Smart OBD System

Team 3 - Cole Arduser, Luke Farmer, Sam Loecke, Samuel Nicklaus

Project Background

Motivation

- Saw need for vehicle diagnostics and analysis tools beyond traditional OBD systems.

Purpose

- Provide real-time dashboard data visualization and a detailed web-based post-trip analytics.

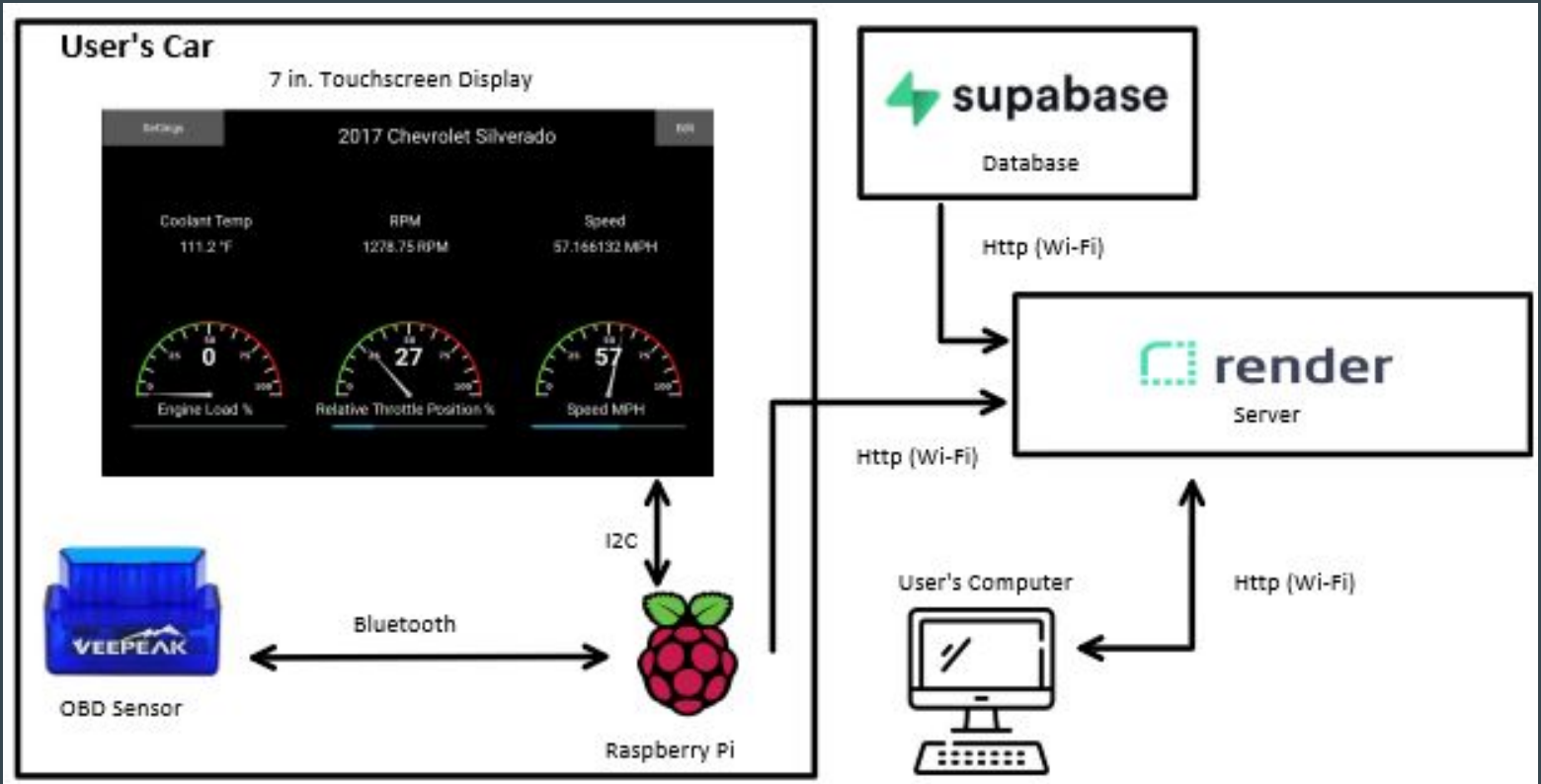
Design Focus

- Creating a user-friendly interface to make advanced vehicle diagnostics for all car owners

Project Goal

- Improve vehicle maintenance and safety by increasing driver awareness and vehicle longevity

System Overview



Reading from the Car's OBD Port

- Using the Veepeak bluetooth OBD sensor to read from the car's OBD port
- The OBD port is located under the steering wheel/driver's side dash
  - All gas cars made after 1996 are required to have an OBD port
- Queries the car's computer for live data along with diagnostic trouble codes (DTCs)
- Sensor is bluetooth enabled to be able to connect to our Raspberry Pi
- Sensor receives commands from the Pi, queries the car, and returns the results

ELM-327 OBD-II Emulator

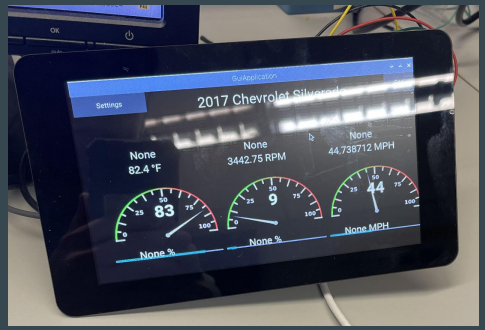
- Emulates a car's OBD port directly on Pi
- Supports basic ELM327 commands and OBD service requests
- Allows for customization of emulated data
- Python library made by Github user Ircama

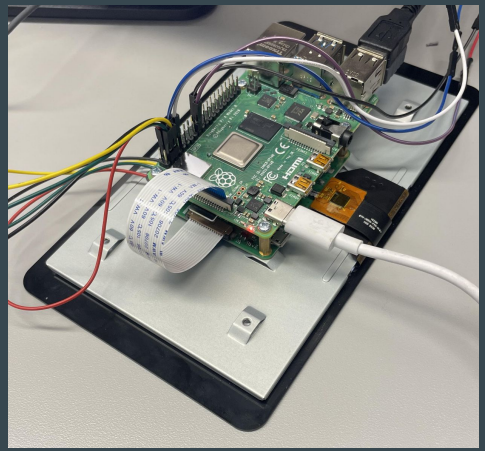
Collecting Data with the Raspberry Pi

- Using the Python OBD library
- First querying the car for all available data points
- Then querying the car's OBD port for all available live data every second
- Passing the received data to the GUI for use in the live dashboard
- Storing the data into the MariaDB database
  - Stored along with a timestamp
- Logging the data to text files as a back up

In Car Dashboard

- 7” Touchscreen Display
- 3 pages
  - Data Display
  - Display Selection
  - Settings
- Uses Kivy
- Allows user to select data types they’d like to see
- Allows user to connect to internet and upload data





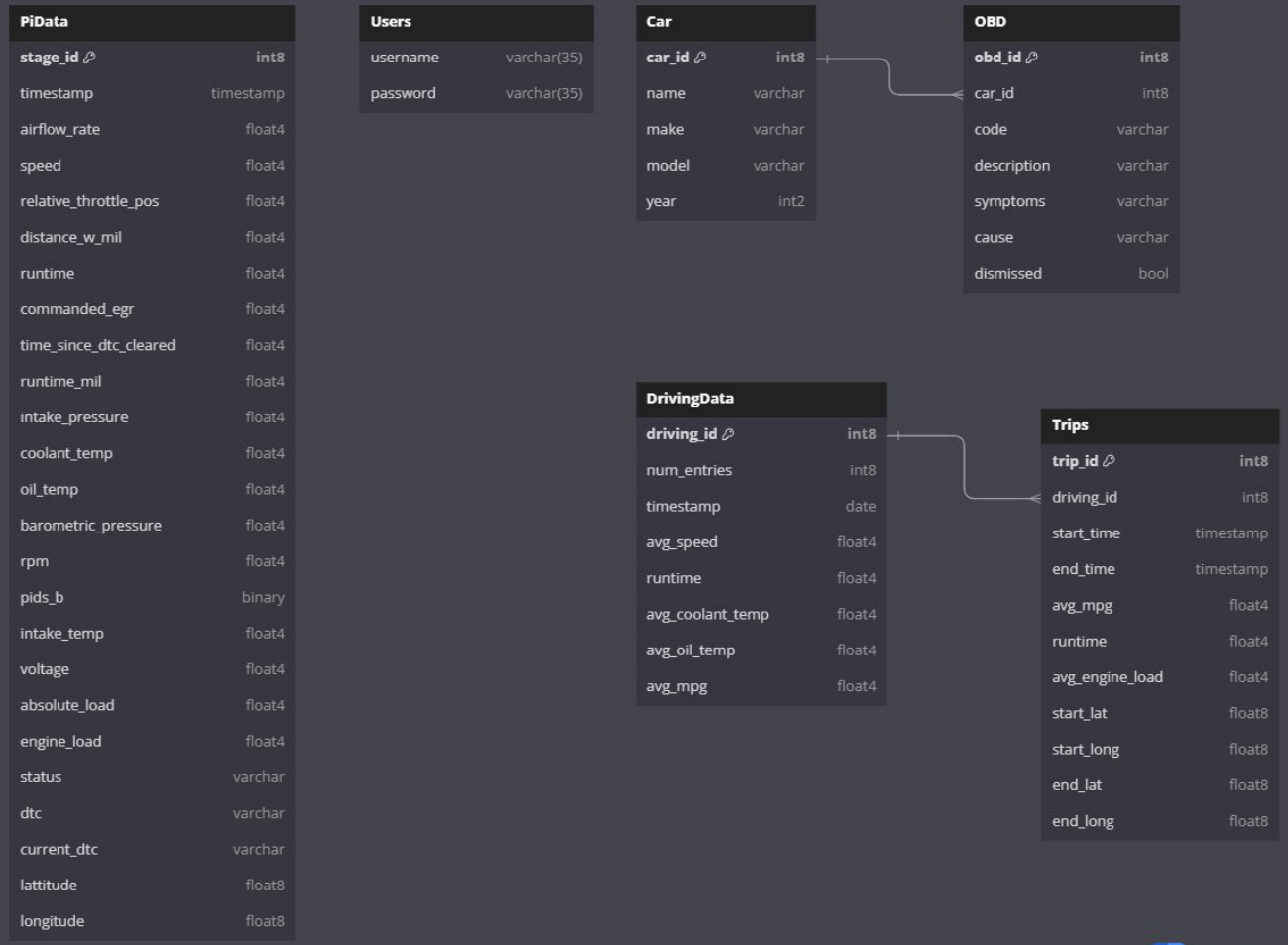
Pi Database

- Uses a local MariaDB database
  - MariaDB is based on a MySQL database
- Saves all available information from the car's OBD port
- Stores data for every second the car is running
- Uploads all of its content to the server via flask apis when connected to the internet

Server

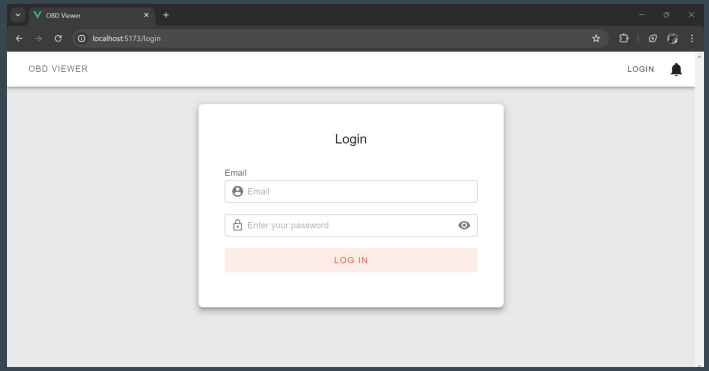
- Python/Flask based implementation
- Connects to Supabase for storage
  - PostgreSQL based
- Server handles the processing of car data
  - Sorts the data by Day
  - Sorts the data into “Trips”
- User auth and verification done through JWT Library

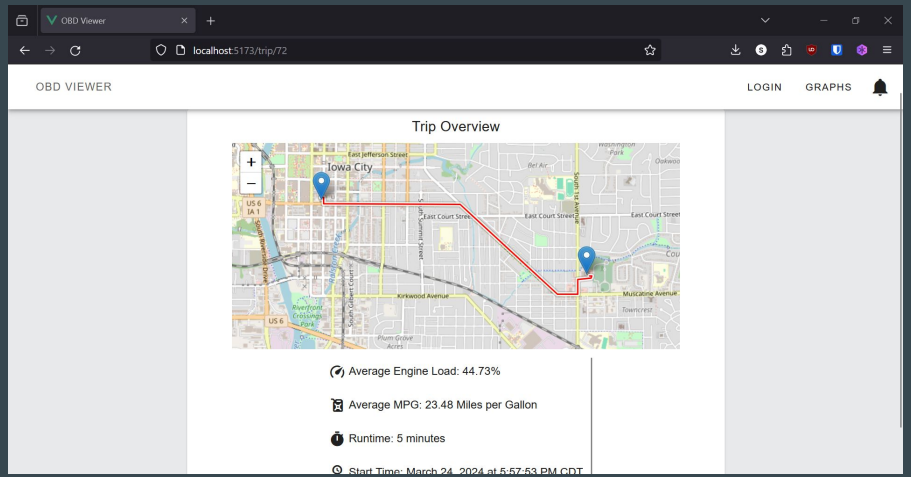
DB Diagram

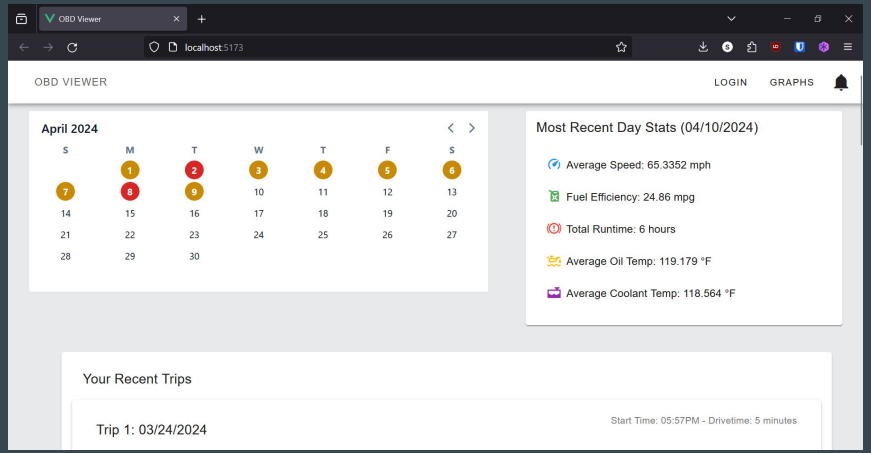

PiData
stage_id int8
timestamp timestamp
airflow_rate float4
speed float4
relative_throttle_pos float4
distance_w_mil float4
runtime float4
commanded_egr float4
time_since_dtc_cleared float4
runtime_mil float4
intake_pressure float4
coolant_temp float4
oil_temp float4
barometric_pressure float4
rpm float4
pids_b binary
intake_temp float4
voltage float4
absolute_load float4
engine_load float4
status varchar
dtc varchar
current_dtc varchar
lattitude float8
longitude float8
Users
username varchar(35)
password varchar(35)
Car
car_id int8
name varchar
make varchar
model varchar
year int2
OBD
obd_id int8
car_id int8
code varchar
description varchar
symptoms varchar
cause varchar
dismissed bool
DrivingData
driving_id int8
num_entries int8
timestamp date
avg_speed float4
runtime float4
avg_coolant_temp float4
avg_oil_temp float4
avg_mpg float4
Trips
trip_id int8
driving_id int8
start_time timestamp
end_time timestamp
avg_mpg float4
runtime float4
avg_engine_load float4
start_lat float8
start_long float8
end_lat float8
end_long float8
dbdiagram.io

Client

- Vuejs Framework
- Communicates to server through HTTP calls
- Handles user management through JWT Token and Cookies
- Allows user to view summarized information about their vehicle data







Questions?

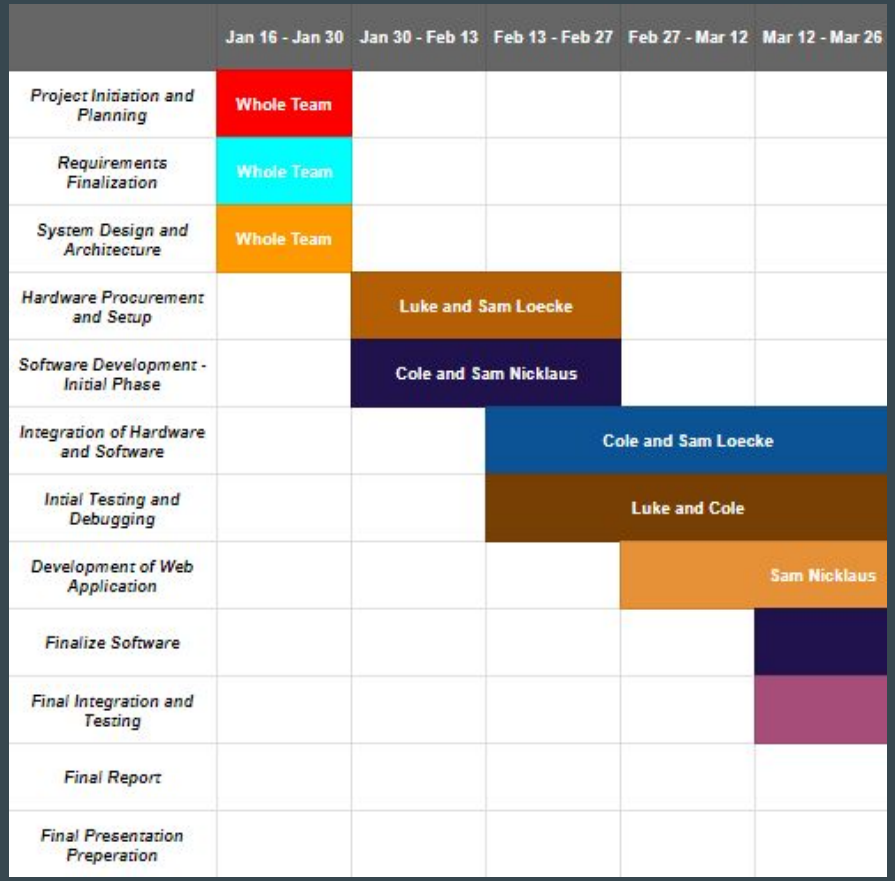
Project Objectives

Completed

- Project Initiation and Planning
- Finalized Requirements
- Hardware Procurement and Setup
- Software Development - Initial Phase

In Progress

- Integration of Hardware and Software
- Initial Testing and Debugging
- Development of Web Application



Future Project Objectives

- Finalize Software
- Final Integration and Testing
- Final Report
- Final Presentation Preparation

Team Member Contributions

- Sam Loecke
  - Sensor and reading data
- Luke Farmer
  - Pi GUI
- Cole Arduser
  - Pi Database to Server Connection
  - OBD Emulator
- Samuel Nicklaus
  - Web Application
  - Server Help