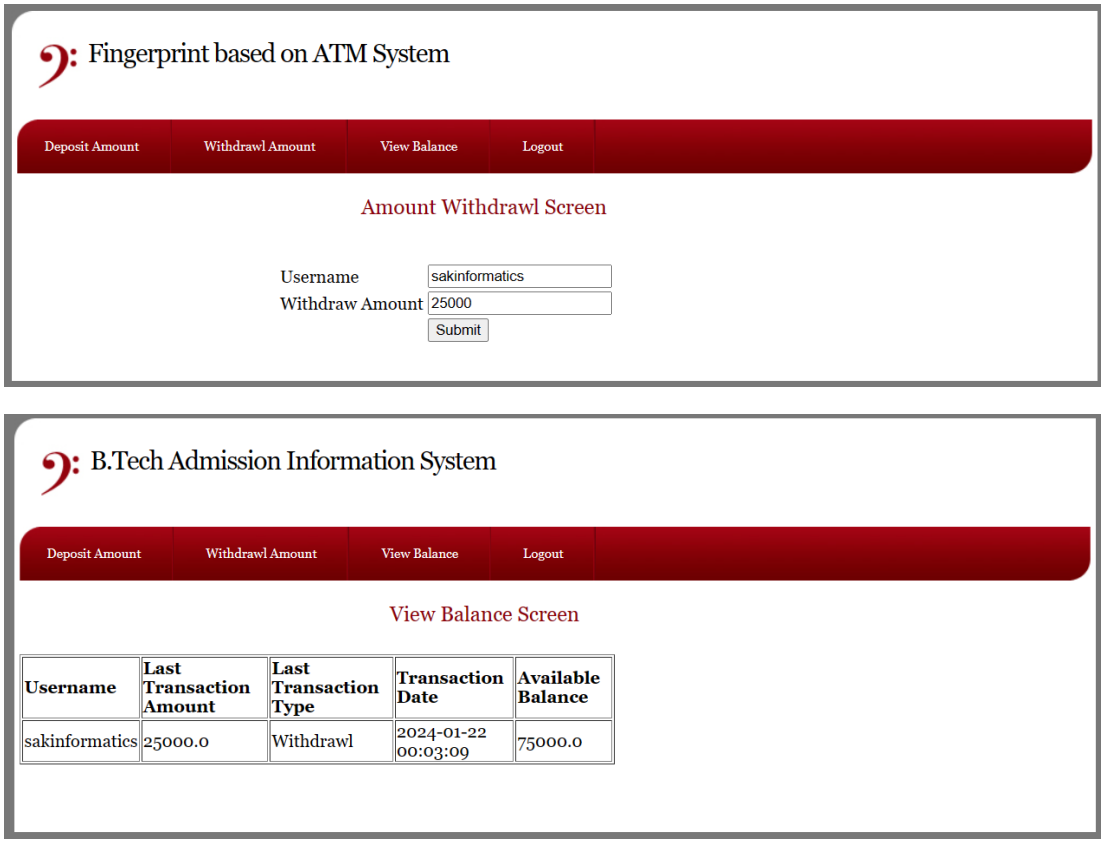


Figure 7: Amount deposit, view balance, amount with drawl and view balance operation screens.

5. CONCLUSION

This research outlines the development of a web-based ATM system, leveraging the Flask web framework and incorporating biometric authentication through fingerprint data. The core functionalities include user registration, login, deposit, withdrawal, and the ability to view account balances. A MySQL database is employed to store user information and transaction details, fostering a structured and organized data management system. The utilization of Flask allows for a modular and scalable web application, providing a user-friendly interface for managing financial transactions. The integration of biometric authentication enhances the security of the system, moving beyond traditional username and password-based methods. The "users" and "transaction" tables in the MySQL database facilitate the storage of user details and transaction records, respectively, creating a robust foundation for tracking financial activities. However, it is essential to note certain aspects for improvement. The code lacks certain security measures crucial for a production-level financial application. Future development should focus on implementing advanced security practices, including password hashing, secure session management, and additional layers of encryption to safeguard sensitive user information and financial transactions.

REFERENCES

[1] S.S, Das and J. Debbarma, “Designing a Biometric Strategy (Fingerprint) Measure for Enhancing ATM Security in Indian e-banking System”, International Journal of Information and Communication Technology Research, vol.1, no. 5, pp.197-203, 2011.

[2] W. W. N. Wan, C. L. Luk, and C.W.C. Chow, “Customers Adoption of Banking Channels in Hong Kong”, International Journal of Bank Marketing, vol. 23, no. 3, pp. 255-272, 2005.

[3] Wikipedia the free encyclopaedia, "Biometrics", Downloaded January 14, 2024 from http://en.wikipedia.org/wiki/Biometrics.
[4] B. Richard and M. Alemayehu, "Developing E-banking Capabilities in a Ghanaian Bank: Preliminary Lessons. Journal of Internet Banking and Commerce, vol. 11, no. 2, 2006.
[5] P. K. Amurthy and M.S. Redddy, "Implementation of ATM Security by Using Fingerprint recognition and GSM", International Journal of Electronics Communication and Computer Engineering vol.3, no. 1, pp. 83-86, 2012.
[6] N. K. Ratha, J.H. Connell, and R.M. Bolle, "Enhancing Security and Privacy in Biometrics-based Authentication Systems, IBM Systems Journal, vol. 40, no. 3, pp. 614-634, 2001.
[7] N. K. Ratha, S. Chikkerur, J.H. Connell and R.M. Bolle. "Generating Cancelable Fingerprint Templates", IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 29, no. 4, 2007.
[8] B. Schouten and B. Jacobs, "Biometrics and their use in e-passport", Image and Vision Computing vol. 27, pp. 305–312. 2009.
[9] S. A. Shaikh and J. R. Rabaiotti, "Characteristic trade-offs in designing large-scale biometric-based identity management systems". Journal of Network and Computer Applications vol. 33, pp. 342–351, 2010.
[10] Christiawan, B. A. Sahar, A. F. Rahardian and E. Muchtar, "Fingershield ATM – ATM Security System using Fingerprint Authentication," 2018 International Symposium on Electronics and Smart Devices (ISESD), Bandung, Indonesia, 2018, pp. 1-6, doi: 10.1109/ISESD.2018.8605473.
[11] D. Anveshini, V. Revathi, A. Eswari, P. Mounika, K. Meghana and D. Aparna, "Pattern Recognition based Fingerprint Authentication for ATM System," 2022 International Conference on Electronics and Renewable Systems (ICEARS), Tuticorin, India, 2022, pp. 1708-1713, doi: 10.1109/ICEARS53579.2022.9751966.
[12] M. N. Narsaiah, G. Lasya, K. Veronica, Abhilash, P. S. Kirthana, M. S. Kumari and A. Pathani, "Fingerprint Recognition for Future ATM Security", E3S Web Conf., 430 (2023) 01167, DOI: https://doi.org/10.1051/e3sconf/202343001167
[13] Website: Biometric ATM: The Future of Secure Financial Transactions (aratek.co), Accessed on 14th January 2024.
[14] Jubayer Ahamed, Maliha Maisha, Zeba Labiba, Md. Ariful Islam, and Dip Nandi. 2022. A review report on the fingerprint-based biometric system in ATM banking. In Proceedings of the 2nd International Conference on Computing Advancements (ICCA '22). Association for Computing Machinery, New York, NY, USA, 522–529. https://doi.org/10.1145/3542954.3543029