**Integració de GitHub amb VS Code.**

**1. Crear un repositori a GitHub**

- Ves a https://github.com/ i inicia sessió amb el teu compte.

- Crea un nou repositori que es digui `projecte-python`.

4. Des del nou repositori, copia l'URL que apareix a la pàgina. Aquest URL és important per clonar el repositori més endavant.

**2. Configurar VS Code**

1. Suposem que tenim Git instal·lat. Si no, descarrega'l i instal·la'l des de https://git-scm.com

2. Obre VS Code i assegura't que tens l'extensió de Python instal·lada.

3. Obre la terminal dins de VS Code (`View > Terminal`) i configura Git amb les teves dades.

 **git config --global user.name "El teu nom"**

 **git config --global user.email "correu@exemple.com"**



 3. Clonar el repositori de GitHub en local



VS Code pot demanar-te que t'autentiquis a GitHub. Pots fer-ho de dues maneres:

 - Automàticament: VS Code obrirà el navegador per iniciar sessió a GitHub.









Després haurem de triar a quina carpeta volem clonar el repositori.

En l’últim pas l’hi direm que ens obri el repositori clonat i estarem situats a la carpeta on l’hem clonat.

- També podem utilitzar un Personal Access Token (PAT). Per generar-lo:

Clica a la imatge del teu perfil i després a “Settings”. Al costat esquerra a baix de tot clica a “developer Settings” aquí seguint les opcions disponibles podrem generar un nou token.



**4. Crear un fitxer i fer un commit**

- Crea un fitxer nou anomenat `main.py` amb el següent contingut:

 **print("Hola, GitHub!")**

- A la barra lateral de VS Code, fes clic a la icona de Source Control (l'icona de branques).

Aquí veurem els fitxers que han canviat i amb els que podem fer un commit (el nou fitxer `main.py`)

Escriu un missatge de commit com "Afegit main.py" i fes clic a Commit.



**5. Enviar els canvis a GitHub (Push)**

- Després de fer el commit, VS Code et mostrarà l'opció de fer un Push per enviar els canvis al repositori remot.



- Fes clic al botó blau per pujar els canvis a GitHub.

Veure informació dels canvis:



**6. Crear una nova branca**



- Dona un nom a la nova branca, per exemple, `nova-funcionalitat`.

- Canvia a la nova branca fent clic a la part inferior esquerra on apareix el nom de la branca actual.



**7. Fer canvis i treballar amb la nova branca**

- Modifica el fitxer `main.py`:

 **print("Hola des de la branca nova!")**

- Fes un commit amb el missatge `"Modificació a la branca nova"`.

- Fes push dels canvis de la nova branca a GitHub.

**8. Fusionar la branca (Merge)**

Ho farem per comandes per veure unes altres possibilitats.

- Torna a la branca principal:

 - Obre el Command Palette i selecciona Git: Checkout to → `main`.



- Fes un merge de la nova branca:

 - A la terminal de VS Code, executa:

**git merge nova-funcionalitat**

- Si no hi ha conflictes, els canvis de la branca `nova-funcionalitat` es fusionaran amb `main`.

- Fes un push per actualitzar el repositori remot amb els canvis fusionats.

**9. Baixar canvis del repositori remot (Pull)**

Si un altre col·laborador ha fet canvis i els ha pujat a GitHub, pots baixar aquests canvis al teu repositori local:

- Fes clic a Pull a la barra lateral de Source Control per sincronitzar els canvis del repositori remot amb el teu local.

Aquí podem veure uns conceptes nous:

1. `git pull`:

 - Descarrega els canvis d'un repositori remot i actualitza automàticament la teva branca local amb aquests canvis. Combina (`merge`) els canvis amb els teus si hi ha diferències.

2. `git pull --rebase`:

 - Fa el mateix que `pull`, però en lloc de fer un merge, aplica els canvis locals sobre la nova base remota, mantenint l'historial més net.

3. `git fetch`:

 - Només descarrega els canvis des del repositori remot, sense modificar la teva branca local. Després, pots decidir què fer amb els canvis (per exemple, fer un `merge` manual).

4. `git fetch --prune`:

 - Descarrega els canvis i al mateix temps elimina les referències a branques remotes que ja no existeixen al servidor. Aquest comando és útil per netejar branques obsoletes.