Software Engineering Seminar
Semester 2025-III
Final Project Definition and Delivery

Eng. Carlos Andrés Sierra, M.Sc.
Full-time Adjunct Professor
Computer Engineering Program
School of Engineering
Universidad Distrital Francisco José de Caldas

Congratulations on reaching the final stage of your *Software Engineering Seminar* course project! This document outlines the requirements for your final project delivery. Your submission should demonstrate a complete, well-documented, and functional web application, integrating the concepts and practices learned throughout the course.

Final Project Scope and Objectives:

- **User Stories and Scrum Implementation:**
  - Provide a comprehensive list of *user stories* for your application.
  - Create a *user story mapping* for planning and prioritization.
  - Include a report describing how your team implemented `Scrum`, including *sprint planning*, *daily stand-ups*, *sprint reviews*, *retrospectives*, and *key learnings.*
  - Discuss the challenges faced during the `Scrum` implementation and how they were addressed.
- **System Analysis and Architecture:**
  - Perform a `system analysis` of your project, identifying *main components*, *actors*, and *interactions.*
  - Design the `software architecture` using *object-oriented* programming principles. Include *diagrams* and rationale for design choices.

Carlos Andrés Sierra, Computer Engineer, M.Sc. in Computer Engineering, Lecturer at Universidad Nacional de Colombia.

Any comment or concern regarding this project can be sent to Carlos A. Sierra at: *casierrav@unal.edu.co*.

- **Backend Implementation:**
  - Develop a `Java` backend for *authentication*, connected to a `MySQL` database.
  - Develop a `Python` backend for *business logic* (students choose the domain), implementing at least *two CRUD operations*, connected to either `PostgreSQL` or `MongoDB`.
  - Ensure both backends expose `REST APIs` for communication.
- **Frontend Implementation:**
  - Develop a simple `web GUI frontend` (any technology, no framework required) that interacts with both backends via `REST APIs`.
- **Testing:**
  - Implement *unit tests* for all backend functions: use `JUnit for Java` and `pytest for Python`.
  - Use `Apache Cucumber` to validate *acceptance criteria* for user stories at the logic level.
  - Use JMeter to perform stress tests on the REST APIs.
- **CI/CD and Deployment:**
  - Provide a basic *CI/CD pipeline* using `GitHub Actions`.
  - Use `Docker` and `Docker Compose` to *containerize* both backends and the frontend.
  - Document *the CI/CD process* and provide evidence of successful builds and deployments.
- **Documentation:**
  - Deliver clear documentation covering user stories, Scrum implementation, system analysis, architecture, API endpoints, database schema, testing strategy, CI/CD setup, and evaluation.
- **Reflection and Evaluation:**
  - Critically evaluate your solution, discussing strengths, limitations, and possible future improvements.

Methodology and Deliverables:

1. **Documentation:**
   - Compile all sections (user stories, Scrum report, system analysis, architecture, API documentation, database schema, testing strategy, CI/CD setup, evaluation) into a single, well-organized PDF.
   - Include diagrams, code snippets, and references as needed.

2. **Project Repository:**
    - Organize all source files, scripts, diagrams, and documentation in a folder named `Final-Project` in your course repository.
    - Provide a `README.md` that explains the structure, setup, and usage of your project.
3. **Project Implementation:**
    - Deliver a functional implementation of your web application using the selected technology stack.
    - Ensure the implementation covers the main features, REST API endpoints, database operations, and frontend interactions described in your documentation.
    - Include clear instructions for setup, execution, and testing in your `README.md`.
4. **Testing and Validation:**
    - Provide unit tests for all backend functions (JUnit for Java, pytest for Python).
    - Include Cucumber feature files and step definitions for acceptance criteria validation.
    - Provide JMeter test plans and results for API stress testing.
5. **CI/CD Demonstration:**
    - Provide working GitHub Actions workflow files (`.github/workflows/ci.yml`) that run tests and build Docker images for your backends and frontend.
    - Include a `docker-compose.yml` file to orchestrate the deployment of all services.
    - Document the CI/CD process and show evidence of successful runs (e.g., screenshots, logs).
6. **Demonstration (Mandatory):**
    - Prepare and deliver a brief presentation or video (5-10 minutes) demonstrating your system's main features, API endpoints, frontend, and CI/CD pipeline.
    - The demo is a required part of the final evaluation.

Project Requirements Checklist:

- List of user stories and user story mapping.
- Scrum implementation report and learnings.
- System analysis and OOP-based architecture design.
- Java backend for authentication (MySQL).
- Python backend for business logic (PostgreSQL or MongoDB).
- REST API communication between frontend and backends.

- Simple web GUI frontend.
- Unit tests (JUnit, pytest).
- Cucumber acceptance tests.
- JMeter stress tests.
- CI/CD pipeline with Docker, Docker Compose, and GitHub Actions.
- Clear documentation and setup instructions.

Examples of Application Ideas:

- Task manager or to-do list application with authentication.
- Simple blog or content management system with user login.
- Basic inventory or product catalog with user roles.
- Contact management or address book with secure access.
- Event registration or booking system with authentication.

Deadline: Friday, December 12th, 2025, at 2:00. Late submissions may affect your grading according to course policies.

Notes:

- All documents must be in **English**.
- Cite any references (articles, tutorials, frameworks) that influenced your design choices.
- Focus on clarity, completeness, and professional presentation.
- This is your opportunity to showcase your ability to design, implement, test, and deploy a robust web application using modern software engineering practices.

Good luck! Your final project is the culmination of your learning and effort throughout the course. Make it count!