Object-Oriented Programming
Semester 2025-II
Workshop No. 4 — Layered Architecture for a Domotic Circuit Simulator (Python)

Eng. Carlos Andrés Sierra, M.Sc.
Lecturer
Computer Engineering
School of Engineering
Universidad Nacional de Colombia

This workshop builds upon **Workshop #1** (conceptual design), **Workshop #2** (technical design), and **Workshop #3** (SOLID principles) to guide you in implementing a **layered architecture** for your domotic circuit simulator. You will structure your Python project into clear layers (presentation, business logic, and data handling), integrate a basic GUI (using PyQt5 or similar), and add file-based persistence for saving/loading circuit designs.

Workshop Scope and Objectives:

- **Layered Architecture:** Organize your code into presentation (UI), business logic (simulation), and data (persistence) layers.
- **Python GUI Implementation:** Develop a simple, functional interface using PyQt5 (or another Python GUI library) to allow users to add components, connect them, and run simulations.
- **File Persistence:** Implement saving and loading of circuit designs using a file format such as JSON or pickle.
- **Integration:** Ensure all layers communicate cleanly, following OOP and SOLID principles.

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

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

Methodology and Deliverables:

1. **Layered Design Review:**
    - Update your class diagrams and design documents to reflect the layered architecture.
    - Clearly indicate which classes belong to each layer and how they interact.
2. **Python GUI Prototype:**
    - Implement a minimal GUI using PyQt5 (recommended) or another Python GUI toolkit.
    - The interface should allow users to add, connect, and simulate basic domotic components (e.g., switches, lights, sensors).
    - Focus on functionality and clarity, not advanced features or aesthetics.
3. **File-based Persistence:**
    - Implement methods to save and load circuit designs to/from a file (e.g., using JSON or pickle).
    - Demonstrate persistence by saving a circuit, closing the program, and reloading it.
4. **Documentation and Submission:**
    - Provide updated UML diagrams (class and/or sequence) showing how each layer communicates.
    - Include code samples or references to new/modified classes for the GUI and data access logic.
    - Write a brief user manual explaining how to run and use your simulator.
5. **Final Deliverables:**
    - A PDF combining diagrams, implementation notes, and usage instructions.
    - A `Workshop-4` folder in your repository containing the code, documentation, and a `README.md` with build/run steps.

Deadline: Friday, November 28th, 2025, at 8:00 PM. Late submissions may affect your grading according to course policies.

Notes:

- Use **English** for all written deliverables.
- Cite any references or tutorials that aided your GUI and file IO implementations.
- This workshop demonstrates your ability to integrate OOP, SOLID, and layered design in a real Python project.

- Focus on maintainability, clarity, and correct use of OOP concepts.

Congratulations on reaching the final step of your OOP journey! Focus on integrating a user-friendly GUI, effective file persistence, and a robust layered design to finalize your domotic circuit simulator project successfully.