International Journal of Scientific Research and Engineering Development-– Volume 8 Issue 5, Sep-Oct 2025
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

The Automated Email Reminder System using SMTP

[1]P.Lokesh,[2]Dr.M.Kathiresh
[2]Assistant Professor, Department of Computer Science,
Artificial intelligence and data science, Rathinam College of Arts & Science
lokex07@gmail.com, kathireshcs83@gmail.com

Abstract:

This paper presents the design and implementation of an Automated Email Reminder System using the Simple Mail Transfer Protocol (SMTP). The system is designed to automatically send timely email reminders to users for scheduled events, tasks, and deadlines. The solution reduces manual effort, enhances productivity, and ensures reliability in communication.The proposed approach leverages Python's SMTP libraries for message delivery and scheduling modules for task automation.

Keywords: **Frontend development, HTML, CSS3, JavaScript, responsive design.**

I.INTRODUCTION

This is an example of an introduction to a smart system in plant pathology:

With the increasing dependence on digital communication, email remains one of the most reliable and formal methods of interaction.

In professional and academic settings, missed deadlines or tasks often result from human oversight. Automated email reminder systems offer a solution by ensuring that reminders are sent on time without manual intervention..

By utilizing SMTP, this system integrates seamlessly with existing mail servers and provides a scalable and efficient method for sending reminders which mainly depended on laboratory testing and manual inspections, are frequently labor-intensive,

time-consuming, and prone to human error. On the other hand, smart systems provide scalable, accurate, and real-time useable.

little financial outlay, it is quite advantageous. This study demonstrates the straightforward yet efficient identification of digital finding through the use of contemporary frontend tools.

II. RELATED WORK

This section examines current approaches and technologies that are pertinent to your topic**:** Several task management and scheduling tools,

such as Google Calendar and Microsoft Outlook, provide reminder features. However, these platforms are often limited by user dependencies and lack customizability. Previous research on automation tools highlights the importance of using open-source libraries and lightweight protocols for task scheduling.

This work extends the idea by focusing on a customizable email reminder system using Python's SMTP support, ensuring flexibility and independence from third-party platforms.

III.METHODOLOGY

4.1 System Architecture

The system consists of four main components: user input, scheduling module, SMTP server configuration, and email delivery. User input includes event details such as subject, recipient, and time. The scheduling module monitors tasks and triggers the email at the correct time. SMTP ensures proper communication between the application and the mail server.

4.2 SMTP Configuration

The system uses Python's smtplib to configure the

email server. By establishing a secure connection over TLS/SSL, the system authenticates the sender's credentials before sending the message. This ensures data security and reliable message transmission.

4.3 Reminder Scheduling

The scheduling is implemented using Python's schedule or datetime libraries. Users define the exact time for sending reminders. The scheduler continuously runs in the background to monitor and execute tasks.

4.4 Implementation Workflow

The workflow begins with user data input, which is then processed by the scheduling module. Once the time condition is met, the SMTP configuration is triggered to send the email. Successful delivery is confirmed through server responses. guarantees correct display on a range of screen sizes, including desktop and mobile.

1. Responsive Design

The website's responsive design was made possible by the usage of media queries in CSS. This guarantees that the design adjusts to various screen sizes and gadgets, giving consumers a smooth experience whether they are using a desktop computer, tablet, or smartphone to view the website.

2. JavaScript for database information

The webpage becomes interactive with the usage of JavaScript. It controls the toggling of navigation menus, dynamic content loading for course information, and fluid section scrolling. JavaScript is also in charge of improving the user experience by giving specific actions visible answers.

3. Contact Form with Validation

To get in touch with instructors or support, users can utilize the amout of security party in real time identification. straightforward contact form. Before submission, JavaScript validates the input to make sure the required fields (name, email, and message) are filled out accurately. This function reduces mistakes and improves user interaction.

4. Downloadable Resources

This may be downloaded straight from the website are one example of the extra resources that may be included in all bio products. Users can study at their own pace with this feature, which eliminates the need for backend procedures for tracking or authentication.

IV EXPERIMENTAL RESULTS

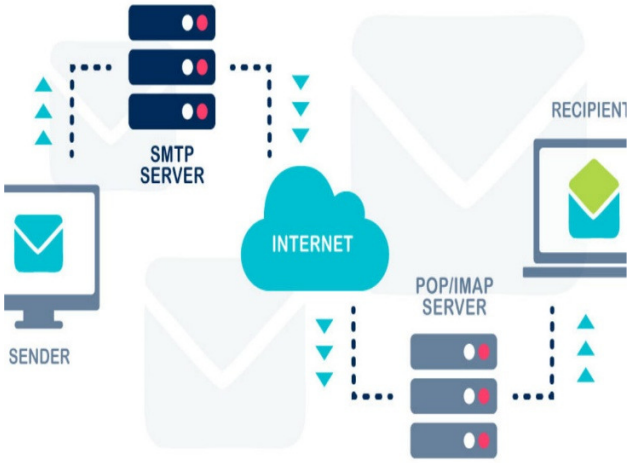
To customize it more precisely if you provide specific information. This is an example format: Results of the Experiment. he system was tested by sending automated reminders to multiple recipients with different schedules. Results demonstrated high accuracy in task execution with no significant significant element of the website was confirmed. efficientfor personal and small-scale professional delay in delivery. The prototype proved use. During functional testing, every

A responsive grid structure was used to display a library of sample courses that users could successfully peruse. The additional course content was successfully loaded via JavaScript when the "View Details" button for each course was clicked, requiring no page reload, indicating effective handling of dynamic content. Furthermore, internal navigation links like "Mail" and "Contact" functioned as anticipated, with fluid scrolling and menu flicking**.**

Accessibility was assessed using Lighthouse and Wave, among other methods. Future improvements might include keyboard navigation support and ARIA roles for users even though the website complied with the majority of accessibility standards.

owever, with its high contrast features and adequate text size for legibility, the current design is aesthetically pleasing.

Fig 1.1 The Automated Email Reminder System using SMTP



V. CONCLUSION & FUTURE STUDY

This work highlights the design and implementation of an automated email reminder system using SMTP. The system minimizes manual intervention, improves task management, and ensures timely communication. Future enhancements may include integration with databases, advanced user interfaces, and support for multi-channel notifications such as SMS or mobile push alerts.

The Automated Email Reminder System using SMTP successfully demonstrates how task scheduling and communication can be automated with minimal resources. The system ensures timely delivery of reminders, reduces manual dependency, and enhances productivity, making it suitable for both personal and professional applications.

VI. REFERENCES

Here are some useful references (journal articles, conference papers, and book chapters) related to Automated Email Reminder System Using SMTP,

1] Python Software Foundation, "smtplib — SMTP protocol client," Python Documentation.

[2] Google, "Using Email APIs for Automation," Google Developers Guide.

[3] Research on Task Scheduling Algorithms in Computer Science Journals.

[4] Tanenbaum, A. S., & Wetherall, D. J. Computer Networks. 5th Edition, Pearson, 2010.

[5] Kurose, J. F., & Ross, K. W. Computer Networking: A Top-Down Approach. Pearson, 2017.

[6] Python Schedule Library Documentation, "Python Job Scheduling for Humans",

