A PROJECT REPORT
ON

"SMART CAR CO-PILOT"
(SCCOP)

Submitted to

GERMAN UNIVERSITY IN CAIRO
DEPARTMENT OF COMPUTER ENGINEERING

In Partial Fulfilment of the Requirement for

INTERNET OF THINGS
ELECTIVE COURSE

BY

MAHMOUD ABDRABO	**19-4032**
YASMINE MAXIMOS	**19-1600**
MOHAMED ABDELMEGEED	
AHMED RADY	**19-1770**
AHMED WAFIK	

UNDER THE GUIDANCE OF
PROF. AMR TALAAT

Contents

Chapter 1

Introduction

1.1 Motive

We live in a busy world where everyone lives a hectic life. Luckily technology played a huge role in making our life more comfortable and easier. But how often does a person get his car checked? Is there a way to make him foresee car troubles?

1.2 Aim and Description

The goal of our project is building an open source OBD2 performance and diagnostic tool that reads engine data (RPM, Speed, Temperature, Throttle, fuel level and Engine Load) from the users vehicle and sends it to a Mobile App via Bluetooth so the user can view live OBD engine data on his Android phone. Also the data will be forwarded from the mobile device to a Web App (Database), where it is stored along with the location retrieved from the GPS of the mobile device. Allowing the user to keep track of the logs of his vehicle along with the location from where it was sent.

Chapter 2

Technical Specification

2.1 Hardware

2.1.1 Arduino Board

The Arduino Uno board was used, it was best choice for the prototype, hence the relatively small size, more powerful than needed, easily programmed and almost unlimited number of compatible stackable shields.

2.1.2 Bluetooth Arduino Shield

A Bluetooth Shield that is compatible with the Arduino board, used for the communication between the arduino and the Android mobile phone/tablet.

2.1.3 CAN-BUS Arduino Shield

A CAN-Bus Shield that is compatible with the Arduino Uno board, used for sending signals to the car's computer to receive the value of the desired parameter.
This shield was not used in the final prototype, because we had trouble receiving the correct signal values from the car, since every car manufacturer have their own standards, we couldn't make it work, but it's a high priority in future work.

2.1.4 Android Operated Device

To install the Android app. an android phone/tablet with bluetooth and internet connectivityis needed, that's about all the devices.

2.2 Software

2.2.1 Arduino Sketch

The code for the micro-controller the arduino is very simple, there is not much processing on it, only initialize the bluetooth connection, run the data simulating loop, and once connected to the android phone, it starts sending those values.

2.2.2 Android Application

The Android application has a very simple, intuitive interface, with only one button, that starts the bluetooth connection with the arduino, starts receiving the simulating data, live updates the screen, with green, yellow and red colors for each parameter to alert the user if anything needs attention, and finally uploads those acquired data to the web app, to be saved in the cloud, for future reference and better view. The app. also uploads location updates from the mobile's GPS or Internet connection (if there's no GPS).

2.2.3 Django Web Application

For the web application, the Django Framework was used. A simple wep app. that provided a couple of APIs to upload status and location data to the online database, additionally it serves as monitor for the newly uploaded data, with more information on display, and a google map that shows the latest location the car uploaded from, the owner can also view all previously logged data (car status and location). The web app. is hosted online on Heroku servers http://sccop.herokuapp.com

2.2.4 Source Code

The whole code is hosted on Github, available for any one to use/improve, code can be found here

Figure 2.1: `https://github.com/mabdrabo/Arduino-SCCOP`

Figure 2.2: `https://github.com/mabdrabo/Android-SCCOP`

Figure 2.3: `https://github.com/mabdrabo/Django-SCCOP`

Chapter 3

Screenshots of Project

3.1 Web Application

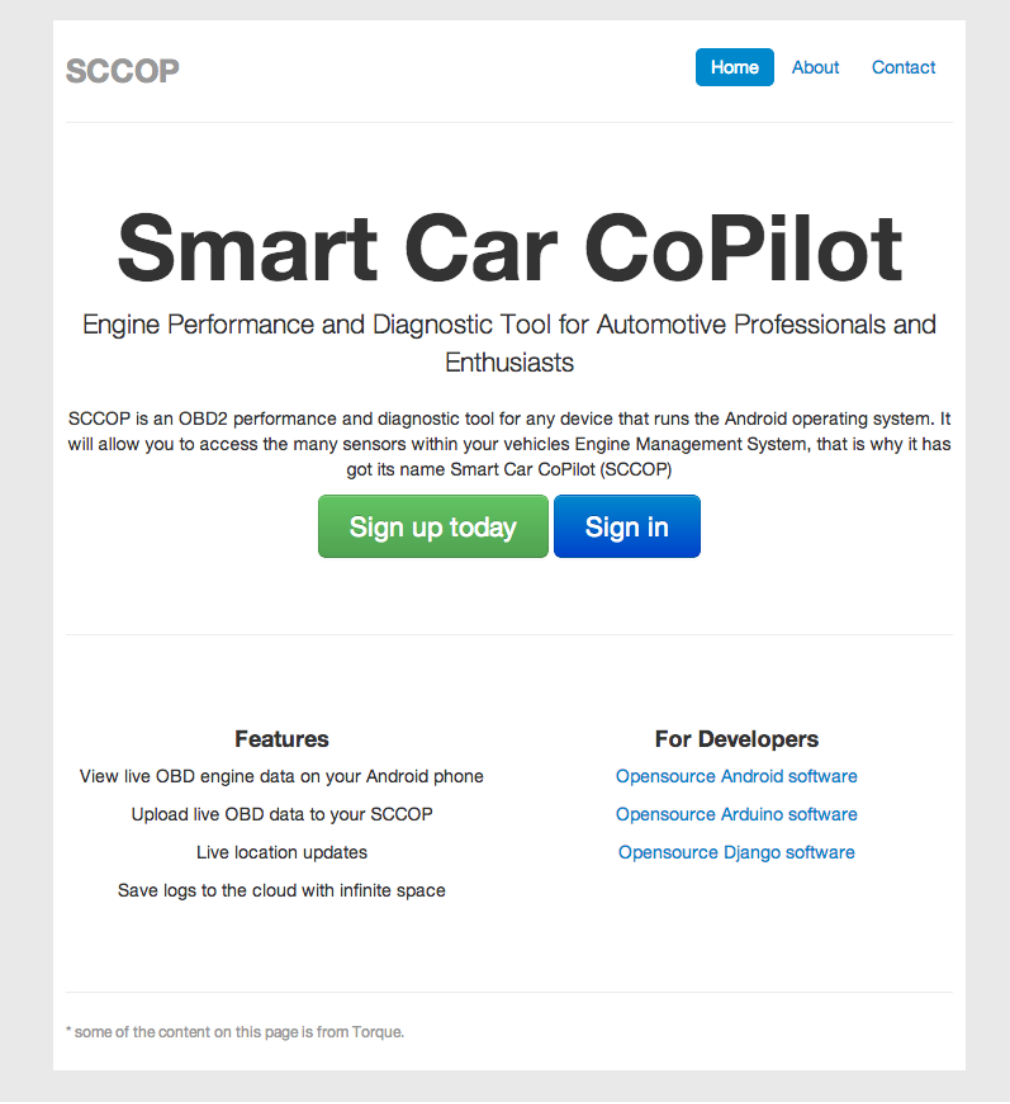

Figure 3.1: SCCOP home page

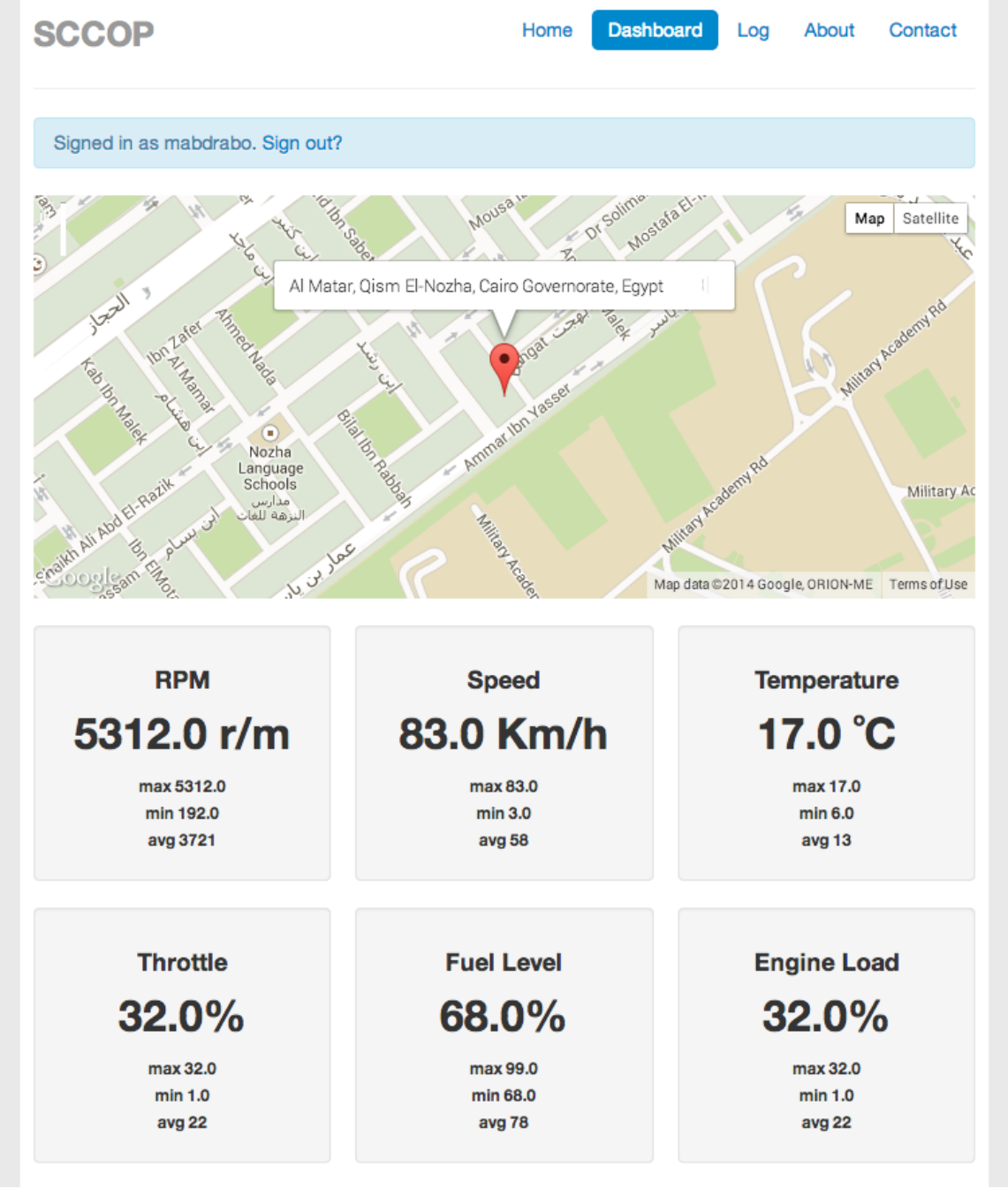

Figure 3.2: user dashboard with last uploaded data and last reported location

15-01-2014 8:58:38 p.m.	1472.0	23.0	17.0	9.0	91.0	9.0
15-01-2014 8:58:37 p.m.	1408.0	22.0	16.0	8.0	92.0	8.0
15-01-2014 8:58:36 p.m.	1408.0	22.0	16.0	8.0	92.0	8.0
15-01-2014 8:58:35 p.m.	1344.0	21.0	15.0	8.0	92.0	8.0
15-01-2014 8:58:33 p.m.	1216.0	19.0	13.0	7.0	93.0	7.0
15-01-2014 8:58:33 p.m.	1152.0	18.0	12.0	7.0	93.0	7.0
15-01-2014 8:58:32 p.m.	1152.0	18.0	12.0	7.0	93.0	7.0
15-01-2014 8:58:31 p.m.	1024.0	16.0	10.0	6.0	94.0	6.0
15-01-2014 8:58:30 p.m.	1024.0	16.0	10.0	6.0	94.0	6.0
15-01-2014 8:58:29 p.m.	960.0	15.0	9.0	5.0	95.0	5.0
15-01-2014 8:58:27 p.m.	832.0	13.0	7.0	5.0	95.0	5.0
15-01-2014 8:58:27 p.m.	768.0	12.0	6.0	4.0	96.0	4.0
15-01-2014 8:58:24 p.m.	640.0	10.0	16.0	3.0	97.0	3.0
15-01-2014 8:58:24 p.m.	576.0	9.0	15.0	3.0	97.0	3.0
15-01-2014 8:58:23 p.m.	512.0	8.0	14.0	3.0	97.0	3.0
15-01-2014 8:58:21 p.m.	448.0	7.0	13.0	2.0	98.0	2.0
15-01-2014 8:58:20 p.m.	384.0	6.0	12.0	2.0	98.0	2.0
15-01-2014 8:58:19 p.m.	320.0	5.0	11.0	1.0	99.0	1.0
MAX	6784.0	106.0	17.0	41.0	99.0	41.0
MIN	320.0	5.0	6.0	1.0	59.0	1.0
AVG	3494	55	12	21	79	21
	RPM	Speed	Temperature	Throttle	Fuel Level	Engine Load

Figure 3.3: saved log of uploaded car state

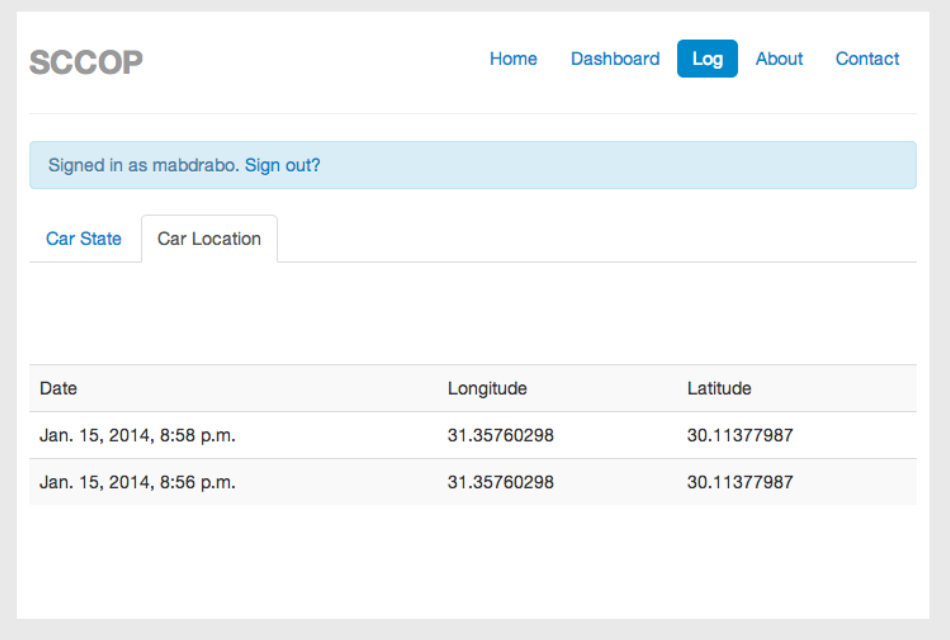

Figure 3.4: saved log of uploaded locations

3.2 Android Application

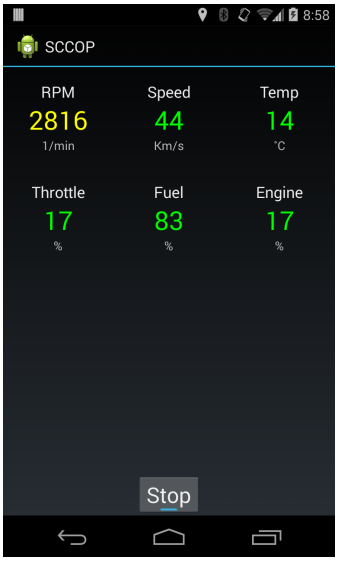

Figure 3.5: showing a yellow alert

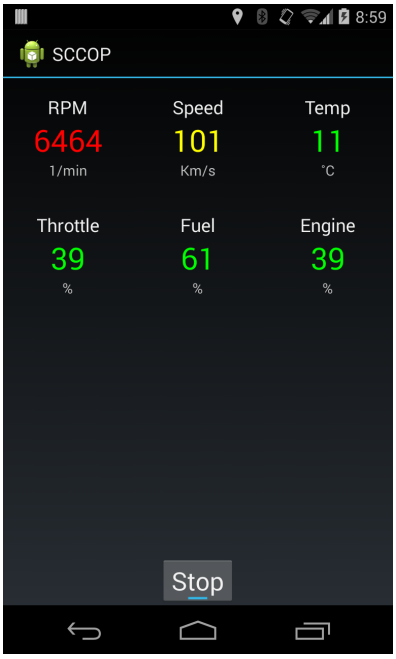

Figure 3.6: showing red and yellow alerts

Chapter 4

Conclusion and Future Scope

4.1 Conclusion

All test cases were a success, only minor deficiencies mostly because of the latency accompanied with sending HTTP requests to a free of charge online server, and not full exception handling for Bluetooth connection.

The idea along with the prototype were accepted by most of the audience, some thought it was just a replica until they knew the estimated cost, plus being all open-source (both hardware and software) giving unlimited access for other developers to improve it in their own way.

4.2 Future Scope

- Being able to retrieve the data from the vehicle using the Can-bus shield instead of simulating the data.
- Enhancing the Mobile App and Web App.
- Retrieving more data from the vehicle (depending on the car model).
- Populate graphs from the saved data, to monitor performance over time.