MailBlaster: Comprehensive Documentation

Project Development Team

January 31, 2025

Contents

1 Project Overview **2**
- 1.1 Introduction . . . 2
- 1.2 System Architecture . . . 2

2 Project Structure **3**
- 2.1 Directory Layout . . . 3

3 Detailed Component Analysis **4**
- 3.1 Application Dependencies . . . 4
  - 3.1.1 requirements.txt Analysis . . . 4
- 3.2 Flask Application (`app.py`) . . . 4
  - 3.2.1 Application Configuration . . . 4
  - 3.2.2 Authentication Mechanism . . . 4
  - 3.2.3 Email Sending Workflow . . . 4

4 HTML Templates **5**
- 4.1 Login Page (`login.html`) . . . 5
  - 4.1.1 Design Philosophy . . . 5
  - 4.1.2 Key Features . . . 5
- 4.2 Main Interface (`main.html`) . . . 5
  - 4.2.1 Functional Sections . . . 5

5 Security Considerations **6**
- 5.1 Authentication . . . 6
- 5.2 File Handling . . . 6

6 Deployment Guidelines **7**
- 6.1 Environment Setup . . . 7

7 Advanced Usage **8**
- 7.1 Template Variables . . . 8

8 Troubleshooting **9**
- 8.1 Common Issues . . . 9

9 Future Enhancements **10**

10 Conclusion **11**

Chapter 1

Project Overview

1.1 Introduction

MailBlaster is a sophisticated Flask-based web application designed for bulk email sending with dynamic templating and secure file management. The project aims to provide an intuitive interface for sending personalized emails to multiple recipients efficiently.

1.2 System Architecture

The application follows a Model-View-Controller (MVC) architectural pattern:

- **Model:** CSV data and email template
- **View:** HTML templates (login and main interface)
- **Controller:** Flask application (`app.py`)

Chapter 2

Project Structure

2.1 Directory Layout

```
mailblaster/

 templates/
    login.html
    main.html

 static/
    uploads/

 app.py
 requirements.txt
 recipients.csv
```

Chapter 3

Detailed Component Analysis

3.1 Application Dependencies

3.1.1 requirements.txt Analysis

```
flask              # Web framework
flask-session      # Session management
python-dotenv     # Environment variable management
```

3.2 Flask Application (app.py)

3.2.1 Application Configuration

```
# Core Flask Configuration
app = Flask(__name__)
app.config['SECRET_KEY'] = 'your-secret-key'
app.config['UPLOAD_FOLDER'] = os.path.join('static', 'uploads')
app.config['SESSION_TYPE'] = 'filesystem'
```

3.2.2 Authentication Mechanism

The application implements a simple session-based authentication:

- Login route captures Gmail credentials
- Stores credentials in server-side session
- Requires App Password for SMTP authentication

3.2.3 Email Sending Workflow

1. Read CSV file
2. Parse recipient details
3. Generate personalized emails
4. Send via SMTP

Chapter 4

HTML Templates

4.1 Login Page (`login.html`)

4.1.1 Design Philosophy

- Responsive mobile-first design
- Gradient background
- Minimalist UI

4.1.2 Key Features

- Gmail address input
- App Password field
- Font Awesome icons
- Gradient button styles

4.2 Main Interface (`main.html`)

4.2.1 Functional Sections

- Email Subject Input
- Dynamic Template Editor
- CSV Recipient Upload
- Attachment Management

Chapter 5

Security Considerations

5.1 Authentication

- Use App Passwords
- No storage of primary credentials
- Session-based access control

5.2 File Handling

- `secure_filename()` for uploads
- Temporary file storage
- Restricted file type uploads

Chapter 6

Deployment Guidelines

6.1 Environment Setup

```
# Create Virtual Environment
python3 -m venv venv
source venv/bin/activate

# Install Dependencies
pip install -r requirements.txt

# Run Application
flask run
```

Chapter 7

Advanced Usage

7.1 Template Variables

Dynamic email templates support `column_name` placeholders:

```
Hello {{name}},
Your unique ID is {{id}}.
```

Chapter 8

Troubleshooting

8.1 Common Issues

- SMTP Authentication Errors
- CSV Parsing Problems
- Email Sending Limitations

Chapter 9

Future Enhancements

- OAuth2 Authentication
- Email Tracking
- Advanced Templating Engine
- Rate Limiting

Chapter 10

Conclusion

MailBlaster provides a robust, secure solution for personalized bulk email communication, with a focus on simplicity and flexibility.