Assignment-9 & 10 for Spring

Subject: CSW2 (CSE 3141)
Session: Jan to May 2025
Branch: CSE
Section: All
Course Outcomes: CO5 and CO6

Learning Levels: Remembering (L1), Understanding (L2), Application (L3), Analysis (L4), Evaluation(L5), Creation (L6)

Q no.	Questions	Learning Levels
Q1.	Create a Student class with private fields name, rollno, and email. Include getter and setter methods for all fields and a display() method to print the student's details (name, rollno, and email). Use appropriate naming conventions for the class and methods. Configure the Student beans using a Spring configuration file (applicationContext.xml). Use setter injection to assign appropriate values for the name, rollno, and email properties for two different student beans. Write a Main class that loads the Spring application context from the XML file, retrieves both Student beans, and demonstrates their usage by printing their details using the display() method or through direct access via getters. Ensure proper package structure and naming conventions. **Note:** Include Java classes, configuration file, and main application with proper naming conventions.	L3, L4
Q2.	Create a Sim interface with two abstract methods: calling() and data(). Then implement this interface in two classes: Airtel and Voda. Each class should provide specific implementations of the calling() and data() methods, printing appropriate messages. Configure two beans (sim1, sim2) in a Spring configuration file (applicationContext.xml) corresponding to the Voda and Airtel classes. Write a Mobile class with a main() method to load the Spring application context, retrieve both Sim beans from the container, and invoke the calling() and data() methods on each. Demonstrate loose coupling by depending on the Sim interface rather than the concrete classes. **Note:** Use proper package structure, naming conventions, and demonstrate interface-based bean injection with Spring.	L3, L4
Q3.	Create a **Vehicle** interface with **start()** and **stop()** methods. Implement this interface in	L3, L4

	two classes: **Car** and **Bike**. Each class should have private fields for **name** and **id**, along with appropriate getters and setters. The **start()** and **stop()** methods should print messages including the **name** and **id**. Configure the **Car** and **Bike** beans in a Spring configuration file (**.xml**). Assign appropriate values to the **name** and **id** properties for each bean. Write a **Transport** class with a **main** method that loads the Spring application context from the configuration file. Retrieve and use the **Vehicle** beans to call the **start()** and **stop()** methods. Note: Write a Spring Framework program with proper naming conventions.	
Q4.	Create a **User** class with name and email fields, along with appropriate getters and setters. Develop a **UserController** class that handles **GET** requests for displaying a user form and **POST** requests to process the form data and display the entered details on a new page. Create two JSP views: **user-form.jsp** for the input form and **user-details.jsp** for displaying the entered details. Configure the **dispatcher-servlet.xml** to enable Spring MVC, map the controller to the /user URL, and set up the view resolver. Use a **web.xml** file to configure the Spring DispatcherServlet. Add necessary dependencies in **pom.xml** for Spring MVC. Write the Spring MVC program with proper naming conventions and include the full set of files for Java classes, Spring configuration, JSP views, and dependencies.	L4, L6
Q5.	Create a simple Spring Boot web application that allows users to submit and display employee details using an HTML form. The application should define an **Employee** class with fields such as **empid**, **name**, **age**, and **salary**. Develop a controller that serves an HTML form at the root URL / for user input. Upon form submission, the data should be captured via a POST request and stored in an in-memory list. The application should then display the list of all submitted employees below the form on the same page. All employee data should be stored in memory. The goal is to demonstrate basic form handling, in-memory data storage, and web page rendering using Spring Boot.	L4, L6
	-END-	