Entorns de Desenvolupament

GitHub-Actions

*Tasca 1*

***Álvaro Gómez Fernández***

Desenvolupament d’aplicacions web

Curs 2024-2025

RA4

Institut Lacetània

**Data d’entrega:** *18/05/2025*

Índex de continguts

[**1.** **Objectiu:** 2](#_Toc198386180)

[**2.** **PART 1:** 3](#_Toc198386181)

[1. Agafeu els fitxers del bloc 3 (testing) corresponents a Pytest o Unittest. 3](#_Toc198386182)

[2. Creeu un nou repositori a GitHub amb aquests fitxers. 4](#_Toc198386183)

[3. Creeu un Workflow de GitHub Actions en el repositori. 6](#_Toc198386184)

[4. Creeu un .yml per a executar les proves d’aquests fitxers Python. 6](#_Toc198386185)

[5. Demostreu el correcte funcionament de les accions mostrant casos d’error (provocats o sense voler) i un últim cas on tot hagi transcorregut correctament. 7](#_Toc198386186)

[**3.** **PART 2:** 8](#_Toc198386187)

[6. Creeu un nou repositori a GitHub. 8](#_Toc198386188)

[7. Creeu un programa en Python que faci el següent: 8](#_Toc198386189)

[a) Obtingui totes les temperatures del dia d’algun poble o ciutat que vulgueu. Per fer-ho, es recomana utilitzar l’API: https://open-meteo.com/ 8](#_Toc198386190)

[b) Calculi la temperatura màxima, la mínima i la mitjana. Cal que ho faci el vostre programa Python, NO utilitzeu l’API per obtenir aquestes dades. 8](#_Toc198386191)

[c) Exporti en un fitxer .json aquestes dades calculades. El nom del fitxer .json ha de contenir la data actual. Per ex: temp\_20250507.json 8](#_Toc198386192)

[8. Creeu un Workflow de GitHub Actions en el repositori. 8](#_Toc198386193)

[9. Creeu un .yml per a executar el programa de Python cada dia a les 23:59. 8](#_Toc198386194)

[10. Cada fitxer .json s’ha de guardar i pujar automàticament al repositori. 8](#_Toc198386195)

[**Bibliografia** 9](#_Toc198386196)

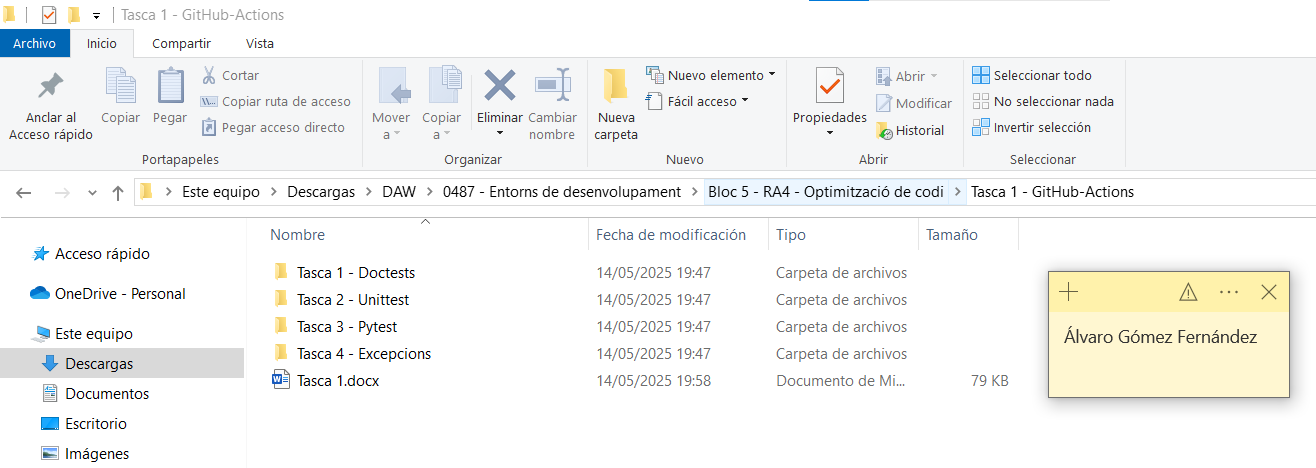
# **Objectiu:**

·Aprendre el correcte funcionament de GitHub Actions.

# **PART 1:**

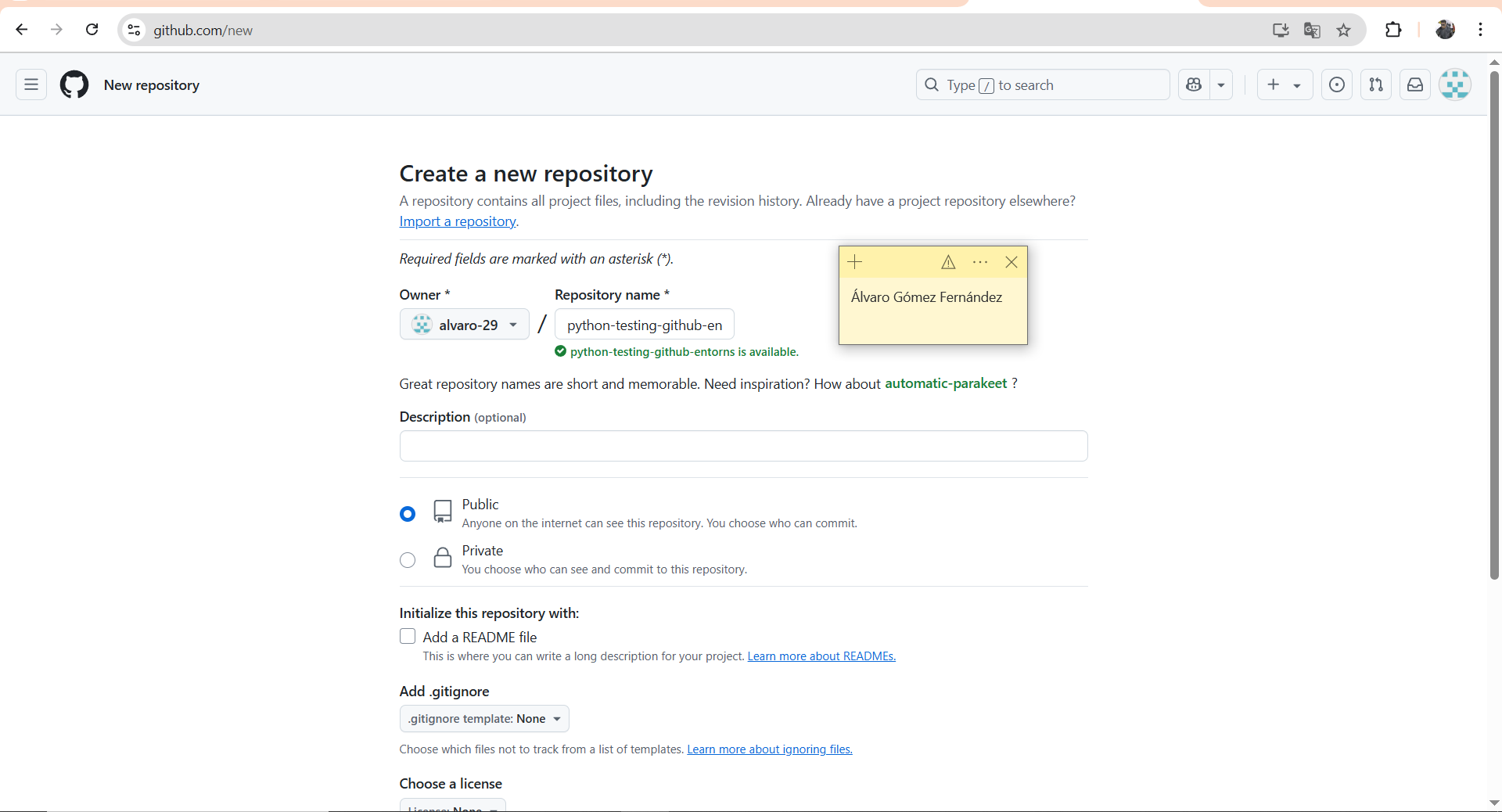
## Agafeu els fitxers del bloc 3 (testing) corresponents a Pytest o Unittest.

·Agafem les carpetes de les activitats del bloc 3:

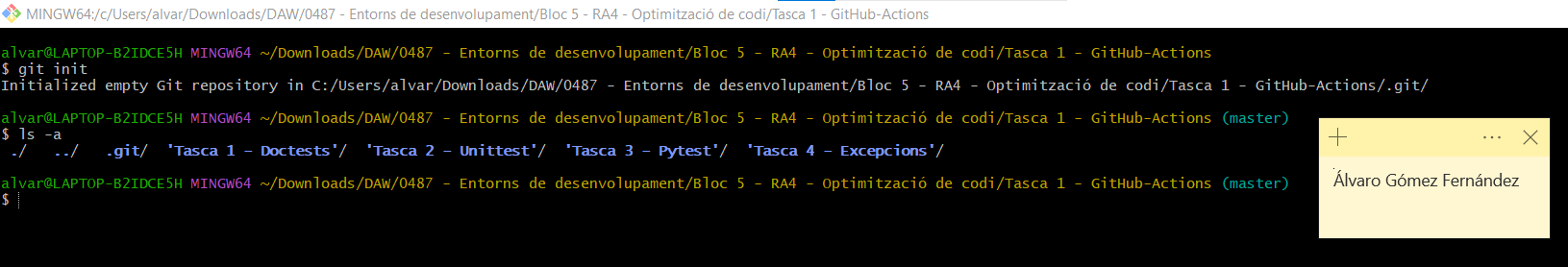


## Creeu un nou repositori a GitHub amb aquests fitxers.

·Anem a ***github.com*** i creem un nou repositori anomenat ***python-testing-github-entorns***:

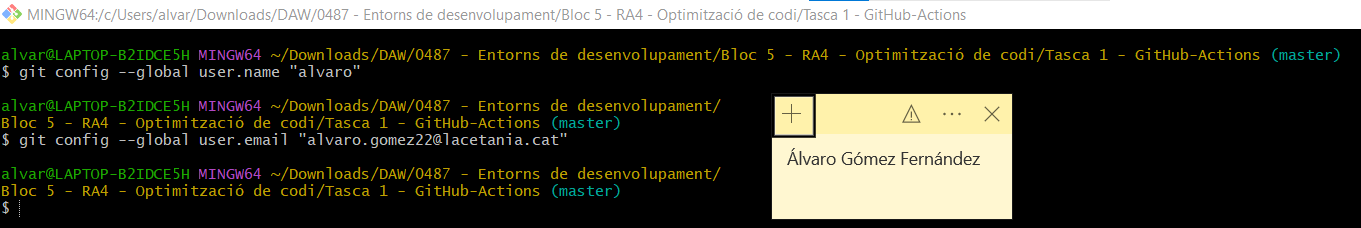


·També utilitzo ***git init*** a la carpeta corresponent per crear un nou repositori localment:

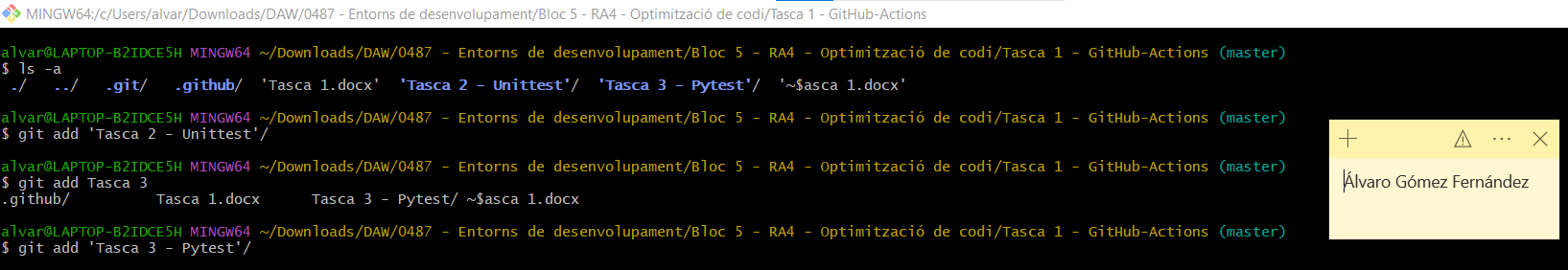


·També hem de configurar la nostra identitat amb les comandes següents:

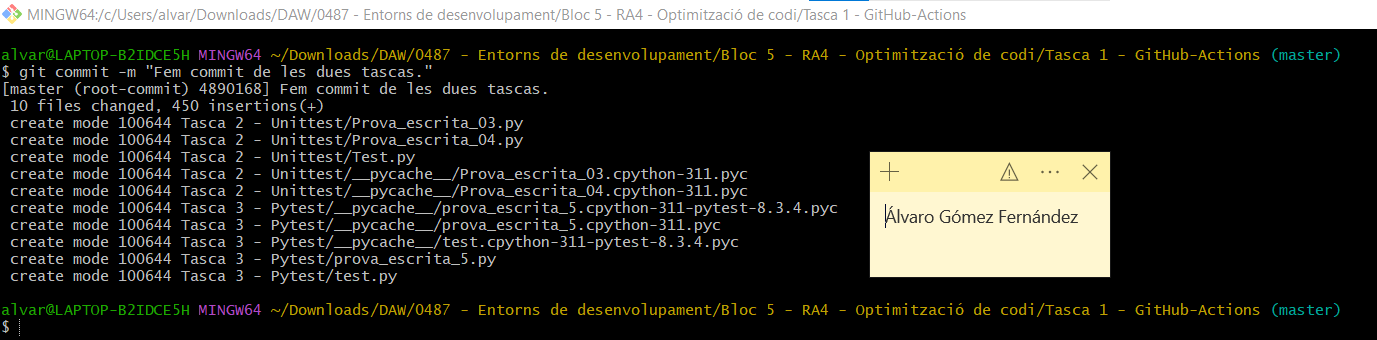
* ***git config –global user.name “alvaro”*** 🡪 nom
* ***git config –global user.email*** [***alvaro.gomez22@lacetania.cat***](mailto:alvaro.gomez22@lacetania.cat)🡪 mail



·Afegim les tasques al repositori amb ***git add …***:

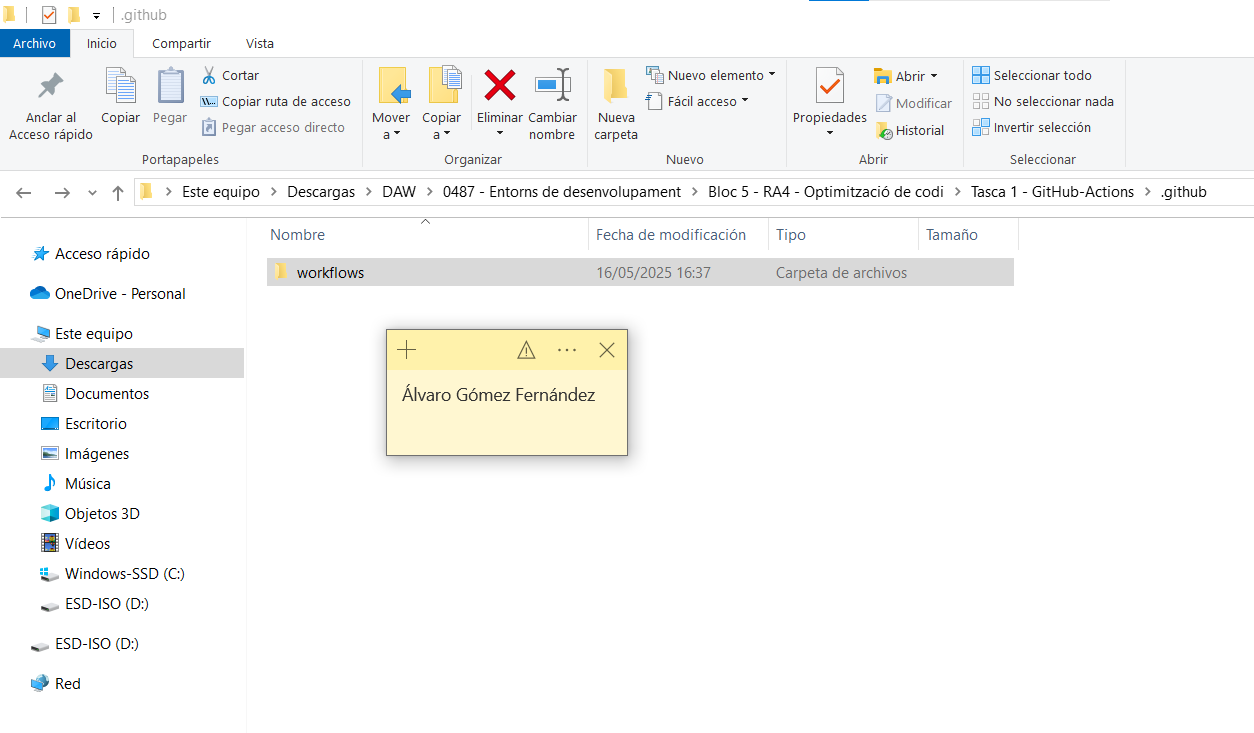


·I també lis fem el commit…



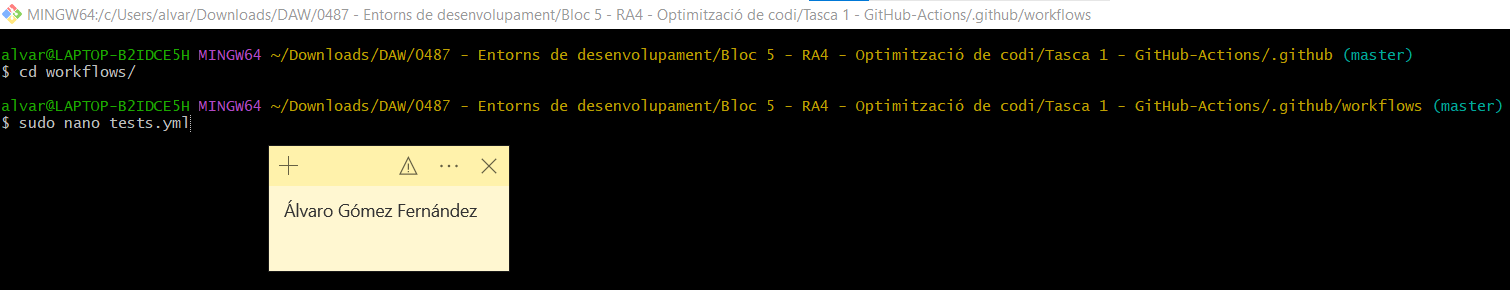
## Creeu un Workflow de GitHub Actions en el repositori.

·Creem la carpeta “***.github/workflows***” a on afegirem el arxiu .***yml***:



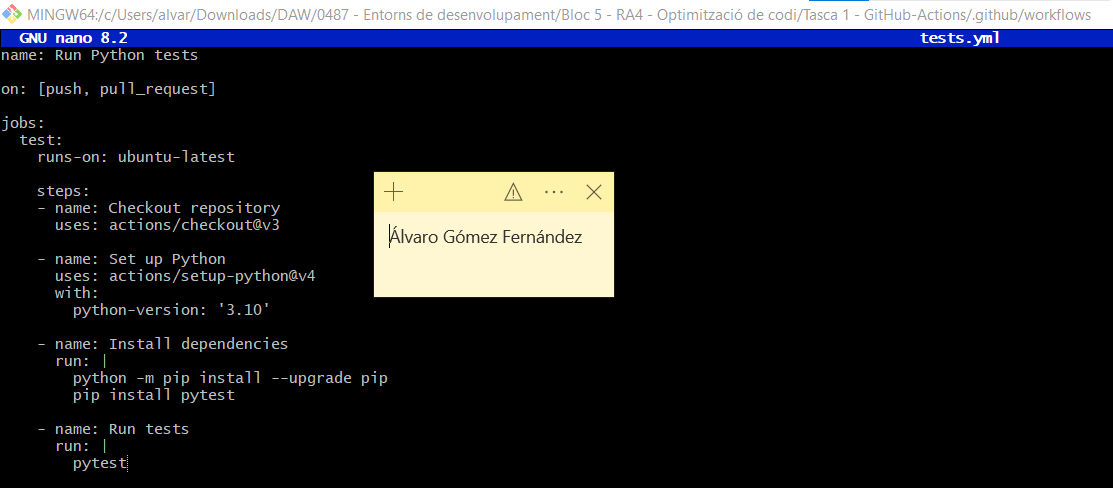
## Creeu un .yml per a executar les proves d’aquests fitxers Python.

·Ho creem des de ***nano***:

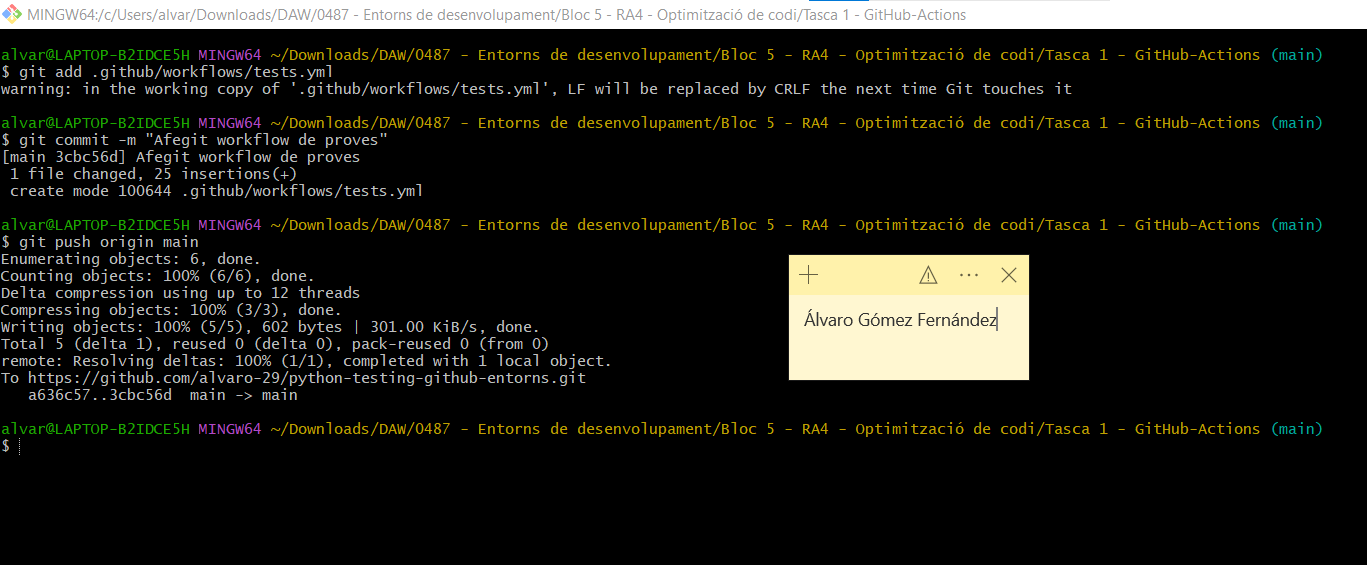


Ara explicaré el que fá el següent arxiu ***.yml***:

* ***name: Run Python tests*** 🡪 Assignem un nom al arxiu.
* ***on: [push, pull\_request*** 🡪 Aquest arxiu s’executarà quan hi ha un ***push*** al repositori o hi ha arxius amb ***pull\_request*** per examinar.
* ***jobs: test: runs-on: ubuntu-latest*** 🡪 S’executarà amb una máquina amb l’última versió d’ubuntu.
* ***steps: - name: Checkout repository uses: actions/checkout@v3:*** Baixa els arxius del repositori per accedir als fitxers python.
* ***- name: Set up Python uses: actions/setup-python@v4 with: python-versions: ‘3.10’:*** S’instal·la Python a la máquina ubuntu.
* ***- name: Install dependencias run: python -m pip install –upgrade pip pip install pytest:*** Instal·lem i agtualitzem tots els paquest necesaris, en aquest cas necesitem solamente pytest.
* ***- name: Run tests run: | pytest:*** Executem pytest als arxius python.



·Ara fem un git ***add***, ***commit*** i ***push*** del arxiu .yml per a activar el workflow:



## Demostreu el correcte funcionament de les accions mostrant casos d’error (provocats o sense voler) i un últim cas on tot hagi transcorregut correctament.

·Ara acabem de sincronitzar els repositoris, local i remot, amb:

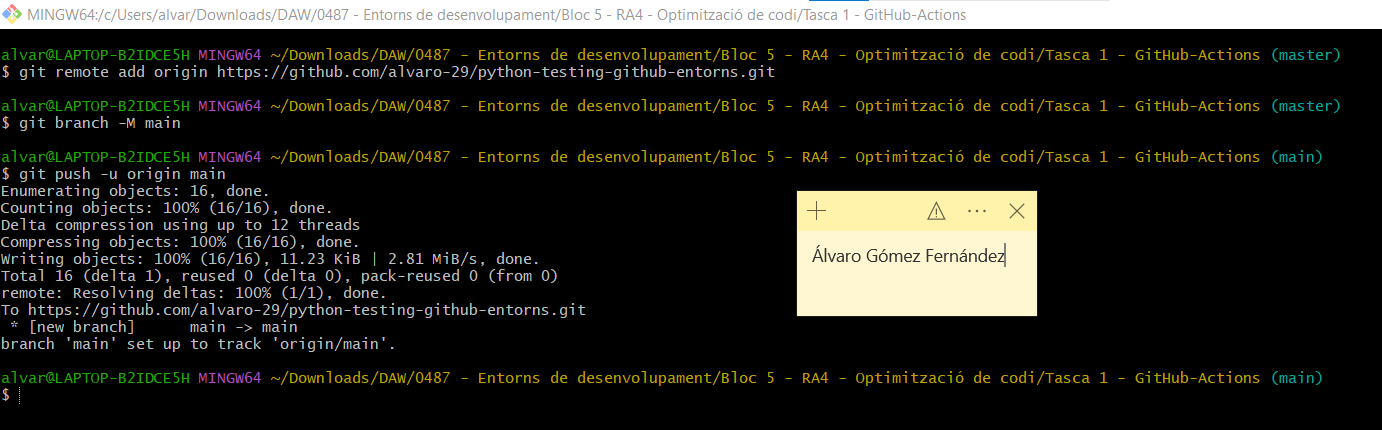
* ***git remote add origin “repositori remot”***.

·Canviem el nom de la branca principal a “***main***” amb:

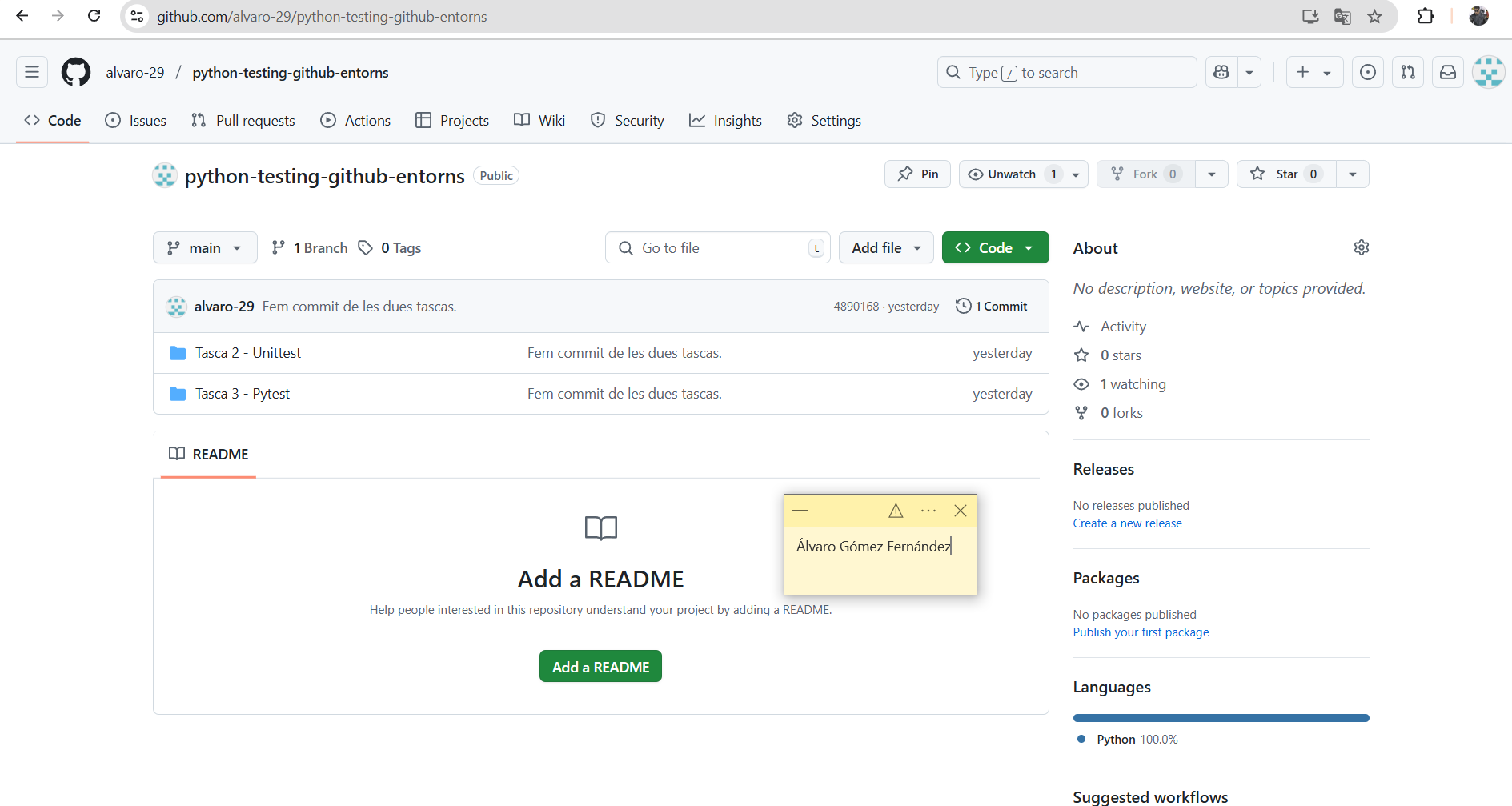
* ***git branch -M main***

·Pujem tots els arxius al repositori remot amb:

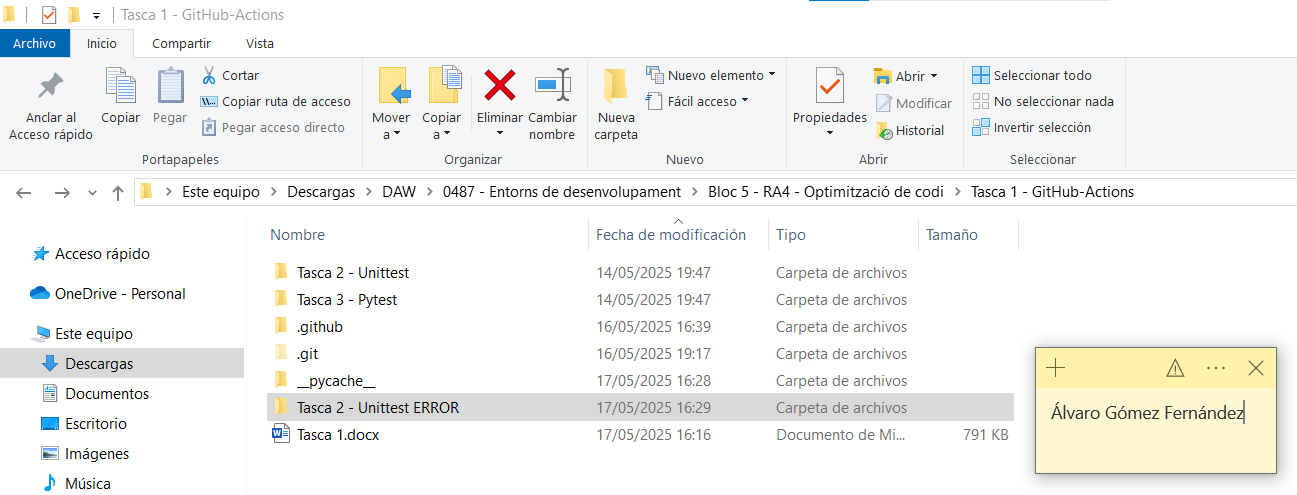
* ***git push -u origin main***

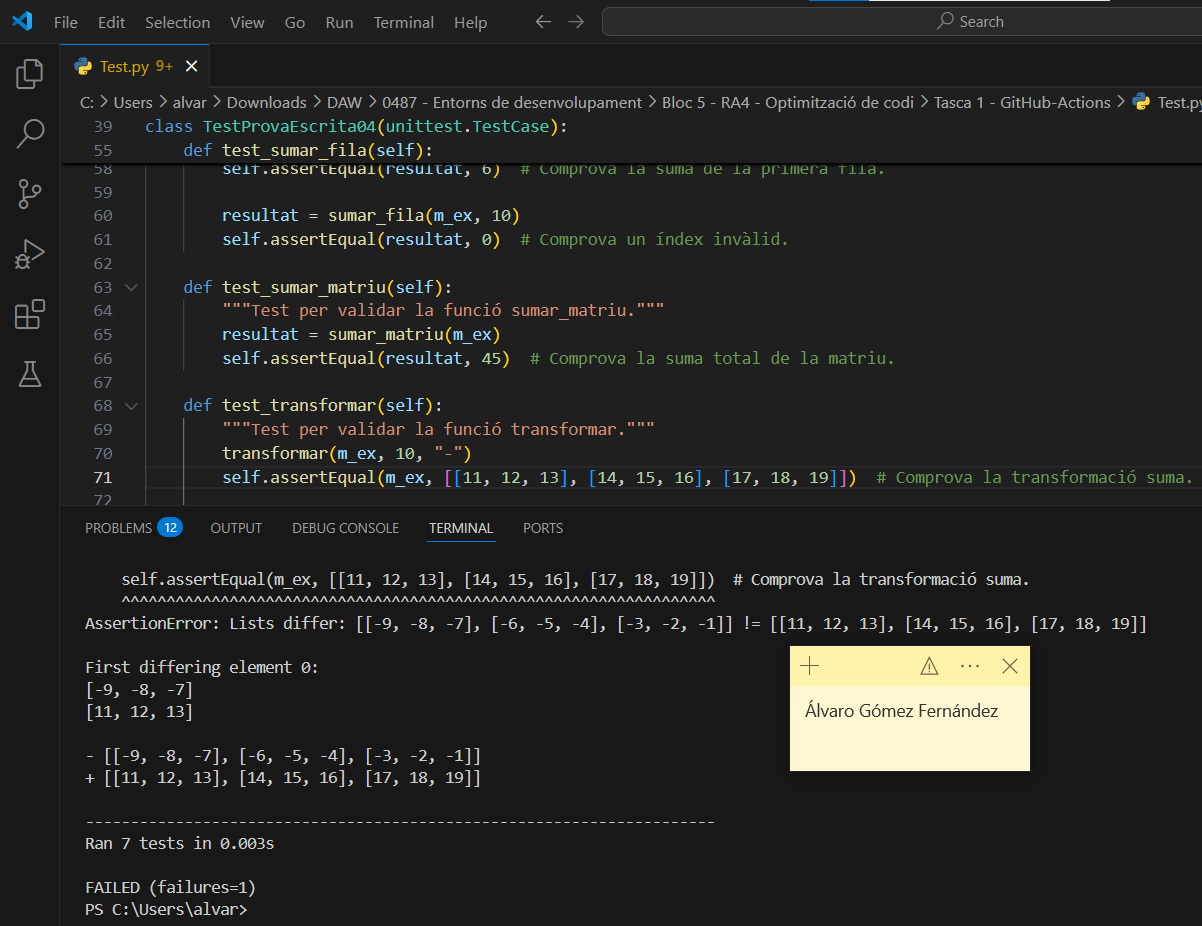


·Ara des del repositori remot veiem que s’han passat els arxius directament de la fase de pull request a estar publicats.



·Això era el cas correcte, si volem provar a fer un error a un unittest per exemple passaria això:

·Error:   




# **PART 2:**

## Creeu un nou repositori a GitHub.

## Creeu un programa en Python que faci el següent:

### Obtingui totes les temperatures del dia d’algun poble o ciutat que vulgueu. Per fer-ho, es recomana utilitzar l’API: <https://open-meteo.com/>

### Calculi la temperatura màxima, la mínima i la mitjana. Cal que ho faci el vostre programa Python, NO utilitzeu l’API per obtenir aquestes dades.

### Exporti en un fitxer .json aquestes dades calculades. El nom del fitxer .json ha de contenir la data actual. Per ex: temp\_20250507.json

## Creeu un Workflow de GitHub Actions en el repositori.

## Creeu un .yml per a executar el programa de Python cada dia a les 23:59.

## Cada fitxer .json s’ha de guardar i pujar automàticament al repositori.

# **Bibliografia**

**No hi ha cap origen al document actual.**