1. Use case: - Enabling Vehicles with Smart Central Locking Mechanisms.

2. Background:

   In modern automotive technology, Central locking systems have become increasingly common, offering convenience and safety by automatically locking all gates/ doors with a single button click. These systems rely on sensors to detect the current status and perform Locking/Unlocking/Alerting Mechanisms.

3. Objective:

   Design and implement a Central locking System that utilizes the set of Gate sensors to detect lock status and perform action accordingly. The system should enhance driver comfort and safety by centralizing control based on real-time conditions.

4. Key Components with Inter-component Connectivity:

- Door Sensor Management Service:

➔ Description: The Door Sensor Management Service oversees the functionality of sensors responsible for detecting the lock status of each door.
➔ Class: DoorSensorManagementService
➔ Methods:
➔ activate_sensor(sensor_id)
➔ deactivate_sensor(sensor_id)
➔ check_sensor_status(sensor_id)
➔ Relationships:
➔ Interacts with the Vehicle Parameters Repository to manage sensor data.
➔ Provides sensor status to the Central Locking System for locking/unlocking actions.

- Central Locking System:

➔ Description: The Central Locking System manages the locking and unlocking of all doors based on sensor inputs and user commands.
➔ Class: CentralLockingSystem
➔ Methods:
➔ lock_all_doors()
➔ unlock_all_doors()
➔ lock_single_door(door_id)
➔ unlock_single_door(door_id)
➔ analyze_lock_status()
➔ generate_alerts()
➔ Relationships:
➔ Receives sensor status from the Door Sensor Management Service to determine lock actions.
➔ Generates alerts if the engine starts and locks are not engaged.

- Vehicle Parameters Repository:

➔ Description: The Vehicle Parameters Repository manages dynamic vehicle data such as engine status, door lock status, and sensor readings.
➔ Class: VehicleParametersRepository
➔ Methods:
➔ update_parameters(vehicle_id, parameters)
➔ get_parameters(vehicle_id)
➔ Relationships:
➔ Stores and retrieves vehicle parameters for access by the Door Sensor Management Service and Central Locking System.

- User Interface Controller:

➔ Description: The User Interface Controller provides a user interface for interacting with the Central Locking System.
➔ Class: UserInterfaceController
➔ Methods:
➔ display_menu()
➔ process_user_input(choice)
➔ display_lock_status()
➔ display_system_alerts()
➔ Relationships:
➔ Interacts with the User Interface Service to display menus and receive user input.
➔ Displays lock status and system alerts retrieved from respective services.

- User Interface Service:

➔ Description: The User Interface Service handles the presentation layer of the Central Locking System.
➔ Class: UserInterfaceService
➔ Methods:
➔ display_menu()
➔ display_lock_status()
➔ display_system_alerts()
➔ Relationships:
➔ Renders menus and status information for display in the User Interface Controller.

- Error Handling Service:

➔ Description: The Error Handling Service manages errors and exceptions within the system.
➔ Class: ErrorHandlingService
➔ Methods:
➔ handle_sensor_failure(sensor_id)
➔ handle_communication_error(error_message)
➔ Relationships:
➔ Provides mechanisms to handle sensor malfunctions, communication errors, or other potential issues encountered by other components.

- Alert Management Service:

➔ Description: The Alert Management Service handles the configuration and sending of alerts for critical events or anomalies detected in the locking system.
➔ Class: AlertManagementService
➔ Methods:
➔ configure_alert_settings(alert_settings)
➔ send_alert(alert_message)
➔ get_all_alerts()
➔ clear_alerts()
➔ Relationships:
➔ Receives alerts triggered by the Central Locking System for further processing and user notification.