Day 4

Report on FastAPI Deployment and Selenium Testing Sessions:

🚀 Report on FastAPI Deployment and Selenium Testing Sessions

Session 1: FastAPI Deployment on Render

This session introduced the process of deploying a portfolio website built using FastAPI on the **Render** platform. The key tasks covered were:

1. Setting Up the Virtual Environment

- A virtual environment was created using:
- python -m venv ~/profile
- source ~/profile/scripts/activate
- FastAPI and its dependencies were installed with:
- python -m pip install "fastapi[standard]"

2. Developing the FastAPI Application

- Navigated to the project folder and created the main application file:
- touch app.py
- code app.py
- The development server was launched using:
- uvicorn app:app --reload
- A templates folder was created to host the index.html frontend file.

3. Deployment on Render

- Configured deployment settings for Render:
    - **Build Command**:
    - pip install fastapi "uvicorn[standard]"
    - **Start Command**:
    - uvicorn app:app --host 0.0.0.0 --port $PORT

4. Version Control Using Git

- Utilized git diff to view code changes before committing them.

5. Introduction to Selenium

- Installed Selenium via:
- python -m pip install selenium

- Used Selenium to control the Chrome browser programmatically, locate web elements, and perform actions using methods like find_element and send_keys.

6. Managing Virtual Environments

- Practiced creating, deleting, and managing multiple virtual environments.
- Used the deactivate command to exit the active environment.

Session 2: Testing and Continuous Integration with GitHub Actions

This session focused on incorporating automated testing and CI/CD pipelines using **GitHub Actions**.

1. Automated Testing with FastAPI and Selenium

- Installed the **FinTest Pro** extension for enhanced testing.
- Created a test_main.py file in the project directory.
- Wrote and executed tests for the homepage using FastAPI's TestClient.

2. CI/CD with GitHub Actions

- Created a GitHub Actions workflow file named main.yml.
- Configured it to trigger on push events to the main branch (or other specified branches).
- Workflow steps included:
    - Installing necessary packages.
    - Running automated tests via:
    - pytest test_main.py

3. Dependency Management

- Updated the requirements.txt file with required packages:
    - fastapi
    - pytest

✅ **Summary**

Over the course of these two sessions, participants acquired hands-on experience in:

- Deploying FastAPI applications on the **Render** platform
- Performing browser automation with **Selenium**
- Writing API tests and managing test suites
- Setting up **GitHub Actions** for continuous integration

These sessions provided a solid foundation in modern backend development, test automation, and deployment workflows, essential for today's software development practices.