Progress Report

- Increment 2 –
Group #9

1. **Team Members**
   Jack Hyland, Gabriel Rigdon, Joab Temotio

2. **Project Title and Description**
   DIY Vehicle Maintenance Manager

   We are developing an application for users working on DIY vehicle projects who wish to manage their part inventory and information. We will offer users an interactive service where they can track their vehicle and part data easily. Beyond just data storage, we want users to be able to receive alerts about expired information, among other time specific issues that may be associated with their data. Using a web interface, users will have an account where they can monitor multiple vehicles at once in a user friendly environment.

3. **Accomplishments and overall project status during this increment**

During this increment, significant progress was made in structuring the backend architecture. The backend follows a controller-enterprise controller-database context model, with PostgreSQL as the database. The controller handles API endpoints for data exchange, while the enterprise controller acts as an intermediary between the API and the database context, which manages queries. Models represent database entities like Cars and Users in a 1:1 mapping with PostgreSQL tables. The "car" module has been fully implemented, enabling end-to-end data flow between the database and API. With these components in place, the application now functions as a full-stack web application.

For the front end, the most notable detail is the complete overhaul to styling. We replaced the beige and green colors with dark blacks and blues. More detailed effects were also added. We also reworked the page structure. There is now a set position for each added vehicle. We also have a heatmap. refactoring was done to the way in which Modals are managed. No longer is the structure held in the page tsx file. We have created dedicated Modal tsx files for both Maintenance Items and Vehicles that derive from Modal.tsx to hold the structure.

4. **Challenges, changes in the plan and scope of the project and things that went wrong during this increment**

We realized we were slightly too ambitious in our plans for increment-2. Establishing a start to both backend and the database when our front end still had much work to be done proved not to be the best path to take. Therefore, we held off on our database development in the interest of moving towards a more finalized front end.

5. **Team Member Contribution for this increment**

Jack Hyland Contributions:

- Progress Report
  - Completed Sections 4 and 7
- Requirements and Design Document
  - Helped update Functional and Nonfunctional requirements
  - Updated Operating Environment detail
- Implementation and Testing Document

        - Helped update Execution Based Functional Testing and Execution Based Nonfunctional Testing
    - Source Code
        - Refactored Modal structure into new files for both AddCarModal and AddMaintenanceModal
            - Moved structure out of page files and into new files
            - Used original Modal.tsx as basis for new Modals
        - Created login page
            - Initially added login page linked to navbar button
            - Removed navbar button and made login page the first page via App.tsx
            - Added styling for login page to match root color schemes
            - Made temporary valid Username: root and Password: 0000

Joab Temotio Contributions:

Sign Out Button on Navigation Bar (#50)

- Added "Sign Out" button to the top navigation bar
- Logs out the user (mock logout)
- Redirects to login page
- Added red border hover effect for visual feedback

Sign Up Page Creation (#49)

- Created new Sign Up page with:
    - Username/Email field
    - Password and Confirm Password fields
- Styled consistently with Login page using shared CSS
- On submit, redirects user back to the login page

- Does not yet store or authenticate user data (mock only)

UserProfile Color Scheme Update (#48)

- Updated styling to match the app's dark theme
- Applied color variables (--color-bg-canvas, --color-text-primary, etc.)
- Improved spacing, layout, and visual consistency with other pages

Contributed to Part6 of the Progress report.

6. Plans for next increment

Our group's next focus will be implementing a PostgreSQL database to store user accounts, car information, and maintenance records. This will allow us to transition from mock data to a fully functional backend system. We also plan to host the database and backend services in the cloud, likely using DigitalOcean, which will enable remote access and ensure the application remains online and available at all times. In addition to backend development, we will be working on improving the overall user experience. Visually, we plan to enhance the interface by adding support for car images, icons, and a

cleaner layout. We also intend to add functionality for users to upload and store receipts or documents related to their vehicle maintenance. Another major component of this phase will be implementing a real login system with authentication and authorization, ensuring user data is secure and only accessible by the rightful owner. Ultimately, these next steps will move us closer to a robust, cloud-hosted application with a secure, user-friendly interface and reliable data handling.

7. Stakeholder communication

Dear all,

We are excited to share the latest progress on our Maintenance Tracker, our application design to help DIY mechanics track vehicle maintenance with ease. We'll begin by discussing the front end development. The most notable change is the complete shift in styling. The original beige and green has been replaced with dark blues and blacks. This color tone both better fits our goal and appears more professional. Some button styling has also changed, including hover effects. Again, this gives the site a more professional look In terms of functionality the first main note is the navigation bar is no longer the access point for the car profile page. Instead, now when a vehicle is added, the resulting information displayed within a car tile logically acts as the link to the car profile page. Additionally, we have added both a login and sign up page. The login page is what appears when the page is initially opened, and the sign up page is accessible through the navbar. The navbar also received a new button with a link to this project's Github page. Although time was spent updating the style of the website, we still expect to finish on time. We believe this change is more than worth it for the user experience.

We have also made significant strides in developing the backend components that power the application's core functionalities. We have established five key HTTP endpoints that facilitate seamless interactions between the client interface and the server. The POST endpoint allows users to create new Car records by sending relevant data to the server, which then processes and stores this information in the database. To retrieve all existing Car records, the GET endpoint allows clients to request a comprehensive list, ensuring users have access to the complete dataset. For accessing specific Car records, the GET/{id} endpoint permits clients to request details of a particular Car by specifying its unique identifier. Updating existing Car records is facilitated through the PUT endpoint, where clients send updated data for a Car, identified by its unique ID, to modify the corresponding entry in the database. Finally, the DELETE endpoint allows for the removal of a specific Car record from the database by specifying its unique identifier.

We look forward to finalizing the website. We will get started on creating all other required endpoints, cleaning up the car and user profile page, creating storage of usernames and passwords for authorization, and completing our database and hosting it in the cloud.

Best,
Team 9

8. Link to video

https://www.youtube.com/watch?v=CEe6t6zw9qM&ab_channel=JoabAlexander