Documentación

1. **Objetivos**

El trabajo busca poder manejar, agilizar y gestionar un cliente de correos, en Python, un lenguaje de POO (programming oriented object / programación orientada objetos) y que este permita la gestión de usuarios, mensajes, carpetas, filtros y las operaciones básicas que se pueden realizar.

1. **Pautas**

Las entregas se harán cada 3 semanas aprox de una manera secuencial donde por cada entrega se van a ir añadiendo nuevas condiciones/funcionalidades al proyecto para que vaya evolucionando.

1. **Requerimientos del sistema**

Para poder correr el código se va a necesitar tener algunos recursos instalados en su computadora

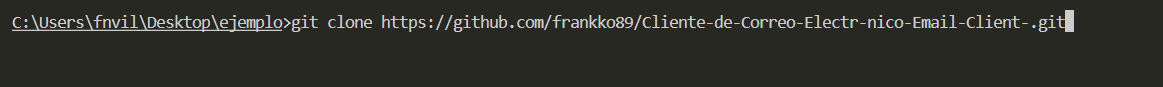
* + Python 3.13.7 o superior
  + Visual Studio Code instalado
  + Tener clonado el repositorio de Git Hub y crear una rama aparte

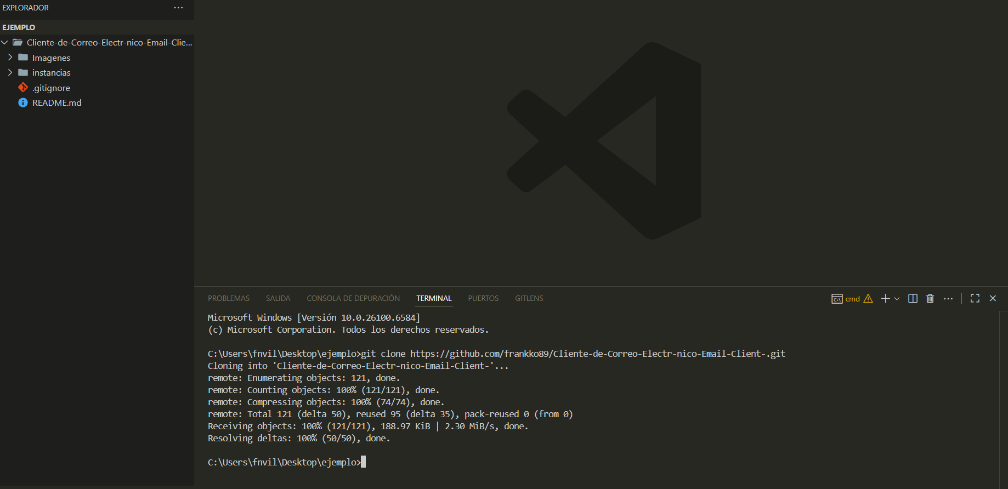
**Pasos para clonar el repositorio y crear una rama**

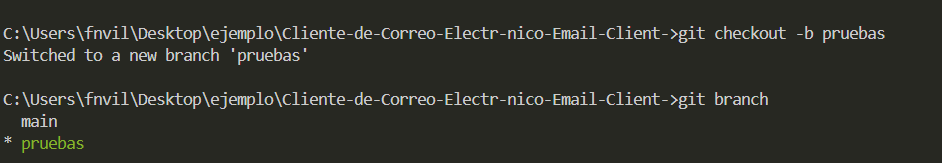
* 1. Entrar al repositorio de Git Hub en el siguiente link

<https://github.com/frankko89/Cliente-de-Correo-Electr-nico-Email-Client->

* 1. Una vez que tengas copiado el URL ir a su terminal de vs code o en el CMD (Ejecutado como administrador) y poner el siguiente comando



* 1. Luego de ejecutar esa línea va a tener clonado el repositorio y se vera de esta forma
  2. Crear una rama para poder trabajar y hacer pruebas sin afectar a la main



Acá creamos la rama para trabajar en una rama aparte y con el comando \* git Branch \* vamos a poder todas las ramas creadas y en cual estamos ubicados para trabajar.

1. **Arquitectura y diseño**

La arquitectura del proyecto se definió por medio de un UML (Lenguaje unificado modelado), el cual se baso en un diagrama de clases que nos ejemplificada de una manera más fácil como son las relaciones y los métodos que van a tener cada parte del proyecto.

**Descripción de Clases:**

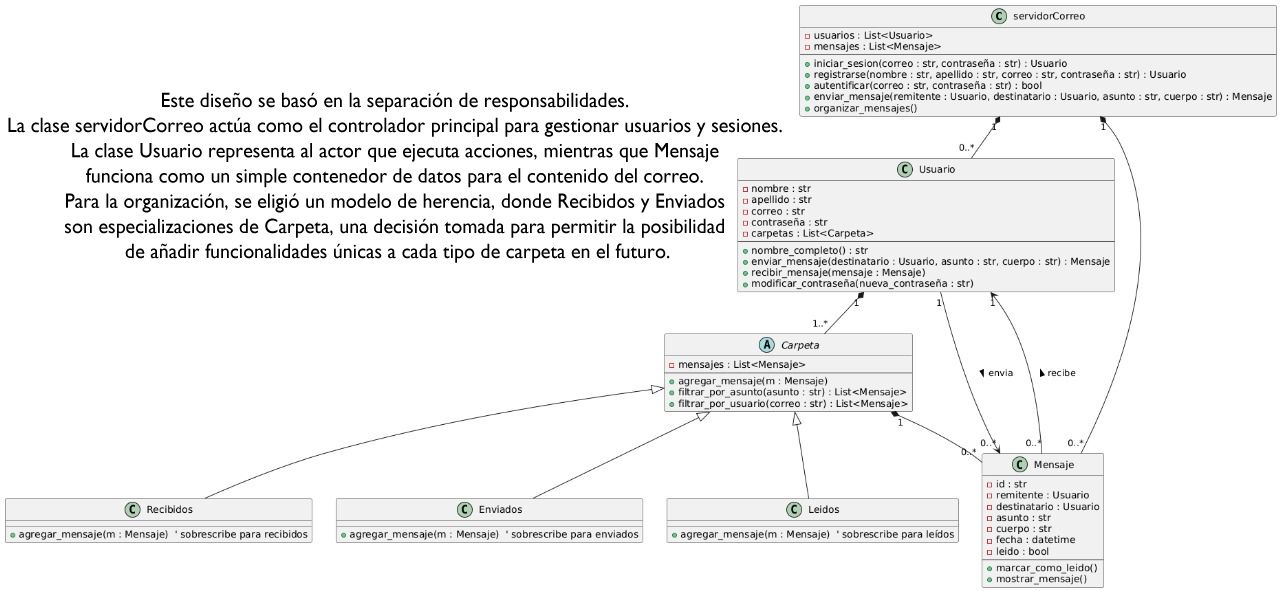
**ServidorCorreo**: La clase central del proyecto. Se encarga de la gestión de usuarios, la autenticación y el enrutamiento de mensajes. Contiene un diccionario de usuarios.

**Métodos principales:** registrarse(), iniciar\_sesion(), enviar\_mensaje().

**Usuario**: Representa a cada usuario del sistema. Contiene información personal y gestiona sus propias carpetas de correo (bandeja\_entrada, bandeja\_salida).

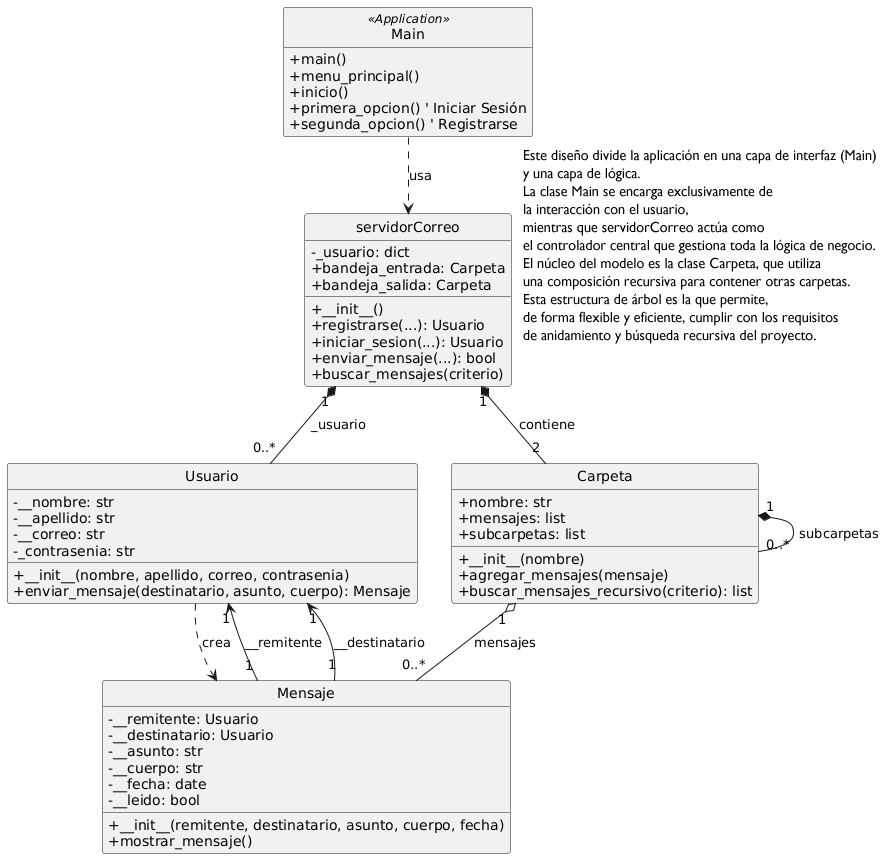
**Mensaje**: Un objeto simple que almacena toda la información de un correo electrónico, como remitente, destinatario, asunto, y cuerpo del mensaje.

**Carpeta**: Una clase que actúa como un contenedor de objetos Mensaje. Proporciona métodos para agregar, listar y filtrar mensajes. Las carpetas Recibidos, Enviados y Leidos son especializaciones de esta clase.

Primer Diseño del UML (Primera Entrega):  


## Segunda Entrega

Decidimos hacer un nuevo UML, agregando main (que maneja la aplicación principal) y modificando Carpeta para que el buscador que funciona por asunto o usuario tenga una sola función de filtrado que funcione de forma recursiva.



A la hora de agregar el sistema de árboles generales, decidimos que para crear las subcarpetas se crearía una carpeta principal para cada usuario, de la cual surgirían tres subcarpetas “Bandeja de Entrada”, “Bandeja de Salida” y “Mensajes Archivados”. Dentro de cada una, los usuarios podrán crear otras subcarpetas con nombres personalizables.

Diseñamos este gráfico para que se entienda mejor la idea:

