Pakiatan Civil Aviation Authority Summer Internship 2023

Automated Email Attachment Download and Sorting System

Submitted By:
Muhammad Owais.
Senior Year student Ghulam Ishaq Khan Institute.
Internee PCAA IT Department..

Catalog

Introduction 3

Limitations and Recent Updates 3

Limitations Working on Microsoft Outlook: 3

Update-Induced Changes and the Importance of Message ID: 3

Permissions and Access Challenges 3

Permission Popups: 3

Adaptation for Other Email Clients 4

Identifying the Target Email Client: 4

Utilizing Libraries and APIs: 4

Storing Data Logs in a Database 4

A Choosing a Database System: 4

Establishing a Database Connection: 4

Running the System on a Server 4

Deploying the System on a Server: 4

Adapting File Paths and Permissions: 5

Handling Email Retrieval: 5

Maintaining Data Logs: 5

Timeline of Progress: 5

Conclusion: 5

Email Attachment Download and Sorting System

Introduction

The Automated Email Attachment Download and Sorting System is a Python-based solution designed to automate the process of downloading email attachments from Microsoft Outlook and organizing them into separate folders based on their file types. This report provides a detailed overview of the system, including its limitations, adaptations for other email clients, storing data logs in a database, and considerations for running the system on a server.
Additionally, it highlights the specific application of the system in the context of private email servers, focusing on the case of the Pakistan Civil Aviation Authority.

Limitations and Recent Updates

Limitations Working on Microsoft Outlook:

The system is primarily developed for use with Microsoft Outlook and utilizes the win32com.client library, which is specifically designed for Outlook integration. Consequently, it may not be directly compatible with other email clients. It is important to note that the system assumes a Windows environment due to its reliance on the Win32 API. Furthermore, changes in the Outlook API or email client behavior may require code adjustments to maintain compatibility.

Update-Induced Changes and the Importance of Message ID:

An update in the Outlook application prompted us to modify the code and introduce new features, such as the inclusion of the Message ID. This unique identifier allows for accurate tracking and prevention of duplicate entries in the email log. By comparing the Message ID with existing entries, the system ensures that previously downloaded attachments are not reprocessed, thereby optimizing efficiency and preventing data redundancy.

Permissions and Access Challenges

Permission Popups:

When accessing Outlook programmatically, users may encounter permission popups that require manual interaction for authentication or authorization. These popups are a security measure implemented by Outlook to ensure secure access to user accounts. Although they can interrupt the automated process, they can be handled by granting the necessary permissions or utilizing techniques such as running the code with elevated privileges.

Adaptation for Other Email Clients

Identifying the Target Email Client:

The system can be adapted for use with other email clients, such as Gmail, Yahoo Mail, or Exchange. However, this would require modifying the code to utilize the appropriate libraries or APIs specific to the chosen email client.

Utilizing Libraries and APIs:

To adapt the system for a different email client, it is essential to explore and utilize Python libraries or APIs designed for the target email client. Popular libraries include imaplib, poplib, smtplib, and email.parser. By leveraging these libraries, users can retrieve emails, parse message content, and handle attachments according to the email client's specifications.

Storing Data Logs in a Database

A Choosing a Database System:

To store data logs in a database, users need to select a suitable database system based on their requirements. Options include SQLite, MySQL, PostgreSQL, or Microsoft SQL Server.

Establishing a Database Connection:

Python database libraries such as sqlite3, MySQLdb, psycopg2, or pyodbc can be used to establish a connection to the selected database. These libraries enable the insertion of log data into the corresponding database tables using SQL statements or object-relational mapping (ORM) techniques provided by the libraries.

Running the System on a Server

Deploying the System on a Server:

To run the system on a server, users must set up a server environment with the necessary dependencies and libraries installed. It is crucial to ensure that the server has access to the email accounts and the required network connectivity.

Adapting File Paths and Permissions:

Code modifications are necessary to specify appropriate file paths on the server for saving downloaded attachments. Adjustments to file permissions should be made to enable the server to write and organize the attachments in the desired folders.

Handling Email Retrieval:

When deploying the system on a server, the code needs to be adjusted to connect directly to the email server using IMAP or POP3 protocols. Emails can be retrieved from the server environment, and the code logic should be updated to process emails on the server rather than relying on the local Outlook application.

Maintaining Data Logs:

Data logs, whether stored in a database or other means, should be accessible to the server environment to record relevant information about the downloaded attachments. The server environment should have appropriate access privileges to read and write data logs as required.

Timeline of Progress:

1. Initial Research and Understanding of Requirements
2. Setting Up the Development Environment
3. Code Development for Email Retrieval and Attachment Handling
4. Testing and Debugging of the System
5. Integration of Message ID and Permission Handling
6. *Adaptation for Other Email Clients and Database Integration*
7. *Server Deployment Considerations and Code Modifications*
8. *Refinement and Optimization of the System*
9. *Documentation and Report Compilation*
10. *Final Review and Submission*

Conclusion:

The Automated Email Attachment Download and Sorting System provides a practical and efficient solution for automating the process of managing email attachments in Microsoft Outlook. While it has certain limitations, particularly in its dependence on Outlook and Windows, the system can be adapted for other email clients with the appropriate modifications.