Car Service and Maintenance Tracker

Component 01: User Management

Description: Manages user-related functionalities, allowing registration, updates, and deletion of user accounts.

CRUD Operations:

- Create: Register a new user and store their details in users.txt.
- Read: Search for a user by username or ID and display their information.
- Update: Modify user details like email, password, or membership type.
- Delete: Remove a user account from the system.

UI Components:

- User Registration Page (HTML, JSP, Bootstrap/Tailwind CSS)
- User Login Page
- User Profile Update Page
- User List Page (Admin View)

OOP Concepts Applied:

- Encapsulation: Secure user data in a User class with getters/setters.
- Inheritance: AdminUser and RegularUser classes inherit from User.
- Polymorphism: Different user authentication mechanisms.

Component 02: Car Management

Description: Handles the addition, modification, and removal of cars and their owners in the service catalog.

CRUD Operations:

- Create: Add new cars to cars.txt with details like car number, owner name, car name, and category (e.g., car, SUV, etc.).
- Read: Search for owners by ID, name, car number, or category and display results.
- Update: Edit service details (e.g., updating service history or car information).
- Delete: Remove cars that are no longer receiving services.

UI Components:

- Car Addition Form
- Car Search Page
- Car Listing Page

OOP Concepts Applied:

- Encapsulation: Car class stores car details.
- Inheritance: SUV and Sedan classes inherit from Car.
- Polymorphism: Different display methods for different car types (e.g., SUV vs. Sedan).

Component 03: Service & Maintenance Tracking

Description: Manages the service and maintenance records for cars, tracking service dates, maintenance tasks, and due dates.

CRUD Operations:

- Create: Record a new service transaction in service_records.txt.
- Read: View all service records and check upcoming maintenance tasks.
- Update: Update service details (e.g., mark a service as completed or reschedule).
- Delete: Remove completed service records from the system.

UI Components:

- Service Booking Page
- Service History Page
- Upcoming Maintenance Page

OOP Concepts Applied:

- Encapsulation: ServiceRecord class for managing service details.
- Abstraction: Methods to check service availability before booking.
- Polymorphism: Different notification methods for regular and premium users.

Component 04: Admin Management

Description: Manages administrative users and their privileges in the system, allowing the creation and management of admin accounts.

CRUD Operations:

- Create: Register a new admin account in admins.txt.
- Read: View a list of all admins and their activity logs.
- Update: Modify admin details, such as permissions or profile updates.
- Delete: Remove admin accounts from the system.

UI Components:

- Admin Dashboard
- Admin Registration Page
- Admin Management Panel

OOP Concepts Applied:

- Encapsulation: Secure admin data in an Admin class.
- Inheritance: AdminUser inherits from User.
- Abstraction: Admin-only methods abstracted from regular users.

Component 05: Owner Management

Description: Manages car owners who use the service system.

CRUD Operations:

- Create: Add new owner details (name, contact info, car details) to owners.txt.
- Read: Search and view owners by name or car number.
- Update: Modify owner details such as contact information or car list.
- Delete: Remove owners who are no longer using the service.

UI Components:

- Owner Registration Page
- Owner List Page
- Owner Profile Edit Page

OOP Concepts Applied:

- Encapsulation: Owner class to store owner details.
- Inheritance: Different classes for IndividualOwner and FleetOwner.
- Polymorphism: Display cars differently based on owner type.

Component 06: Feedback and Review Management

Description: This module allows users to submit feedback and reviews for services they have received. It manages the collection, display, and moderation of user reviews.

CRUD Operations:

- Create: Users can submit new reviews or feedback for services, stored in reviews.txt.
- Read: Display existing reviews for specific services to other users.
- Update: Allow users to edit their submitted reviews or ratings.
- Delete: Provide users and admins the option to delete inappropriate or outdated reviews.

UI Components:

- Service Review Submission Page
- View Service Reviews Page
- Admin Review Moderation Panel

OOP Concepts Applied:

- Encapsulation: Review class secures feedback data.
- Inheritance: PublicReview and VerifiedReview classes.
- Polymorphism: Different display methods for admin and regular users.

This revised project structure now aligns with a Car Service and Maintenance Tracker, incorporating the necessary components and OOP concepts for managing cars, owners, services, and feedback.