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:	12-09-23
Roll No:	9772
Team Members:	Nash, Prince, Glen, Ayush

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 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.

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

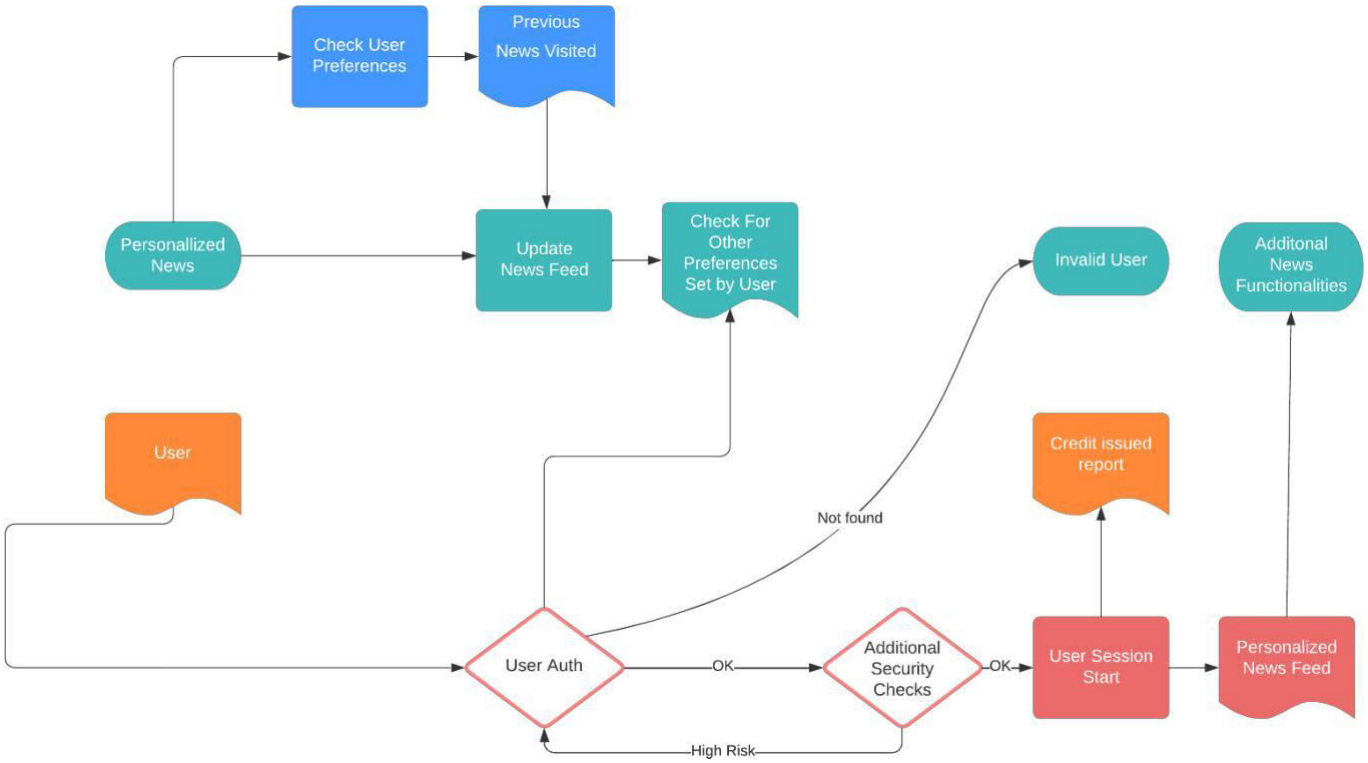
Materials and Resources:

- 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.

Nash Dabre
9767
TE COMPS B

SE EXP 7: Architecture



SHORTCOMINGS For "Samachaar" - Online News Portal:

1. **Limited Content Diversity:** "Samachaar" may have a limited range of news sources, potentially leading to a lack of diverse perspectives.
2. **Slow Loading Times:** Depending on the server and user's internet connection, the app may suffer from slow loading times, which can frustrate users.
3. **Browser Compatibility:** Compatibility issues with certain web browsers could exclude some users from accessing the portal effectively.
4. **Security Concerns:** Personalized news features may raise concerns about the security of user data, such as browsing history or preferences.
5. **Mobile Responsiveness:** The app may not be fully optimized for mobile devices, which could hinder the experience for users on smartphones and tablets.
6. **Limited Personalization:** The personalization algorithm might not be very accurate, leading to irrelevant or redundant news recommendations.
7. **Accessibility Issues:** The app may not meet all accessibility standards, making it challenging for individuals with disabilities to use effectively.
8. **Inaccurate Content:** The app may occasionally present inaccurate or outdated news content due to delays in content updates.
9. **Limited Interactivity:** It might lack interactive features for users to engage with news content beyond just reading.

10. **Advertisements:** Users may find the amount and placement of ads intrusive and disruptive to their reading experience.
11. **Resource Intensive:** The app could be resource-intensive, consuming a significant amount of system resources and potentially slowing down the user's device.
12. **Limited Offline Access:** Users may not have the ability to access previously personalized news content while offline, limiting their convenience.

UPDATES For “Samachaar” - Online News Portal:

1. **Expanded News Sources:** Include a broader range of news sources and categories to provide users with a more comprehensive news experience.
2. **Faster Loading Times:** Optimize the app's performance for quicker loading, ensuring a smoother and more efficient user experience.
3. **Enhanced Browser Compatibility:** Ensure full compatibility with popular web browsers to maximize the accessibility of the web app.
4. **Improved Security Measures:** Strengthen data security and privacy protocols to build trust with users, especially concerning personalization features.
5. **Mobile Optimization:** Implement responsive design and mobile app versions to cater to users on various devices and screen sizes.
6. **Refined Personalization Algorithm:** Continuously improve the personalization algorithm to deliver more accurate and relevant news recommendations.
7. **Accessibility Compliance:** Make the app fully accessible, adhering to accessibility standards to accommodate users with disabilities.
8. **Real-Time Updates:** Ensure that news content is updated in real-time to provide users with the latest information.
9. **Interactive Features:** Introduce interactive elements like commenting, sharing, and voting to encourage user engagement with news content.
10. **Balanced Advertisements:** Optimize ad placements and frequency to ensure that ads are unobtrusive and do not disrupt the user experience.
11. **Resource Efficiency:** Optimize resource usage to make the app run smoothly on a wider range of devices without straining system resources.
12. **Offline Access:** Allow users to access and save personalized news content for offline reading, improving convenience and accessibility.

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, clear 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 (Inherits from Vehicle):
   - Additional Properties: numDoors, fuelType
   - Additional Methods: lockDoors, unlockDoors

3. Motorcycle (Inherits from Vehicle):
   - Additional Properties: hasHelmetStorage
   - Additional Methods: putOnHelmet, takeOffHelmet

4. Truck (Inherits from Vehicle):
   - Additional Properties: cargoCapacity
   - Additional Methods: loadCargo, unloadCargo

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