Optional Step: CI/CD with GitHub Actions

University: Tek-Up
Group: ING-4-SDIA

Tutor: Haythem Ghazouani

1 Introduction

In professional software development, we don't just "push and hope". We use **CI/CD** (Continuous Integration / Continuous Deployment) to automate testing and building.

The Pipeline Advantage

Every time you push code to GitHub, a "bot" starts a virtual machine, pulls your code, runs your tests, and builds your Docker images. If something fails, you are notified immediately.

2 GitHub Actions Basics

GitHub provides a service called **Actions**. It uses `.yaml` files located in the `.github/workflows/` directory.

2.1 The Trigger

Our pipeline starts automatically when:

- You **push** to the `master` branch.
- Someone creates a **Pull Request**.

3 Workflow Breakdown

Look at the file `.github/workflows/main.yml`:

1. **lint-python:** Uses `flake8` to check if your code follows PEP8 standards (clean code).
2. **build-docker-backend:** Verifies that the Docker image for the API can be built without errors.
3. **build-docker-frontend:** Verifies the React build process.

DevOps Concept: Breaking the Build

If a student pushes code with a syntax error, the "lint" job will fail, and a **Red X** will appear next to the commit in GitHub. The Docker images won't even try to build, saving time and resources.

4 How to monitor?

1. Push your changes to GitHub.
2. Go to your repository on the web.
3. Click on the **Actions** tab at the top.
4. You will see a list of "Workflow runs". Click on the latest one to see the real-time progress of each job.

5 Exercise (Bonus)

- **Slack Integration:** Research how to add a step that sends a message to a Discord or Slack channel whenever a build fails.
- **Security:** Add a step called `Safety` or `Bandit` to check your Python dependencies for known vulnerabilities.