COMP 3125 SOFTWARE ENGINEERING (3 credit hours)

Elmira College

SPRING 2025

Required Text:

1. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides (1994), ***Design Patterns: Elements of Reusable Object-Oriented Software***, 1st Edition, Addison-Wesley.
2. Grady Booch, James Rumbaugh, Ivar Jacobson (2005), ***The Unified Modeling Language User Guide***, 2nd Edition, Addison-Wesley.
3. Roger S. Pressman (2015), ***Software Engineering: A Practitioner's Approach***, 8th Edition, McGraw-Hill.

Pre-requisites: COMP 2112 Data Structures and Algorithms

Course Description

This course provides an in-depth exploration of both object-oriented and traditional software engineering methodologies, building upon the foundational analysis and design concepts previously introduced. It encompasses a comprehensive study of the entire software development lifecycle, from requirements gathering and system design to implementation, testing, and maintenance. The curriculum places significant emphasis on object-oriented principles and the application of the Unified Modeling Language (UML) to model and document software systems. Key topics include the fundamentals of software engineering, such as requirements engineering, software design patterns, system architecture, and quality assurance. The course also covers essential aspects of project management, including planning, scheduling, and risk assessment, to equip students with practical skills for real-world software development projects. Through a combination of lectures, hands-on projects, and case studies, students will gain a thorough understanding of modern software engineering practices and the ability to apply them in diverse development environments.

Course Objectives and Goals

- Gain a deep understanding of object-oriented concepts, including encapsulation, inheritance, and polymorphism.
- Get prepared for careers in software development by fostering critical thinking, problem-solving abilities, and a strong foundation in both theoretical and practical aspects of software engineering.
- Be able to create various UML diagrams, such as class diagrams, sequence diagrams, use case diagrams, and activity diagrams, to represent system structure and behavior.
- Acquire hands-on experience in implementing software systems using object-oriented programming languages.
- understand the importance of software quality and learn techniques for quality assurance, such as code reviews, refactoring, and performance optimization.

Evaluation of Performance

Your grade will be based upon your performance on exams, assignments, and participation.

4 Assignments	20%
2 Quizzes	15%
6 Labs	30%
Project	10%
Midterm Exam	10%
Final Exam	15%
Total	100%

Grades will be assigned as follows:

A	93% and above	B-	80 - 82%	D+	67 - 69%
A-	90 - 92%	C+	77 - 79%	D	63 - 66%
B+	87 - 89%	C	73 - 76%	D-	60 - 62%
B	83 - 86%	C-	70 - 72%	F	59% or below

Withdrawal Policy: Please see Elmira College Bulletin for information on this policy.

Academic Honesty: Please read the section on Academic Honesty in the **Code of Conduct**. Briefly, academic dishonesty includes: cheating, fabrication, facilitating academic dishonesty, and plagiarism. Ask if you have any questions on whether something constitutes as academic dishonesty. All work must be original and new. Past assignments from current or other courses will not be accepted. Academic dishonesty will not be tolerated. It will result in zero on the assignment, and a report will be filed with the school. Continued practice will result in failure of the class. Institutional penalties may also apply with repeated acts of academic honesty.

Student Responsibility:

- It is your responsibility to keep track of assignments and due dates.
- You should ask questions concerning assignments and lectures, if you need any clarifications.
- If you are struggling in class, have concerns, and/or unsure about expectations, please stop by during office hours or make an appointment for another time.

Tentative Schedule of Topics

Topic	Materials	Tasks & Evaluations
Introduction to Software Engineering	Chapter 1	
Object-Oriented Principles and UML Basics	Chapter 2	Assignment 1
The Software Development Lifecycle	Chapter 3	
UML Diagrams: Use Case, Class, and Interaction Diagrams	Chapter 4	Lab 1
Requirements Engineering	Chapter 5	Lab 2
Architectural Design and Object-Oriented Analysis	Chapter 6	Quiz 1
Design Patterns: Creational, Structural, and Behavioral	Chapter 7	Assignment 2

Agile Methodologies and Testing Strategies	Chapter 8	Lab 3
Component-Based Development and Formal Methods	Chapter 9	
Web Engineering and Project Management	Chapter 10	Midterm Exam
UML Diagrams and Best Practices	Chapter 11	
Design Patterns and Case Studies	Chapter 12	Lab 4
Agile Practices and Tools	Chapter 13	Assignment 3
Testing Techniques and Automation	Chapter 14	Lab 5
Component-Based Development	Chapter 15	Quiz2
Formal Methods and Verification	Chapter 16	Assignment 4
Advanced Web Engineering and Security	Chapter 17	Lab 6
Project Management and Risk Assessment	Chapter 18	
Comprehensive Application of Design Patterns in Software Architecture	Chapter 19	Project
Course Review and Future Trends	Chapter 20	Final Exam