**Examen Primer Bimestre**

**Integrante:** Jonatan Claudio

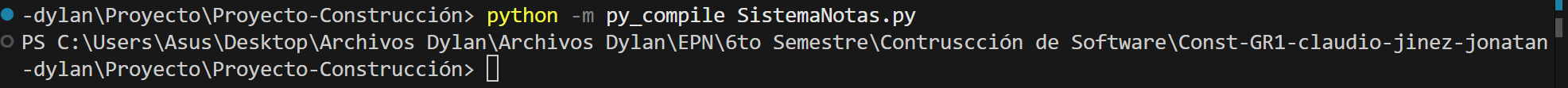
**Examen:** Sistema Simple de Registro de Notas

**Actividades del Examen:**

Realizar los "scripts" necesarios para establecer un flujo de calidad del código con los ejemplos que tenemos en clase, dependiendo del proyecto de cada grupo. Los grupos del proyecto son máximos de 2 estudiantes.

* **Compilar**

Dado que Python es un interprete no requiere de compilación, sin embargo, se realizó una verificación de error de sintaxis. Para ello se usó el comando: *python -m py\_compile SistemaNotas.py.*



Cuando se ejecutó esa línea de código se creo un archivo .pyc y además no se encontró ningún error.

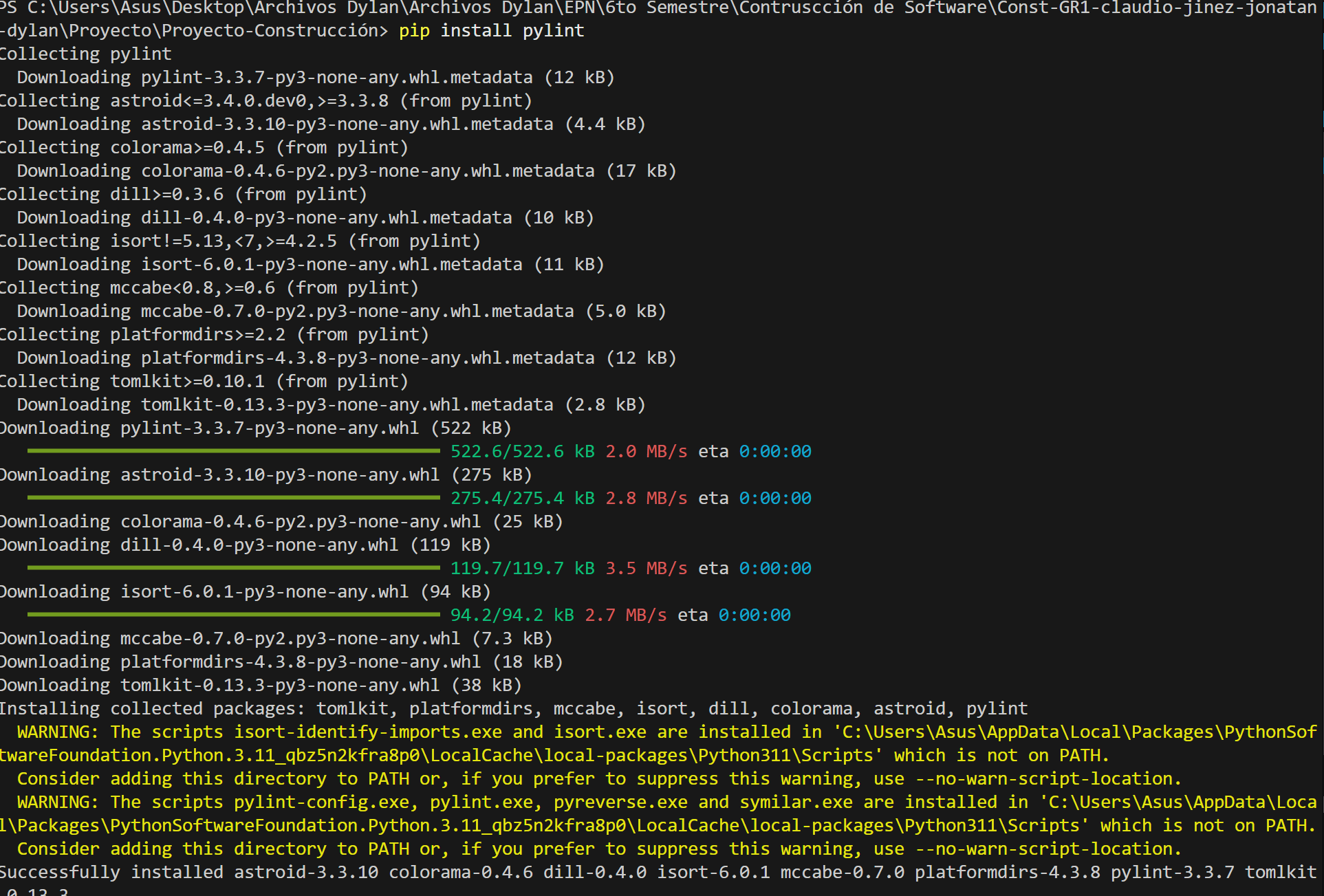
A screenshot of a computer

AI-generated content may be incorrect.

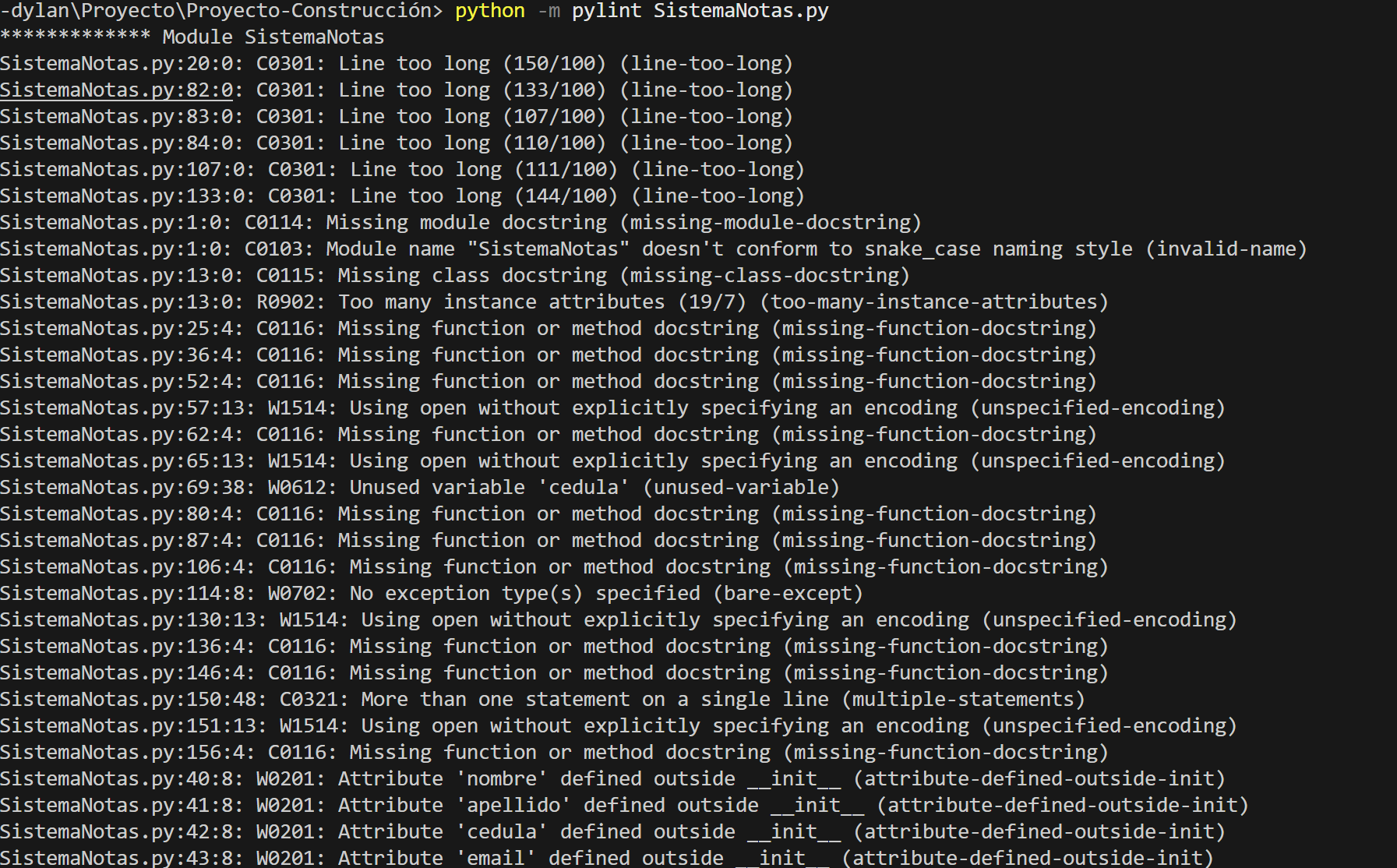
* **Lint**

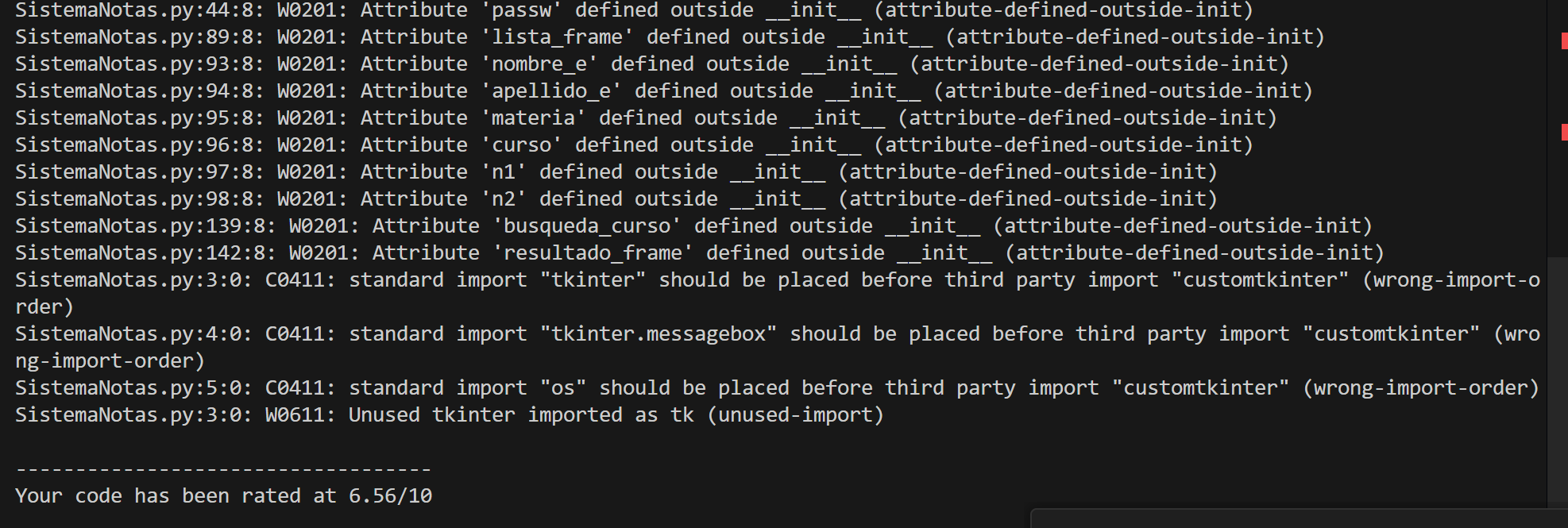
Para hacer la revisión de estilo y calidad usando lint en Python se ejecutaron varios comandos.

Primero instalamos Pylint usando el comando *pip install pylint.*



Luego, se ejecutó el comando *python -m pylint SsistemaNotas.py*





Se pidió a chatgpt que interpretara cada uno de los comentarios obtenidos.

**¿Qué significa el resultado?**

Tu código obtuvo una calificación de 6.56/10, lo cual es un buen punto de partida, pero puede mejorar mucho con pequeñas correcciones. Aquí están los puntos clave:

**Errores y advertencias más comunes**

Líneas muy largas (C0301): Trata de no superar los 100 caracteres por línea.

Falta de docstrings (C0114, C0115, C0116): Cada clase y función debería tener una pequeña descripción.

Demasiados atributos en una clase (R0902): Podrías considerar dividir responsabilidades

Archivos abiertos sin codificación (W1514): Siempre usa encoding="utf-8" al abrir archivos.

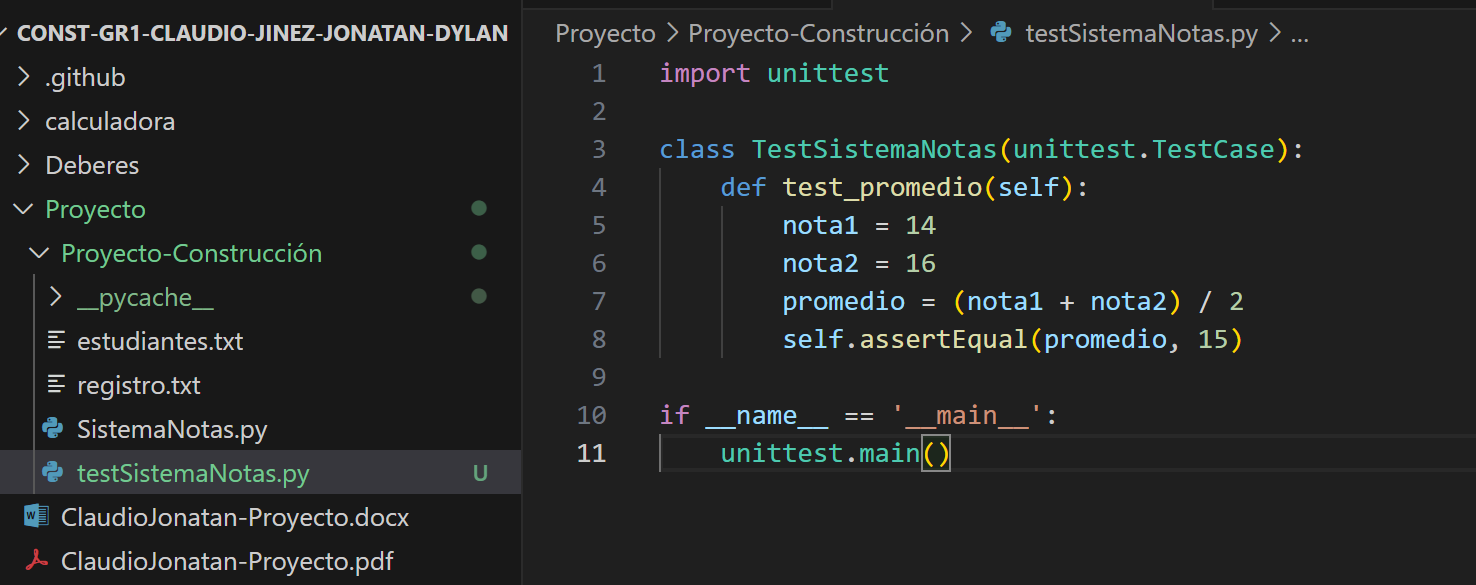
Mal orden de imports (C0411): Primero van los estándares (tkinter, os), luego terceros (customtkinter).

Atributos definidos fuera de \_\_init\_\_ (W0201): Mejor definir todo en el constructor.

Importaciones no usadas (W0611): Como tkinter as tk, si no lo usas, elimínalo.

* **Pruebas unitarias y/o de integración**

Ahora, para realizar las pruebas unitarias creamos un archivo de pruebas llamado testSistemaNotas.py con el siguiente código.

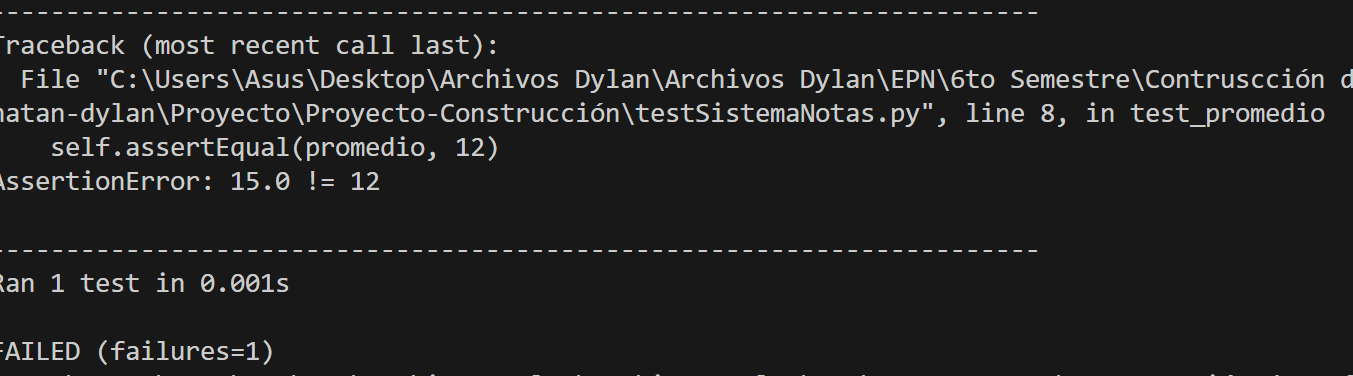


Una vez creado el archivo de pruebas unitarias, se lo ejecutara con el comando python -m unittest testSistemaNotas.py.

A black screen with white text

AI-generated content may be incorrect.

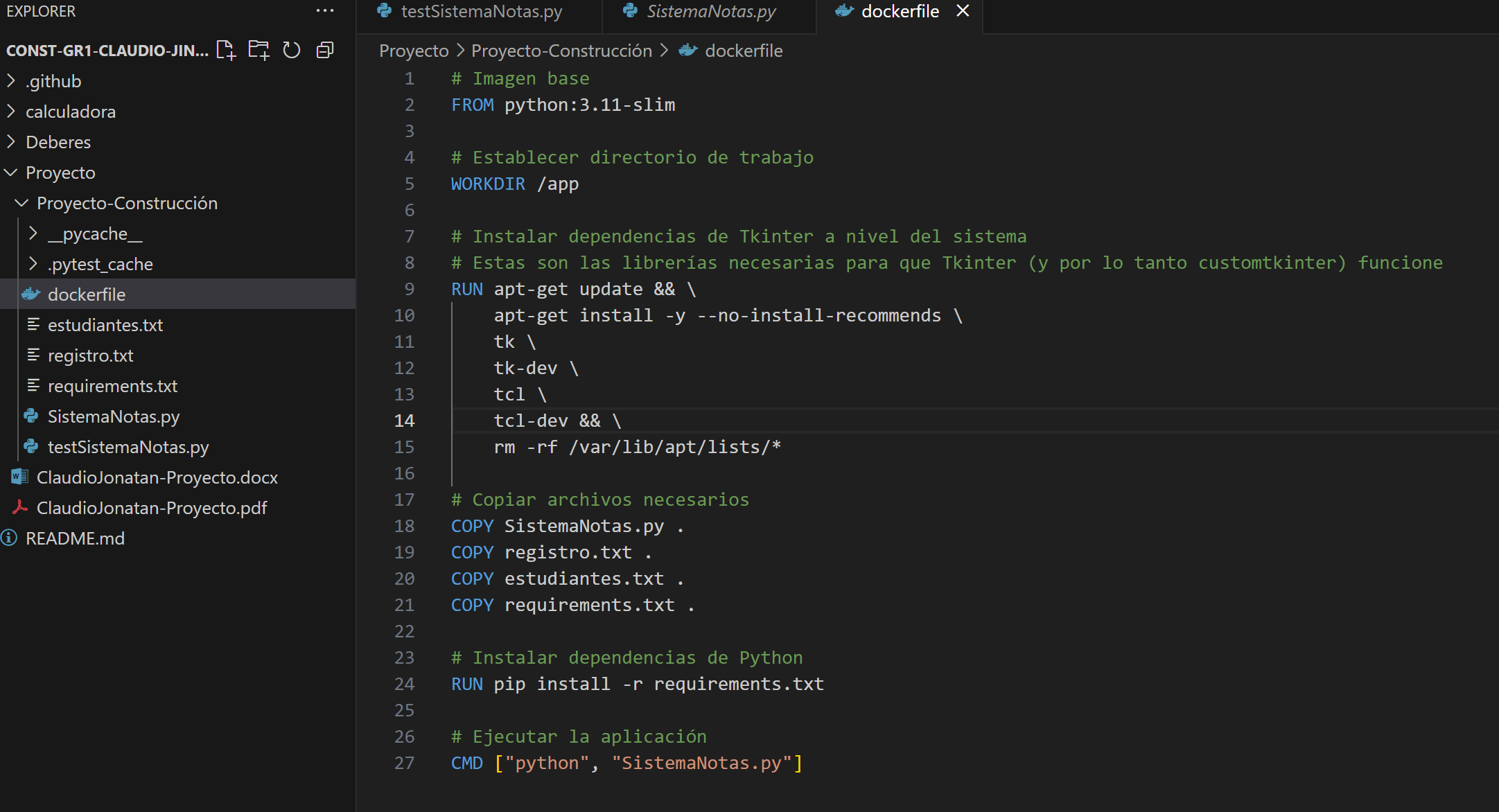
Ahora si se modifica para que el resultado sea incorrecto.



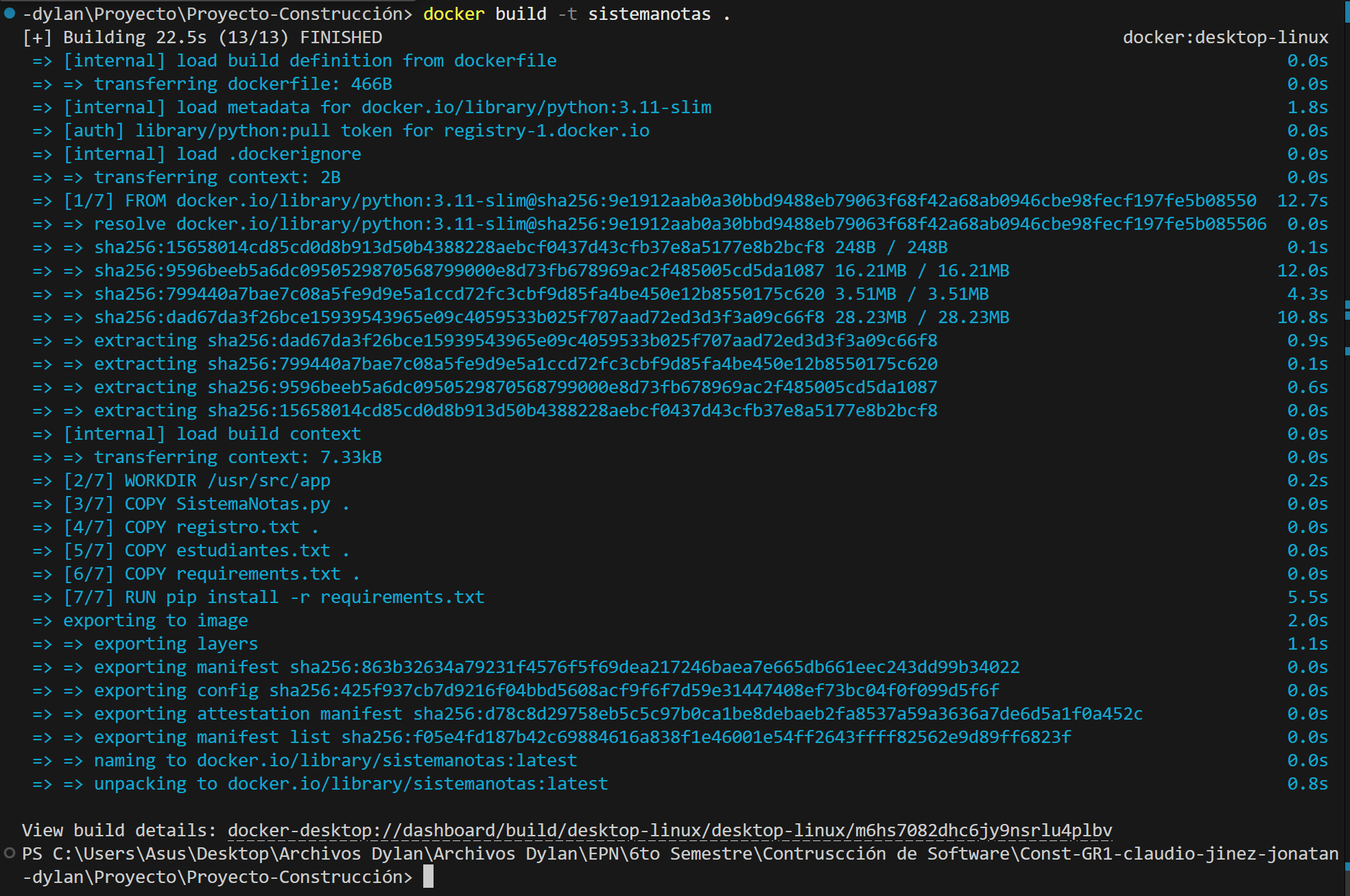
De esta forma las pruebas unitarias cumplen con su objetivo y además se sabe que el sistema cumple su función correctamente.

* **Dockerfile para despliegue**

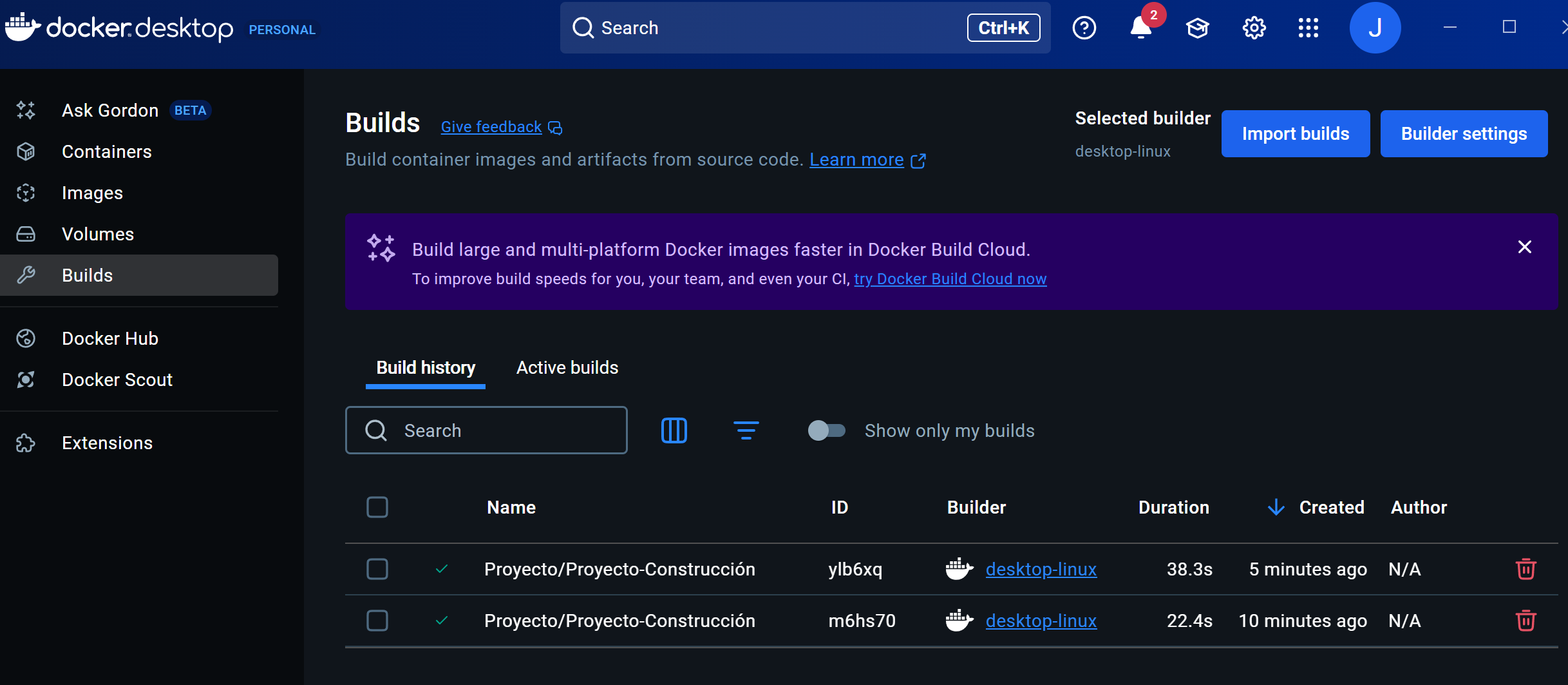
Creamos el archivo dockerfile.



Y lo ejecutamos usando el comando docker build -t sistemanotas .



Finalmente, se usó este comando: docker run -it sistemanotas .



**Sin embargo, cabe destacar que, al ser una aplicación con interfaz gráfica, Docker no la puede abrir. Por lo que se pensara en migrar la aplicación a web.**

* **Generar archivo. yml**

Por último, se creó el archivo yml

