Email Client Application Report
Computer Networks Lab 2

Abdelrahman Omar Abouroumia 8368
Group 1 Section 1

March 14, 2025

1 Introduction

The objective of this project is to develop a Python-based email client application capable of sending and receiving emails using the `smtplib` and `imaplib` libraries. The application should establish a TCP connection with an email server, communicate using SMTP and IMAP protocols, send email messages, fetch unread emails, and close the connection.

2 System Requirements

2.1 Software Requirements

- Python 3.x
- Required libraries:
  - `smtplib`
  - `imaplib`
  - `email`
  - `tkinter`
  - `threading`
- SMTP and IMAP servers (smtp.gmail.com, imap.gaim.com)

3 Implementation Details

3.1 Sending Emails

The application uses `smtplib` and `email.mime.multipart` to send emails. The following parameters are required:

- Sender email and password
- Recipient email
- Email subject and body

Listing 1: Sending Emails

```
def send_email():
    sender_email = sender_email_entry.get()
    sender_password = sender_password_entry.get()
    recipient_email = recipient_email_entry.get()
    subject = subject_entry.get()
    body = body_text.get("1.0", tk.END)

    try:
        msg = MIMEMultipart()
        msg['From'] = sender_email
        msg['To'] = recipient_email
        msg['Subject'] = subject
        msg.attach(MIMEText(body, 'plain'))

        server = smtplib.SMTP(SMTP_SERVER, SMTP_PORT)
        server.starttls()
        server.login(sender_email, sender_password)
        server.sendmail(sender_email, recipient_email, msg.as_string())
        server.quit()
        messagebox.showinfo("Success", "Email sent successfully!")
    except Exception as e:
        messagebox.showerror("Error", f"Failed to send email: {e}")
```

3.2 Receiving Emails

The application fetches unread emails from the inbox using `imaplib`. The email body is extracted and displayed in the inbox area.

Listing 2: Receiving Emails

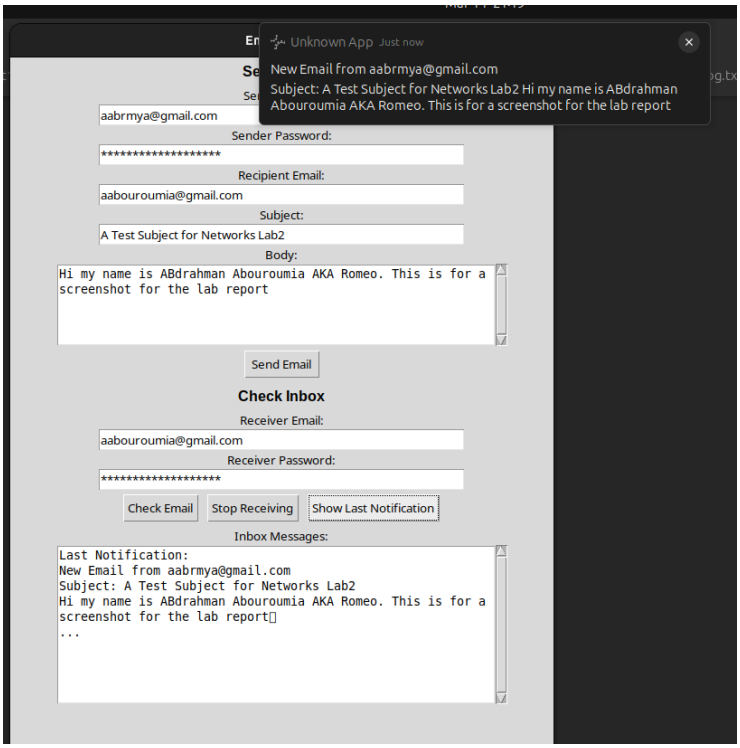
```
def receive_email():
    global stop_receiving
    stop_receiving = False

    def fetch_emails():
        email_address = receiver_email_entry.get()
        email_password = receiver_password_entry.get()

        try:
            mail = imaplib.IMAP4_SSL(IMAP_SERVER, IMAP_PORT)
```

```
            mail.login(email_address, email_password)
            mail.select('inbox')

            while not stop_receiving:
                result, data = mail.search(None, 'UNSEEN')
                email_ids = data[0].split()

                for email_id in email_ids:
                    result, msg_data = mail.fetch(email_id, '(RFC822)')
                    raw_email = msg_data[0][1]
                    msg = email.message_from_bytes(raw_email)
                    sender = msg['From']
                    subject = msg['Subject']
                    body = ""

                    if msg.is_multipart():
                        for part in msg.walk():
                            if part.get_content_type() == "text/plain":
                                body = part.get_payload(decode=True).decode()
                                break
                    else:
                        body = msg.get_payload(decode=True).decode()

                    show_notification(f"New Email from {sender}", f"Subject: {su

                mail.logout()
        except Exception as e:
            messagebox.showerror("Error", f"Failed to receive email: {e}")

    threading.Thread(target=fetch_emails, daemon=True).start()
```

4 Graphical User Interface (GUI)

The application features a GUI built with `tkinter`. Users can:

- Input sender and receiver email credentials
- Compose and send emails
- Fetch unread emails
- Receive push notifications for new emails



Screenshot of the Email Client GUI

5 Testing and Results

The application was tested with multiple email accounts and SMTP/IMAP servers:

- Successfully sent and received emails
- Implemented error handling for incorrect credentials
- Push notifications worked correctly

6 References

- SMTP Protocol: `https://tools.ietf.org/html/rfc5321`
- IMAP Protocol: `https://tools.ietf.org/html/rfc3501`