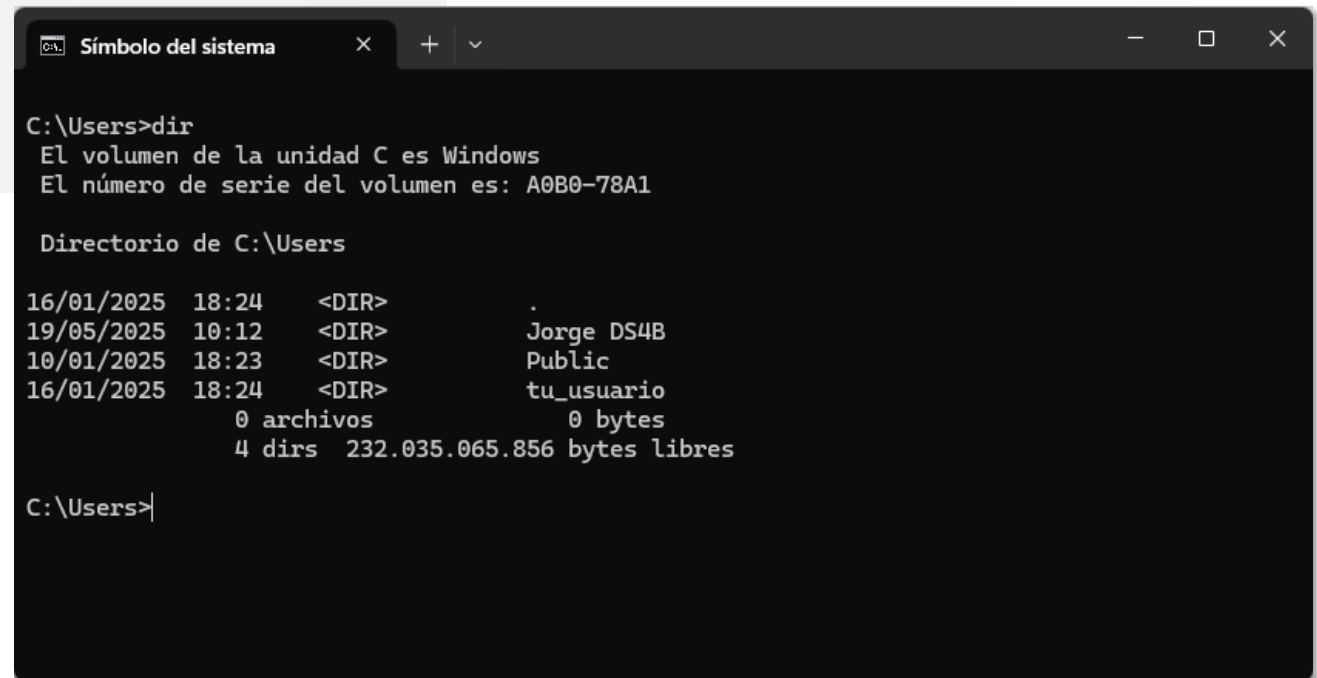


## ¿Para qué nos puede servir?

- Instalación y gestión de librerías y entornos.
- Organización de proyectos - Estructuras de carpetas.
- Ejecución de scripts.
- Lanzar apps en local.
- Trabajar con Git



```
C:\Users>dir
El volumen de la unidad C es Windows
El número de serie del volumen es: A0B0-78A1

Directorio de C:\Users

16/01/2025  18:24    <DIR>          .
19/05/2025  10:12    <DIR>          Jorge DS4B
10/01/2025  18:23    <DIR>          Public
16/01/2025  18:24    <DIR>          tu_usuario
                0 archivos                0 bytes
                4 dirs  232.035.065.856 bytes libres

C:\Users>
```

## Inconveniente:

Existen diferentes tipos de terminal, en función del sistema operativo y cada una con comandos diferentes.



### **PowerShell (Windows):**

Acceso rápido: (Win + R) y escribir powershell

---

Terminal avanzada de Windows con sintaxis orientada a objetos, ideal para automatización y scripting profesional.



### **Símbolo del sistema - CMD (Windows)**

Acceso rápido: (Win + R) y escribir CMD

---

Terminal clásica con comandos básicos, útil para tareas sencillas de archivos y navegación.



### **Terminal (macOS/Linux)**

Acceso rápido: (Cmd + espacio) y escribir terminal

Consola por defecto en sistemas basados en Unix, como macOS y Linux, muy utilizada para desarrollo, automatización y manejo del sistema.

## 1. Instalación y gestión de librerías

- Crear entornos

***conda create -n entorno1 python=3.11***

- Activar entornos

***conda activate entorno1***

- Instalar librerías:

***conda install pandas matplotlib seaborn***

```
Anaconda Prompt - conda x + v
(base) C:\Users\Jorge DS4B>conda activate pdsm_foundations

(pdsm_foundations) C:\Users\Jorge DS4B>conda list
# packages in environment at C:\Users\Jorge DS4B\.conda\envs\pdsm_foundations:
#
# Name                          Version          Build      Channel
_openmp_mutex                   4.5              2_gnu      conda-forge
anyio                           4.9.0            pyh29332c3_0 conda-forge
argon2-cffi                     23.1.0           pyhd8ed1ab_1 conda-forge
```

## 2. Organización de proyectos

- Crear y estructurar carpetas para un nuevo proyecto.
- Navegar y moverse entre carpetas.
- Ver y comprobar dónde estás (pwd, cd, ls, mkdir, tree, dir).

### *Ejemplos:*

1. Crear una carpeta proyecto\_ds y después cambiar el directorio para trabajar sobre esta carpeta recién creada:

***mkdir proyecto\_ds && cd proyecto\_ds***

2. Crear 4 carpetas al mismo nivel dentro del directorio actual:

***mkdir datos scripts notebooks resultados***

### 3. Ejecución de scripts

- Ejecutar un script de Python sin necesidad de abrir Jupyter Notebook ni otro IDE

#### *Ejemplo:*

1. Ejecutar el script de entrenamiento del modelo de ML, ubicado en la carpeta procesos:

***python procesos/entrenamiento.py***

#### **Variables de entorno - PATH**

En este punto entra en juego el concepto de **PATH**, una variable de entorno que almacena una lista de rutas donde la terminal busca los programas o comandos que intentamos ejecutar.

Si el intérprete de Python, o herramientas como pip o conda, no están incluidas en el PATH, la terminal no podrá ejecutarlas, aunque estén correctamente instaladas en el sistema.

## Variables de entorno PATH

### ¿Cómo solucionarlo?

En Windows existen dos alternativas:

- Incluir manualmente la ruta de instalación del programa (Python, conda, etc) a la variable de entorno PATH.
- Usar una terminal ya configurada, como Miniconda, donde Python y sus herramientas ya las tenemos disponibles directamente.

En macOS normalmente al instalar Miniconda , se añaden automáticamente sus rutas al PATH, lo que permite ejecutar los comandos sin problemas.

Si usamos VS Code, podremos configurar el entorno desde el editor para que reconozca automáticamente la ruta correcta del intérprete de Python.

## 4. Lanzar Apps en local

- Hacer pruebas de desarrollo simulando un servidor en local (Streamlit o FastApi)

### Streamlit:

Librería que permite crear **Interfaces web interactivas** (formularios, visualizaciones).  
Es ideal para compartir modelos, análisis o prototipos sin necesidad de programar HTML o JS.

### FastAPI:

Es un framework que actúa como **punto entre aplicaciones**, permitiendo que se comuniquen de forma rápida y segura.

### *Ejemplo:*

1. Lanzar la app comparador.py para probar las configuraciones efectuadas y comprobar su interfaz:

***streamlit run comparador.py***

## 5. Trabajar con Git

- Inicializar repositorio y versionar archivos (git init, git add, git commit).
- Subir código a GitHub (git push).
- Descargar proyectos (git clone).

### *Ejemplos:*

1. Guardar el cambio del archivo analisis.py en el índice de Git:

***git add scripts/analisis.py***

2. Crear un punto de control en el historial de Git con todos los archivos añadidos y con el mensaje explicativo: “Versión final del análisis”:

***git commit -m "Versión final del análisis"***



Acción	macOS/Linux (Terminal)	Windows (CMD)	Windows (PowerShell)
<b>Mostrar ubicación actual</b>	<code>pwd</code>	<code>cd</code>	<code>pwd</code>
<b>Listar archivos/directorios</b>	<code>ls</code>	<code>dir</code>	<code>ls, dir</code>
<b>Cambiar directorio</b>	<code>cd ruta</code>	<code>cd ruta</code>	<code>cd ruta</code>
<b>Limpiar pantalla</b>	<code>clear</code>	<code>cls</code>	<code>Clear</code>
<b>Ver carpeta del usuario actual</b>	<code>echo \$HOME</code>	<code>echo %USERPROFILE%</code>	<code>echo \$env:USERPROFILE</code>
<b>Crear carpeta</b>	<code>mkdir nombre</code>	<code>mkdir nombre</code>	<code>mkdir nombre</code>
<b>Crear archivo vacío</b>	<code>touch archivo</code>	<code>type nul &gt; archivo</code>	<code>New-Item archivo</code>
<b>Añadir texto a un archivo</b>	<code>echo texto &gt;&gt; archivo</code>	<code>echo texto &gt;&gt; archivo</code>	<code>Add-Content archivo -Value "texto"</code>
<b>Mover/renombrar archivo</b>	<code>mv origen destino</code>	<code>move origen destino</code>	<code>Move-Item origen destino</code>
<b>Copiar archivo</b>	<code>cp origen destino</code>	<code>copy origen destino</code>	<code>Copy-Item origen destino</code>
<b>Visualizar contenido archivo</b>	<code>cat archivo</code>	<code>type archivo</code>	<code>Get-Content archivo</code>
<b>Ver árbol de carpetas y archivos</b>	<code>tree</code>	<code>tree /f</code>	<code>tree</code>
<b>Eliminar archivo</b>	<code>rm archivo</code>	<code>del archivo</code>	<code>Remove-Item archivo</code>
<b>Eliminar carpeta (vacía)</b>	<code>rmdir carpeta</code>	<code>rmdir carpeta</code>	<code>Remove-Item carpeta</code>
<b>Eliminar carpeta (con contenido)</b>	<code>rm -r carpeta</code>	<code>rmdir carpeta /s /q</code>	<code>Remove-Item carpeta -Recurse -Force</code>
<b>Buscar archivo por nombre</b>	<code>find . -name archivo</code>	<code>dir archivo /s</code>	<code>Get-ChildItem -Recurse archivo</code>
<b>Buscar contenido dentro de archivo</b>	<code>grep "texto" archivo.txt</code>	<code>find "texto" archivo.txt</code>	<code>Select-String -Pattern "texto" -Path archivo.txt</code>