Design Document

# 1. Introduction

The project is to create a driver monitor system that monitors your driving performance per journey. The systems uses a raspberry pi to connect to a OBDII dongle which pull all the data .Once the journey has concluded a report is generated and available to view on your smart phone. The system should store every report generated in the last month.

This design document presents the designs used or intended to be used in implementing the project. The designs described, follow the requirements specified in the Project Requirements Specification document given to us at the start of semester 2.

## Purpose

The purpose of this document is to present a detailed description of the designs of the driver monitor system , created for the Professional Practice in IT module project. Firstly, this document is intended for the team, to use the designs as guidelines to implement the project. Lastly, this document is also one of the project requirements.

# 2. System Requirements

The system requirements for this system are as follows

* A Raspberry Pi running Raspbian and a laptop with either Intellij Idea or Eclipse EE to run the Spring application
* Either a Android or Apple device to run the Ionic Application
* An OBDII Dongle

# 3. Technology Used and Why

The application running on the Raspberry Pi will be implemented in Spring boot (Java). The reason we have decided to program this in Spring Boot is because Spring has a great way of connecting to a database. In our case we are using MongoDB and both work very well together. Our Spring application connects to a car by a OBDII dongle. The connection is done by the COM ports between P.C and OBDII dongle.

On the server side of things we will be running MongoDB to store the data from the Spring Boot application. MongoDB stores data as documents in a binary representation called BSON (Binary JSON). This makes life a lot easier when creating an API as the API needs to be in JSON format. For creating the API we will be using Node.js. Node will be using Mongoose to connect to the MongoDB, Express to create the sever, Body Parser which parses the data from the MongoDB to JSON format. Lastly Cors will be used to assign the appropriate headers so applications can make requests to the API. The reasons for using Node.js is because it is fast, NPM is the Node.js packet manager and it is really good to use. It does a great job specifying and installing project dependencies, but also obscures a great deal of complexities. All this technology will be hosted on a Digital Ocean Droplet. Digital Ocean make it easy to set up a server which already has MongoDB pre-installed. You can create a student account which gives you $50 free credit.

For the mobile application side we have decided to go with Ionic. Ionic is an HTML5 mobile app development framework targeted at building hybrid mobile apps. Hybrid apps are essentially small websites running in a browser shell in an app that have access to the native platform layer. Reasons for using Ionic are that we wont have to develop different mobile applications to run on different mobile operating systems. We can just develop the one Ionic app and this will up on all mobile operating systems.

# 4. Architecture of the Solution

Architecture of the solution goes here

# 5. Design Methodology

Design methodology goes here

# 6. Features of the Implementation

Features of the implementation goes here

# 7. Limitations

Limitations goes here

# 8. Known Bugs

Known bugs goes here

# 9. Recommendations for Future Development

Recommendations for future development goes here

# 10. Conclusions

Conclusions goes here