



Alexandria National University

Faculty of Computing and Data Science

Computer Networks – Final Project

Project Title:

Email Client Application

Submitted by:

- Ahmed Hesham – ID: 2305277
- Youssef Hesham – ID: 2305205
- Hager Amr – ID: 2305197
- Adel Maged – ID: 2305291

Instructor: Dr. Mohamed Rezk

TA: Eng. Ahmed Ashraf / Eng. Miar Mamdouh / Eng. Salma Magdy

Date: 18/12/2025

1- Introduction

- ❑ This project implements an email notification system that integrates **SMTP, IMAP, and TCP socket programming**.

The system allows sending emails using SMTP and monitoring incoming emails using IMAP.

Whenever a new email is detected, a **real-time TCP notification** is sent from the server to the client.

- ❑ The goal of this project is to demonstrate practical usage of **application layer protocols** and **transport layer communication**.

2- System Architecture

The system consists of four main components:

- **SMTP Client:** Sends emails using Gmail SMTP server.
- **IMAP Client:** Monitors the inbox and detects new incoming emails.
- **TCP Notification Server:** Sends notifications to connected clients.
- **TCP Notification Client:** Receives real-time notifications

Architecture Flow:

1. Email is sent using SMTP.
2. IMAP client checks the inbox.
3. If a new email is detected, the TCP server sends a notification.
4. The TCP client receives and displays the notification.

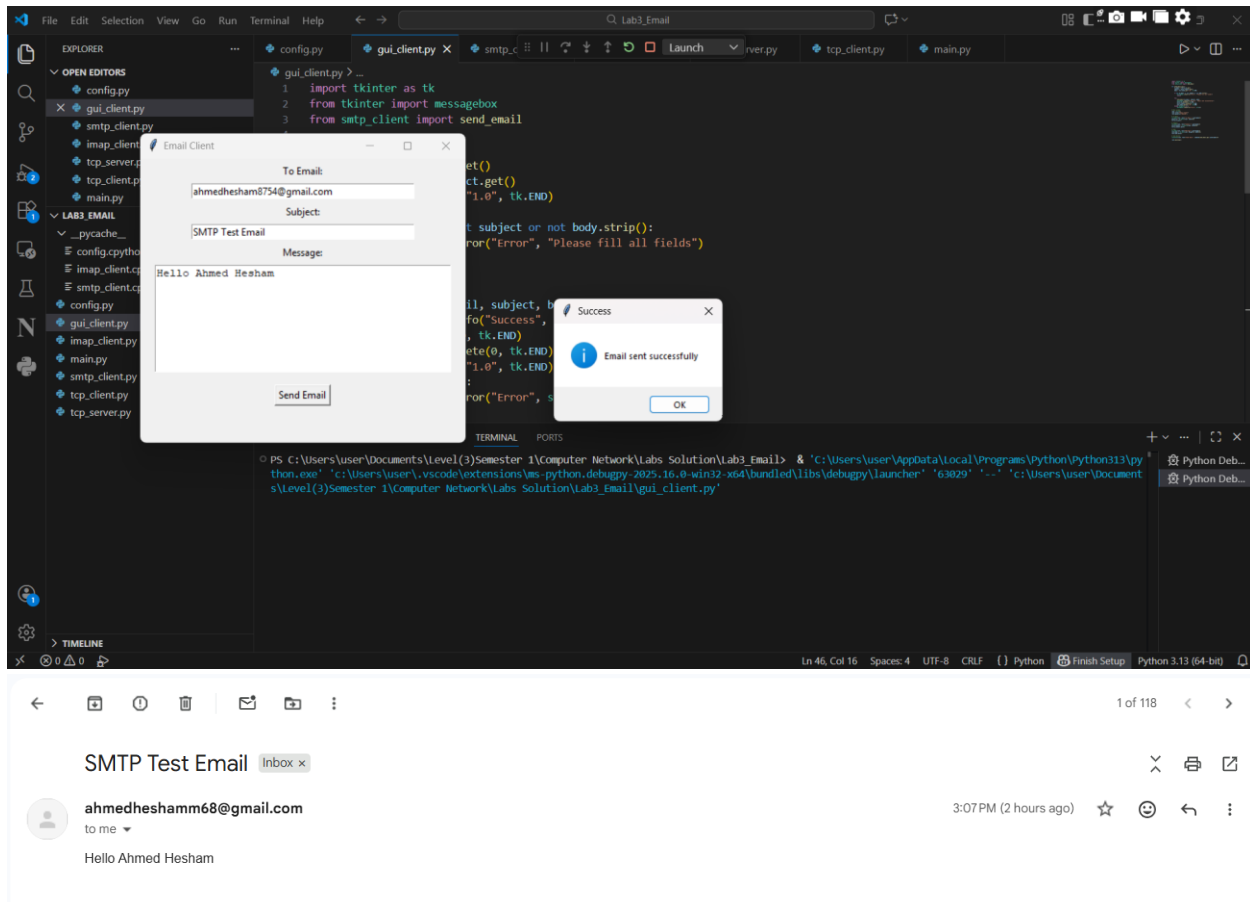
3- Technologies Used

- Programming Language: **Python**
- Protocols:
 - SMTP (Email Sending)
 - IMAP (Email Receiving)
 - TCP (Notification System)
- Tools:
 - Wireshark
 - Gmail (App Password)
 - Python socket library

4- Implementation Details

4.1 SMTP Client

- Uses Gmail SMTP server.
- Sends email securely using SSL.
- App Password is used instead of the real Gmail password.



smtp_capture (Wire shark)>>(tcp.port == 465)

		Info	Length	Protocol	Destination	Source	Time	No.
		Application Data 139		TLSv1.3	192.168.10.12	209.85.202.108	507.078063	7186
		Application Data 113		TLSv1.3	209.85.202.108	192.168.10.12	507.078591	7187
		Seq=5117 Ack=865 Win=269312 Len=0		TCP	192.168.10.12	209.85.202.108	507.164256	7188
		[ACK] 56248 → 465 54		TCP	192.168.10.12	209.85.202.108	507.173425	7189
		Application Data 139		TLSv1.3	192.168.10.12	209.85.202.108	507.173808	7190
		Application Data 82		TLSv1.3	209.85.202.108	192.168.10.12	507.303778	7191
		Seq=5202 Ack=893 Win=269312 Len=0		TCP	192.168.10.12	209.85.202.108	507.343929	7192
		[ACK] 56248 → 465 54		TCP	192.168.10.12	209.85.202.108	507.344596	7193
		Application Data 139		TLSv1.3	192.168.10.12	209.85.202.108	507.427626	7194
		Application Data 470		TLSv1.3	192.168.10.12	209.85.202.108	507.796251	7195
		Seq=5287 Ack=1309 Win=269056 Len=0		TCP	192.168.10.12	209.85.202.108	507.796505	7196
		[ACK] 56248 → 465 54		TCP	192.168.10.12	209.85.202.108	507.870073	7197
		Application Data 151		TLSv1.3	192.168.10.12	209.85.202.108	507.886447	7198
		Application Data 82		TLSv1.3	192.168.10.12	209.85.202.108	507.886656	7200
		Seq=5384 Ack=1337 Win=269056 Len=0		TCP	192.168.10.12	209.85.202.108	507.897097	7201
		[ACK] 56248 → 465 54		TCP	192.168.10.12	209.85.202.108	507.963334	7202
		Application Data 155		TLSv1.3	192.168.10.12	209.85.202.108		
		Seq=1337 Ack=5486 Win=64768 Len=0		TCP	192.168.10.12	209.85.202.108		
		[FIN, ACK] 56248 → 465 54		TCP	192.168.10.12	209.85.202.108		
		Seq=1337 Ack=5486 Win=64768 Len=0		TCP	192.168.10.12	209.85.202.108		
		[FIN, ACK] 465 → 56248 54		TCP	192.168.10.12	209.85.202.108		
		Seq=5486 Ack=1338 Win=269056 Len=0		TCP	192.168.10.12	209.85.202.108		
		[ACK] 56248 → 465 54		TCP	192.168.10.12	209.85.202.108		

0000	78 44 76 9b 89 44 8c f8	c5 8d 81 be 08 00 45 00	xDr...D... ..E..	ts), 66 bytes captured (528 bits) on interface \Device\NPF_{64EABC48-0637-4087-A0B8-54CCDA07D0B4}, id 0 <
0010	00 34 c8 fc 40 00 00 06	00 00 c0 a8 0a 0c d1 55	4-@... ..U	Ethernet II, Src: Intel_E8:81:be (8c:f8:c5:8d:81:be), Dst: ZioncomElect_9b:89:44 (78:44:76:9b:89:44) <
0020	ca 6d d1 04 01 d1 16 12	f8 c9 00 00 00 00 00 02	m-.....	Internet Protocol Version 4, Src: 192.168.10.12, Dst: 209.85.202.108 <
0030	ff ff 66 9e 00 00 02 04	05 b4 01 03 00 01 01	...F.....	Transmission Control Protocol, Src Port: 53508, Dst Port: 465, Seq: 0, Len: 0 <
0040	04 02		..	

4.2 IMAP Client

- Connects to Gmail IMAP server.
- Checks the inbox for the latest email.
- Extracts the email subject.

```

main.py
1 from imap_client import check_latest_email
2
3 subject = check_latest_email()
4 print("latest email subject:", subject)
5

```

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\User\Documents\Level(3)\Semester 1\Computer Network\Lab3_email> & "C:\Users\User\AppData\Local\Programs\Python\Python313\python.exe" -c "C:\Users\User\AppData\Local\Programs\Python\Python313\python.exe" "C:\Users\User\Documents\Level(3)\Semester 1\Computer Network\Lab3_email\gui_client.py"
PS C:\Users\User\Documents\Level(3)\Semester 1\Computer Network\Lab3_email> c: cd 'C:\Users\User\Documents\Level(3)\Semester 1\Computer Network\Lab3_email' & "C:\Users\User\AppData\Local\Programs\Python\Python313\python.exe" "C:\Users\User\Documents\Level(3)\Semester 1\Computer Network\Lab3_email\main.py"
Latest email subject: PHP Test Email
PS C:\Users\User\Documents\Level(3)\Semester 1\Computer Network\Lab3_email>

```

Info	Length	Protocol	Destination	Source	Time	No
Application Data 1466	1466	TLSv1.3	192.168.10.12	64.233.167.108	882.288501	12016
Application Data 1466	1466	TLSv1.3	192.168.10.12	64.233.167.108	882.288501	12017
Application Data 1466	1466	TLSv1.3	192.168.10.12	64.233.167.108	882.288501	12018
Application Data 1318	1318	TLSv1.3	192.168.10.12	64.233.167.108	882.288501	12019
Seq=802 Ack=16295 Win=65280 Len=0 [ACK] 993 → 58525 54		TCP	64.233.167.108	192.168.10.12	882.288779	12020
Application Data 103	103	TLSv1.3	64.233.167.108	192.168.10.12	882.290099	12021
Seq=16295 Ack=851 Win=269312 Len=0 [ACK] 58525 → 993 54		TCP	192.168.10.12	64.233.167.108	882.356698	12022
Application Data 963	963	TLSv1.3	192.168.10.12	64.233.167.108	882.434992	12023
Seq=851 Ack=17204 Win=64512 Len=0 [ACK] 993 → 58525 54		TCP	64.233.167.108	192.168.10.12	882.483483	12024
Application Data 94	94	TLSv1.3	192.168.10.12	64.233.167.108	882.796374	12025
Application Data 90	90	TLSv1.3	64.233.167.108	192.168.10.12	882.827855	12026
Seq=17244 Ack=887 Win=269312 Len=0 [ACK] 58525 → 993 54		TCP	192.168.10.12	64.233.167.108	882.912607	12027
Application Data 132	132	TLSv1.3	192.168.10.12	64.233.167.108	882.946445	12028
Seq=887 Ack=17322 Win=64256 Len=0 [FIN, ACK] 993 → 58525 54		TCP	64.233.167.108	192.168.10.12	882.947240	12029
Seq=17322 Ack=887 Win=269312 Len=0 [FIN, ACK] 58525 → 993 54		TCP	192.168.10.12	64.233.167.108	882.979633	12030
Seq=888 Ack=17323 Win=64256 Len=0 [ACK] 993 → 58525 54		TCP	64.233.167.108	192.168.10.12	882.979730	12031
Seq=17323 Ack=888 Win=269312 Len=0 [ACK] 58525 → 993 54		TCP	192.168.10.12	64.233.167.108	883.015805	12032

0000	78 44 76 9b 89 44 8c f8 c5 8d 81 be 00 00 45 00	xDv...D...:.....E
0010	00 34 e9 bd 40 00 80 06 00 00 c0 a8 0a 0c 40 e9	4: @...:.....@
0020	a7 6c fb 93 03 e1 58 97 6e 56 00 00 00 00 00 02	-1...X: nV.....
0030	ff ff b3 30 00 00 02 04 05 b4 01 03 03 00 01 01	--0...:.....
0040	04 02	..

Profile: Default | Packets: 12595 - Displayed: 151 (1.2%)

Wireshark Wi-FiT83PH3.pcapng

4.3 TCP Notification Server

- Listens on port **5050**.
- Waits for a client connection.
- Sends a notification only when a **new email** is detected.
- Prevents duplicate notifications by tracking the last received email.

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\user\Documents\Level(3)\Semester 1\Computer Network\Labs Solution\Lab3_Email> python tcp_server.py
Notification Server started...
Client connected: ('127.0.0.1', 50241)

```

4.4 TCP Notification Client

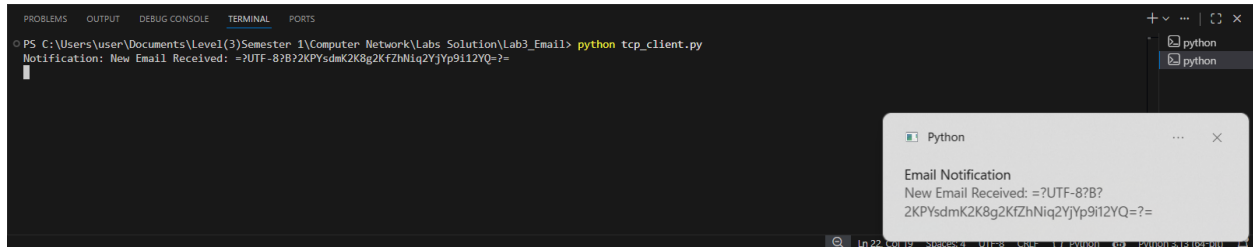
- Connects to the TCP server.
- Listens continuously for incoming notifications.
- Displays the received email subject.
-

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\user\Documents\Level(3)\Semester 1\Computer Network\Labs Solution\Lab3_Email> python tcp_client.py
Notification: New Email Received: SMTP Test Email

```

A push notification system was implemented using the Plyer library. When a new email arrives, the client displays a desktop notification instead of only printing the message to the console.



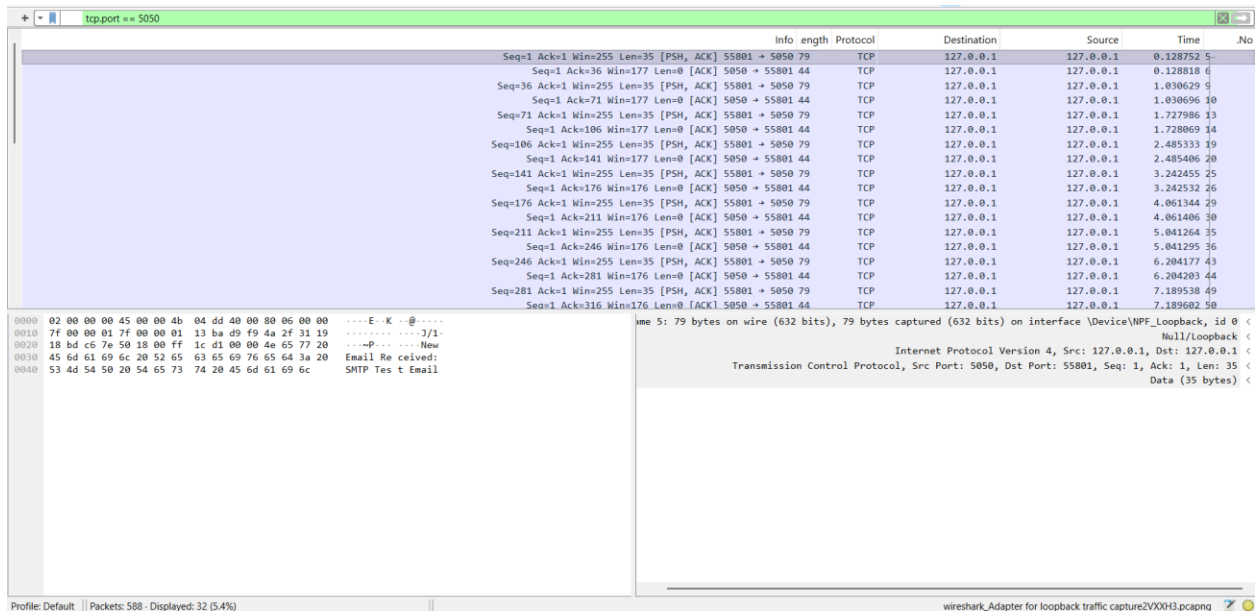
5- Wireshark Analysis

Wireshark was used to capture and analyze TCP traffic between the notification server and client.

- Since both client and server run on the same machine, **Loopback Adapter** was used.
- TCP packets were filtered using:

```
tcp.port == 5050
```

This confirmed successful TCP communication and data transmission.



6- Table (Protocol Summary)

Component	Protocol	Port	Description
Email Sending	SMTP	465	Sends emails using Gmail SMTP server
Email Receiving	IMAP	993	Checks inbox for new emails
Notification Server	TCP	5050	Sends real-time email notifications
Notification Client	TCP	5050	Receives notifications from server

7- Conclusion

- ❑ This project successfully demonstrates the integration of email protocols with TCP-based client-server communication.

- ❑ The system provides real-time notifications upon receiving new emails and verifies data transmission using Wireshark.
- ❑ It reflects a practical understanding of networking concepts and application layer protocols.

Scan QR_Code To Get Source Code:

