SOFTWARE ENGINEERING LABORATORY (22IS43) TASK EXECUTION SHEET

Name:Aaditya Rao	**USN: 1NT23IS003**	**Date:09/05/2025**
Lab Activity # / Task #: LA-05	**Document name:Software Design**	**Submitted details:**

LA-05 : SOFTWARE DESIGN

Task 1: Design of Modules for Each Function

1. **Module Design**:
    - **Navigation Module**: It handles the vehicle's positioning, path planning, and updation of routes.
    - **Obstacle Detection Module**: Processes the input from sensors (LIDAR, radar, camera) to detect obstacles in real-time.
    - **Decision-Making Module**: Decides on actions based on sensor data, including collision avoidance, speed adjustment, and path planning.
    - **Control Module**: Interfaces with the vehicle's hardware to control steering, braking, and acceleration.
2. **Module Interfaces**:
    - **Navigation Module Interface**: Receives GPS and sensor data as input and provides updated routes.
    - **Obstacle Detection Interface**: Takes in sensor data and outputs the detected obstacles and their position relative to the vehicle.
    - **Decision-Making Interface**: Takes in the navigation data, obstacle data, and vehicle status and outputs control commands.
    - **Control Module Interface**: Receives the commands for steering, braking, etc., and sends feedback on vehicle status.

Task 2: Apply Abstraction and Identify Subsystems

1. **Subsystems**:
    - **Perception Subsystem**: Includes the obstacle detection and sensor fusion modules.
    - **Navigation Subsystem**: Includes the route planning and path updating modules.
    - **Control Subsystem**: Includes modules that manage steering, braking, and acceleration commands.

SOFTWARE ENGINEERING LABORATORY (22IS43) TASK EXECUTION SHEET

Name:Aaditya Rao	USN: 1NT23IS003	Date:09/05/2025
Lab Activity # / Task #: LA-05	Document name:Software Design	Submitted details:

- **Decision-Making Subsystem**: Includes the logic for making decisions based on sensor inputs, vehicle status, and the planned routes.

2. **Subsystem Interfaces**:
   - **Perception to Decision-Making**: Provides the detected obstacles and environmental data to inform the decision-making module.
   - **Navigation to Control**: Passes the updated route information and velocity commands to the control module.
   - **Decision-Making to Control**: Sends commands for braking, steering, and acceleration.

Task 3(a): Architectural Design Using Structured Design Principles

1. **Architecture**:
   - Define the high-level system structure:
     - **Input Layer**: Collects sensor data from cameras, LIDAR, radar, GPS.
     - **Processing Layer**: Perception, navigation, and decision-making modules.
     - **Output Layer**: Sends commands to the vehicle's control systems.
   - **Module Interaction**:
     - The Perception Module sends data to the Decision-Making Module, which in turn interacts with the Control Module.
     - Communication between subsystems occurs through well-defined interfaces (e.g., sensor data, control commands).
2. **Interface Specifications**:
   - **Perception Module Interface**: Receives sensor data, processes it, and sends detected obstacles and environmental data to the decision-making module.
   - **Navigation Module Interface**: Receives map and route information, processes it, and updates the navigation route.
   - **Control Module Interface**: Receives speed, steering, and braking commands from the decision-making module and sends feedback on vehicle status.

SOFTWARE ENGINEERING LABORATORY (22IS43) TASK EXECUTION SHEET

Name:Aaditya Rao	**USN: 1NT23IS003**	**Date:09/05/2025**
Lab Activity # / Task #: LA-05	**Document name:Software Design**	**Submitted details:**

Task 3(b): Architectural Design Using Object-Oriented Design Principles

1. **Objects and Class Definitions**:
    - **Vehicle Object**: Coordinates the entire system, providing a high-level interface for controlling vehicle actions.
    - **Sensor Object**: Represents individual sensors (LIDAR, cameras, radar) and provides methods for data collection.
    - **Control Object**: Manages low-level control of vehicle components (steering, acceleration).
    - **PathPlanning Object**: Encapsulates all methods for determining the vehicle's path based on input data.
    - **ObstacleDetection Object**: Processes sensor data to detect obstacles and provide hazard information.
2. **Object Interaction**:
    - **Vehicle Object** interacts with **Control Object** to manage real-time driving actions.
    - **PathPlanning Object** communicates with **Vehicle Object** to update the navigation and routing system.
    - **ObstacleDetection Object** feeds data into the **Vehicle Object** to influence decision-making regarding obstacles.

Task 4: Prepare Architecture Test Plan and Report Template

1. **Test Plan**:
    - **Test Strategy**: Perform both functional and non-functional testing (e.g., performance, stress).
    - **Test Cases**:
        - Validate module interfaces to ensure proper communication between modules.
        - Simulate failure scenarios (e.g., sensor failure, control malfunction) and check for proper fault handling.
        - Perform latency and load tests for real-time performance.

SOFTWARE ENGINEERING LABORATORY (22IS43) TASK EXECUTION SHEET

Name:Aaditya Rao	USN: 1NT23IS003	Date:09/05/2025
Lab Activity # / Task #: LA-05	Document name:Software Design	Submitted details:

- **Test Levels**:
  - **Unit Testing**: Test each module independently.
  - **Integration Testing**: Test the interaction between modules.
  - **System Testing**: Test the entire system under operational conditions.
  - **Acceptance Testing**: Validate the system against safety and regulatory standards.

2. **Report Template**:
   - **Test Case ID**
   - **Description**
   - **Expected Result**
   - **Actual Result**
   - **Pass/Fail**
   - **Comments**

Task 5: Conduct Architecture Design Testing and Report Results

1. **Testing Execution**:
   - Execute all defined test cases.
   - Identify and report any failures or discrepancies in the system.
   - Log defects and their corresponding resolutions.
2. **Test Results Report**:
   - **Test Summary**: Overview of testing activities, including pass/fail rates.
   - **Defect Summary**: Number and severity of defects found during testing.
   - **Test Coverage**: Percentage of requirements covered by tests.
   - **Test Log**: Detailed logs of each test case execution.

SOFTWARE ENGINEERING LABORATORY (22IS43) TASK EXECUTION SHEET

Name:Aaditya Rao	USN: 1NT23IS003	Date:09/05/2025
Lab Activity # / Task #: LA-05	Document name:Software Design	Submitted details:

Task 6: Analyze Architecture for Maintainability, Cohesion, and Modularity

1. **Maintainability**:
    - The architecture is designed to be flexible, with each module having clear responsibilities, which allows for easy updates and bug fixes.
    - Interfaces are well-defined to minimize dependencies between modules, enabling easier maintenance.
2. **Cohesion**:
    - Each module is designed to perform a single, well-defined task, ensuring high cohesion within modules.
    - For example, the **Obstacle Detection Module** only handles obstacle detection, while the **PathPlanning Module** only handles navigation.
3. **Modularity**:
    - The system is broken down into clearly defined subsystems (perception, navigation, control, decision-making), each of which can be independently updated or replaced.
    - This modularity makes it easier to swap components (e.g., upgrading sensor types) without disrupting the entire system.

Task 7: Review Design and Document Review Output

1. **Review Process**:
    - Organize design reviews with stakeholders, including system architects, developers, and safety engineers.
    - Gather feedback on design correctness, modularity, and compliance with standards (ISO 26262).
2. **Review Output**:
    - Document the feedback, including suggested design improvements.
    - Create action items based on the review and assign responsibilities for addressing them.
    - Summarize any changes made to the design following the review.

SOFTWARE ENGINEERING LABORATORY (22IS43) TASK EXECUTION SHEET

Name:Aaditya Rao	**USN: 1NT23IS003**	**Date:09/05/2025**
Lab Activity # / Task #: LA-05	**Document name:Software Design**	**Submitted details:**

Task 8(a): Implement High-Level Pseudocode with Documentation

1. **Pseudocode Example for Obstacle Detection Module**:

```
Function detect_obstacle(sensor_data):
    Obstacles = []
    For each sensor in sensor_data:
        If sensor detects obstacle:
            obstacles.append(sensor.get_obstacle_data())
    Return obstacles
```

2. **Documentation**:
    - **Inputs**: Sensor data from cameras, LIDAR, and radar.
    - **Outputs**: List of obstacles with positions and distance from the vehicle.
    - **Algorithm**: Iterates through sensor data and collects obstacles if detected.

Task 8(b): Apply Instructed Coding Guidelines

1. **Coding Guidelines**:
    - **Naming Conventions**: Use descriptive names for variables and functions, following camelCase for functions and variables.
    - **Indentation**: Use 4 spaces per indentation level.
    - **Commenting**: Provide comments for each function, class, and complex logic block.
2. **MISRA C Compliance**:
    - Adhere to the MISRA C guidelines for automotive safety-critical software, ensuring safe and predictable behavior.

SOFTWARE ENGINEERING LABORATORY (22IS43) TASK EXECUTION SHEET

Name:Aaditya Rao	**USN: 1NT23IS003**	**Date:09/05/2025**
Lab Activity # / Task #: LA-05	**Document name:Software Design**	**Submitted details:**

Task 9: Conduct Unit Tests and Report Testing Status

1. **Unit Testing**:
    - Execute tests on each module using predefined test cases.
    - Record the status of each test (pass/fail), and include any issues encountered.
2. **Test Reporting**:
    - **Test Coverage**: Provide details on what portions of the code have been tested.
    - **Issues Found**: List and categorize any issues (critical, minor).
    - **Test Results**: Document whether each unit test passed or failed, with links to any detailed logs or reports.