Anthony Chavez

Professor Sun

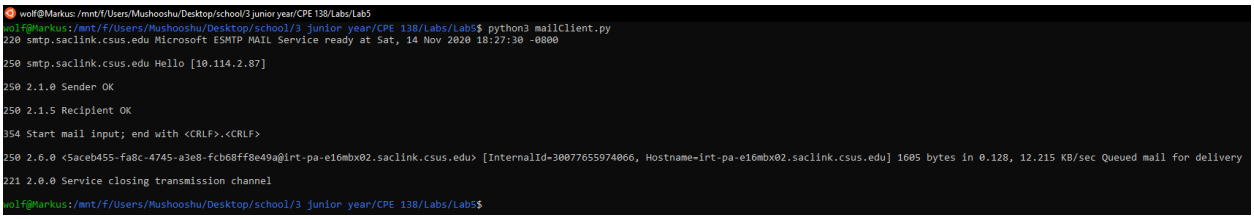
Socket Programming 3

mailClient Source Code

```
from socket import *

msg = "\r\n I love computer networks!"
endmsg = "\r\n.\r\n"

# Choose a mail server (e.g. Google mail server) and call it mailserver
mailServer = "smtp.csus.edu"
serverPort = 25

# Create socket called clientSocket and establish a TCP connection with mailserver
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((mailServer, serverPort))

recv = clientSocket.recv(1024).decode()
print(recv)
if recv[:3] != '220':
        print('220 reply not received from server.')

# Send HELO command and print server response.
heloCommand = 'HELO Alice\r\n'
clientSocket.send(heloCommand.encode())
recv1 = clientSocket.recv(1024).decode()
print(recv1)
if recv1[:3] != '250':
        print('250 reply not received from server.')

# Send MAIL FROM command and print server response.
mailFromCommand = 'MAIL FROM: aechavez@csus.edu\r\n'
clientSocket.send(mailFromCommand.encode())
recv1 = clientSocket.recv(1024).decode()
print(recv1)
if recv1[:3] != '250':
        print('250 Reply not Received')

# Send RCPT TO command and print server response.
rcptToCommand = 'RCPT TO: aechavez@csus.edu\r\n'
clientSocket.send((rcptToCommand).encode())
recv1 = clientSocket.recv(1024).decode()
print(recv1)
if recv1[:3] != '250':
        print('250 Reply not Received')
```

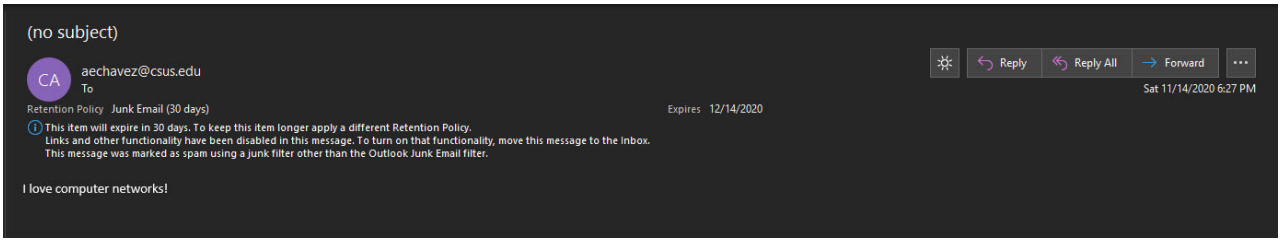
```
# Send DATA command and print server response.
dataCommand = 'DATA\r\n'
clientSocket.send((dataCommand).encode())
recv1 = clientSocket.recv(1024).decode()
print(recv1)
if recv1[:3] != '354':
        print('354 Reply Not Received')

# Send message data.
# Message ends with a single period.
clientSocket.send((msg+endmsg).encode())
recv1 = clientSocket.recv(1024).decode()
print(recv1)
if recv1[:3] != '250':
        print('250 Reply Not Received')

# Send QUIT command and get server response.
quitCmd = 'QUIT\r\n'
clientSocket.send((quitCmd).encode())
recv1 = clientSocket.recv(1024).decode()
print(recv1)
if recv1[:3] != '221':
        print('221 Reply Not Received')
clientSocket.close()
```

Terminal Run



Here we have the execution of the mail client program in the BASH terminal. The first few lines, we can see the 3-way handshaking occur. Then the message is sent after a successful handshaking phase is completed. Finally, the socket is closed after the message is sent.

Displaying Message in Outlook



Here we see the message sent by the mail client program in the user agent, Outlook. In the code we had the message, “I love computer networks!” which we see was successfully sent and received.