Course Title: Software Construction & Development
Course Code: SE 3001
Credit Hours: 3+1
Instructor: Abeeha Sattar

Course Objectives:

By the end of the course, you will be able to:

- Convert OO design into code
- Use version controlling for their source code
- Use issue tracking system for dividing tasks among team members and tracking them
- Use logging in their software for error reporting
- Write unit test cases in JUnit
- Understand various code smells and refactorings
- Understand GoF design patterns and their corresponding code

Reference Books:

1. Clean Code: A Handbook of Agile Software Craftsmanship 1st Edition by Robert C. Martin
2. Head First Design Patterns by Eric Freeman, Elisabeth Robson
3. Refactoring - Improving the Design of Existing Code by Martin Fowler, with Kent Beck
4. Software Architecture Patterns by Mark Richards, O'Reilly Media, Inc., 2015

Other Reference Links:

1. https://docs.oracle.com/en/java/
2. https://www.martinfowler.com/eaaDev/uiArchs.html
3. https://refactoring.com/
4. https://refactoring.guru/
5. https://courses.cs.washington.edu/courses/cse331/15au/tools/versioncontrol.html

Tentative Weekly Plan:

1.	• Intro to the course • Requirements elicitation for the project. • Java introduction
2.	• OOP Concepts in Java • Intro to MVC architecture, Layered architecture
3.	• Java basics: static & dynamic binding, super keyword in Java, Java interfaces • Event-driven programming: events and event handlers, callbacks etc.
4.	• JavaFX (Text, TextField, Button, TableView etc.) • Java exception handling: try, catch, throw, throws, finally keywords
5.	• Java exception handling: try-with-resources, chained exceptions • Observer design pattern, Singleton design • pattern
6.	Midterm I
7.	• MVC in code. maintaining logs using log4j2. • Java Collections

8.	• MVC, MVVM, MVP flavors • Version control using Git (add, commit etc.) • Git: branching and merging • Testing basics, unit testing
9.	• Writing test cases in Junit • Case Studies
10.	• Writing test cases in Junit • Case Studies
11.	Midterm II
12.	• Code Smells & Refactoring
13.	• Martin Fowler's Refactoring Concepts, Refactoring Strategies
14.	• Design Patterns (tentative: Composite, Interpreter, Strategy)
15.	• Design Patterns (tentative: Template, Visitor, Factory, Abstract Factory, Chain of Responsibility)
16.	• Project Presentations & Demonstrations
17.	Final Exam

Grading Scheme

Assignments	5 %
Quizzes	5 %
Project	10 %
Midterms (2)	15+15 %
Final	50 %