Project Work: Java

Course Code: CMP 175 | **Course Title**: Object Oriented Programming (Java)
Semester: II | **Program**: BCSIT
Assignment Type: Group Project
Project Title: Java Swing Application with CRUD Functionality
Submission Deadline: TBD

Objective

The goal of this assignment is to provide students with hands-on experience in building a complete Java application using **Swing for GUI**, implementing **object-oriented programming concepts**, **event-driven programming**, **exception handling**, **file operations**, and optionally **generics**. Each group will collaboratively design, develop, and test a mini-project that demonstrates their understanding of real-world software development using Java.

Project Theme (Choose One)

Each group can choose **one** of the following types of applications. Original ideas are also welcomed, but must be approved beforehand.

1. **Student Management System**
    - Manage records: Add, View, Update, Delete student data
    - Fields: Name, Roll No, Program, Semester, Contact, Email, Address
    - Store/retrieve data using file I/O
    - Provide search and sort functionality
2. **Library Management System**
    - CRUD operations for books and borrowers
    - Fields for books: Title, Author, ISBN, Quantity
    - Fields for borrowers: Name, ID, Contact Info
    - Issue/Return management with date tracking

3. **Simple Inventory Management System**
    - Manage product inventory with name, category, quantity, price
    - Support adding new products, updating stock, removing items
    - Report generation (low stock, out-of-stock items)
4. **Swing-based Game (Optional but I prefer Advanced Teams)**
    - Eg: Tic-Tac-Toe, Shooter Game, PongGame, Memory Card Flip, Number Puzzle
    - Should include GUI, basic rules/logic, and state persistence (optional)
    - Requires instructor's approval before proceeding

Project Requirements

The final application **must include** the following features:

1. User Interface (UI)

- Built using **Swing components** like `JFrame`, `JPanel`, `JLabel`, `JButton`, `JTextField`, `JTextArea`, `JTable`, etc.
- Must include navigation/menu controls for each functionality
- Basic styling/layout using layout managers (`BorderLayout`, `GridLayout`, `BoxLayout`, etc.)

2. Functionalities

- **CRUD** operations (Create, Read, Update, Delete)
- **Search/Filter** capabilities (e.g., search by name or ID)
- **File Handling**: Save/load data to/from text or binary files (using streams)

- **Validation** and **Exception Handling**: Proper input validation and custom exception handling
- Use of **classes**, **objects**, **inheritance**, and **method overloading/overriding**
- Use of **constructors**, `this` keyword, and **static members** where appropriate
- Optional: Use **generics** in utility or data manager classes

Group Formation

- Each group must consist of **4 students**
- All members **must understand all parts** of the project
- Each student must be prepared to:
    - Present their project in class
    - Explain **any part** of the source code during viva

Deliverables

Each group must submit the following:

1. **Project Source Code** (well-commented and properly structured)

2. **Project Report** (10–15 pages) in the following format:
    - Cover Page
    - Table of Contents
    - Project Overview and Objective

- Technology Used
- Class Diagrams or UI Flow (optional but appreciated)
- Key Screenshots
- Description of Each Module
- Code Snippets (important logic)
- Challenges Faced and Solutions
- Contribution of Each Member

3. **Viva Presentation** (Each group will present and be individually assessed)

Evaluation Criteria

Criteria	Weightage
Functionality & Features	20%
GUI Design & Usability	10%
Code Quality (OOP Principles, Comments)	10%
File Handling & Exception Handling	10%
Report Completeness & Quality	10%
Individual Viva Performance	20%
Presentation & Team Collaboration	10%
Creativity or Extra Features	10%

General Guidelines

- **Plagiarism will not be tolerated**. Groups found copying code from others or from the internet without proper understanding will receive a penalty.
- Each group must **demonstrate the project** live during the viva.
- Work division must be fair and transparent among all members.
- Late submissions will receive marks deduction unless pre-approved.

Instructor's Note

This project is not just about code, but **understanding Java as a tool** for solving problems, working in teams, and applying what you've learned. Focus on clarity, usability, and maintainable code. Do not hesitate to ask for help during project development. Regular check-ins or demos may be scheduled during lab sessions.

"Programming isn't just typing code. It's about thinking about how your software interacts with people and systems."

APPENDIX:

Rubric

Criteria	Excellent (Full Marks) *Score: 5/5 or 4/4*	Good *Score: 4/5 or 3/4*	Satisfactory *Score: 3/5 or 2/4*	Needs Improvement *Score: 2 or below*
1. Project Design & Functionality (5 marks)	Application is complete, all features work perfectly, intuitive UI	Most features work as expected, minor UI flaws	Core functionality implemented, some bugs present	Major features missing or buggy, unclear UI
2. OOP Concepts (5 marks)	Excellent use of classes, inheritance, encapsulation, polymorphism	Good use of OOP, minor design issues	Basic class usage, minimal use of OOP principles	Poor OOP structure, procedural style code
3. Swing Components & Event Handling (5 marks)	GUI is well-designed, all events handled correctly	Mostly functional GUI, events handled adequately	Basic GUI, limited interaction, some handlers missing	Poor or incomplete GUI, no event handling
4. Code Quality & Organization (5 marks)	Clean, readable, well-commented, modular code	Some inconsistency, minor organization issues	Code is readable but lacks modularity/comments	Messy code, difficult to follow, poor structure
5. Exception Handling & I/O (5 marks)	Robust exception handling and file I/O implemented	Some exceptions handled, basic file operations	Minimal handling, limited file usage	No exception handling, no file operations

6. Individual Understanding & Viva (10 marks)	Student explains code clearly, demonstrates full understanding	Good explanation, answers most questions	Partial understanding, relies on teammates	Unclear answers, unaware of codebase