Software Implementation and Testing Document

For

Group 9

Version 1.0

Authors:
Jack Hyland
Gabriel Rigdon
Joab Temotio

1. Programming Languages

- TypeScript (Frontend - React)
    - Improves code quality with static typing and better maintainability.
- JavaScript (Frontend - React)
    - Ensures flexibility and broad compatibility with web technologies.
- C# (Backend - ASP.NET)
    - Provides a secure, efficient, and scalable backend solution.
- SQL (PostgreSQL)
    - PostgreSQL is reliable for structured data.
- JSON (Data Exchange)
    - Standard for transmitting data between frontend and backend.

2. Platforms, APIs, Databases, and other technologies used

Platforms

- Vite + React (Frontend)
    - Faster development and optimized build times.
- ASP.NET (Backend)
    - Scalable and high-performance for API handling.

APIs & Services

- Swagger (API Documentation)
    - Simplifies API testing and documentation.
- JWT (Authentication)
    - Secures API access and user authentication.

Databases (Planned Implementation)

- PostgreSQL OR MongoDB
    - PostgreSQL supports structured data, while MongoDB offers flexibility.

3. Execution-based Functional Testing

- Unit Testing (Frontend & Backend)
    - Verified that buttons like “Add Car” and “Add Maintenance Item” trigger the correct actions.

- Manual UI Testing
    - Navigated through the Home Page, Car Profile Page, and User Profile Page to ensure all pages load correctly and are accessible via the navigation bar.
    - Submitted forms to verify that the entered data appears correctly.
- Edge Case Testing
    - Checked how the system handles empty or incorrect form inputs.

4. Execution-based Non-Functional Testing

- Conducted manual visual checks to ensure a consistent font and color scheme across all pages.
    - Verified that buttons have hover effects for user feedback.
    - Ensured that form validation messages appear correctly when fields are left blank.

- Reliability Testing

    - Simulated invalid inputs (e.g., empty fields, incorrect data types) to ensure the system does not crash.

- Maintainability & Scalability Testing

    - Reviewed the component-based structure of the React frontend to confirm modularity.

5. Non-Execution-based Testing

- Code Reviews

    - Team members checked each other's GitHub pull requests.

- Walkthroughs & Inspections

    - Regular team conversations discuss key parts of the project, including UI design, API structure, and database schema.
    - Walked through the user journey to identify potential UX issues before development.