CSCI-351 Project: SMTP Email Server

Members: Jake Edelstein (jee1623@rit.edu)

SMTP implementation details:

My project aims to create a basic email server using the SMTP protocol. It uses nearly all of the same steps as seen in a real mail server that uses SMTP as its mail protocol of choice. First, The client and server must establish a connection with each other, and this is done through the HELO signal from the client with code 220. Next, the client tells the server who the mail is coming from, who the mail should go to, and the data associated with the email message. Each of these steps is followed up with an Ok with code 250. Finally, the client will send a period (.) on a line by itself to signal the end of the message data, which is interpreted as a message with code 354. Once the message content is sent, the server will send a Bye signal with code 221 to the client, which will then close its end of the connection. Here is a screenshot of the SMTP protocol from its Wikipedia page, for reference:

```
S: 220 smtp.example.com ESMTP Postfix
C: HELO relay.example.org
S: 250 Hello relay.example.org, I am glad to meet you
C: MAIL FROM:<bob@example.org>
S: 250 Ok
C: RCPT TO:<alice@example.com>
S: 250 Ok
C: RCPT TO:<theboss@example.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: From: "Bob Example" <bob@example.org>
C: To: "Alice Example" <alice@example.com>
C: Cc: theboss@example.com
C: Date: Tue, 15 Jan 2008 16:02:43 -0500
C: Subject: Test message
C:
C: Hello Alice.
C: This is a test message with 5 header fields and 4 lines in the message body.
C: Your friend,
C: Bob
C: .
S: 250 Ok: queued as 12345
C: QUIT
S: 221 Bye
{The server closes the connection}
```

My program matches the real SMTP protocol very closely. For simplicity, my program does not include a Cc or Bcc feature or a date. Each email is saved as a file in the mailbox, and the filename is the subject of the email. Each user gets their own subdirectory in the mailbox so that multiple users can receive an email with the same subject. Additionally, in the real SMTP, the server will close its own connection after the Bye signal is sent, but in my implementation, I allowed the server to keep the connection open so that it is easier to send multiple test emails without needing to restart the program or the connection every single time. In summary, my implementation matches the real protocol on these steps:

- HELO signal (220)
- MAIL FROM message

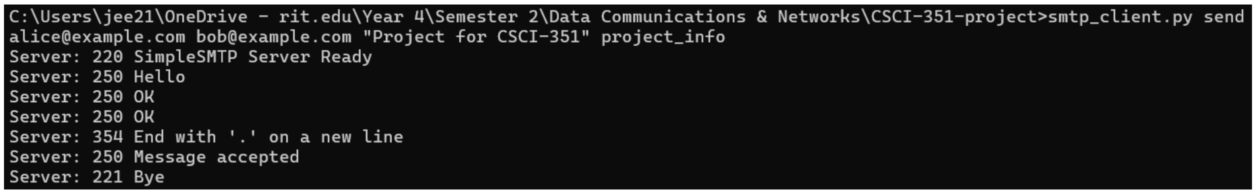
- RCPT TO message
- DATA message
- Ok signal (250)
- End data with . (354)

Minor changes for simplicity occur at these steps:

- No Cc, Bcc, or date
- Email subject is the filename when saved
- Server connection stays open after Bye (221)

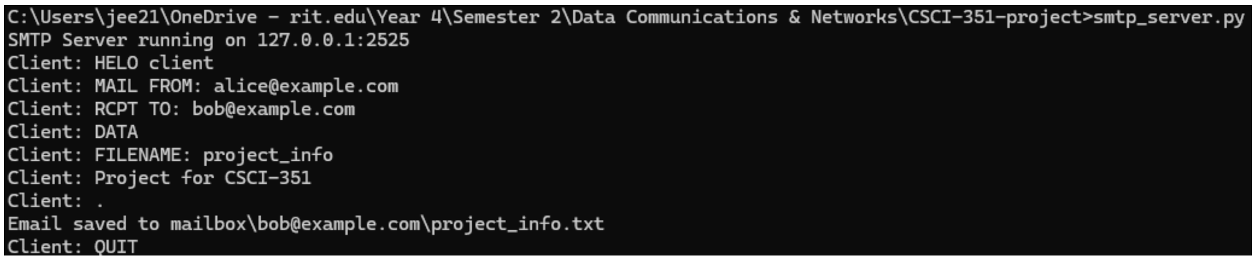
i. Sending an email to the server

Client view:



```
C:\Users\jee21\OneDrive - rit.edu\Year 4\Semester 2\Data Communications & Networks\CSCI-351-project>smtp_client.py send
alice@example.com bob@example.com "Project for CSCI-351" project_info
Server: 220 SimpleSMTP Server Ready
Server: 250 Hello
Server: 250 OK
Server: 250 OK
Server: 354 End with '.' on a new line
Server: 250 Message accepted
Server: 221 Bye
```

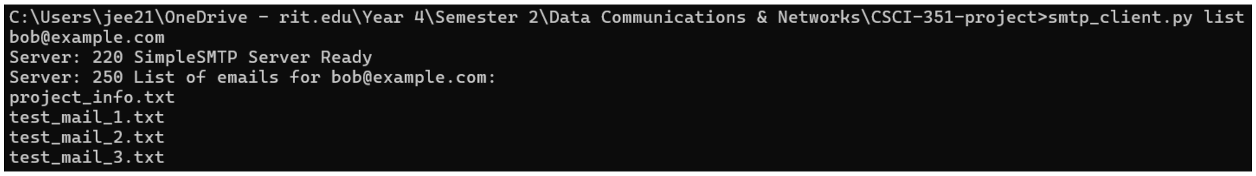
Server view:



```
C:\Users\jee21\OneDrive - rit.edu\Year 4\Semester 2\Data Communications & Networks\CSCI-351-project>smtp_server.py
SMTP Server running on 127.0.0.1:2525
Client: HELO client
Client: MAIL FROM: alice@example.com
Client: RCPT TO: bob@example.com
Client: DATA
Client: FILENAME: project_info
Client: Project for CSCI-351
Client: .
Email saved to mailbox\bob@example.com\project_info.txt
Client: QUIT
```

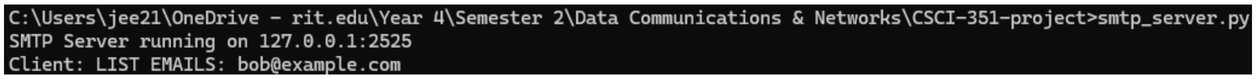
ii. Requesting all emails for a user

Client view:



```
C:\Users\jee21\OneDrive - rit.edu\Year 4\Semester 2\Data Communications & Networks\CSCI-351-project>smtp_client.py list
bob@example.com
Server: 220 SimpleSMTP Server Ready
Server: 250 List of emails for bob@example.com:
project_info.txt
test_mail_1.txt
test_mail_2.txt
test_mail_3.txt
```

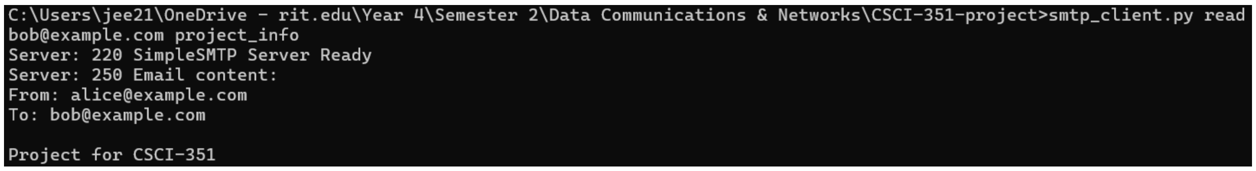
Server view:



```
C:\Users\jee21\OneDrive - rit.edu\Year 4\Semester 2\Data Communications & Networks\CSCI-351-project>smtp_server.py
SMTP Server running on 127.0.0.1:2525
Client: LIST EMAILS: bob@example.com
```

iii. Reading an individual email

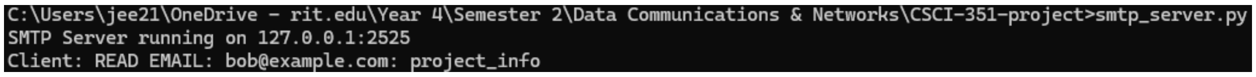
Client view:



```
C:\Users\jee21\OneDrive - rit.edu\Year 4\Semester 2\Data Communications & Networks\CSCI-351-project>smtp_client.py read
bob@example.com project_info
Server: 220 SimpleSMTP Server Ready
Server: 250 Email content:
From: alice@example.com
To: bob@example.com

Project for CSCI-351
```

Server view:



```
C:\Users\jee21\OneDrive - rit.edu\Year 4\Semester 2\Data Communications & Networks\CSCI-351-project>smtp_server.py
SMTP Server running on 127.0.0.1:2525
Client: READ EMAIL: bob@example.com: project_info
```