# Object-Oriented Programming
# Semester 2025-I
# Final Course Project — A Transactional Application

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

This **Final Course Project** brings together concepts and practices from:

- **Workshop #1:** Object-Oriented Analysis & Conceptual Design
- **Workshop #2:** Technical Design and Inheritance-Polymorphism
- **Workshop #3:** SOLID Principles and Architecture Refinements
- **Workshop #4:** Layers Architecture, Java FX, and File-Based Persistence

You will develop a *simple transactional application of your choice*, finalizing all OOP principles, design artifacts, and best practices covered throughout the semester.

### Project Scope and Objectives

- **Complete OOP Solution:** Demonstrate a robust class hierarchy incorporating inheritance, polymorphism, encapsulation, and SOLID principles.
- **Layered Architecture:** Maintain clear separation into presentation, business logic, and data/persistence layers.
- **User Interface:** Include a functional Java FX GUI that clearly illustrates real-world usage scenarios for your transactional system.
- **Data Handling:** Provide file-based persistence (or an alternative data store if you wish to exceed requirements) to store and retrieve core application data.

---

Carlos Andrés Sierra, Computer Engineer, M.Sc. in Computer Engineering, Titular Professor at Universidad Distrital Francisco José de Caldas.

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

## Methodology and Deliverables

1. **Refined System Design:**
   - Present conceptual design updates from Workshop #1, including refined requirements, CRC cards, user stories, and mockups.
   - Present updated UML diagrams (class, sequence, or other relevant diagrams) reflecting your final architecture and technical design.
   - Confirm you have applied OOP best practices: clear responsibilities, minimal coupling, and maximum cohesion.
   - Explicitly show how your design incorporates improvements and refinements from Workshop #3, especially regarding SOLID principles and architectural decisions.
   - Demonstrate how your layered architecture (from Workshop #4) supports separation of concerns and enhances maintainability.
2. **Implementation:**
   - Implement your code in Java (preferably) or another OOP language, integrating the layers:
     (a) *Presentation Layer:* Java FX forms, dialogues, or panels.
     (b) *Business Layer:* Classes, services, or controllers encapsulating core logic.
     (c) *Data Layer:* Serializing/storing data in files, or an alternative storage solution if desired.
   - Demonstrate the usage of SOLID principles in your classes and interfaces.
   - Clearly document any architectural changes or refinements made as a result of Workshop #3.
   - Show how your implementation supports the layered architecture defined in Workshop #4.
3. **Testing & Validation:**
   - Outline test cases or scenarios ensuring your system handles normal and edge cases (e.g., invalid inputs, repeated transactions).
   - Present logs, screenshots, or short videos (if feasible) highlighting typical workflows (create transaction, view items, store data, etc.).
4. **Documentation:**
   - Compile all relevant README or how-to-run instructions into a `README.md`.
   - Provide a brief user manual or instruction set explaining how an end-user can operate the interface.
   - Reference any external libraries or tutorials you relied upon (indicate version numbers, if applicable).

5. **Submission Requirements:**
    - Place your final application in a dedicated folder named `FinalProject` in your course repository.
    - Supply a PDF report containing the design, diagrams, code overview, and usage instructions.
    - Ensure the project compiles and runs on a standard environment (e.g., Java 11+).

## Deadline & Format

**Final Submission Date:** Refer to your course syllabus or official announcements for the exact deadline.

**Notes:**

- Keep your deliverables **in English**.
- Cite any external references, libraries, or tutorials used in your final implementation.
- The project should emphasize practical OOP knowledge; strive for clarity, maintainability, and correctness.
- Your final design and implementation should clearly demonstrate the improvements and refinements made in Workshop #3, especially regarding SOLID principles and architectural quality.
- Your layered architecture should be evident in both your design documentation and code structure, as developed in Workshop #4.

*Congratulations on reaching the final stage of this course! This project consolidates everything you've learned about object-oriented design and implementation. Best of luck, and we look forward to reviewing your complete, well-architected transactional solutions.*