Fleet Monitoring System

Design Document

Team #12
Lofti Ben Othmane
Joe Herrera - Mobile Developer
Marco Yepez - Embedded System and Mobile Developer
Nicolas De la Cruz - Back-end Developer
Lorenzo Chavarria - Web Application Developer
sddec20-12@iastate.edu
http://sddec20-12.sd.ece.iastate.edu/

Executive Summary

Development Standards & Practices Used

- Agile Development Standards
- JavaScript Coding Standards
- Google Java Style Standards
- Oracle PLSQL Coding Guidelines

Summary of Requirements

- Develop Android and iOS mobile applications for Fleet Monitoring System
- Develop a web application for Fleet Monitoring System
- Provide GPS location of each vehicle in a fleet
- Provide data analytics of vehicle data
- Provide client chat
- Provide SMS notifications

Applicable Courses from Iowa State University Curriculum

List all Iowa State University courses whose contents were applicable to your project.

- CPR E 288
- COM S 309
- COM S 363
- COM S 352
- S E 319
- S E 329
- S E 339

New Skills/Knowledge acquired that was not taught in courses

- Server configuration
- Rest API creation
- Hardware equipment knowledge, e.g. resistors

Table of Contents

List of figures/tables/symbols/definitions (This should be the similar to the project plan)

1 Introduction

1.1 Acknowledgment

Acknowledgment to team SDMay18-2018, the team with the first version of this project.

1.2 Problem and Project Statement

With UPS, FedEx, and numerous fleet companies out there. We need something to monitor and manage these fleets. With our fleet monitoring system, a company would be able to use our user-friendly solution. Our solution revolves around web and mobile applications to view GPS locations of each vehicle, SMS notifications, client chat, and data analytics. Our solution would be based on MongoDB for our database and a server running Node.js Rest API for microservices. The fleet would need a PiCan for each vehicle to be able to send data from the vehicle to the server. Through the connectivity of each component, our solution would provide clients a fleet monitoring system.

1.3 Operational Environment

The conditions for this product are regular conditions inside the car for the PiCan device. Regular conditions would be restricted to the device being dry, avoiding all liquids.

1.4 Requirements

- Develop Android and iOS mobile applications for Fleet Monitoring System
- Develop a web application for Fleet Monitoring System
- Provide GPS location of each vehicle in fleet
- Provide data analytics of vehicle data
- Provide client chat
- Provide SMS notifications

1.5 Intended Users and Uses

- Managers
    - Add users to groups
    - Accept new driver account request
    - Analyze data from vehicles
- Drivers
    - Submit driver account request
    - Indirectly produce usage data while driving

1.6 Assumptions and Limitations

Assumptions:

-There will be one manager and many groups that will consist of drivers. The manager will be able to control the different groups and see different data analytics.

Limitations:

-The idea of hardware is that it has to be in a small workspace.

-PiCan device can only be tested and used on cars of the year 2008 and newer.

-Limited knowledge of hardware and server.

1.7 Expected End Product and Deliverables

Deliverable 1: Web Application - This web page will allow a user (manager or driver) to log in. As a manager, they would be able to view real-time tracking of all the vehicles currently being driven. A chat system will be implemented so that the manager can communicate with drivers.

Deliverable 2: Mobile Application - The mobile application, in practice will be similar to the web application. There are performance bottlenecks with mobile devices as they don't have the computing power of a computer. Therefore, the mobile may contain only a smaller amount of data analytics and may not have the performance to run real-time tracking.

Deliverable 3: PiCan - The PiCan is the hardware that will be installed on the vehicles to extract data and send it to the server. Data that is extracted from the vehicle includes mileage, fuel, speed, etc.

Deliverable 4: Server - The server will be a Node.js Rest API to provide Microservices for PiCan data posting and the client can request and post data to the database.

Deliverable 5: Database - Mongo Database to store user information, registered vehicles, registered drivers, and vehicle data.

The end product would include the web application, mobile application, PiCan, server, and database that all work together to create a fleet monitoring system. The fleet monitoring system includes hardware that must be installed on a vehicle. The hardware will communicate data to the server so that the manager and drivers can use the web/mobile application to view data analytics. The Fleet Monitoring System is set to be completed by December 2020.

2. Specifications and Analysis

2.1 Proposed Approach

Approach:

- Use MongoDB for scalability, as we need to collect extreme amount of data from vehicles
- Rest API for Microservices in Node.js using Microsoft Azure Web API Design
- Dynamic web page creation using Node.js and Express.js
- Google Java Style Standards

2.2 Design Analysis

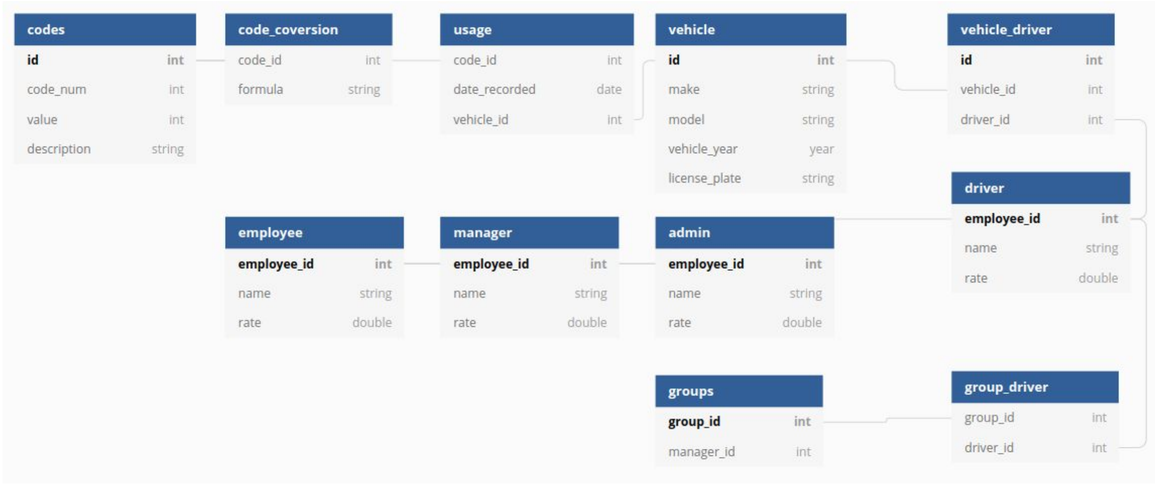
In these weeks, we have met with our adviser and client to discuss what the previous team from a past semester completed. We discussed what and how we should implement for a successful project. Our team has been able to produce design prototypes for an Android and website app. We have also been able to successfully replicate what the previous team completed for the raspberry pi device and vehicle. In the end, we want to implement these front-end designs, refractor the embedded system program, and create a successful, sustainable backend. Our strengths will be that we are most familiar with how our product and the software will work. Our weakness is mostly that we lack some knowledge of hardware and server.

2.3 Development Process

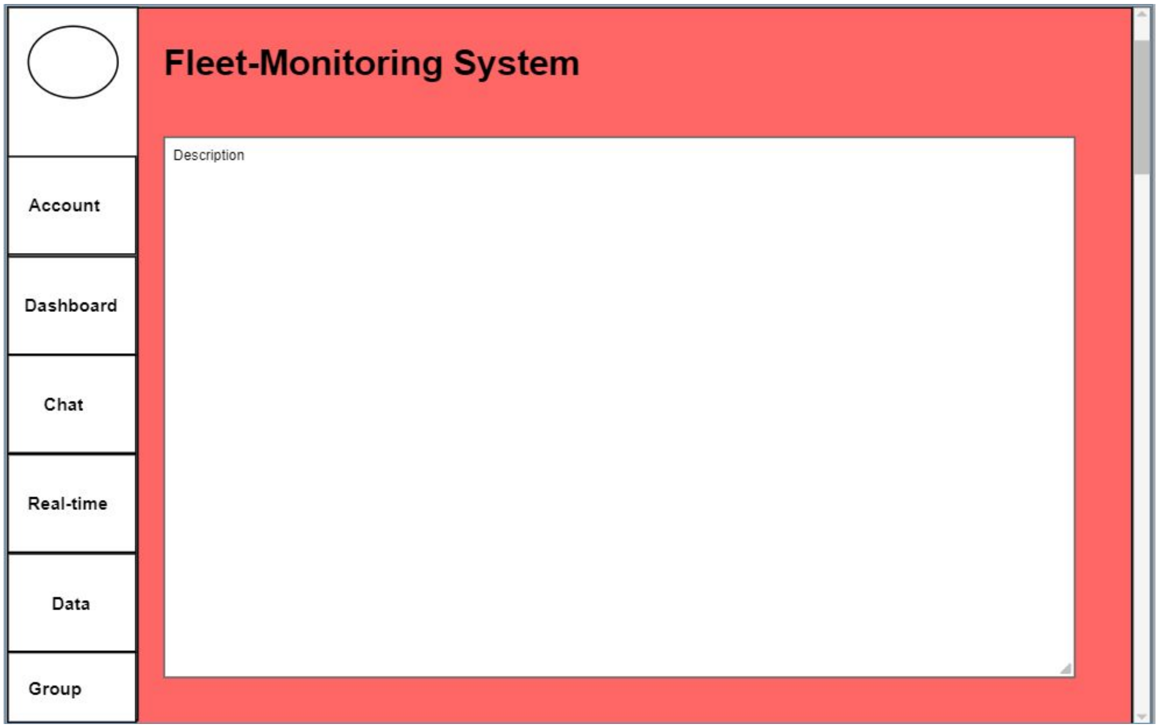
Our team will follow the Agile software development process, our development sprints will be three-weeks long. We meet with our advisor and team weekly for updates, questions asked, and any other relevant topics for our project.

2.4 Conceptual Sketch

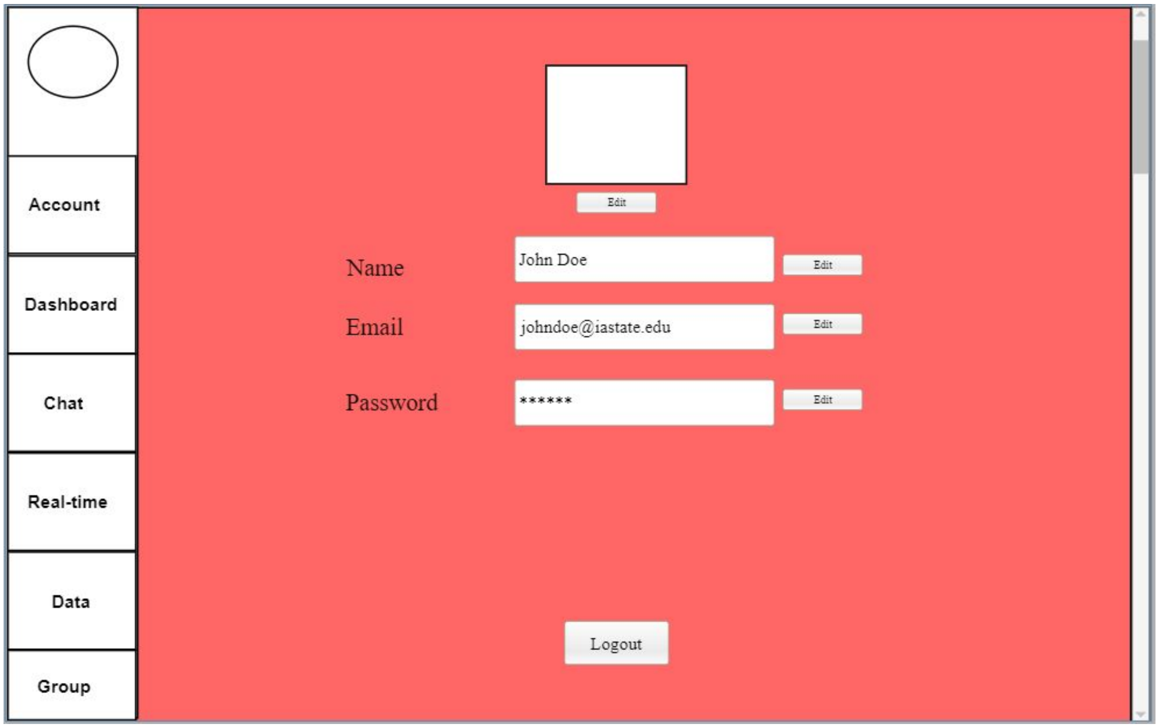
Database Architectural Design:



Web Application Design:



Home Page

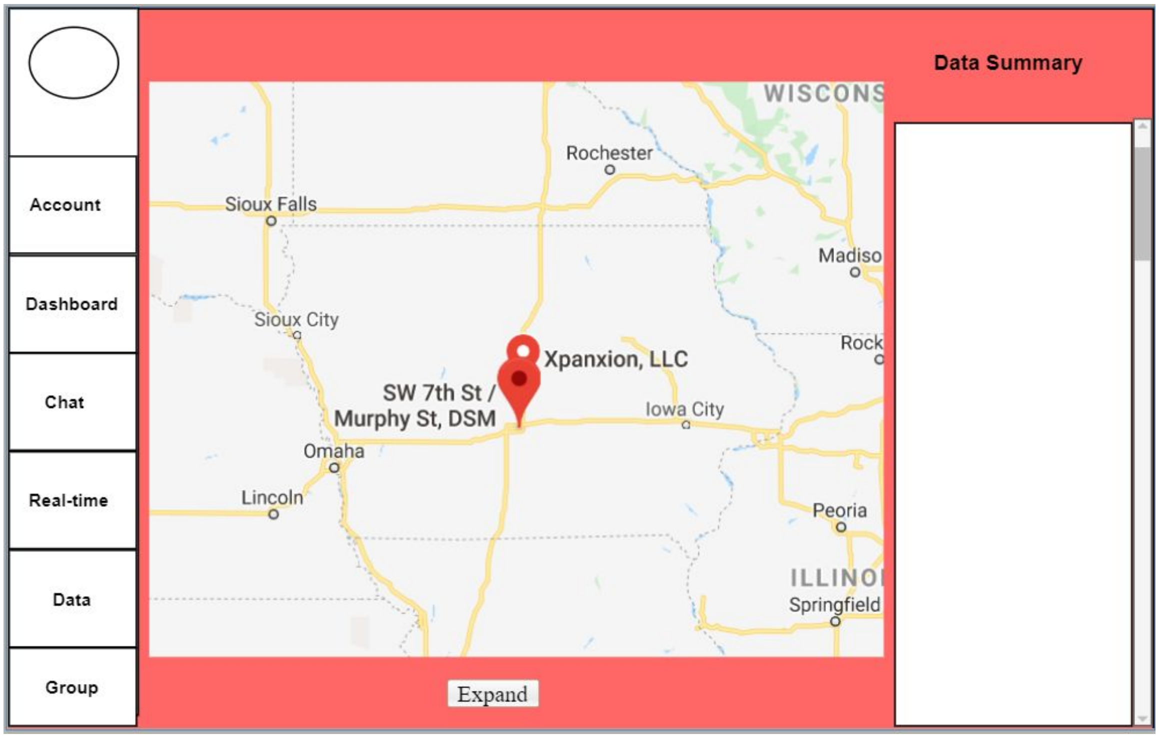


Name: John Doe

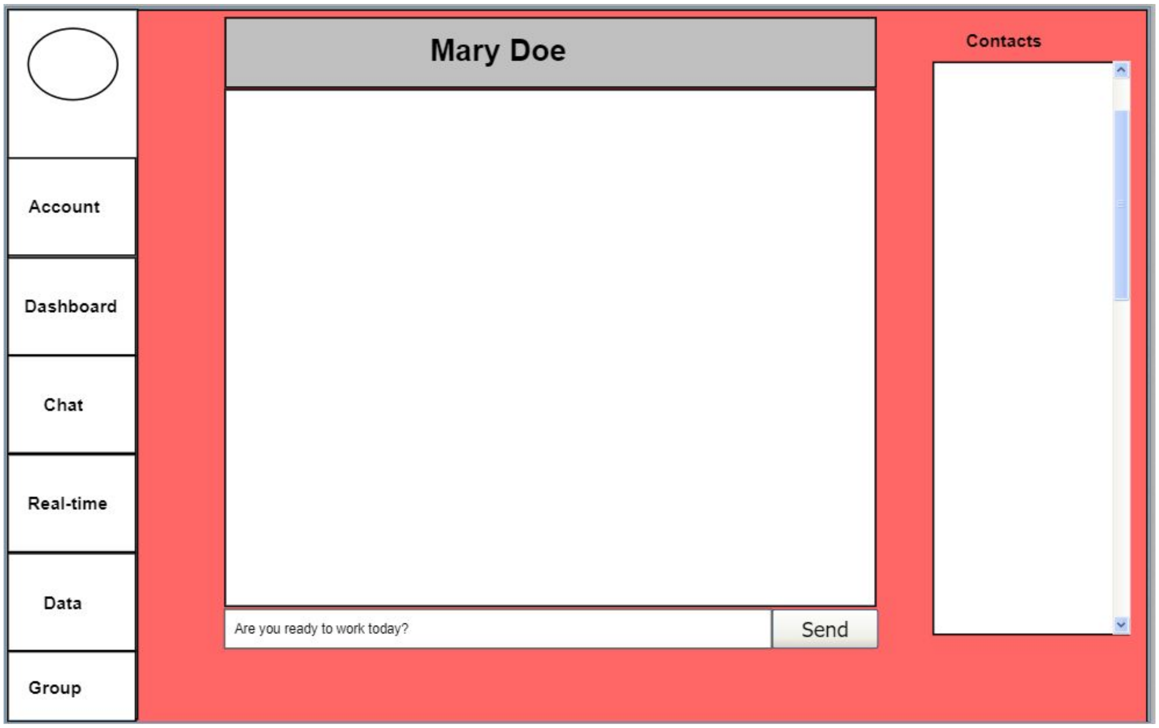
Email: johndoe@iastate.edu

Password: ******

Account Page

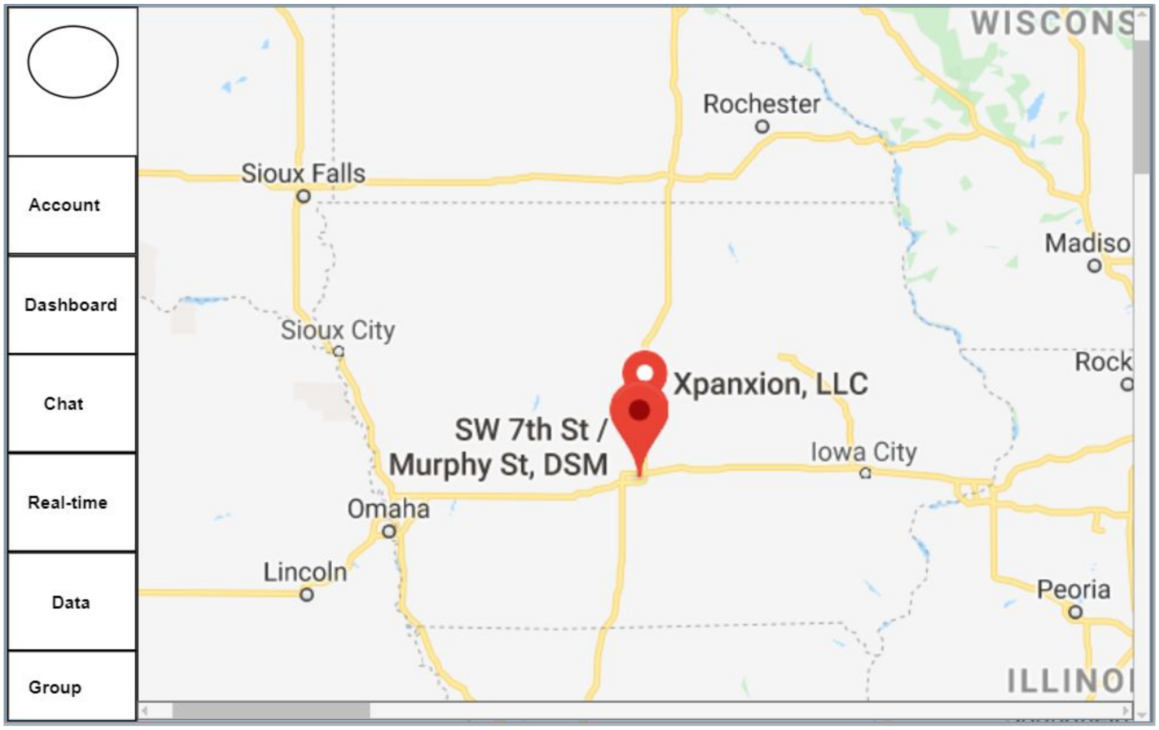


Dashboard Page

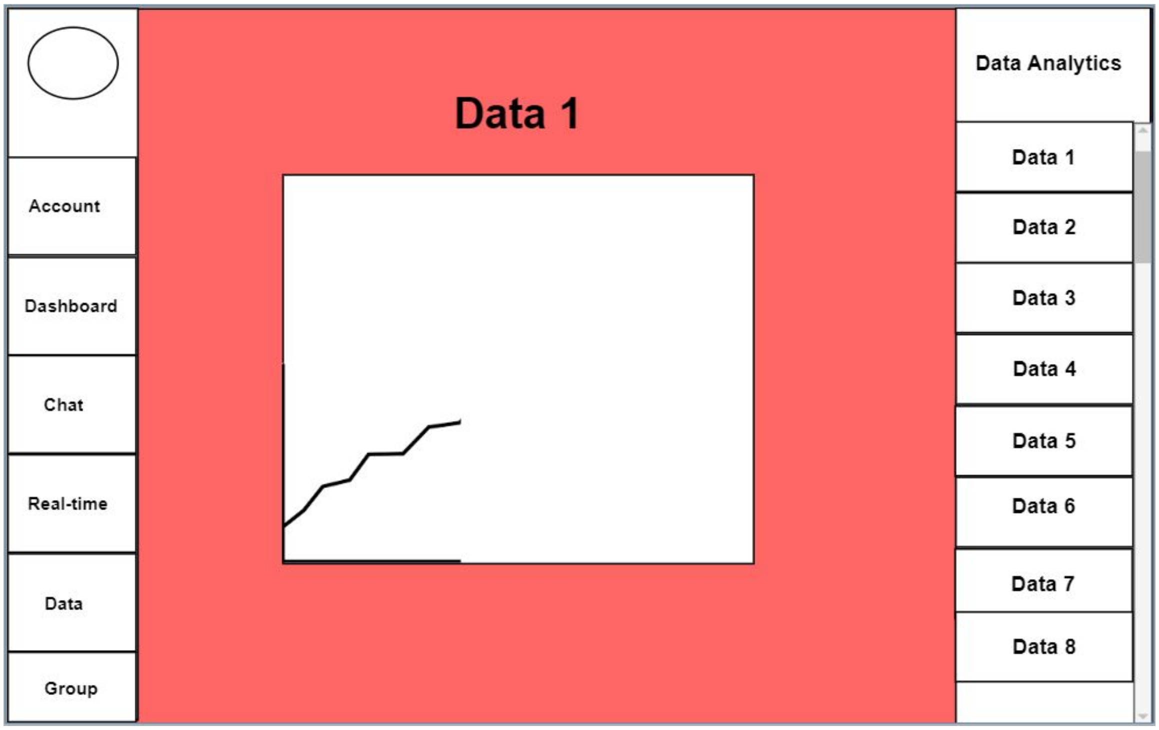


Are you ready to work today?

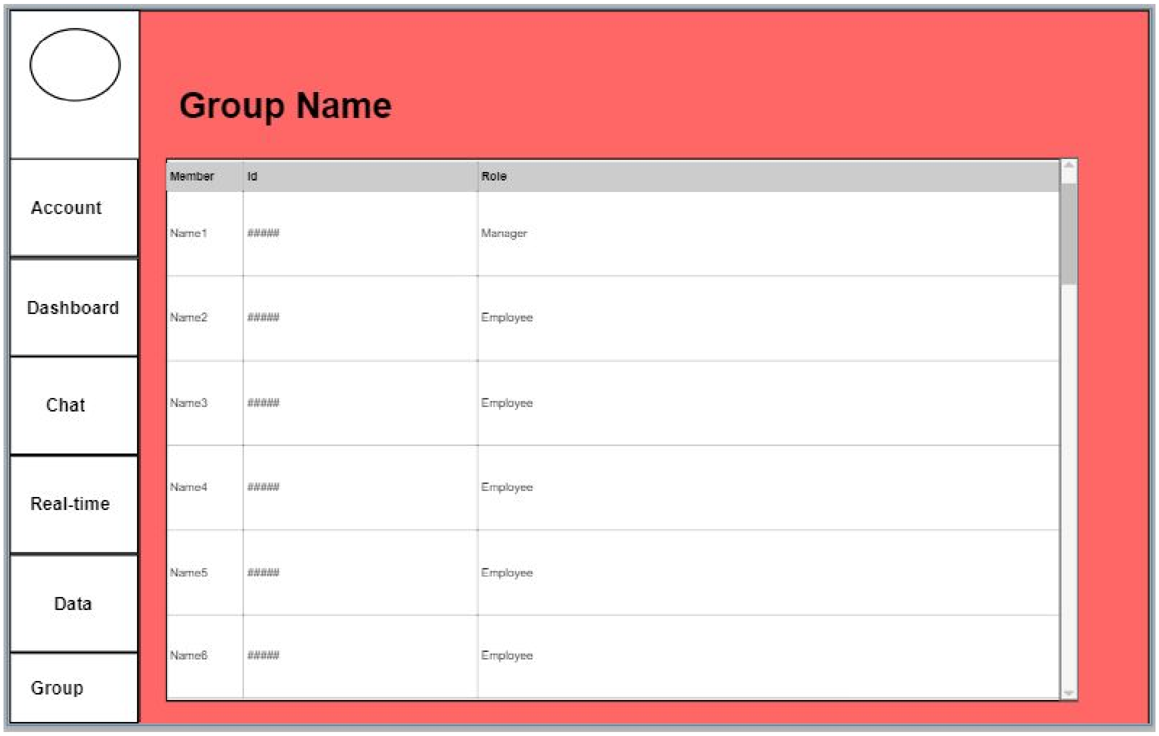
Chat Page



Real-time Data Page



Data Analytics

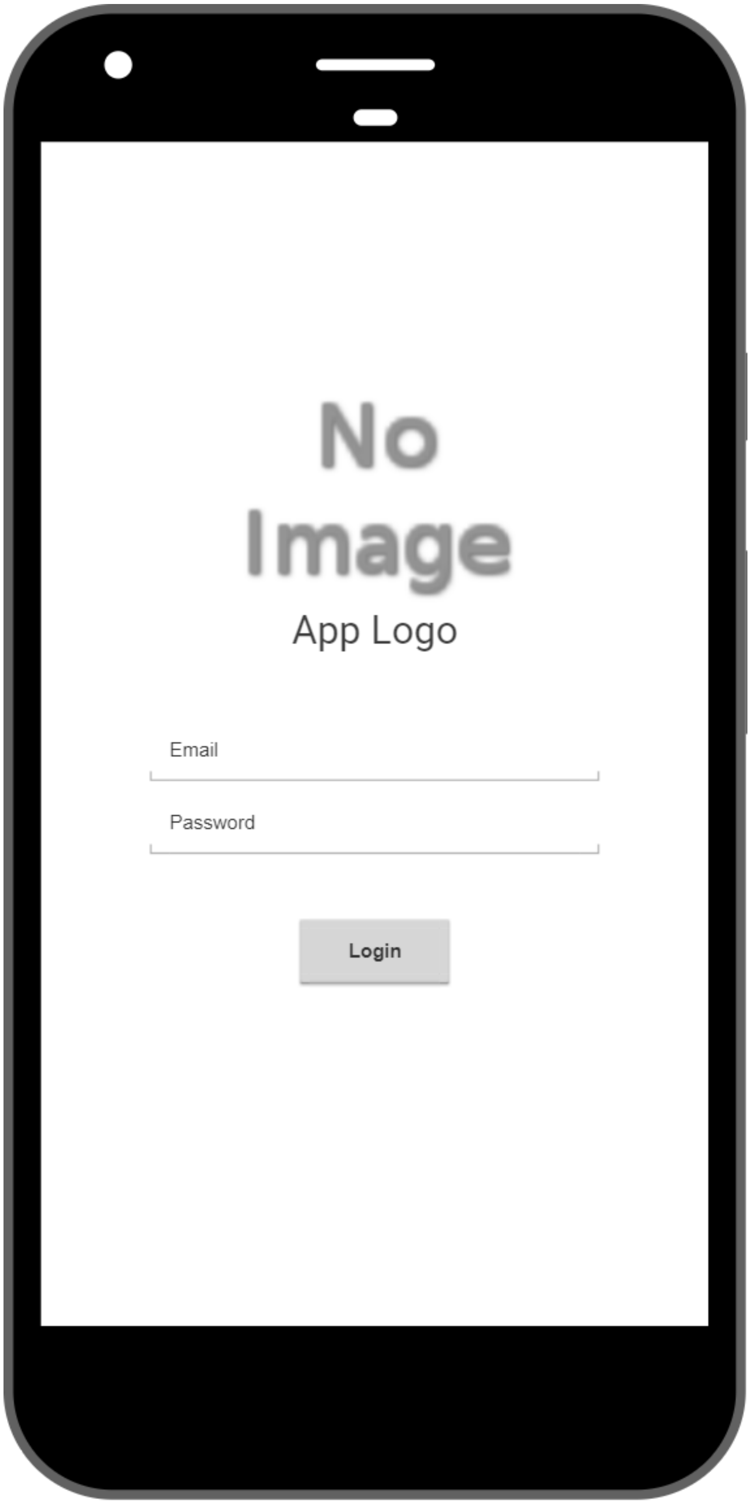


Group Name

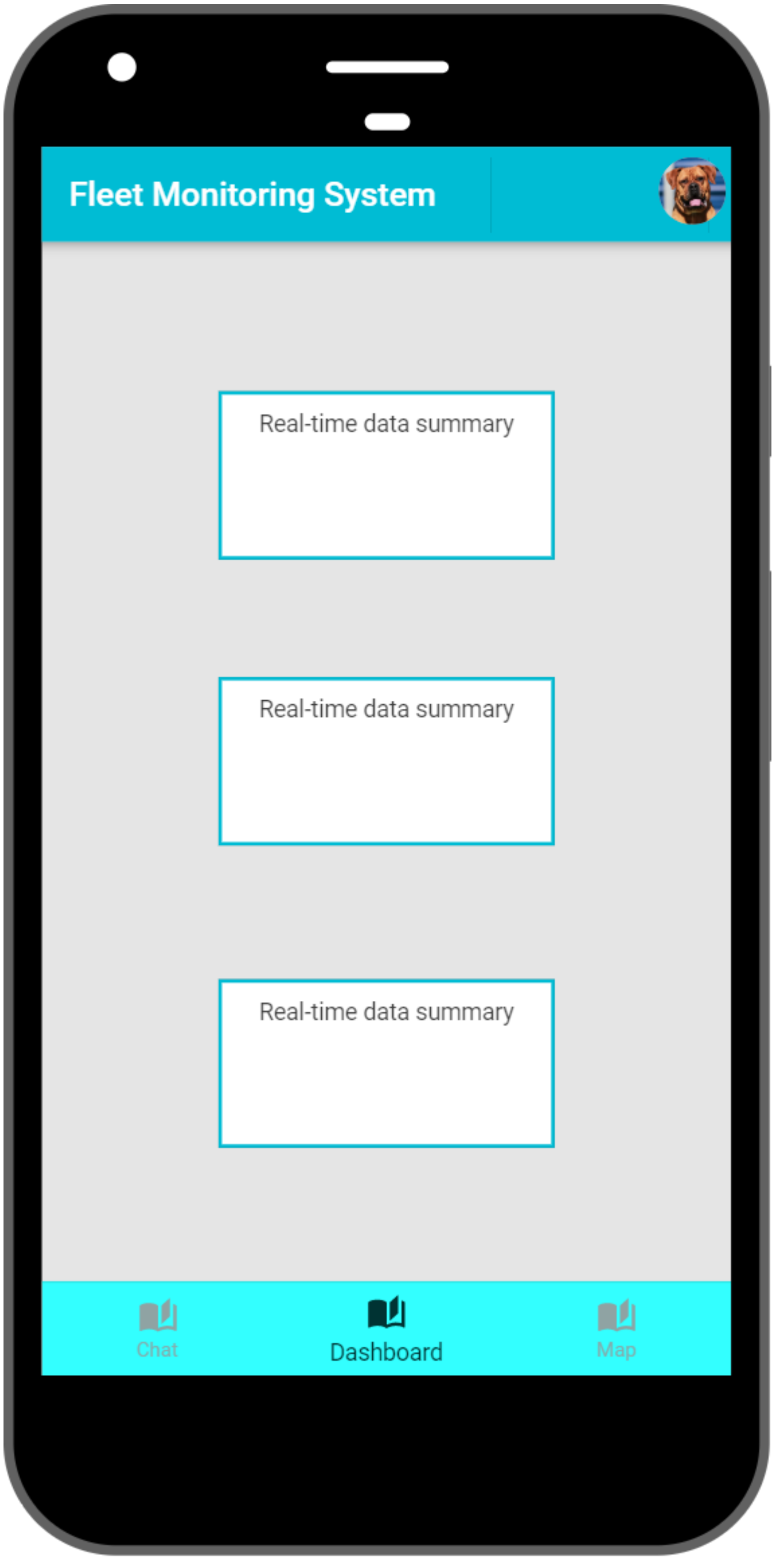
Member	Id	Role
Name1	#####	Manager
Name2	#####	Employee
Name3	#####	Employee
Name4	#####	Employee
Name5	#####	Employee
Name6	#####	Employee

Group Page

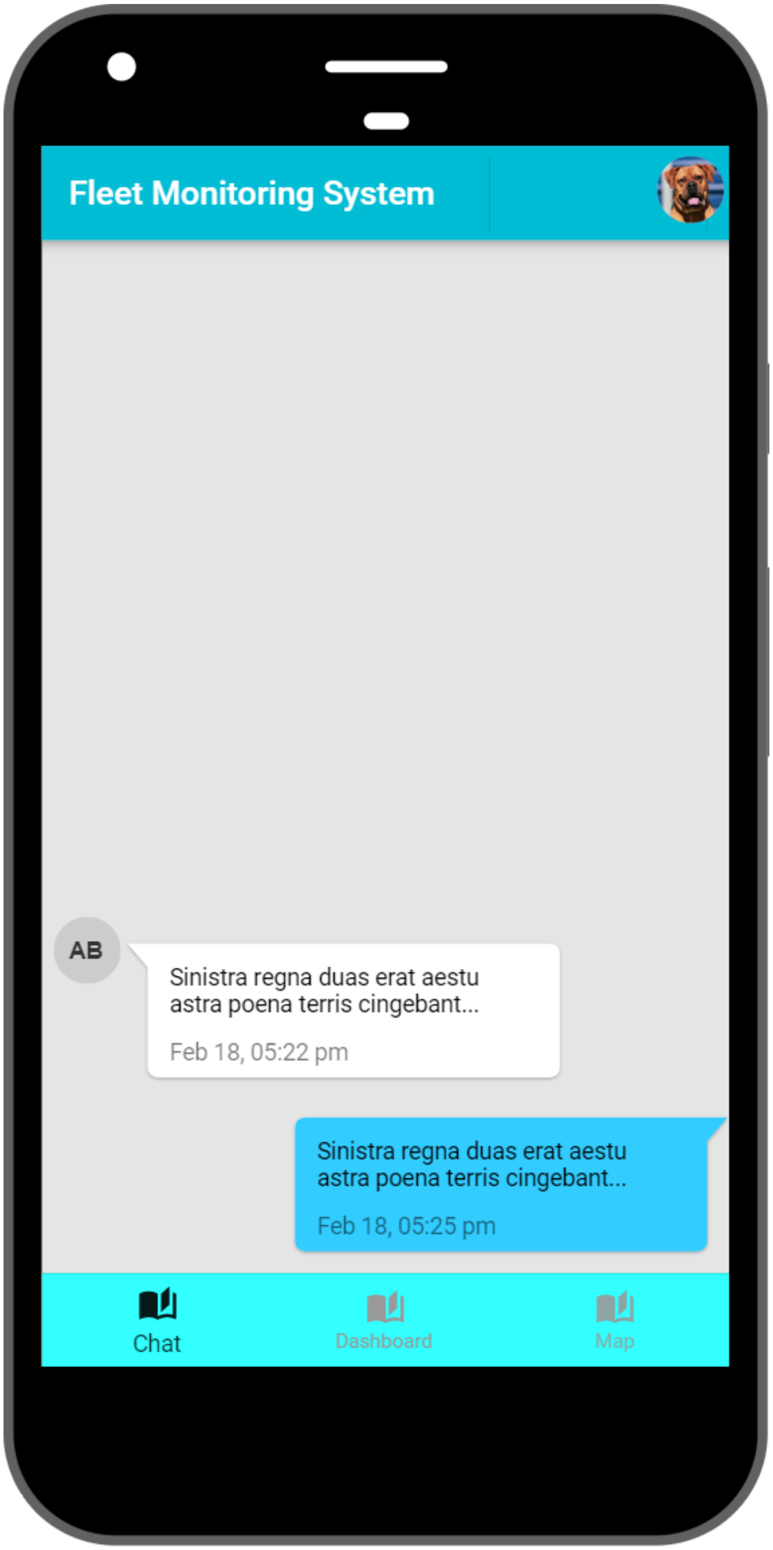
Mobile Application Design:



Login View



Dashboard View



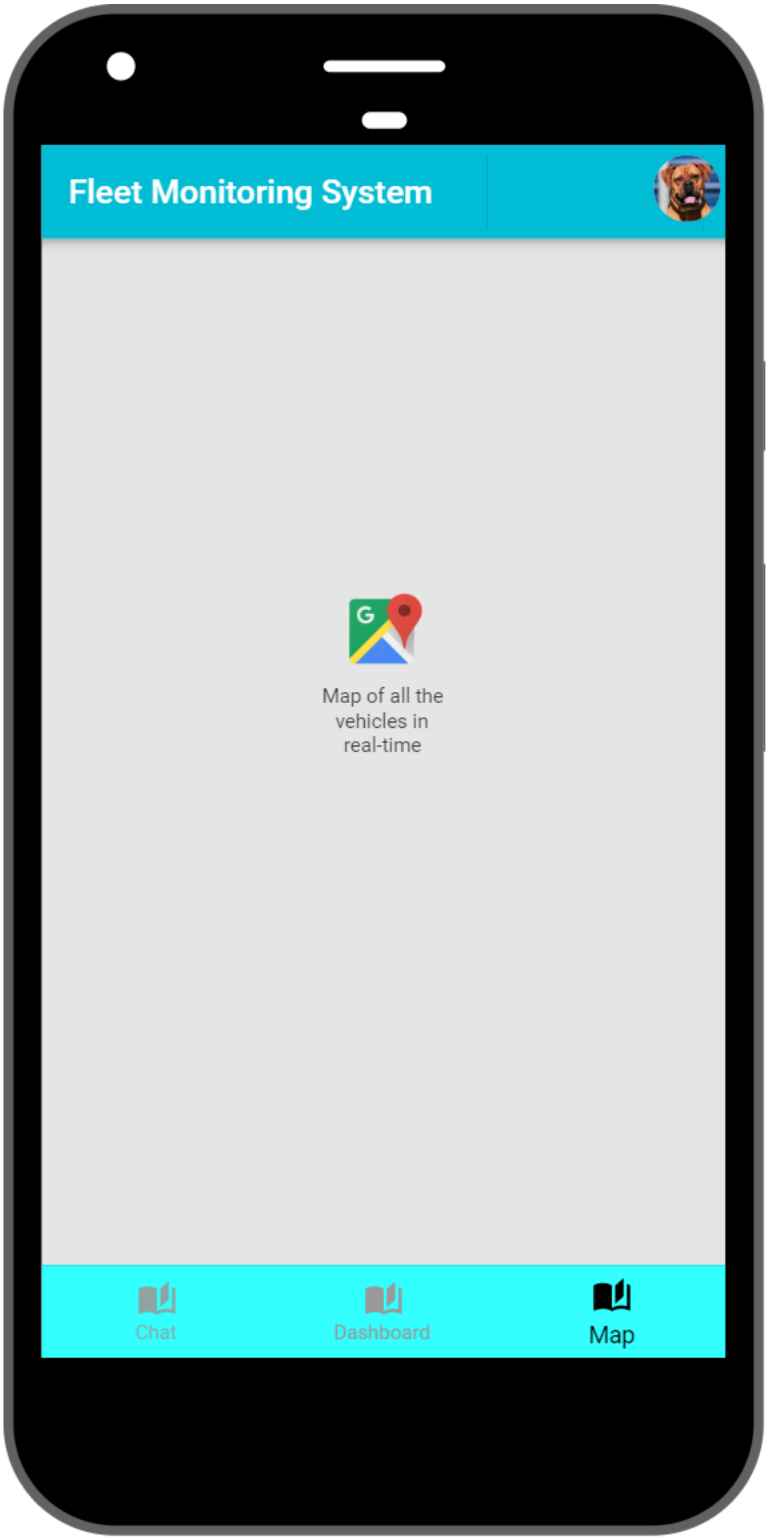
Sinistra regna duas erat aestu astra poena terris cingebant...

Feb 18, 05:22 pm

Sinistra regna duas erat aestu astra poena terris cingebant...

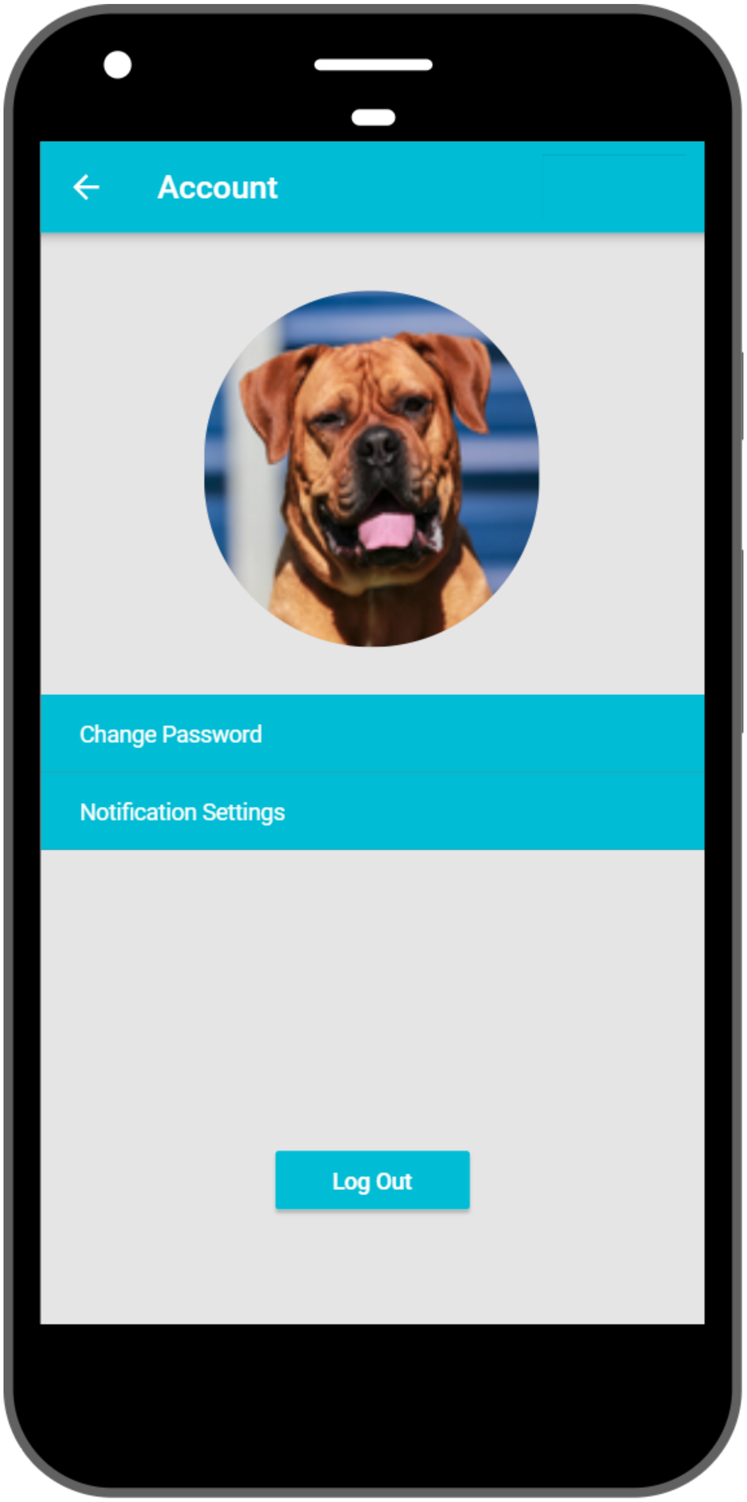
Feb 18, 05:25 pm

Chat View



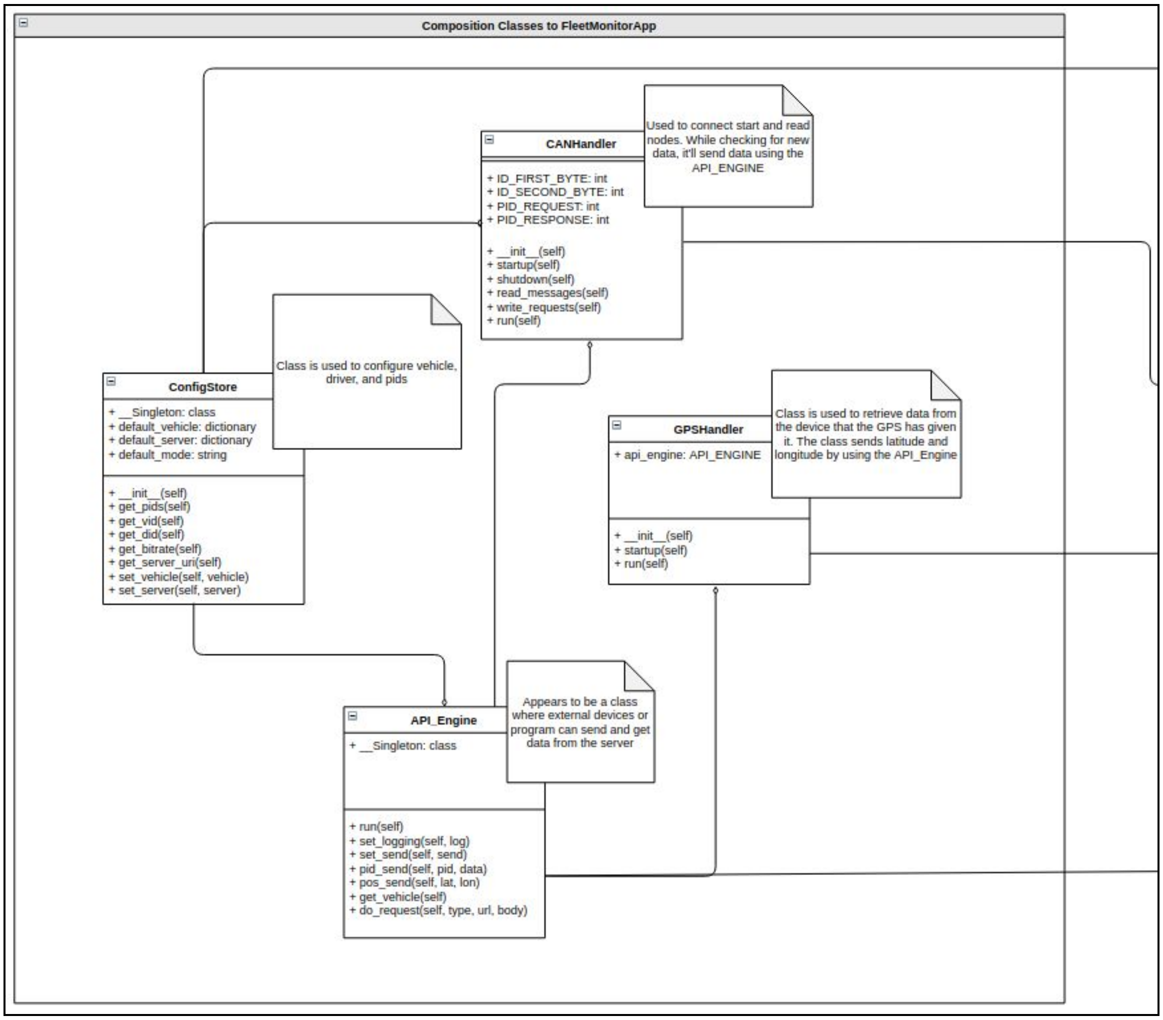
Fleet Monitoring System
Map of all the vehicles in real-time
Chat
Dashboard
Map

Map View

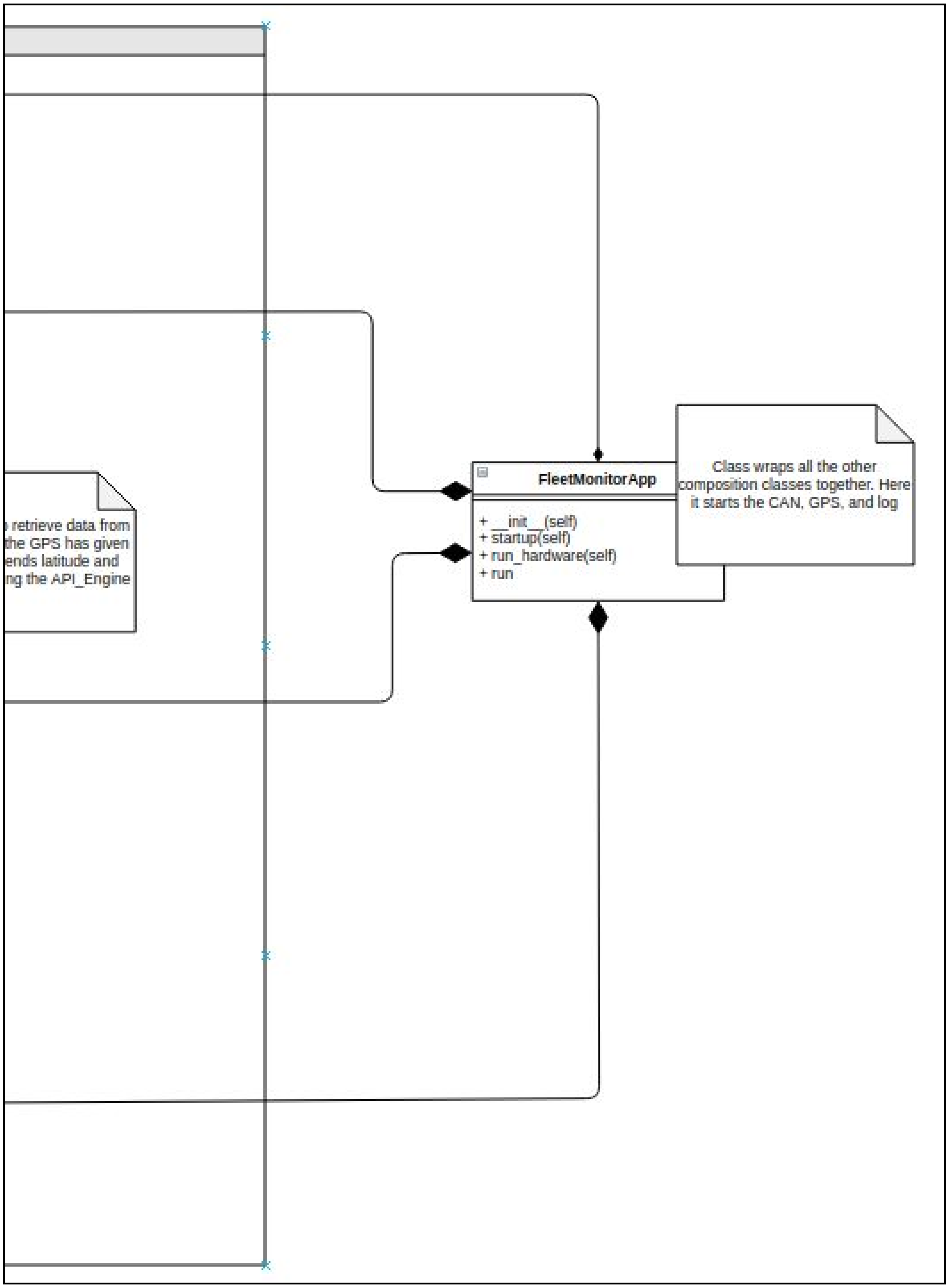


Profile Settings View

Embedded System Class Diagram:


Composition Classes to FleetMonitorApp
CANHandler
+ ID_FIRST_BYTE: int
+ ID_SECOND_BYTE: int
+ PID_REQUEST: int
+ PID_RESPONSE: int
+ __init__(self)
+ startup(self)
+ shutdown(self)
+ read_messages(self)
+ write_requests(self)
+ run(self)
Used to connect start and read nodes. While checking for new data, it'll send data using the API_ENGINE
ConfigStore
+ __Singleton: class
+ default_vehicle: dictionary
+ default_server: dictionary
+ default_mode: string
+ __init__(self)
+ get_pids(self)
+ get_vid(self)
+ get_did(self)
+ get_bitrate(self)
+ get_server_uri(self)
+ set_vehicle(self, vehicle)
+ set_server(self, server)
Class is used to configure vehicle, driver, and pids
GPSHandler
+ api_engine: API_ENGINE
+ __init__(self)
+ startup(self)
+ run(self)
Class is used to retrieve data from the device that the GPS has given it. The class sends latitude and longitude by using the API_Engine
API_Engine
+ __Singleton: class
+ run(self)
+ set_logging(self, log)
+ set_send(self, send)
+ pid_send(self, pid, data)
+ pos_send(self, lat, lon)
+ get_vehicle(self)
+ do_request(self, type, url, body)
Appears to be a class where external devices or program can send and get data from the server

Left-side of diagram (next image is right-side)



Right-side of diagram (previous image is left-side)

3. Statement of Work

3.1 Previous Work And Literature

Include relevant background/literature review for the project

– If similar products exist in the market, describe what has already been done

– If you are following previous work, cite that and discuss the **advantages/shortcomings**

– Note that while you are not expected to "compete" with other existing products / research groups, you should be able to differentiate your project from what is available

Detail any similar products or research done on this topic previously. Please cite your sources and include them in your references. All figures must be captioned and referenced in your text.

3.2 Technology Considerations

Highlight the strengths, weakness, and trade-offs made in technology available.

Discuss possible solutions and design alternatives

3.3 Task Decomposition

In order to solve the problem at hand, it helps to decompose it into multiple tasks and to understand interdependence among tasks.

3.4 Possible Risks And Risk Management

Include any concerns or details that may slow or hinder your plan as it is now. These may include anything to do with costs, materials, equipment, knowledge of area, accuracy issues, etc.

3.5 Project Proposed Milestones and Evaluation Criteria

What are some key milestones in your proposed project? Consider developing task-wise milestones. What tests will your group perform to confirm it works?

3.6 Project Tracking Procedures

What will your group use to track progress throughout the course of this and next semester?

3.7 Expected Results and Validation

What is the desired outcome?

How will you confirm that your solutions work at a **High level**?

4. Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline

• A realistic, well-planned schedule is an essential component of every well-planned project

• Most scheduling errors occur as the result of either not properly identifying all of the necessary activities (tasks and/or subtasks) or not properly estimating the amount of effort required to correctly complete the activity

• A detailed schedule is needed as a part of the plan:

– Start with a Gantt chart showing the tasks (that you developed in 3.3) and associated subtasks versus the proposed project calendar. The Gantt chart shall be referenced and summarized in the text.

– Annotate the Gantt chart with when each project deliverable will be delivered

• Completely compatible with an Agile development cycle if that's your thing

How would you plan for the project to be completed in two semesters? Represent with appropriate charts and tables or other means.

Make sure to include at least a couple paragraphs discussing the timeline and why it is being proposed. Include details that distinguish between design details for present project version and later stages of project.

4.2 Feasibility Assessment

Realistic projection of what the project will be. State foreseen challenges of the project.

4.3 Personnel Effort Requirements

Include a detailed estimate in the form of a table accompanied by a textual reference and explanation. This estimate shall be done on a task-by-task basis and should be based on the projected effort required to perform the task correctly and not just “X” hours per week for the number of weeks that the task is active

4.4 Other Resource Requirements

Identify the other resources aside from financial, such as parts and materials that are required to conduct the project.

4.5 Financial Requirements

If relevant, include the total financial resources required to conduct the project.

5. Testing and Implementation

Testing is an **extremely** important component of most projects, whether it involves a circuit, a process, or a software library

Although the tooling is usually significantly different, the testing process is typically quite similar regardless of CprE, EE, or SE themed project:

1. Define the needed types of tests (unit testing for modules, integrity testing for interfaces, user-study for functional and non-functional requirements)
2. Define the individual items to be tested
3. Define, design, and develop the actual test cases
4. Determine the anticipated test results for each test case 5. Perform the actual tests
6. Evaluate the actual test results
7. Make the necessary changes to the product being tested 8. Perform any necessary retesting
9. Document the entire testing process and its results

Include Functional and Non-Functional Testing, Modeling and Simulations, challenges you’ve determined.

5.1 Interface Specifications

– Discuss any hardware/software interfacing that you are working on for testing your project

5.2 Hardware and software

– Indicate any hardware and/or software used in the testing phase

– Provide brief, simple introductions for each to explain the usefulness of each

5.3 Functional Testing

Examples include unit, integration, system, acceptance testing

5.4 Non-Functional Testing

Testing for performance, security, usability, compatibility

5.5 Process

– Explain how each method indicated in Section 2 was tested

– Flow diagram of the process if applicable (should be for most projects)

5.6 Results

– List and explain any and all results obtained so far during the testing phase

- – Include failures and successes
- – Explain what you learned and how you are planning to change it as you progress with your project
- – If you are including figures, please include captions and cite it in the text

• This part will likely need to be refined in your 492 semester where the majority of the implementation and testing work will take place

-**Modeling and Simulation**: This could be logic analyzation, waveform outputs, block testing. 3D model renders, modeling graphs.

-List the **implementation Issues and Challenges**.

6. Closing Material

6.1 Conclusion

Summarize the work you have done so far. Briefly re-iterate your goals. Then, re-iterate the best plan of action (or solution) to achieving your goals and indicate why this surpasses all other possible solutions tested.

6.2 References

This will likely be different than in project plan, since these will be technical references versus related work / market survey references. Do professional citation style(ex. IEEE).

6.3 Appendices

Any additional information that would be helpful to the evaluation of your design document.

If you have any large graphs, tables, or similar that does not directly pertain to the problem but helps support it, include that here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc. PCB testing issues etc. Software bugs etc.