II.3525 – Tests & Tests Automation

2025-2026

Final Project: Automated Testing Campaign

Group Work: 3-4 students per team

Duration: 6 weeks with 3 supervised sessions of 3h

Start Date: December 1, 2025

Teams composition & Project choice : Fill this file to define your team member and choose your project before **December 8, 2025**

Defense Date: **January 12, 2026**

Overview

You will enhance an existing application by implementing a comprehensive automated testing strategy, integrating it into a CI/CD pipeline.

Requirements

Option Selection

You must choose **ONE** of the following options:

Option 1: Use Your Own Application

Select an application you have previously developed that includes:

- A frontend (web or mobile) OR an API
- At least 5-10 functional features
- Source code accessible via Git
- Examples: Shopping cart app, Todo List manager, any CRUD application

Option 2: Use an Existing Open Source Application

Select an open source project with similar characteristics to those listed above.

Deliverables

1. Manual Testing Campaign

Create a comprehensive test plan document including:

- Application description and testing scope
- Testing objectives and acceptance criteria
- Complete list of features to be tested
- Test environment
- Detailed manual test cases with:
  - Unique test case ID
  - Clear description
  - Preconditions
  - Step-by-step execution instructions
  - Expected results
  - Priority classification (high/medium/low)

2. Test Automation

Develop automated test scripts covering:

- **Minimum 70% coverage** of your manual test cases as unit tests
- **UI Testing:** Frontend testing using Selenium or equivalent
- **API Testing:** Backend testing using Postman, pytest, or similar tools

3. CI/CD Integration

Deploy your tests in a CI/CD pipeline using **Jenkins** or **GitLab CI/CD**:

- Pipeline configuration file (Jenkinsfile or .gitlab-ci.yml)
- Automated execution triggered by:
  - Code commits (push)
  - Scheduled runs (running at night)
  - Manual triggers
- Pipeline stages must include:
  - Build/Setup
  - Unit tests
  - Integration tests
  - UI/API tests
  - Test reporting and results dashboard

4. Quality Metrics & Analysis

Measure, track, and document the following metrics:

- **Code coverage:** Percentage of code tested
- **Test execution time:** Duration of complete test suite
- **Pass/fail rate:** Percentage of successful tests
- **Defects found:** Number and severity of bugs discovered

5. TDD/BDD Implementation

Apply modern testing methodologies:

- **Test-Driven Development (TDD):**
  - Implement at least 3 features using the TDD approach
  - Document the Red → Green → Refactor cycle for each feature
  - Show code evolution in the final document
- **Behavior-Driven Development (BDD):**
  - Write test scenarios in Gherkin syntax (Given-When-Then)
  - Implement scenarios using Cucumber, Behave, or Pytest-BDD

6. Documentation & Reports

Prepare comprehensive project documentation:

- Testing strategy and approach
- Test automation framework architecture
- CI/CD pipeline deployment plan
- Test results and quality metrics reports
- Lessons learned and recommendations

7. Final Presentation

Prepare a presentation including:

- **Pitch:** Project context and challenges
- **Live Demonstration:**
  - Application quick walkthrough
  - Live pipeline execution trigger
  - Real-time test results and reports
  - Show metrics

Submission Guidelines

All deliverables must be submitted via Moodle with:

- Complete documentation
- Working CI/CD pipeline configuration
- The slides used for the presentation

Add rayan.karkour@ext.isep.fr and maurras.togbe@isep.fr to your GitHub repository