Title of Course: Advanced Programming Lab-3 **Course Code: 18B17CI673**

L-T-P scheme:0-0-2 **Credit:2**

Prerequisite:

- Students should have a strong understanding of at least one programming language (e.g., Java, Python, C++).
- Proficiency in fundamental data structures (arrays, linked lists, stacks, queues) and algorithms is essential,
- Understanding of OOP concepts like classes, objects, inheritance, polymorphism, and encapsulation.
- Familiarity with basic software development concepts, including version control (e.g., Git), debugging, and testing.
- Basic knowledge of databases and SQL
- Familiarity with web development concepts (HTML, CSS, basic JavaScript) can be beneficial.

Objectives:

1. To empower students with advanced programming skills by immersing them in the principles of software design. By the end of the course, students will demonstrate the ability to apply these concepts in real-world scenarios, resulting in the creation of well-designed, modular, and maintainable software solutions.
2. To equip students with a comprehensive understanding and practical proficiency in modern software development concepts, focusing on Docker and microservices, enabling them to design, implement, and deploy scalable and maintainable software solutions.

Learning Outcomes:

Course Outcome	Description
CO1	Students will demonstrate the ability to follow systematic software design steps as instructed by the instructor. This includes understanding and adhering to a structured approach in the software development process.
CO2	Students will apply software design concepts, including principles of abstraction, modularity, and encapsulation, in practical scenarios. They will showcase the ability to translate theoretical knowledge into practical application.
CO3	Students will demonstrate adaptability to emerging technologies in the software development landscape, showcasing competence in the use of modern tools and methodologies
CO4	Students will demonstrate strong problem-solving skills and critical thinking, applying these skills to propose effective solutions for complex engineering problems.
CO5	Students will adhere to ethical considerations and demonstrate their communication skills in their coding practices, project presentations, and collaborative work.

Course Content:

Unit-1: Understanding Software Design, Definition of software design, Importance of good design in software development, Basic Principles of Software Design, Abstraction, Modularity, Encapsulation, Separation of Concerns (SoC)

Unit-2: Cohesion Types of cohesion (functional, sequential, communicational, procedural, temporal, logical), Strategies to achieve high cohesion, Coupling, Types of coupling (data coupling, control coupling, stamp coupling, data-structure coupling, control-structure coupling), Strategies to achieve loose coupling

Unit-3: Introduction to Design Patterns, Benefits of using design patterns, Common Design Patterns, Creational Patterns (Singleton, Factory, Abstract Factory), Structural Patterns (Adapter, Decorator, Proxy), Behavioral Patterns (Observer, Strategy, Command)

Unit-4: Introduction to Docker, Benefits of containerization, working with Docker, Docker containers, Docker images, Docker Compose

Unit-5: Microservices Overview, Characteristics and advantages of microservices architecture, Building Microservices, Design considerations, Communication between microservices, Deployment strategies for microservices, Docker and Microservices, Containerizing microservices, Orchestration with Docker Swarm or Kubernetes,

Evaluation Scheme:

Exams		Marks	Coverage
P-1		15 Marks	Uni1-1 & Unit-2
P-2		15 Marks	Unit-3, Unit-4 & Unit-5
Day-to-Day Work	Viva	20 Marks	70 Marks
	Demonstration	20 Marks	
	Lab Record	15 Marks	
	Attendance & Discipline	15 Marks	
Total		100 Marks	

Resources:

- "Clean Code: A Handbook of Agile Software Craftsmanship" by Robert C. Martin
- "Design Patterns: Elements of Reusable Object-Oriented Software" by Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides
- Docker documentation: https://docs.docker.com/
- Microservices architecture: Various online resources and case studies.