

**Para trabajar con Python podemos hacerlo de tres formas diferentes ( en realidad existen más ), con environments, anaconda y en colab, ahora en tu ordenador utiliza las 3 formas y en tu github crea un documento donde explique como lo has hecho, incluye capturas de pantalla y explica como hacerlo, el documento debe quedar como si fuera un manual que utilizará un amigo tuyo, así que indica la url de descarga, los pasos a seguir y las notas que consideres necesarias.**

### **1. Crea dos enviroment en tu PC:**

**Para poder hacer las siguientes tareas tendrás que buscar en internet como instalar un paquete de python, como ver los paquetes que tenemos instalados de python.**

- **lista los paquetes python instalados en el primer environment**
- **instala un paquete en el 2º environment y lista los paquetes**

### **2. Instala Anaconda en tu PC**

- **Descarga e instala Anaconda**
- **Crea un notebook de jupyter, en esta**
  - **crea un titulo que diga mi primer programa**
  - **crea un codigo que imprima "mi primer programa"**

### **3. En tu hoja de github explica las diferencias (cons y pros) entre:**

- **Environment**
- **Anaconda**
- **Colab**

## 1. Crea dos enviroment en tu PC:

**Para poder hacer las siguientes tareas tendrás que buscar en internet como instalar un paquete de python, como ver los paquetes que tenemos instalados de python.**

- **lista los paquetes python instalados en el primer environment**
- **instala un paquete en el 2º environment y lista los paquetes**

-Para poder hacer las siguientes tareas tendrás que buscar en internet como instalar un paquete de python, como ver los paquetes que tenemos instalados de python.

# Crear el primer entorno virtual

```
python -m venv env1
```

# Activar el primer entorno virtual1

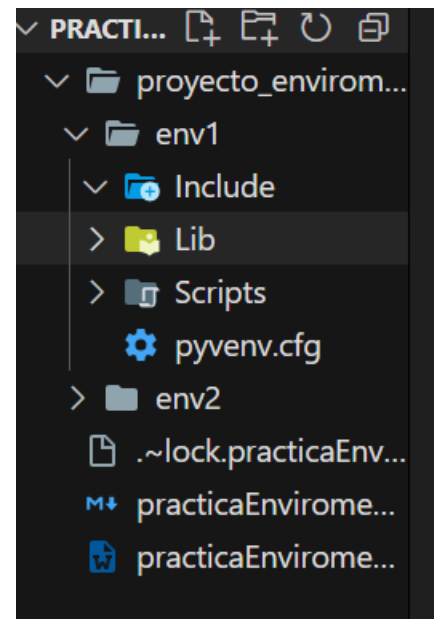
```
env1\Scripts\activate
```

# Crear el segundo entorno virtual

```
python -m venv env2
```

# Activar el segundo entorno virtual 2

```
env2\Scripts\activate
```



-Listar paquetes del primer enviroment entro en la carpeta de env1 y hago el siguiente comando pip list

Package	Version
-----	-----
aiohttp	3.8.5
aiosignal	1.3.1
async-timeout	4.0.2
attrs	23.1.0
cachetools	5.3.1
certifi	2023.7.22
charset-normalizer	3.2.0
click	8.1.7
colorama	0.4.6
contourpy	1.1.1
cycler	0.11.0
distlib	0.3.7
filelock	3.12.4
fonttools	4.42.1
frozenlist	1.4.0
google-auth	2.23.2
idna	3.4
importlib-metadata	6.8.0
importlib-resources	6.1.0
joblib	1.3.2
keras	2.14.0
keras-util	0.0.2
keras-utils	1.0.13
kiwisolver	1.4.5
Markdown	3.4.4
markdown-it-py	3.0.0
MarkupSafe	2.1.3
matplotlib	3.8.0
mdurl	0.1.2
multidict	6.0.4
numpy	1.26.0
oauthlib	3.2.2
openai	0.28.1
package-name	0.1
packaging	23.1
pandas	2.1.1
Pillow	10.0.1
pip	21.2.3
platformdirs	3.11.0
protobuf	4.24.4
pyasn1	0.5.0
pyasn1-modules	0.3.0
Pygments	2.16.1

Instalo pandas con pip install pandas

-Instala un paquete en el 2º enviroment y lista los paquetesentro en la carpeta de env2 y hago el siguiente comando pip list

Instalo matlib con pip install matlib

Asegúrate de estar dentro del entorno virtual adecuado cuando ejecutes estos comandos. Puedes desactivar un entorno virtual en cualquier momento con el comando:

Comando: deactivate

joblib	1.3.2
keras	2.14.0
keras-util	0.0.2
keras-utils	1.0.13
kiwisolver	1.4.5
Markdown	3.4.4
markdown-it-py	3.0.0
MarkupSafe	2.1.3
matplotlib	3.8.0
mdurl	0.1.2
multidict	6.0.4
numpy	1.26.0
oauthlib	3.2.2
openai	0.28.1
package-name	0.1
packaging	23.1
pandas	2.1.1
Pillow	10.0.1
pip	21.2.3
platformdirs	3.11.0
protobuf	4.24.4
pyasn1	0.5.0
pyasn1-modules	0.3.0
Pygments	2.16.1
pyarsing	3.1.1
python-dateutil	2.8.2
pytz	2023.3.p
requests	2.31.0
requests-oauthlib	1.3.1
rich	13.6.0
rsa	4.9
scikit-learn	1.1.3
scipy	1.11.2
setuptools	57.4.0
shellingham	1.5.3
six	1.16.0
somepackage	1.2.3
tensorboard-data-server	0.7.1
threadpoolctl	3.2.0
tqdm	4.65.0
typer	0.9.0
typing_extensions	4.8.0
tzdata	2023.3
urllib3	2.0.4
utils	1.0.1
virtualenv	20.24.5
Werkzeug	3.0.0
wheel	0.41.2
yaml	1.9.2
zipp	3.17.0

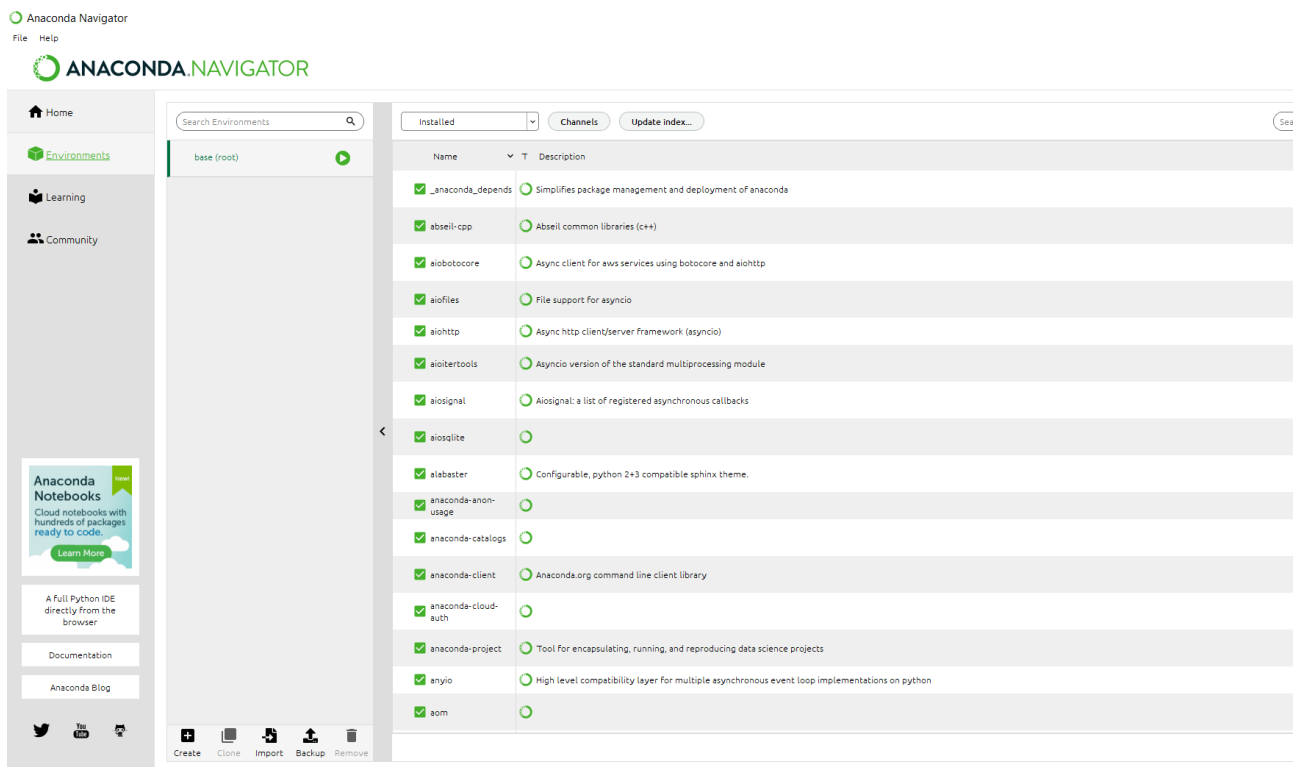
## 2. Instala Anaconda en tu PC

- **Descarga e instala Anaconda**
- **Crea un notebook de jupyter, en esta**
  - **crea un titulo que diga mi primer programa**
  - **crea un codigo que imprima "mi primer programa"**

Primero de todo instalamos el anaconda buscando por Google.

Después entramos en la pagina Oficial de Anaconda y descargamos el .exe.

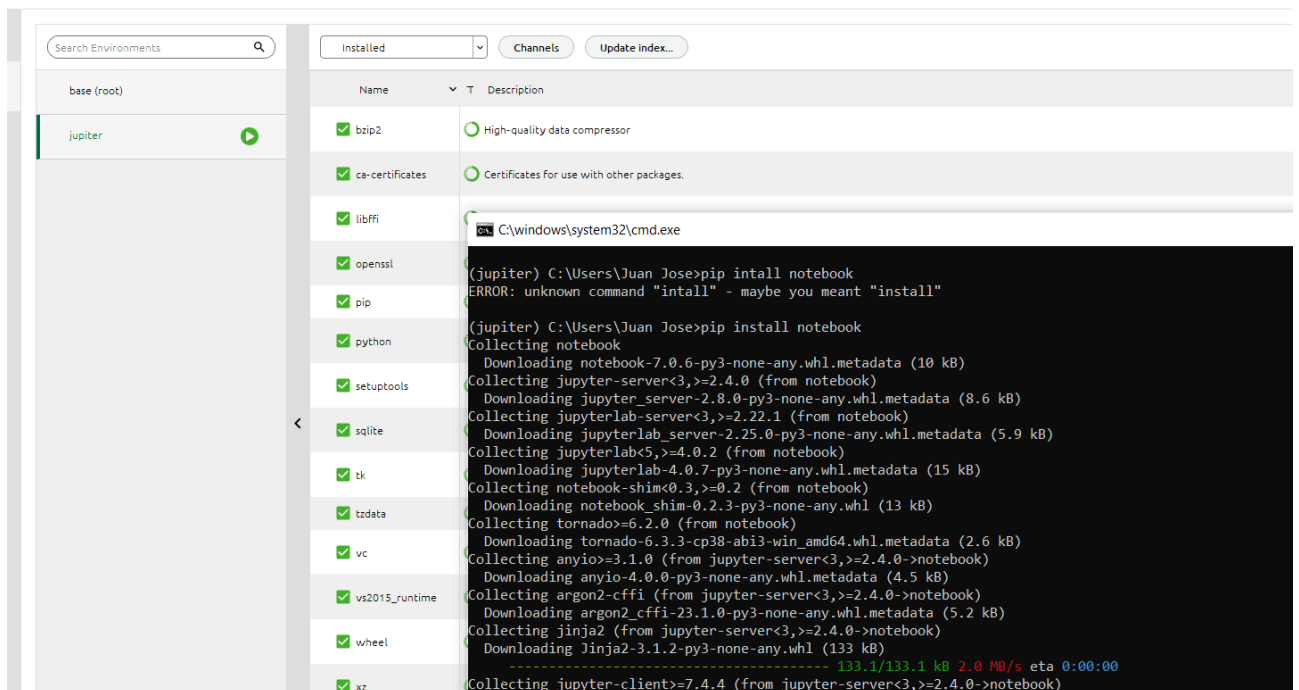
Le damos a Siguiente → Siguiente.



Y ya esta instalado el anaconda.

Creo un enviroment llamado Jupiter

NDA.NAVIGATOR



Para entrar en la terminal le doy al boton play.

Siguiente punto

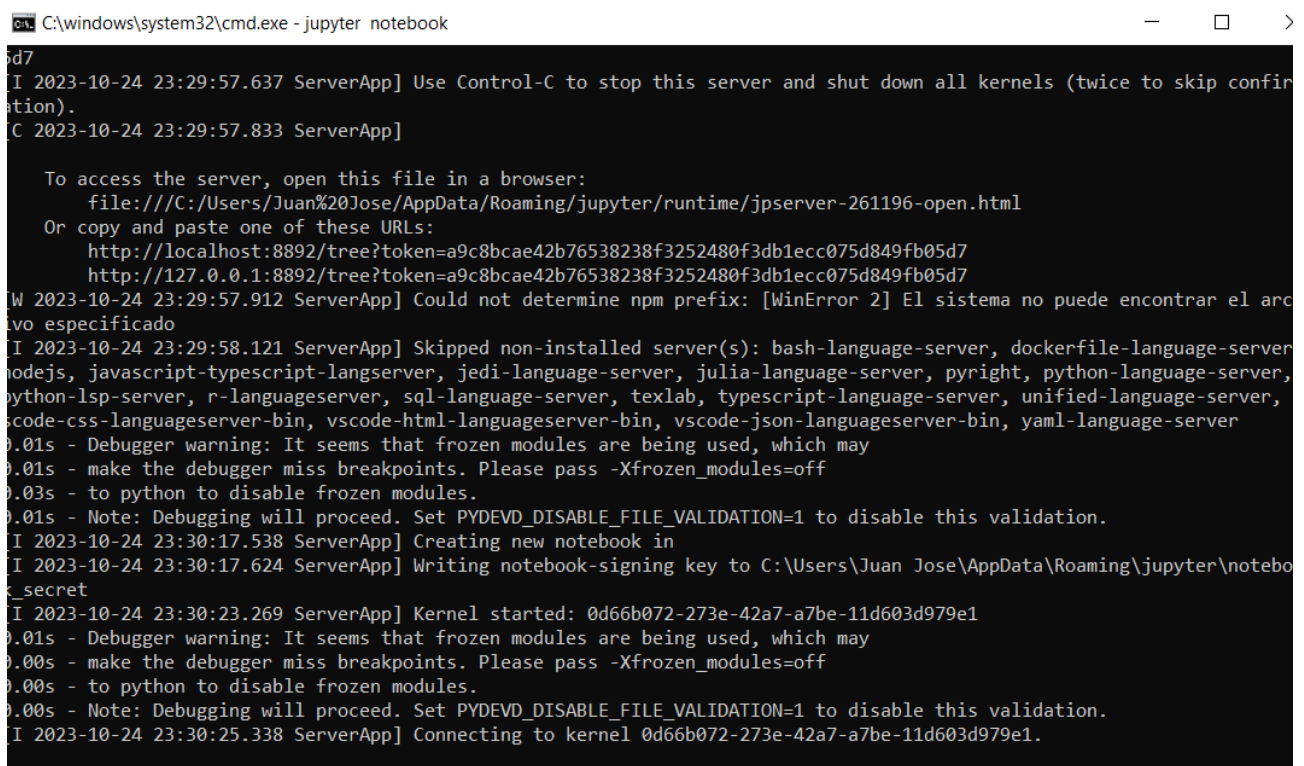
- **Crea un notebook de jupyter, en esta**
  - **crea un titulo que diga mi primer programa**
  - **crea un codigo que imprima "mi primer programa"**

Instalamos Jupiter notebook con el siguiente comando

```
pip install notebook
```

Después para desplegarlo

```
jupyter notebook
```

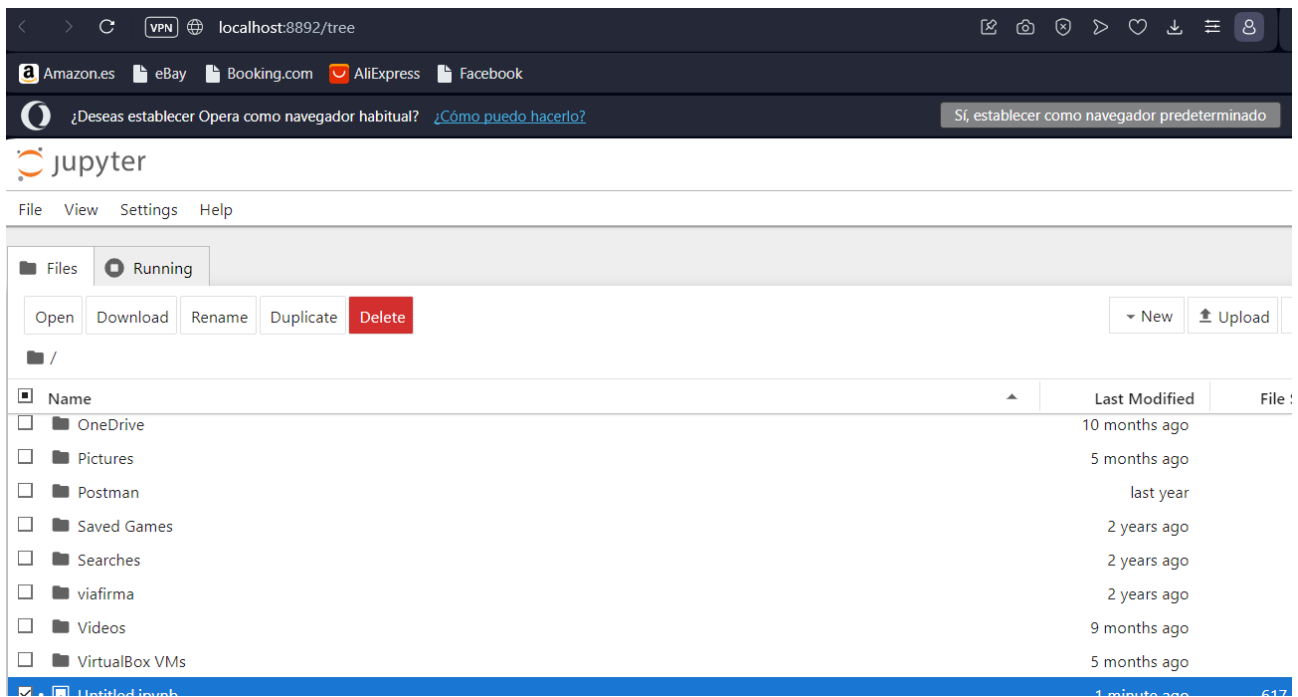


```
C:\windows\system32\cmd.exe - jupyter notebook
d7
[I 2023-10-24 23:29:57.637 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confir
ation).
C 2023-10-24 23:29:57.833 ServerApp]

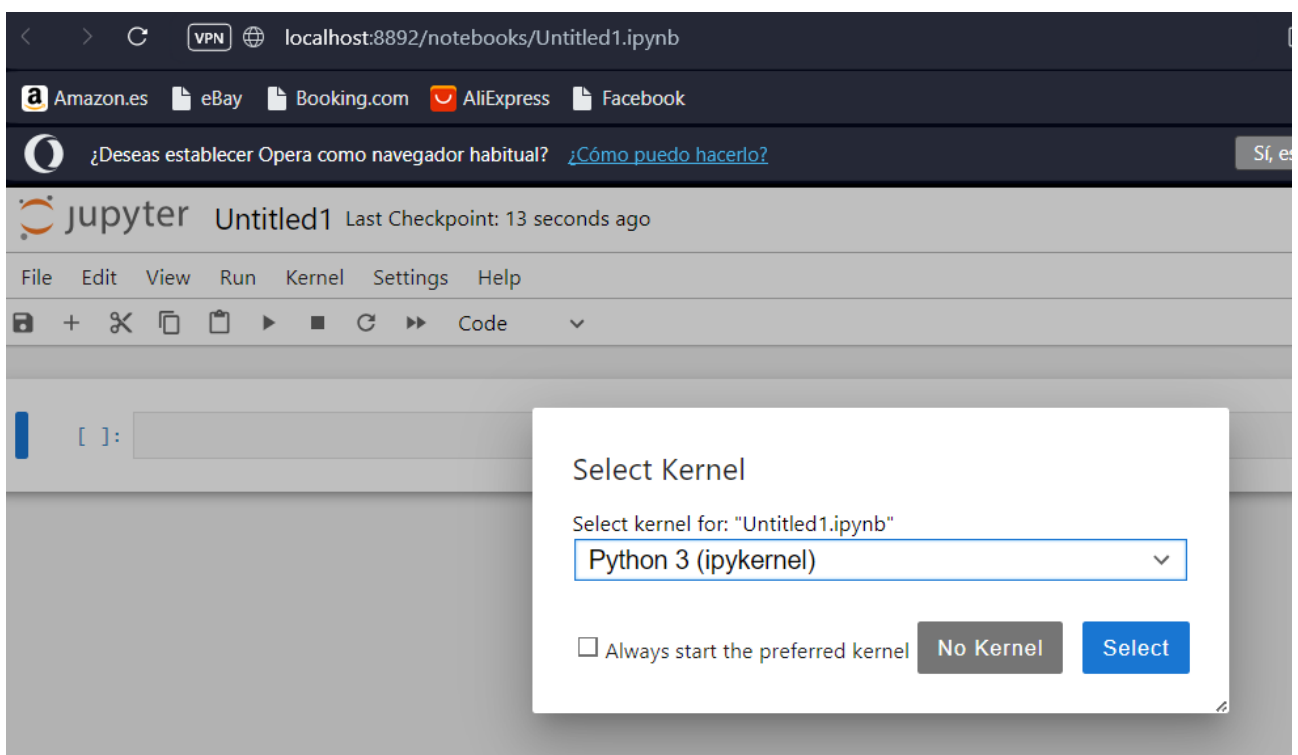
To access the server, open this file in a browser:
file:///C:/Users/Juan%20Jose/AppData/Roaming/jupyter/runtime/jpserver-261196-open.html
Or copy and paste one of these URLs:
http://localhost:8892/tree?token=a9c8bcae42b76538238f3252480f3db1ecc075d849fb05d7
http://127.0.0.1:8892/tree?token=a9c8bcae42b76538238f3252480f3db1ecc075d849fb05d7
[W 2023-10-24 23:29:57.912 ServerApp] Could not determine npm prefix: [WinError 2] El sistema no puede encontrar el arc
ivo especificado
[I 2023-10-24 23:29:58.121 ServerApp] Skipped non-installed server(s): bash-language-server, dockerfile-language-server
nodejs, javascript-typescript-langserver, jedi-language-server, julia-language-server, pyright, python-language-server,
python-lsp-server, r-languageserver, sql-language-server, texlab, typescript-language-server, unified-language-server,
vscode-css-languageserver-bin, vscode-html-languageserver-bin, vscode-json-languageserver-bin, yaml-language-server
0.01s - Debugger warning: It seems that frozen modules are being used, which may
0.01s - make the debugger miss breakpoints. Please pass -Xfrozen_modules=off
0.03s - to python to disable frozen modules.
0.01s - Note: Debugging will proceed. Set PYDEVD_DISABLE_FILE_VALIDATION=1 to disable this validation.
[I 2023-10-24 23:30:17.538 ServerApp] Creating new notebook in
[I 2023-10-24 23:30:17.624 ServerApp] Writing notebook-signing key to C:\Users\Juan Jose\AppData\Roaming\jupyter\notebo
k_secret
[I 2023-10-24 23:30:23.269 ServerApp] Kernel started: 0d66b072-273e-42a7-a7be-11d603d979e1
0.01s - Debugger warning: It seems that frozen modules are being used, which may
0.00s - make the debugger miss breakpoints. Please pass -Xfrozen_modules=off
0.00s - to python to disable frozen modules.
0.00s - Note: Debugging will proceed. Set PYDEVD_DISABLE_FILE_VALIDATION=1 to disable this validation.
[I 2023-10-24 23:30:25.338 ServerApp] Connecting to kernel 0d66b072-273e-42a7-a7be-11d603d979e1.
```

Esta captura es la muestra del comando anterior nos llevará al

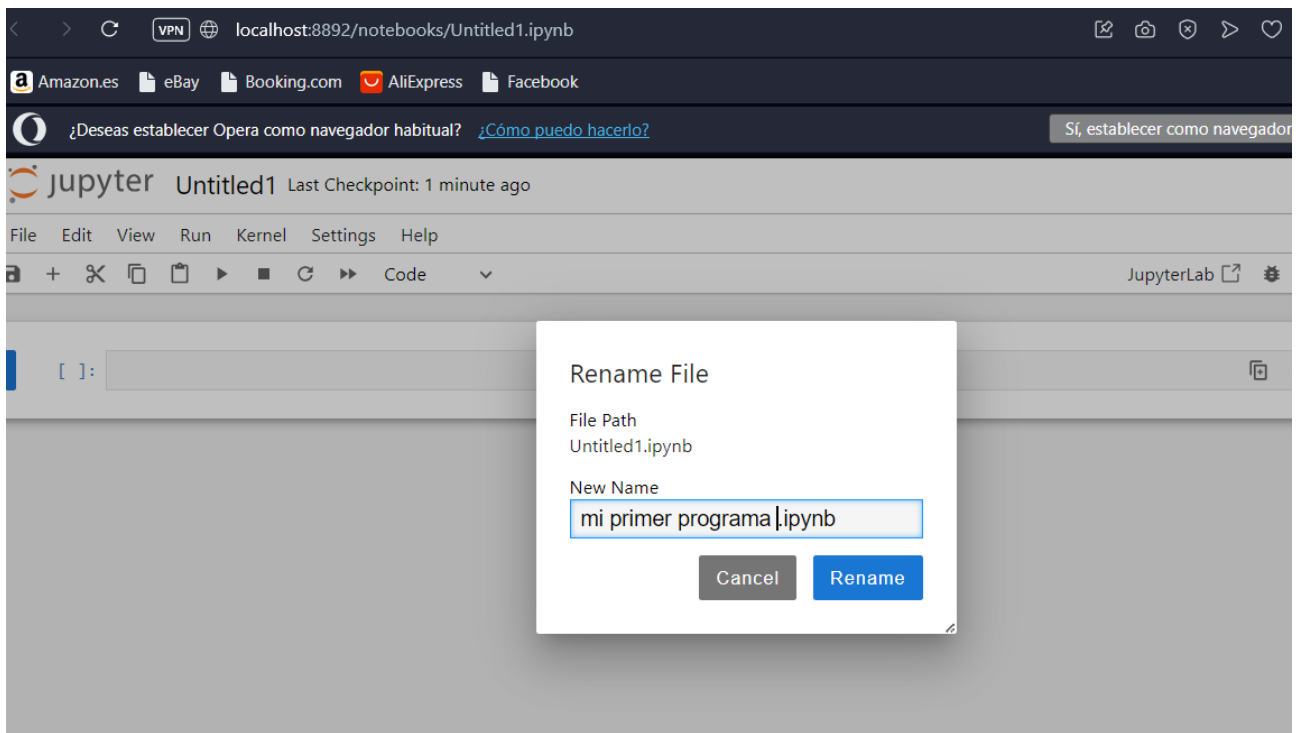
<http://localhost:8892/notebooks/Untitled.ipynb>



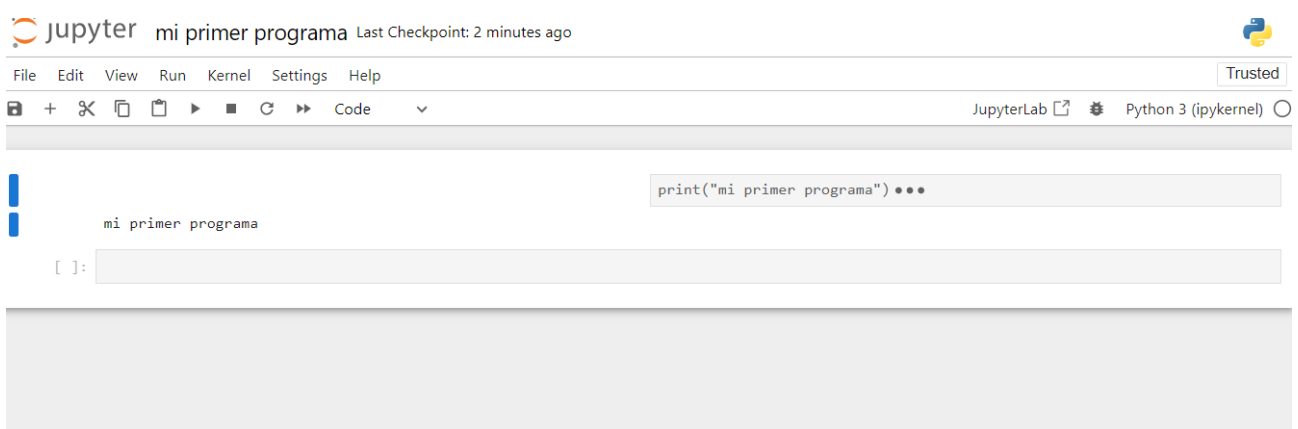
Ahora una vez dentro le damos a new → notebook y creamos un cuaderno.



Una vez que lo tengamos claro, seleccionamos el kernel.



Le ponemos el titulo del archivo.ipynb



programamos en python con print(“mi primer programa”) y ejecutamos dandole el botón azul de la izquierda.



### 3. En tu hoja de github explica las diferencias (cons y pros) entre:

- **Environment**
- **Anaconda**
- **Colab**

#### **Environment:**

- **Pros:**
  - Un "environment" (entorno) se refiere a un ambiente de desarrollo o ejecución de software aislado. Puede configurarse de forma personalizada con paquetes y bibliotecas específicos para un proyecto.
  - Proporciona control total sobre las bibliotecas, versiones y configuraciones utilizadas en un proyecto.
  - Ideal para proyectos en los que es importante administrar y controlar todas las dependencias.
- **Contras:**
  - Configurar y mantener "environments" puede requerir tiempo y experiencia técnica.
  - Puede ser complicado cuando se trabaja en múltiples proyectos con diferentes dependencias.

#### 2. **Anaconda:**

- **Pros:**
  - Anaconda es una distribución de Python que incluye una amplia gama de bibliotecas científicas y herramientas de análisis de datos.
  - Ofrece una gestión de "environments" fácil de usar a través de conda, lo que facilita la creación de entornos aislados con paquetes específicos.
  - Ampliamente utilizado en ciencia de datos y aprendizaje automático.
  - Disponible en múltiples sistemas operativos.
- **Contras:**
  - Puede ocupar espacio en disco debido a la gran cantidad de paquetes incluidos.

#### 3. **Colab (Google Colaboratory):**

- **Pros:**
  - Colab es una plataforma en línea basada en Jupyter Notebook que ofrece acceso gratuito a recursos de GPU y TPU de Google.
  - No se requiere configuración ni instalación, lo que lo hace conveniente para la colaboración y el acceso desde cualquier lugar con conexión a Internet.
  - Adecuado para la ejecución de código, análisis de datos y desarrollo de modelos de aprendizaje automático.
- **Contras:**

- La cantidad de recursos disponibles en Colab puede ser limitada en comparación con las instalaciones locales.
- Los entornos no son permanentes; los recursos se reinician después de un tiempo de inactividad.
- No es tan flexible como un "environment" local en términos de personalización y control de versiones de bibliotecas.