

# Tarea 08: Tarea Python con tkinter y Gmail

Alejandra Magalí López Miranda, 201600085<sup>1,\*</sup>

<sup>1</sup>*Escuela de Ingeniería Mecánica Eléctrica, Facultad de ingeniería, Universidad de San Carlos, Guatemala.*  
(Dated: 13 de septiembre de 2025)

## I. INTRODUCCIÓN

Desarrollar una aplicación de correo electrónico en Python que permita a los usuarios enviar y recibir mensajes a través de Gmail, mediante una interfaz gráfica sencilla y funcional.

## II. OBJETIVOS

### A. General

- Desarrollar una aplicación de correo electrónico en Python que permita a los usuarios enviar y recibir mensajes a través de Gmail, mediante una interfaz gráfica sencilla y funcional.

### B. Específicos

- \* Implementar el envío de correos electrónicos: Crear una función que permita al usuario redactar y enviar correos desde su cuenta de Gmail de forma segura utilizando SMTP.
- \* Implementar la recepción de correos electrónicos: Desarrollar una función que recupere los correos más recientes del usuario mediante IMAP y los muestre en la interfaz gráfica.
- \* Diseñar una interfaz gráfica intuitiva: Utilizar Tkinter para crear una GUI que facilite la interacción del usuario con las funciones de envío y recepción de correos, mostrando la información de manera clara y organizada.

## III. FUNCIONALIDAD

### A. Inicio del programa

- Se importa Tkinter para la interfaz gráfica.
- Se importan smtplib e imaplib para enviar y recibir correos.
- Se define la configuración de Gmail (SMTP para enviar, IMAP para recibir) y las credenciales del usuario.

### B. Funcion para enviar correos

- Obtener destinatario, asunto y mensaje desde la interfaz.
- Conectarse al servidor SMTP de Gmail y establecer conexión segura con starttls().
- Iniciar sesión con la cuenta y contraseña.
- Crear el correo con asunto y cuerpo y enviarlo al destinatario.
- Mostrar un mensaje de éxito o error según corresponda.

### C. Funcion para recibir correos

- Conectarse al servidor IMAP de Gmail y seleccionar la bandeja de entrada.
- Buscar los últimos correos
- Para cada correo: extraer remitente, asunto y cuerpo.
- Mostrar los correos recuperados en un área de texto de la interfaz.
- Cerrar sesión y manejar posibles errores.

### D. Interfaz grafica (Tkinter)

- Crear campos de entrada para remitente (solo lectura), destinatario y asunto.
- Crear un área de texto para escribir el mensaje.
- Botones para “Enviar” y “Recibir correos”.
- Área de texto para mostrar los correos recibidos.
- Ejecutar el bucle principal de la ventana (main-loop).

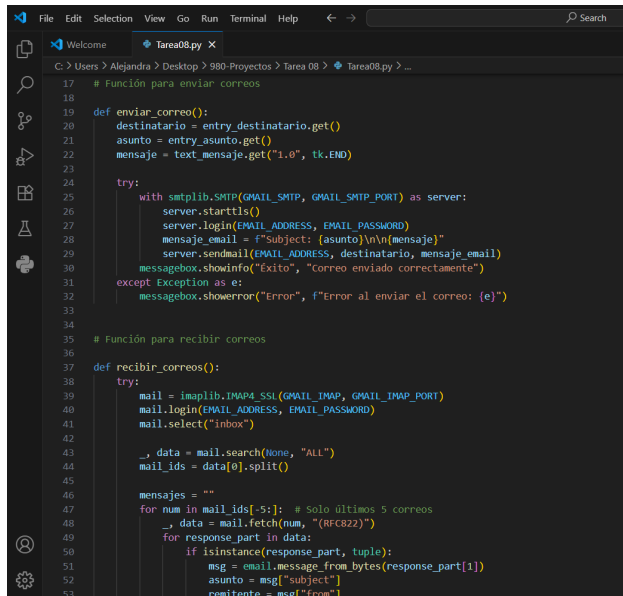
---

\* 2816733570108@ingenieria.usac.edu.gt

## IV. RESULTADOS

### A. Código en Python

Figura 1



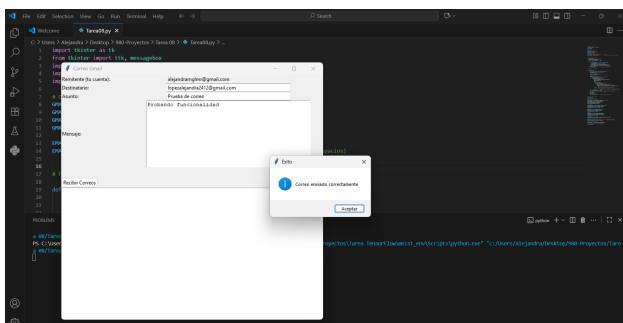
```

17 # Función para enviar correos
18
19 def enviar_correo():
20     destinatario = entry_destinatario.get()
21     asunto = entry_asunto.get()
22     mensaje = text_mensaje.get("1.0", tk.END)
23
24     try:
25         with smtplib.SMTP(GMAIL_SMTP, GMAIL_SMTP_PORT) as server:
26             server.starttls()
27             server.login(EMAIL_ADDRESS, EMAIL_PASSWORD)
28             mensaje_email = f"Subject: {asunto}\n\n{mensaje}"
29             server.sendmail(EMAIL_ADDRESS, destinatario, mensaje_email)
30             messagebox.showinfo("Éxito", "Correo enviado correctamente")
31     except Exception as e:
32         messagebox.showerror("Error", f"Error al enviar el correo: {e}")
33
34
35 # Función para recibir correos
36
37 def recibir_correos():
38     try:
39         mail = imaplib.IMAP4_SSL(GMAIL_IMAP, GMAIL_IMAP_PORT)
40         mail.login(EMAIL_ADDRESS, EMAIL_PASSWORD)
41         mail.select("inbox")
42
43         data = mail.search(None, "ALL")
44         mail_ids = data[0].split()
45
46         mensajes = ""
47         for num in mail_ids[-5:]: # Solo últimos 5 correos
48             data = mail.fetch(num, "(RFC222)")
49             for response_part in data:
50                 if isinstance(response_part, tuple):
51                     msg = email.message_from_bytes(response_part[1])
52                     asunto = msg["subject"]
53                     remitente = msg["from"]

```

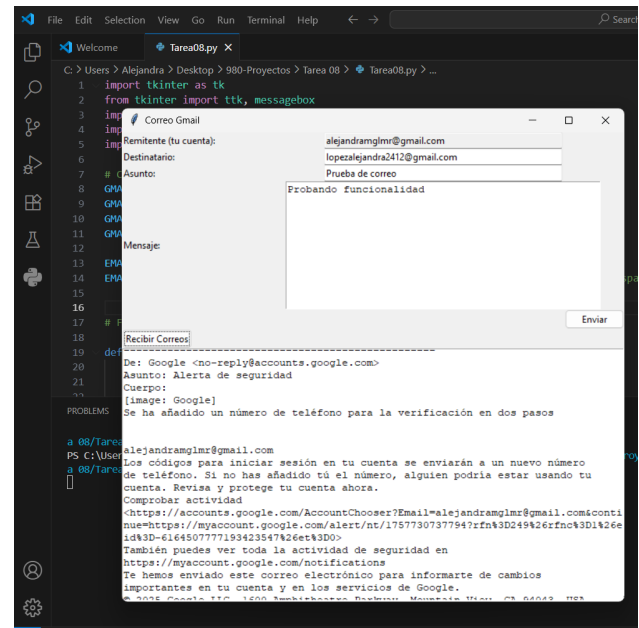
### B. Correo enviado

Figura 2



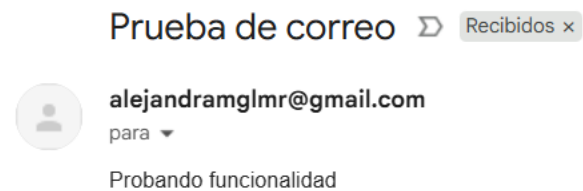
### C. Correos recibidos

Figura 3



### D. Correo recibido en gmail

Figura 4



## V. CONCLUSIONES

- \* Se desarrolló con éxito una aplicación capaz de enviar y recibir mensajes a través de Gmail, evidenciando cómo Python puede integrarse eficientemente con los protocolos de correo electrónico SMTP e IMAP.
- \* La interfaz gráfica construida con Tkinter mejora la experiencia del usuario, permitiendo manejar funciones complejas de correo de forma sencilla y sin necesidad de conocimientos técnicos avanzados.
- \* El uso de contraseñas de aplicación permite que las aplicaciones externas accedan a la cuenta de correo de manera segura, sin exponer la contraseña principal, garantizando la privacidad y el correcto funcionamiento.

cionamiento de los protocolos de envío y recepción de correos.

## **VI. ANEXOS**

Link de repositorio: <https://github.com/AlejandraMLM/980-Proyectos.git>

---