International Journal of Advanced Research and Review

www.ijarr.org

Email Notification System using AWS SES

[1]Shaik Afrozah, [2]Talasila Jyothi, [3]Yetukuri Sradha, [4]Dr. P.V.S. Sarma

[1,2,3]U. G Student, Dept COMPUTER SCIENCE AND ENGINEERING, St.Ann's College Of Engineering and Technology, Nayunipalli (V), Vetapalem (M), Chirala, Bapatla Dist, Andhra Pradesh – 523187, India

[4]Associate professor, COMPUTER SCIENCE AND ENGINEERING, St.Ann's College Of Engineering and Technology, Nayunipalli (V), Vetapalem (M), Chirala, Bapatla Dist, Andhra Pradesh – 523187, India

ABSTRACT

The Email Notification System using AWS SES is a Flask-based web application that manages user registration, authentication, and email verification. Many platforms lack in security for verifying the emails of the users during the registration process, often leading to fake registration or unauthorized accounts. To address this issue, the Email Notification System using AWS SES provides a secure solution for the email verification and user authentication. This web application is developed using Python Flask, HTML and MongoDB integrated with Amazon Simple Email Service (AWS SES), to send reliable verification emails with unique token links. New users must verify their email before accessing the dashboard, ensuring security. Only verified users can access the dashboard.

KEY WORDS

Amazon Web Services (AWS), Simple Email Service (SES), Flask Framework, MongoDB, Email Verification, Dashboard Management.

INTRODUCTION

The Email Notification System with AWS SES is designed to securely handle user registration and verification in online applications. On many online systems, preventing false accounts and unauthorized access requires email validation and user authentication. The main programming language for this project is Python; the backend framework is Flask; and efficient data storage and retrieval is made possible by MongoDB. Through integration with Amazon Simple Email Service (AWS SES), the solution ensures reliable and scalable

distribution of verification emails to users. Before they may access the dashboard, each user who registers receives a unique verification link to activate their account. The system additionally has session management, password hashing, and a modular frontend design constructed with HTML and CSS to ensure security and usability. All things considered, this demonstrates how cloud-based email services may be combined with modern web technologies.

LITERATURE REVIEW

Recent years I explored some related papers to Email Verification. In that I have observed some limitations. To overcome them I studied some related papers. In that first paper is automated notification proposed by Balachander, but in this paper they have not explored about automated notification using emails. I also took another paper which focused on authentication protocols proposed by Sharma and Gupta, but they have not explored about user authentication. In the final paper I studied about Secure Email Service by J. H. Zeng, X. Li, and Q. Wang, but they have not explored about verification links. These limitations are solved by the proposed methods.

RELATED WORK

This project uses email verification using AWS Simple Email Service (SES) for sending reliable and cost-effective emails. MongoDB for storing the passwords of the users. Python Flask framework is used for building proof-of-concept applications due to its simplicity and modularity. For secure authentication, libraries like Werkzeug are commonly used to handle password hashing, ensuring user credentials are stored safely. Html, CSS are used for frontend. By combining web frameworks with cloud services to demonstrate end-to- end functionality such as registration, login, and secure email communication.

EXISTING METHOD

In traditional email systems, applications often rely on SMTP servers for sending emails, which can be difficult to configure, prone to deliverability issues, and less scalable when handling bulk or transactional messages. Similarly, user authentication in older systems is sometimes managed using plain-text credentials or weak encryption methods, leaving applications vulnerable to security breaches. Many existing solutions also rely on relational databases that require rigid schema definitions, making it harder to adapt to evolving user data requirements. Additionally, integrating these systems with cloud services can be complex, requiring significant manual setup and maintenance. As a result, existing systems may suffer from security limitations, poor scalability, and high maintenance.

PROPOSED METHOD

The proposed system is a Flask-based proof- of-concept email notification system that integrates AWS SES for reliable and scalable email delivery and MongoDB for flexible user management. Unlike traditional SMTP based systems, AWS SES provides high deliverability, reduced configuration and seamless cloud scalability. The system ensures secure authentication using hashed passwords with Werkzeug and employs session management to control access to protected routes such as the user dashboard. The verified users only can access the dashboard.

SYSTEM ARCHITECTURE

Fig-1: Block Diagram of Email Notification System using AWS SES

METHADOLOGY DESCRIPTION

Creating Identities in AWS SES: In Amazon Simple Email Service AWS SES, identities are email addresses or domains that you verify with SES to send or receive emails.

Verifying the Emails: To verify an email with AWS SES, open the verification email sent by AWS, click the link inside within 24 hours, and the email will be marked as "Verified" in the SES console.

User Registration: The user should register with their credentials.

Email Verification: Once the user registers a verification link is sent to their verified email. The user needs to verify the email by clicking on the link.

User Login: A user begins by entering their credentials, such as a username and password, on the login page.

Dashboard Access: After login is verified, the system grants access to the dashboard.

RESULTS AND DISCUSSION

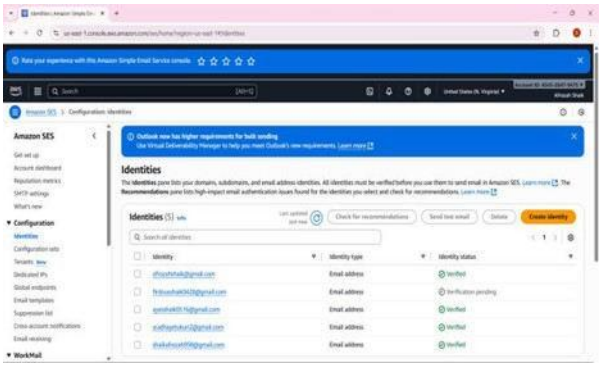

Fig 2: Creating Identities in AWS SES In Amazon Simple Email Service AWS SES, identities are email addresses or domains that you verify with SES to send or receive emails. These identities help AWS SES ensure you're authorized to use them and also help improve deliverability and trustworthiness.

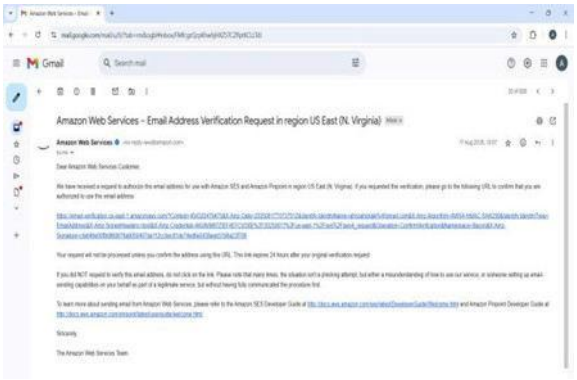

Fig-3: Verifying the email via link sent by AWS SES

To verify an email with AWS SES, open the verification email sent by AWS, click the link inside within 24 hours, and the email will be marked as "Verified" in the SES console.

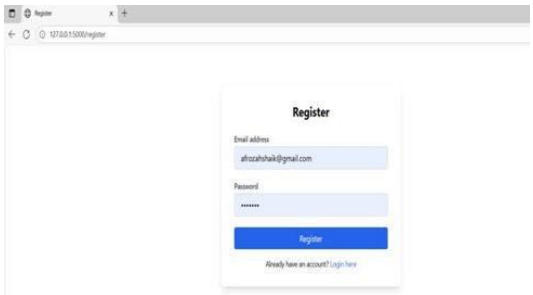

Fig-4: Registration Form

It shows a user registration interface from a web application running on the local server. This is typically part of a web-based authentication system where users are required to create an account before accessing the application's features.

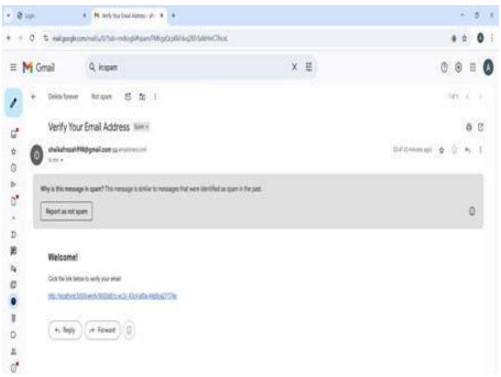

Fig-5: verifying the email

After registration a verification link is send to the registered email. The user need to click on the link to verify.

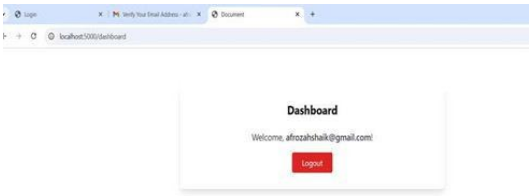

Fig-6: Access to Dashboard

A user begins by entering their credentials, such as a username and password, on the login page. Once verified, the system grants access to the dashboard. The dashboard acts as a central hub where all important features, tools, and information are displayed in an organized manner.

CONCLUSION

In conclusion, the project shows how login and dashboard systems play an important role in today's digital platforms. By combining security with ease of use, it ensures that only authorized users can access information while also providing a simple way to manage tasks. Such systems reflect the growing need for safe, efficient, and user-friendly technology in education, business, and daily life.

FUTURE SCOPE

The future scope of the Email Notification System using AWS SES includes integrating AI-driven personalization to enhance user engagement through context-aware and targeted email delivery. It can be expanded with analytics dashboards to monitor delivery rates, user interactions, and campaign effectiveness in real time. Additionally, incorporating multi-channel communication such as SMS or push notifications can make the system more versatile and efficient for enterprise-scale applications.

REFERENCES

[1] J. H. Zeng, X. Li, and Q. Wang, "Secure

Email Service Based on Cloud Computing,"
Procedia Engineering, vol. 15, pp. 3203–
3207, 2011.

[2] Surekha, G., Kumari, Y. S., & Harini, P. Cosdes: A Collaborative Spam Detection System with a Novel E-Mail Abstraction Scheme.

[3] L. Richardson and S. Ruby, *RESTful Web Services*, O'Reilly Media, 2008.

[4] A. Grinberg, *Flask Web Development: Developing Web Applications with Python*, 2nd ed., O'Reilly Media, 2018.

[5] K. Chodorow, *MongoDB: The Definitive Guide: Powerful and Scalable Data Storage*, 3rd ed., O'Reilly Media, 2019.

[6] Amazon Web Services, "Amazon Simple Email Service (SES) – Developer Guide," Amazon, 2023.

[7] M. Reitz and B. Schlusser, *The Hitchhiker's Guide to Python: Best Practices for Development*, O'Reilly Media, 2016.

[8] S. Garfinkel and G. Spafford, *Web Security, Privacy and Commerce*, 2nd ed., O'Reilly Media, 2002.

[9] U. A. Butt, R. Amin, H. Aldabbas, S. Mohan, B. Alouffi, and A. Ahmadian, "Cloud-based email phishing attack using machine and deep learning algorithm," *Applied Intelligence*, 2023.

[10] K. Shen, C. Wang, M. Guo, X. Zheng,
C. Lu, B. Liu et al., "Weak Links in Authentication Chains: A Large-scale Analysis of Email Sender Spoofing Attacks," *arXiv preprint*, 2020.

[11] I. Vazquez Sandoval, A. Atashpendar,
G. Lenzini, and P. Y. A. Ryan, "PakeMail: Authentication and Key Management in Decentralized Secure Email and Messaging via PAKE," *arXiv preprint*, 2021.

[12] Qxf2 Services, "Sending Email through Amazon SES with Flask App," *Qxf2 Blog*, 2023.

[13] L. Ghimpu, "Flask and Amazon SES: Build Your Own SendGrid," *Medium*, 2023. [14] "How to Use MongoDB in a Flask Application," *DigitalOcean Tutorials*, 2022. [15] Mailtrap.io, "Flask Email Verification Tutorial," *Mailtrap Blog*, 2023.

[16] A. Alshamrani and K. Bahattab, "Email Security Using Multi-Factor Authentication and Cloud-Based Services," *International Journal of Computer Applications*, vol. 975, no. 8887, pp. 25–31, 2015.

[17] "Amazon SES Python SDK Code Examples," *AWS Code Library*, 2024.

[18] Incentius, "Avoiding Common Pitfalls: The Best Approach to AWS SES Integration in Flask," *Incentius Tech Blog*, 2023.

[19] Edstem, "Cloud-Native Email Automation with AWS SES, Python, Flask, and Jinja2," *Edstem Blog*, 2024.

[20] TestDriven.io, "Flask SES RQ – Sending Confirmation Emails with Flask, Redis, and AWS SES," *GitHub Repository*, 2023.