Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 2, No. 4, February : 2024

BANKING SIGNATURE MATCH VERIFIER: ENSURING SECURE TRANSACTIONS WITH SIGNATURE VERIFICATION USING PYTHON

Dr. K.Soumya, Guide, Department of Computer Science, Andhra University College of Engineering for Women, Visakhapatnam, 530017, India Email: soumyacf@andhrauniversity.edu.in
Bongi Sai Geetha, Budda Dipika, **Byri Prasanthi, Chavali Lakshmi Gowri Veera Phani Sri, Devarapu Satyaveni,** Student, Department of Computer Science, Andhra University College of Engineering for Women, Visakhapatnam, 530017, India

1. ABSTRACT
The application allows users to register with their credentials including a phone number and signature image. Upon registration, users receive an OTP via SMS for phone number verification. After successful registration, users can log in using their credentials and upload an image containing their signature for verification. The system uses Twilio for phone number verification and OTP generation. It also utilizes CSV files to store user data including passwords, phone numbers, and signature image paths. Image comparison is performed to verify the uploaded signature against the registered signature image. Overall, the system provides basic banking functionalities with an emphasis on secure user authentication and signature verification.
Keywords: Handwriting Recognition, Signature Verification, Encryption for OTP Transmission, Feature Extraction for Signatures, Fraud Detection in Signature Verification

2. INTRODUCTION
The Banking Signature Match Verifier project, built on the Flask framework, represents a sophisticated web application tailored for signature verification through image analysis. Leveraging Twilio's SMS services [4], the system implements robust phone number verification mechanisms to bolster user authentication and ensure the integrity of the verification process. Beyond mere verification, the platform offers a comprehensive suite of functionalities including user registration, login capabilities, and the ability to upload signature images for comparative analysis.

This enables users to securely manage their accounts and perform signature verifications with ease. Central to the project's security architecture is the generation and verification of one-time passwords (OTPs), serving as an additional layer of authentication to fortify user access control. By implementing OTP-based authentication, the system enhances security and safeguards against unauthorized access.

Moreover, the application maintains detailed statistics for both daily and monthly signature comparisons [6]. This feature not only enhances security by tracking user activity but also provides valuable insights into usage patterns, aiding in performance optimization and user behaviour analysis. The project also demonstrates robust file upload handling capabilities, efficient session management functionalities, and comprehensive error handling mechanisms. These features collectively contribute to the system's reliability, usability, and resilience against potential vulnerabilities.

In summary, the Banking Signature Match Verifier project exemplifies a sophisticated web application that combines advanced image analysis techniques, secure authentication mechanisms, and comprehensive user management functionalities to deliver a seamless and secure platform for signature verification.

3. SYSTEM ARCHITECTURE AND REQUIREMENTS
3.1 SYSTEM OVERVIEW:
The Signature Match Verifier system is a web-based application designed to authenticate users through signature image comparison. Users register and upload their signature images, which are stored securely in a database. Twilio integration [4] enables phone number verification via SMS for user authentication. The system compares uploaded signature images using an image comparison service to

determine match percentages. Error handling and security measures are implemented to ensure reliable and secure operation. Additionally, the system maintains statistics on verification attempts for user management and audit purposes.

3.1.1 Frontend:

The front end of the application is developed using HTML, CSS, and JavaScript. It provides the user interface for interacting with the system, including registration, login, signature image upload, and result display.

3.1.2 Backend:

The backend is implemented using Python and the Flask web framework [3]. It handles user requests, processes data, interacts with the database, and communicates with external services such as Twilio for phone number verification and image comparison services.

3.1.3 Database:

The system utilizes a relational database management system (RDBMS) such as SQLite, MySQL, or PostgreSQL to store user data, signature images, and verification statistics. The database schema includes tables for user information, signature images, verification logs, and statistical data.

3.1.4 Twilio Integration:

Twilio's SMS services are integrated into the system for phone number verification using one-time passwords (OTPs). This integration requires a Twilio account SID, authentication token, and phone number.

3.1.5 Image Comparison Service:

The system incorporates an image comparison service to compare uploaded signature images for verification purposes. This service uses algorithms to analyse image similarity and determine the match percentage.

3.1.6 File Storage:

The system requires a file storage mechanism to store uploaded signature images securely. This could be implemented using a local file system or cloud storage services like Amazon S3 or Google Cloud Storage.

3.1.7 Security:

The system implements various security measures including encryption of sensitive data, secure communication using HTTPS, protection against SQL injection and cross-site scripting (XSS) attacks, and proper authentication and authorization mechanisms [5].

3.1.8 Error Handling:

Comprehensive error handling mechanisms are implemented to gracefully handle errors and exceptions, providing informative error messages to users and logging errors for system administrators.

3.1.9 Deployment:

The system is deployed on a web server with the necessary configurations to support Python, Flask, and other dependencies. Deployment options include traditional web hosting providers, cloud platforms like AWS or Azure, or containerized deployments using Docker.

3.1.10 Scalability:

The system architecture is designed to be scalable to accommodate growing user loads and data volumes. This may involve horizontal scaling by adding more server instances or vertical scaling by upgrading server resources.

3.2 SYSTEM MODULES

The functions of different modules are:

3.2.1 User Authentication Module:

Handles user registration, login, and authentication processes using one-time passwords (OTPs) sent via SMS [2].

3.2.2 Signature Upload Module:

Allows users to upload their signature images for comparison and verification.

3.2.3 Image Comparison Module:
Compares uploaded signature images with registered ones using image processing algorithms to determine a match or mismatch [4].

3.2.4 Match Statistics Module:
Maintains records of daily and monthly match statistics for each user, providing insights into their signature verification history.

3.2.5 Twilio Integration Module:
Integrates Twilio API for phone number verification via SMS, ensuring secure user authentication [2].

3.2.6 Error Handling Module:
Manages error handling throughout the system, providing appropriate feedback to users in case of any issues or discrepancies.

3.2.7 User Interface Module:
Provides a user-friendly interface for interacting with the system, including web pages for registration, login, signature upload, and result display.

3.2.8 Session Management Module:
Manages user sessions to maintain state and ensure secure interactions between the user and the system.

3.3 SYSTEM REQUIREMENTS

3.3.1 Software Requirements:
Python
Flask
Twilio API
CSV Module
OpenCV
NumPy
PIL

3.3.2 Hardware Requirements:
CPU
GPU
RAM
Storage

3.4 IMPLEMENTATION

The implementation of the Banking Signature Match Verifier involves developing a web application using Flask and integrating Twilio [4] for secure phone number verification via SMS. Users can register, login, and upload signature images for comparison, with one-time passwords (OTPs) used for authentication. The application maintains match statistics for daily and monthly comparisons [6], enhancing security. Additionally, it includes features for file upload handling, session management, and error handling to ensure a robust and user-friendly experience.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 2, No. 4, February : 2024

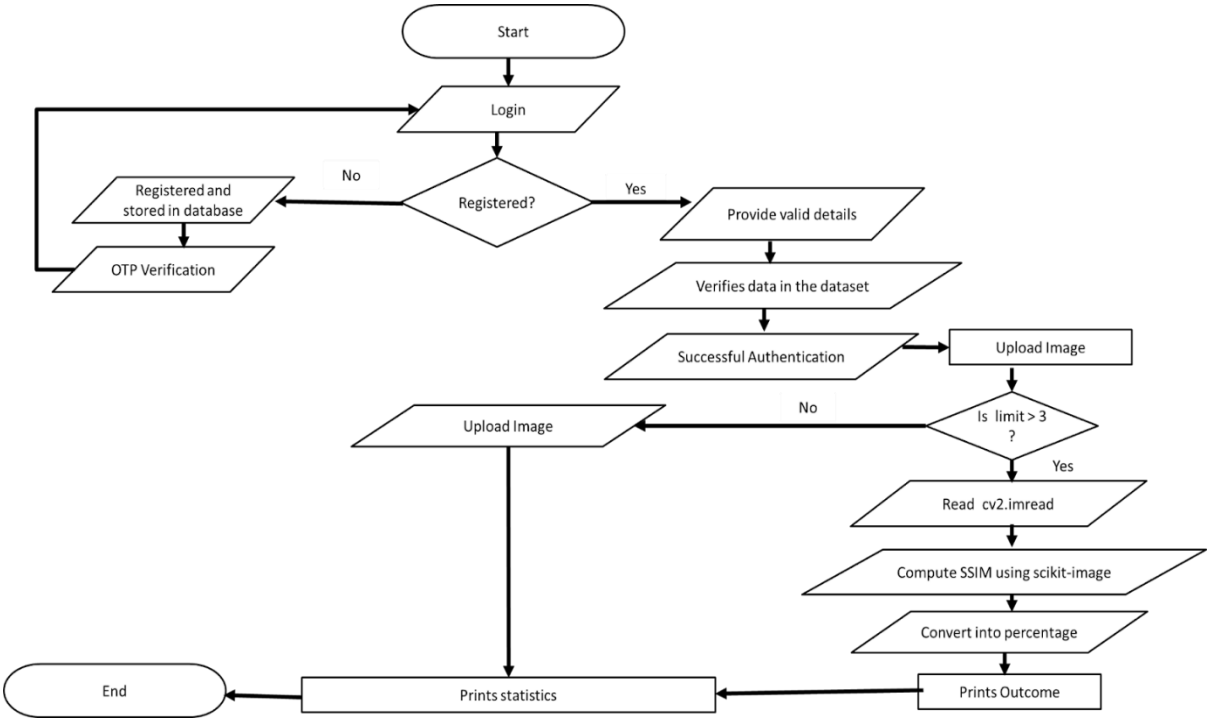

Fig. 1 Flowchart of Banking Signature Match Verifier

4. RESULTS:

The Banking Signature Match Verifier project is a web application developed using Flask, a Python web framework, to verify signatures through image comparison. It utilizes Twilio for phone number verification via SMS [4], ensuring secure user authentication. Users can register, login, and upload signature images for comparison. The application generates and verifies one-time passwords (OTPs) to authenticate users. It maintains match statistics for daily and monthly comparisons [6], enhancing security and user management. With features like file upload handling, session management, and error handling, the project provides a robust and user-friendly interface for signature verification.

WORKFLOW:

Step - 1: The user enters the Home page of the website that enables the user to navigate to the Register page and Login page as shown in Fig. 2

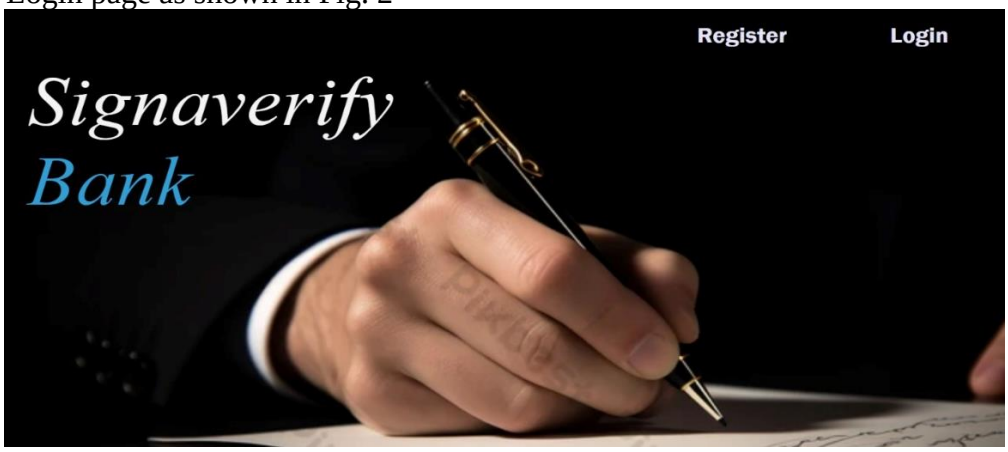

Fig. 2 Signature Match Verifier (Home Page of the Website)

Step - 2: The user enters the Registration Page that enables the user to enter the details like mobile number, password, Id and helps to upload the signature of the user as shown in Fig. 3

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 2, No. 4, February : 2024

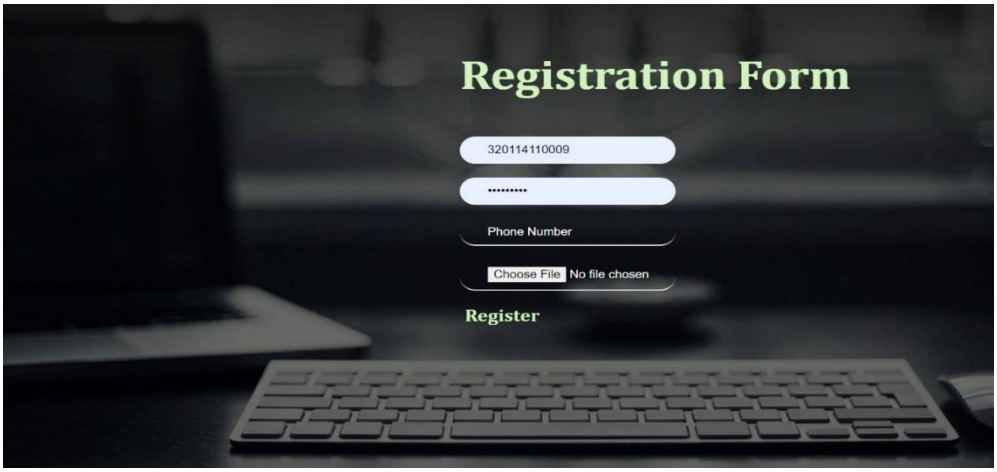

Fig. 3 User can Register through this page

Step - 3: The Login Page is used to help the user to log in to the system by validating the details provided by the user as shown in Fig. 4

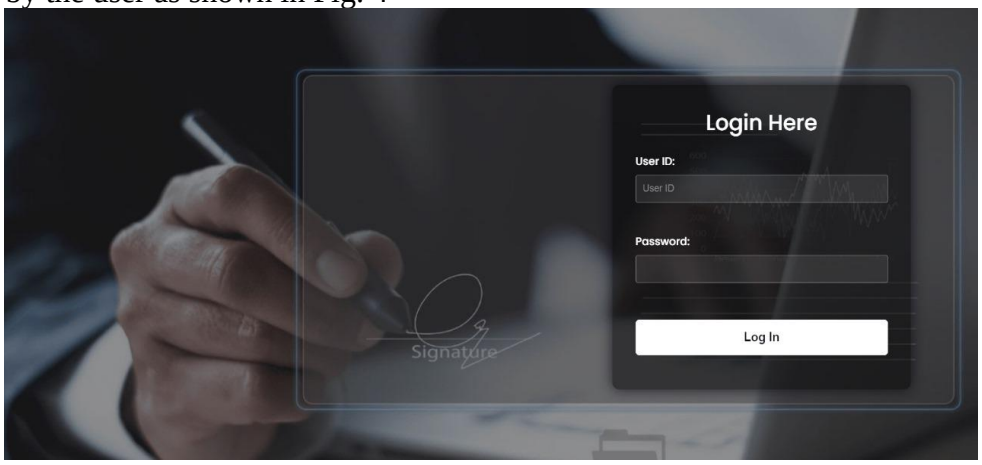

Fig. 4 User can Login to the Website

Step - 4: If Transaction amount is less than 1 Lakh then there will be no tax and signature check is done as shown in Fig. 5

Fig. 5 Signature Check without Tax

Step - 5: If Transaction amount is greater than 1 Lakh then there will be tax at the rate of 0.01 and signature check is done as shown in Fig. 6

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 2, No. 4, February : 2024

Fig. 6 Signature Check with Tax

Step - 6: After the evaluation and verification, the system checks daily and monthly matches and as shown in Fig. 7

Fig. 7 User can see their Monthly and Daily Signature Matches

Step - 7: The user can login for only three times to the system and if the user exceeds the limit, then there is an alert displayed on the screen as shown in Fig. 8

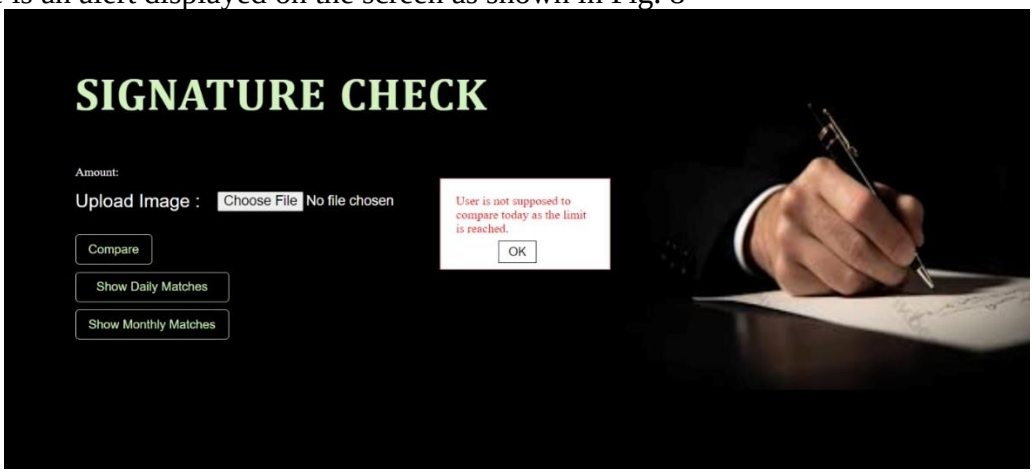

Fig. 8 User cannot verify their Signature image after limit is reached
(Alert message displayed)

5. FUTURE SCOPE:

5.1 Multi-factor Authentication (MFA):

Integrating additional authentication factors such as biometrics (face, iris, fingerprint), voice recognition, or hardware tokens can strengthen security and provide users with more options for authentication.

5.2 Machine Learning for Signature Recognition:
Implementing machine learning algorithms to improve signature recognition accuracy and handle variations in writing styles, enhancing the system's effectiveness in verifying signatures.

5.3 Mobile Application Development:
Creating a mobile application version of the system to provide users with more convenience and accessibility, allowing them to verify signatures and perform authentication tasks on their smartphones.

5.4 Enhanced Reporting and Analytics:
Developing advanced reporting and analytics features to provide administrators with insights into user authentication patterns, suspicious activities, and compliance metrics, enabling proactive security measures.

5.5 Integration with Document Management Systems:
Integrating the signature verification system with document management platforms to streamline document authentication processes and ensure the integrity of digitally signed documents.

5.6 Continuous Security Auditing:
Implementing automated security auditing mechanisms to regularly assess the system's security posture, detect vulnerabilities, and apply necessary security patches and updates.

5.8 Enhanced User Experience:
Focusing on improving the user interface, accessibility, and user experience to make the system more intuitive and user-friendly for both administrators and end-users.

6. LITERATURE ON REVIEW

The Signature Match Verifier project encompasses research articles, academic papers, and industry reports related to signature verification, biometric authentication, and secure user authentication methods. It delves into topics such as machine learning algorithms for signature recognition, advancements in biometric authentication technologies, and the integration of multi-factor authentication systems. Additionally, it explores the challenges and limitations of existing signature verification methods, including variability in signature styles and susceptibility to fraud. The review highlights the importance of robust authentication mechanisms in various sectors, including banking, finance, and digital identity management, emphasizing the need for reliable and secure signature verification solutions. Furthermore, it examines recent trends and innovations in the field, such as the adoption of blockchain technology for secure document authentication and the emergence of mobile-based authentication solutions. Through comprehensive analysis and synthesis of relevant literature, the review contributes to the understanding of key concepts, methodologies, and advancements in signature verification and user authentication, guiding the development and implementation of the Signature Match Verifier project.

7. CONCLUSION:

In conclusion, the Banking Signature Match Verifier project presents a comprehensive and sophisticated solution for signature verification through image analysis. Leveraging the Flask framework and integrating Twilio [4] for secure phone number verification, the system ensures robust user authentication and integrity of the verification process. With features such as user registration, login capabilities, signature image upload, and one-time password authentication, users can securely manage their accounts and perform signature verifications with ease. The project emphasizes security and user management, maintaining detailed statistics for daily and monthly comparisons [6] to track user activity and detect potential fraud.

References

[1] D. Bertolini, L. S. Oliveira, E. Justino, and R. Sabourin. "Reducing forgeries in writer-independent off-line signature verification through ensemble of classifiers." *Pattern Recognition*, 43(1), January 2010.

[2] H. Baltzakis and N. Papamarkos. "A new signature verification technique based on a two-stage neural network classifier." *Engineering Applications of Artificial Intelligence*, 14(1):95–103, February 2001.
[3] J. Coetzer, B.M. Herbst, J.A. du Preez. "Offline Signature Verification Using the Discrete Radon Transform and a Hidden Markov Model". *EURASIP Journal on Applied Signal Processing*, 2004(4):559–571, 2004.
[4] Luana Batista, Eric Granger, and Robert Sabourin. "Dynamic selection of generative–discriminative ensembles for off-line signature verification." *Pattern Recognition*, 45(4):1326–1340, April 2012.
[5] S. Sayeed, N.S. Kamel, R. Besar. "A Sensor-Based Approach for Dynamic Signature." *Add relevant publication details*. Put references
[6] W. Hou, X. Ye, K. Wang. "A Survey of Off-line Signature Verification". *Proceedings of the 2004 International Conference on intelligent Mechatronics and Automation*. Chengdu, China, 2004.
[7] Yoshua Bengio. "Learning Deep Architectures for AI." *Foundations and Trends in Machine Learning*, 2(1):1–127, January 2009.