Goal

Setup the **Iris MLOps pipeline** in a **GitHub repository** with `main` and `dev` branches.

Add **evaluation and data validation unit tests** using `pytest`.

Configure **Continuous Integration (CI)** using **GitHub Actions** to:

- Fetch model and dataset from **DVC (Google Cloud Storage)**.
- Run validation and evaluation tests automatically.

Push changes to the `dev` branch and raise a **Pull Request (PR)** to `main`.

Each branch must have its own **CI pipeline** triggered on push or PR.

Perform a **sanity test** using GitHub Actions and **CML** to post test results as a PR comment.

1️⃣ Project Setup

- Navigated to project folder: `cd ~/iris_dvc_pipeline`
- Activated virtual environment: `source venv/bin/activate`
- Created required directories: `.github/workflows` and `tests`

2️⃣ Copied Week 4 Files from GCS

- Pulled `ci.yml`, test scripts, README, and PR template using `gsutil cp` commands.
- Verified files inside respective folders.

3️⃣ Git Initialization & Repository Setup

- Initialized Git repository and committed all Week 4 setup files.
- Linked local repo with GitHub remote:
  `https://github.com/23f3001764/iris_dvc_pipeline_mlops.git`
- Created and pushed two branches: `main` and `dev`.

4️⃣ Configured GitHub Secrets

- Added `GCP_SA_KEY` in GitHub → Settings → Secrets → Actions.
- This allows CI pipeline to access DVC remote in Google Cloud.

5️⃣ Triggered CI Pipeline

- Added a small change (`trigger.txt`) and pushed to `dev`.
- CI ran automatically via **GitHub Actions**:
  - Downloaded data/model from DVC remote
  - Executed `pytest` validation and evaluation scripts
  - Confirmed accuracy ≥ 0.8
  - Posted test report as a PR comment (via CML)

6️⃣ Pull Request & Merge

- Opened PR from `dev` → `main`.
- Verified successful CI run on PR.

Understanding `ci.yml` (CI: pytest + DVC + sanity test)

This file defines your **Continuous Integration (CI) workflow** for the Iris MLOps project.
It automates testing every time you **push code or open a Pull Request**.

gh pr create --title "Week 4 CI & Tests" --body "Added pytest + DVC CI" --base main --head dev

This command **creates a Pull Request (PR)** on GitHub directly from your terminal using the **GitHub CLI tool** (`gh`).

Instead of going to the GitHub website and clicking “Compare & pull request,” we’’re doing it **programmatically** through the command line.

test_validation_and_evaluation.py

What it checks

Why it matters

test_data_file_exists

Data file is available

Ensures DVC or dataset sync works

test_data_columns_and_nulls

Dataset has all columns & no missing data

Ensures data schema is consistent

test_model_file_exists

Model file is present

Confirms training or DVC pull worked

test_model_prediction_shape

Model produces correct number of predictions

Verifies model usability

test_metrics_file_and_accuracy

Accuracy ≥ 0.8

Detects model quality issues early