Modernizing XYZ Car Dealership's Ordering System

XYZ Car Dealership needs a modern ordering system overhaul. Your task is to design and implement this system using object-oriented programming principles.

Vehicle Types:

- Sedan
- Truck
- SUV
- Minivan

Dealer Information:

- Name
- Address

Vehicle Details:

- Model (String)
- Base Price (Fixed based on type) - $30,000 (Sedan), $35,000 (Truck), $40,000 (SUV), $45,000 (Minivan)
- Color (String)
- Model Year (Integer)

Additional Attributes (Type Specific):

- Truck: Cargo Bed Size (String - e.g., Short Bed, Long Bed)
- SUV: Roof Rack Type (String - e.g., Standard, Heavy Duty)

Optional Features:

- Enhanced Safety Features ($3,000)
- Security ($1,000)
- Entertainment System ($2,000)
- Sunroof ($2,500)

Requirements:

- Object-Oriented Design: Utilize classes with proper relationships (inheritance, aggregation/composition), access modifiers (private, protected, public), getters and setters.
- Inventory Management: Allow preconfigured vehicles with options to be stored and managed.
- Search Functionality: Search for vehicles by various criteria (most/least expensive, specific features).
- Vehicle Customization: Enable users to create custom vehicles with desired options and calculate the final price.
- Basic Algorithms: Implement functions for adding, removing, searching, and displaying available vehicles.
- Repositories: consider implementing a VehicleRepository class to store vehicle inventory and an OrderRepository class to store orders. This would allow you to save and load vehicle and order data between program executions
- Class Diagram: Create a visual representation of the class structure using a tool like Draw.io.

This modernizing system will equip XYZ Car Dealership with a user-friendly and efficient ordering system, enhancing customer experience and inventory control.