

Bulk Emailing with in-built Mail Merge

Contents:

1. Introduction
 2. Background and Motivation
 3. Problem Statement
 4. Our Goals and Objectives
 5. Brief Overview
 6. System Design and Architecture for Bulk Email Application
 7. System Design and Architecture for Mail Merge Application
 8. Technical Stack for Bulk Email Application
 9. Technical Stack for Mail Merge Application
 10. Features Implemented in Bulk Email Application
 11. Features Implemented in Mail Merge Application
 12. Steps to Run the Bulk Email Application
 13. Steps to Run the Mail Merge Application
 14. Video Demonstration
 15. Screenshots
 16. Conclusion
-

1. Introduction:

Bulk emailing and mail merge are indispensable tools for efficient and personalized communication across industries. Bulk emailing enables organizations to reach large audiences quickly, supporting use cases such as marketing campaigns, newsletters, and internal updates. Meanwhile, mail merge allows for the creation of tailored messages at scale, enhancing engagement and response rates through personalization. Recognizing the limitations of existing solutions, including high costs and daily sending caps, we have developed an integrated bulk emailing and mail merge software. This solution empowers users to send personalized, scalable messages effectively, overcoming traditional constraints and optimizing communication efforts.

2. Background and Motivation:

Bulk emailing and mail merge are essential tools that, when used together, significantly enhance communication across various industries. Bulk emailing allows organizations to reach large audiences efficiently, whether for email marketing campaigns, newsletters, event invitations, or internal communications. However, many bulk emailing solutions have high costs or limitations, such as daily sending caps imposed by free email services like Gmail.

Mail merge complements bulk emailing by enabling users to create personalized communications at scale. By merging a template with a data source, such as a spreadsheet or database, users can generate customized emails and documents tailored to each recipient. This personalization is crucial for improving engagement and response rates, making communications more effective.

Recognizing the challenges associated with existing solutions, we have developed our own integrated bulk emailing and mail merge software. This powerful combination provides users with an accessible and flexible option to send personalized messages to large groups without the constraints of traditional platforms. By leveraging mail merge within bulk emailing, users can enhance their outreach efforts, ensuring that each message resonates with its intended audience while maintaining efficiency and scalability.

3. Problem Statement:

Existing email services are unsuitable for high-volume communication needs like newsletters and notifications. This project aims to develop a scalable and secure bulk emailing software that overcomes these limitations, offering high-performance delivery, advanced features, and greater flexibility for efficient email operations.

4. Our Goals and Objectives:

4.1. Enhanced Bulk Processing

To enable seamless delivery of thousands of emails per hour without the artificial throttling or daily limits imposed by services like Gmail, ensuring users can communicate efficiently at scale.

4.2. Advanced Email Management

To provide sophisticated features such as concurrent email processing, batch sending with customizable rate limits, detailed error tracking and logging, retry mechanisms for failed deliveries, and support for large attachments, offering a versatile and reliable solution.

4.3. Technical Superiority

To leverage advanced asynchronous processing techniques using asyncio, implement background task management, and ensure robust error handling to optimize performance and prevent server overload.

5. Brief Overview:

Our software is a high-performance bulk emailing solution designed to efficiently process and send thousands of emails using the SMTP protocol. It validates incoming requests, constructs MIME messages with formatted bodies and attachments, and processes recipients in concurrent batches with retry logic and delivery tracking. Security is ensured through SMTP authentication, TLS encryption, and data protection measures like secure file handling and input validation. Optimized for performance, the system utilizes connection pooling, memory-efficient streaming, and background task processing. Comprehensive testing and monitoring ensure reliability, tracking email delivery, logging errors, and providing detailed performance metrics.

6. System Design and Architecture Bulk Email Application:

6.1. API Layer (main.py)

- Provides RESTful endpoints for email operations.
- Implements rate limiting to prevent abuse.
- Handles background task processing for efficient execution.
- Manages file uploads for attachments.
- Formats consistent error responses.

6.2. Email Processing (email_utils.py)

- Constructs email messages with MIME format.
- Processes and attaches files securely.
- Manages SMTP connections with pooling and retry logic.
- Executes batch processing for large recipient lists.
- Logs errors for troubleshooting and analysis.

6.3. Data Models (email_schema.py)

- Validates email request inputs and ensures data integrity.
- Defines schemas for attachments and recipients.
- Handles optional fields gracefully for flexibility.

7. System Design and Architecture Mail Merge Application:

```
├── app.py                # Main Flask application file that handles routing
                           and email sending logic.
├── bulk_emailer.py       # Contains the BulkEmailer class for sending emails
                           and managing attachments.
├── email_template.py     # Manages email templates, including creation,
                           rendering, and storage.
├── email_validation.py   # Validates email addresses and checks for
                           valid domains.
├── README.md             # Documentation for the project, including setup
                           and usage instructions.
├── requirements.txt      # Lists the Python packages required to run the
                           application.
├── static                # Contains static files such as JavaScript and CSS.
    ├── main.js           # JavaScript file for client-side functionality.
    └── styles.css        # CSS file for styling the application.
├── templates             # Contains HTML templates and JSON files for email
                           templates.
    ├── email_templates.json # JSON file for storing email templates.
    └── index.html        # Main HTML file for the application interface.
├── text_cleaner.py       # Contains the TextCleaner class for sanitizing
                           input text.
└── uploads               # Directory for storing uploaded files
                           (attachments).
```

8. Technical Stack for Bulk Email Application:

8.1. Core Technologies

- **SMTP Protocol:** SMTP protocol via the asynchronous library aiosmtplib to securely send emails with features like TLS encryption, connection pooling, and robust error handling.
- **FastAPI:** Modern web framework with automatic API documentation, type checking, validation, and asynchronous support.
- **Pydantic:** Provides data validation, type checking, schema definition, and automatic documentation.
- **SlowAPI:** Enables rate limiting, request throttling, and IP-based restrictions.

8.2. Supporting Tools

- **Python-dotenv:** Manages environment variables, secure credential storage, and configuration.
- **Uvicorn:** High-performance ASGI server with WebSocket and HTTP/2 support.

9. Technical Stack for Mail Merge Application:

Backend:

Flask: A lightweight web framework for building the web application.

Python 3.13(Latest Version Required): Used for server-side logic and email handling.

SMTP: For sending emails through a Gmail SMTP server.

Frontend:

HTML/CSS: For structuring and styling the web application.

Bootstrap: A front-end framework for creating responsive and visually appealing user interfaces.

Database:

JSON Files: Used for storing email templates in a structured format.

Libraries:

Flask-WTF: For form handling and validation.

email-validator: For validating email addresses.

python-dotenv: For managing environment variables securely.

dnspython: For checking domain MX records to validate email domains.

Requirement.txt:

flask>=3.0.0

flask-wtf>=1.2.1

typing>=3.7.4

python-dotenv>=1.0.0

werkzeug>=3.0.0

email_validator>=2.1.0

dnspython>=2.4.2

10. Features Implemented in Bulk Email Application:

10.1. Efficient Email Processing

- Validates requests, checks rate limits, and processes attachments.
- Constructs MIME messages with formatted bodies, embedded links, and attachments.
- Splits recipients into batches, executes batches concurrently, and tracks delivery status with retry logic.

10.2. Robust Error Handling

- Manages connection errors, invalid recipients, attachment issues, and rate limit violations.

10.3. Security Enhancements

- Ensures SMTP authentication, TLS encryption, and secure credential storage.
- Implements IP-based rate limiting, request throttling, and concurrent request controls.
- Protects data with secure file handling, input validation, and error sanitization.

10.4. Performance Optimization

- Executes emails in configurable concurrent batches with memory efficiency.
- Manages connections with pooling, automatic reconnections, and timeout handling.
- Handles attachments and tasks using memory-efficient streaming and background processing.

11. Features Implemented in Mail Merge Application:

User Interface: The application features a clean and responsive UI built with Bootstrap. Users can easily navigate through the application to send emails or manage templates.

Email Form: Users can fill out a form to specify the sender's email, password, subject, body, recipients, CC, BCC, and attachments. The form includes validation to ensure that all required fields are filled out correctly.

Template Management: Users can create and manage reusable email templates. Templates can include variables for personalization, allowing users to send tailored messages to each recipient.

Email Validation: The application validates email addresses to ensure they are correctly formatted and that their domains are capable of receiving emails. This helps prevent sending to invalid addresses.

Sending Emails: When the user submits the form, the application processes the input, validates the email addresses, and sends the emails using the SMTP protocol. It supports sending attachments and includes options for CC and BCC recipients.

Error Handling: The application provides feedback to users in case of errors, such as invalid email addresses or issues with sending emails.

Security: The application emphasizes security by using environment variables for sensitive data, such as email credentials, and encourages the use of Gmail App Passwords instead of regular passwords.

12. Steps to Run the Bulk Email Application:

12.1. Configure SMTP Credentials

- Update the SMTP credentials with your own in the configuration file present inside the Bulk_email directory.

12.2. Install Dependencies

- Open your terminal and run the following command to install the required packages:
pip install -r requirements.txt

12.3. Open Two Terminals in the Bulk_email Directory(.i.e something/BulkEmail_MailMerge/Bulk_email)

Terminal 1: Start the FastAPI Server

- Run the following command to start the FastAPI server:

uvicorn app.main:app --host 0.0.0.0 --port 8000 --reload

(Alternatively, use **python -m uvicorn app.main:app --host 0.0.0.0 --port 8000 --reload**)

Terminal 2: Run the Bulk Email Script

- Add the email addresses of recipients to recipients.txt.
- Make the script executable (if not already):
chmod +x bulk_tester.sh
- Ensure that the file paths are correctly set in bulk_tester.sh.
- Run the script using the following command:
./bulk_tester.sh

13. Steps to Run the Mail Merge Application:

Go to the Mail_Merge Directory of the project directory and follow below steps:

1. Open a terminal window in the extracted folder and run the following command
 - a. **python3.13 -m venv .CN_Project_Mail_Merge**
 - b. **source .CN_Project_Mail_Merge/bin/activate**
 - c. **pip install -r requirements.txt**
 - d. **python app.py** : This will start the server: Running on **http://127.0.0.1:5000**
2. Copy the address into a browser. You will a page as shown below:

3. A default template Welcome is provide within the Email Template
4. More can be created or edited using this webpage. Instructions for the same are detailed in the Web Page itself

14. Video Demonstration:

<https://drive.google.com/file/d/1dxwO8zb7Q70nI3y6tby0WAHb7EMANDpb/view>

<https://drive.google.com/file/d/1LYKqrMmqhGrHjHQNWGEK-z5s9yzQTGYJ/view?usp=sharing>

15. Screenshots:




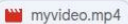











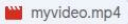

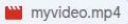
- 15.1. Command **uvicorn app.main:app --host 0.0.0.0 --port 8000 --reload** has been run, the server is run, and emails have been successfully sent to 97 recipients (95 from recipient.txt and 2 from the script file) in 10 batches of 10 recipients each.

```
satyam@satyam-ROG-Strix-G513IH-G513IH:~/Bulk_email$ uvicorn app.main:app --host 0.0.0.0 --port 8000
INFO: Started server process [49108]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
INFO: 127.0.0.1:37924 - "POST /send-email/ HTTP/1.1" 200 OK
INFO:app.email_utils:Batch 1: Email sent successfully to 10 recipients
INFO:app.email_utils:Batch 2: Email sent successfully to 10 recipients
INFO:app.email_utils:Batch 3: Email sent successfully to 10 recipients
INFO:app.email_utils:Batch 4: Email sent successfully to 10 recipients
INFO:app.email_utils:Batch 5: Email sent successfully to 10 recipients
INFO:app.email_utils:Batch 6: Email sent successfully to 10 recipients
INFO:app.email_utils:Batch 7: Email sent successfully to 10 recipients
INFO:app.email_utils:Batch 8: Email sent successfully to 10 recipients
INFO:app.email_utils:Batch 9: Email sent successfully to 10 recipients
INFO:app.email_utils:Batch 10: Email sent successfully to 7 recipients
INFO:app.email_utils:Email send summary: Successful batches: 10, Failed batches: 0
```

15.2. The script is made executable, and the script is run using the command `./bulk_tester.sh`


```
satyam@satyam-ROG-Strix-G513IH-G513IH:~/Bulk_email$ chmod +x bulk_tester.sh
satyam@satyam-ROG-Strix-G513IH-G513IH:~/Bulk_email$ ./bulk_tester.sh
{"message": "Email is being sent to 95 recipients", "recipients_count": 95}satyam@satyam-ROG-Strix-G513IH-G513IH:~/Bulk_email$
```

15.3. These are the emails sent from the sender:

<input type="checkbox"/> ☆ To: 220010060 ... 15	Video and Screenshot Attached - Please find the attached video and screenshot. Additional Links: https://google.com  
<input type="checkbox"/> ☆ To: 220010038 ... 3	Video and Screenshot Attached - Please find the attached video and screenshot. Additional Links: https://google.com  
<input type="checkbox"/> ☆ To: 220010027 ... 3	Video and Screenshot Attached - Please find the attached video and screenshot. Additional Links: https://google.com  
<input type="checkbox"/> ☆ To: 220010017 ... 4	Video and Screenshot Attached - Please find the attached video and screenshot. Additional Links: https://google.com  
<input type="checkbox"/> ☆ To: 220010049 ... 2	Video and Screenshot Attached - Please find the attached video and screenshot. Additional Links: https://google.com  
<input type="checkbox"/> ☆ To: 220010007 ...	Video and Screenshot Attached - Please find the attached video and screenshot. Additional Links: https://google.com  
<input type="checkbox"/> ☆ To: 220120026 ...	Video and Screenshot Attached - Please find the attached video and screenshot. Additional Links: https://google.com  
<input type="checkbox"/> ☆ To: 220120012 ...	Video and Screenshot Attached - Please find the attached video and screenshot. Additional Links: https://google.com  
<input type="checkbox"/> ☆ To: 220120001 ...	Video and Screenshot Attached - Please find the attached video and screenshot. Additional Links: https://google.com  


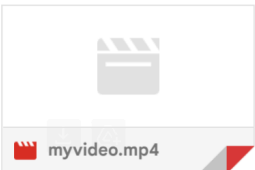
15.4. The email did not go in the SPAM section. These are the emails received by the receivers:

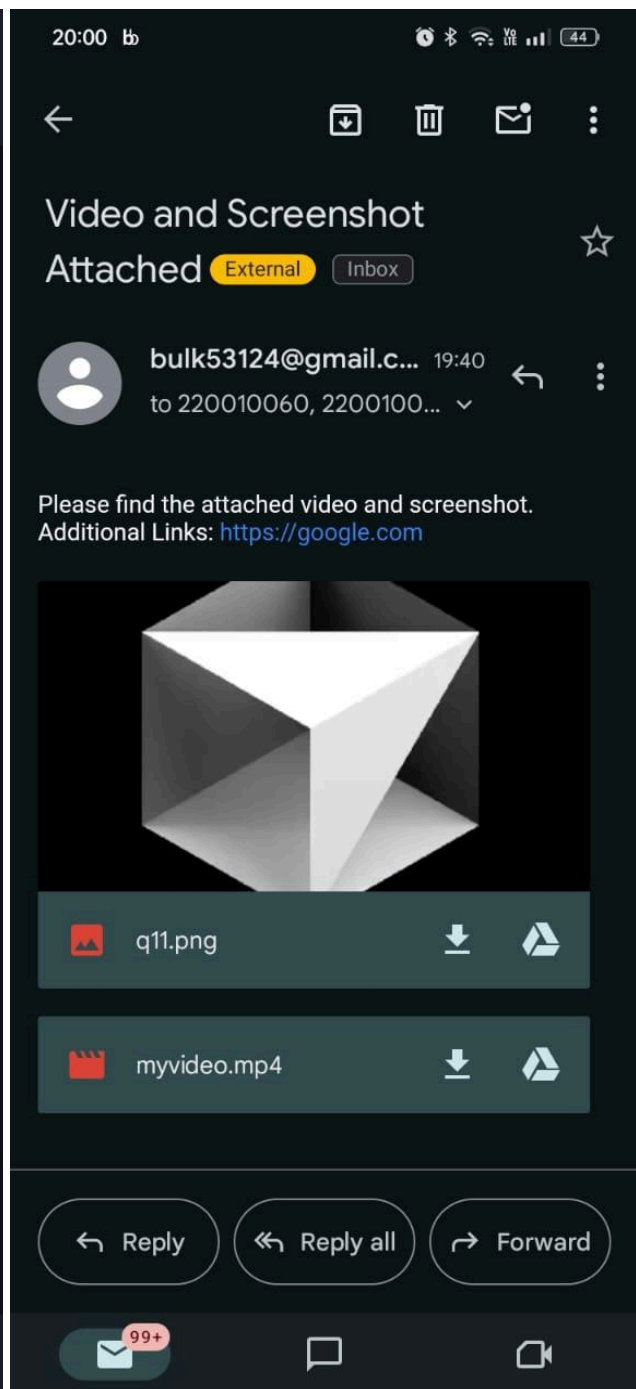
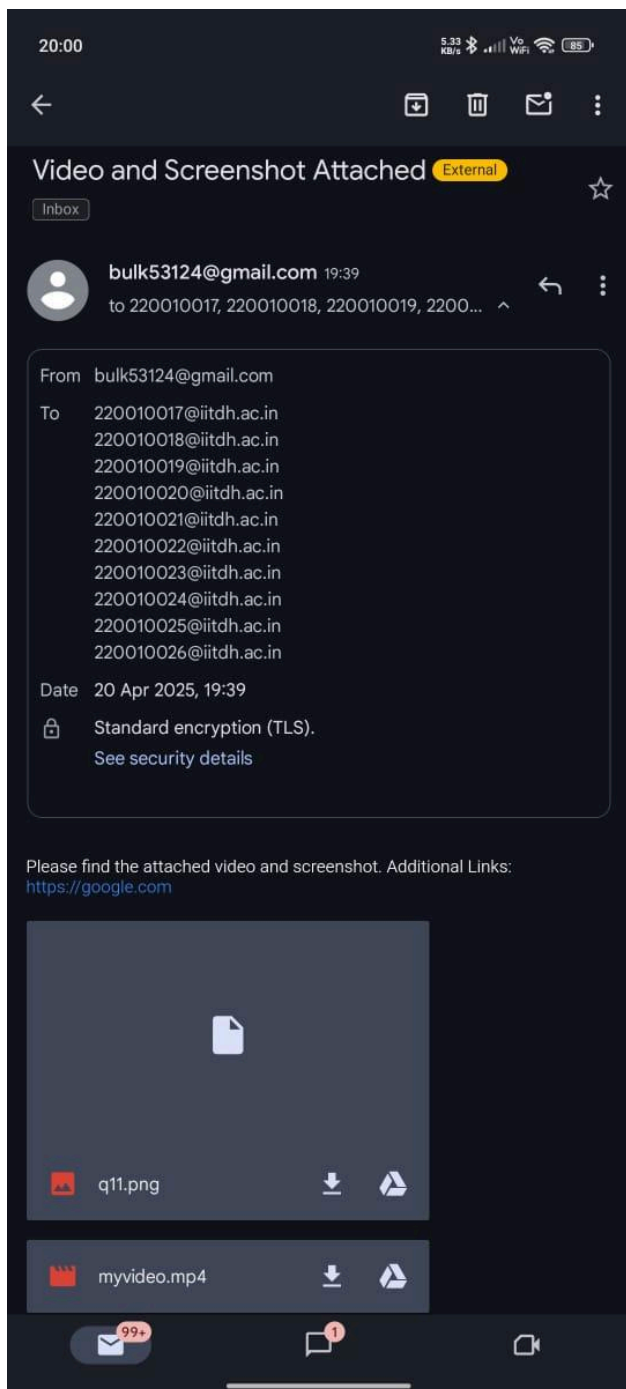
Video and Screenshot Attached External Inbox x ⌵ 🖨 🔗

 **bulk53124@gmail.com** 🗨 7:40 PM (3 hours ago) ☆ ↶ ⋮
to 220010060, 220010061, 220010062, 220010063, 220020029, 220020056, 220030003, 220090011, 220110008, me ▼

Please find the attached video and screenshot. Additional Links: <https://google.com>

2 Attachments • Scanned by Gmail ⓘ ⬇ 🔗



16. Conclusion:

This bulk emailing software overcomes the limitations of existing services by offering scalable, secure, and efficient high-volume email delivery. With features like batch processing, error tracking, and robust security, it ensures reliable performance. Built on modern frameworks, the system addresses real-world challenges, providing a practical solution for diverse email communication needs.