Department of Computer Engineering

Academic Term: First Term 2023-24

Class: T.E /Computer Sem – V / Software Engineering

Practical No:	**7**
Title:	**Design Using Object-Oriented Approach with Emphasis on Cohesion and Coupling in Software Engineering**
Date of Performance:	12/9/2023
Roll No:	9644
Team Members:	Mohtashim Ali, Aditya Dhikale, Siddhant Murade (9644) (9531) (9625)

Rubrics for Evaluation:

Sr. No	Performance Indicator	Excellent	Good	Below Average	Total Score
1	On time Completion & Submission (01)	01 (On Time)	NA	00 (Not on Time)	
2	Theory Understanding(02)	02(Correct)	NA	01 (Tried)	
3	Content Quality (03)	03(All used)	02 (Partial)	01 (rarely followed)	
4	Post Lab Questions (04)	04(done well)	3 (Partially Correct)	2(submitted)	

Signature of the Teacher:

Department of Computer Engineering

Academic Term: First Term 2022-23

Class: T.E /Computer Sem – V / Software Engineering

Signature of the Teacher:

Lab Experiment 07

Experiment Name: Design Using Object-Oriented Approach with Emphasis on Cohesion and Coupling in Software Engineering
Objective: The objective of this lab experiment is to introduce students to the Object-Oriented (OO)
approach in software design, focusing on the principles of cohesion and coupling. Students will gain
practical experience in designing a sample software project using OO principles to achieve high
cohesion and low coupling, promoting maintainable and flexible software.
Introduction: The Object-Oriented approach is a powerful paradigm in software design, emphasizing the organization of code into objects, classes, and interactions. Cohesion and Coupling
are essential design principles that guide the creation of well-structured and modular software.
Lab Experiment Overview:

1. Introduction to Object-Oriented Design: The lab session begins with an introduction to the Object-
Oriented approach, explaining the concepts of classes, objects, inheritance, polymorphism, and encapsulation.
2. Defining the Sample Project: Students are provided with a sample software project that requires design and implementation. The project may involve multiple modules or functionalities.
3. Cohesion in Design: Students learn about Cohesion, the degree to which elements within a module or class belong together. They understand the different types of cohesion, such as functional, sequential, communicational, and temporal, and how to achieve high cohesion in their design.
4. Coupling in Design: Students explore Coupling, the degree of interdependence between modules or classes. They understand the types of coupling, such as content, common, control, and stamp coupling, and strive for low coupling in their design.
5. Applying OO Principles: Using the Object-Oriented approach, students design classes and identify their attributes, methods, and interactions. They ensure that classes have high cohesion and are loosely coupled.
6. Class Diagrams: Students create Class Diagrams to visually represent their design, illustrating the relationships between classes and their attributes and methods.
7. Design Review: Students conduct a design review session, where they present their Class Diagrams and receive feedback from their peers.
8. Conclusion and Reflection: Students discuss the significance of Object-Oriented Design principles, Cohesion, and Coupling in creating maintainable and flexible software. They reflect on their experience in applying these principles during the design process.
Learning Outcomes: By the end of this lab experiment, students are expected to:
Understand the Object-Oriented approach and its core principles, such as encapsulation, inheritance, and polymorphism. Gain practical experience in designing software using OO principles with an emphasis on Cohesion and Coupling.

Learn to identify and implement high cohesion and low coupling in their design, promoting

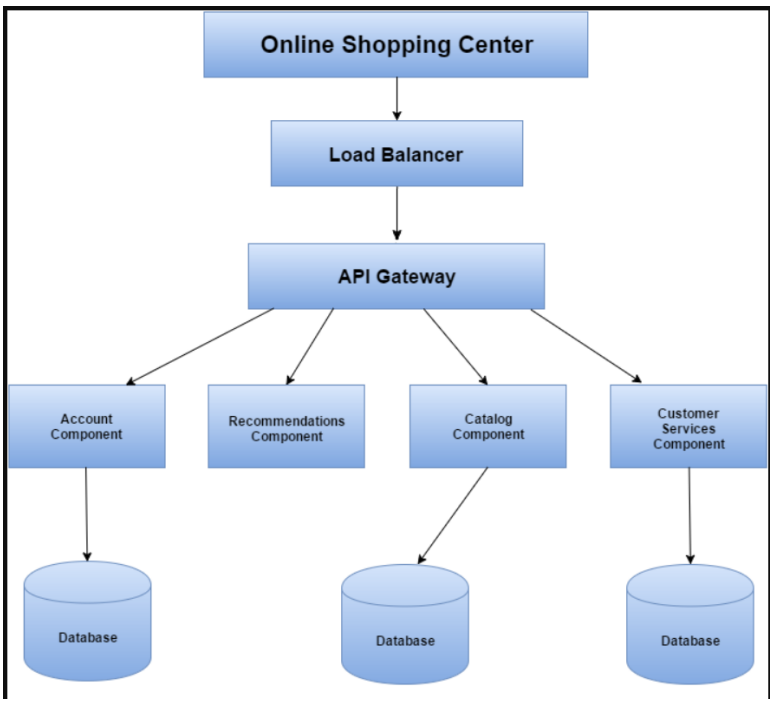
modular and maintainable code. Develop skills in creating Class Diagrams to visualize the relationships between classes. Appreciate the importance of design principles in creating robust and adaptable software.

Pre-Lab Preparations: Before the lab session, students should review Object-Oriented concepts, such as classes, objects, inheritance, and polymorphism. They should also familiarize themselves with the principles of Cohesion and Coupling in software design.

Materials and Resources:

Project brief and details for the sample software project
Whiteboard or projector for creating Class Diagrams
Drawing tools or software for visualizing the design

Conclusion: The lab experiment on designing software using the Object-Oriented approach with a focus on Cohesion and Coupling provides students with essential skills in creating well-structured and maintainable software. By applying OO principles and ensuring high cohesion and low coupling, students design flexible and reusable code, facilitating future changes and enhancements. The experience in creating Class Diagrams enhances their ability to visualize and communicate their design effectively. The lab experiment encourages students to adopt design best practices, promoting modular and efficient software development in their future projects. Emphasizing Cohesion and Coupling in the Object-Oriented approach empowers students to create high-quality software that meets user requirements and adapts to evolving needs with ease.



Shortcoming

Inability to Physically Inspect Products: When shopping online, you can't physically touch or try on the products, which can lead to disappointment if the item doesn't meet your expectations in terms of quality, size, or appearance.

Shipping Delays: Delays in shipping can be frustrating, especially when you need an item by a certain date. Weather, logistical issues, or unexpected events can cause delays in the delivery process.

Security Concerns: Online shopping involves sharing personal and financial information. Security breaches can lead to identity theft and unauthorized use of credit card information.

Sizing and Fit Issues: Online clothing shopping can be challenging because sizing can vary between brands. It's common to receive items that don't fit as expected, leading to additional returns and exchanges.

Hidden Costs: Some online retailers may add unexpected charges, such as shipping fees, taxes, or additional service charges, during the checkout process.

Overwhelming Choices: The vast selection of products available online can be overwhelming and make it difficult to make a decision. This abundance of choices can lead to decision fatigue.

Environmental Concerns: The convenience of online shopping often means increased packaging waste and transportation-related emissions, contributing to environmental concerns.

Updates :

Mobile Optimization: Ensure that your website is mobile-friendly and responsive. More and more people are shopping on their smartphones, so having a seamless mobile experience is crucial.

I**mproved User Experience (UX):** Continually refine your website's layout, navigation, and design to make it intuitive and user-friendly. A smooth and enjoyable user experience can lead to higher conversion rates.

Personalization: Implement personalized product recommendations based on a user's browsing and purchase history. This can enhance the shopping experience and encourage additional purchases.

Faster Load Times: Optimize your website's speed to reduce bounce rates and improve SEO rankings. Fast-loading pages are essential for keeping users engaged.

High-Quality Product Images and Videos: Provide multiple high-resolution images and videos for each product to give customers a more comprehensive view of what they are buying.

Customer Reviews and Ratings: Enable and promote customer reviews and ratings to build trust and provide social proof for your products.

Security Enhancements: Regularly update your website's security measures to protect customer data and build trust. Implement secure payment gateways and SSL certificates.

POSTLABS:

a) Analyse a given software design and assess the level of cohesion and coupling, identifying potential areas for improvement:
The software design of the "Samachaar website demonstrates reasonably good levels of cohesion and coupling Cohesion is apparent in well-defined components with distinct functions, while loose coupling allows for flexibility and minimal interdependence between components. To improve the design, the website can benefit from further enhancing cohesion by minimizing overlapping functions and ensuring each component has a single, dar responsibility. Additionally, refining interfaces and interactions between components, focusing on modularity, and conducting regular testing and refactoring efforts are recommended for ongoing software quality and maintainability.

b) Apply Object-Oriented principles, such as encapsulation and inheritance, to design a class hierarchy for a specific problem domain. In the "Vehicle" domain, we establish a class hierarchy using Object-Oriented principles:

1. Vehicle (Base Class)
-Properties: make, model, year
- Methods: start, stop, accelerate, brake

2 Car (Inberts from Vehide)
-Additional Properties: numDoors, fuelType
-Additional Methods: lockDoors, unlockDoors

3. Motorcycle (Inherits from Vehide):
-Additional Properties: hasHelmet Storage
-Additional Methods: putOnHelmet takeOffHelmet

4. Truck (Inhents from Vehide)
-Additional Properties: cargoCapacity
-Additional Methods: loadCargo, unload Cargo

This hierarchy exemplifies encapsulation, where properties and methods are contained within each dass, and inheritance, which allows specialized classes to inherit properties and methods from the base dass, promoting code reusability and structure.

c) Evaluate the impact of cohesion and coupling on software maintenance, extensibility, and reusability in a real-world project scenario. In a real-world project, cohesion and coupling have significant effects:

-Software Maintenance: High cohesion simplifies changes, and low coupling reduces unintended impacts during maintenance.

-Software Extensibility: High cohesion and low coupling ease the addition of new

features and components.

-Software Reusability: Well-structured, cohesive, and loosely coupled code is more reusable in various contexts.

In practice, striking a balance between cohesion and coupling is crucial for business agility, cost savings, team collaboration, and quality assurance in long-term projects.