THUNDER DRONE

Project: Reading and Sending data from sensors on a Phantom Drone to a Mobile Phone Application

Team number: 5

Client name(s):

Dean Mohamedally	(affiliated with UCL)
Peli de Halleux	(affiliated with Microsoft)

Team members:

Garrett May

Omer Mohammed Mirza

Iacovos Pantekhis

1. Contents

2. Abstract

The proposal set for our project is thus: to be able to use a drone of sorts, and have some method of taking sensors on board it, transmitting the data over a connection, and sending it to a mobile application for the displaying of data in a convenient form. Preferably, the data should be sent over Wi-Fi, and, importantly, the mobile application should be coded in a language known as TouchDevelop. Using this proposal, we (the team) should endeavour to come as close as possible to the requirements set. The intended application for this proposal is to be able to use the drone as a way of extracting information whilst in stormy weather.

Project Video: https://www.youtube.com/watch?v=NAdk99IfQFA

3. Overview

The mobile phone application (app) displays the data of the sensors on the Phantom Drone sent to it and displays the data via, for example, a graph.

The app is an imaginative experiment for Microsoft, intended to show the uses and applications drones can have to benefit society.

To deliver this project, we will place an Engduino and a Raspberry Pi 2 on the Phantom Drone. Sensors will be connected to the Engduino, allowing the Engduino to read data from it. This data will then be transferred to the Raspberry Pi, which in turn will send the data to the mobile phone via Wi-Fi, by using a Wi-Fi Adaptor. The app will review the data, and then display it in the form of, for example, a graph. Depending on how much data is sent, there may be methods to switch to different reviews of data.

4. Context

Microsoft have asked us to use a Phantom Drone and sensors placed upon it to show a possible use for it. Their main aim is to use a new programming language, TouchDevelop, as the platform for making the app. TouchDevelop is a new experimental language, with an aim to be relatively simple to code in and 'touch-friendly', allowing even children to hopefully grasp the concept of programming in a fun, interactive way. Peli de Halleux is affiliated with Microsoft and TouchDevelop, and is providing us with help in its implementation. Dean Mohamedally is our mediator, who communicates between Peli de Halleux and us, and has briefed us on our project.

The project is not truly based upon a specific problem and thus a solution to solve it; rather, the project is based around a specific study - drones - and the hypothesised ability to help and assist us in more ways than we can imagine. It is an open-ended project, which allows us to choose any possible problem we like, and prove how using drones, and our app, a solution can be created.

Our choice for the project is to take advantage of the drone and see whether it can provide help in detecting storms, hurricanes, tornadoes, and the sort. If the project successfully works, it will be of great benefit to many people, as these hazardous natural weather might be able to be foretold much earlier in advance than before, allowing more time for preparation of people, buildings and structures, animals, and the environment.

5. Team Skills and Roles

5.1. Garrett May

- StrengthsFinder Skills:
  - Context
  - Deliberative
  - Belief
  - Consistency
  - Relator
- Roles:
  - Team Leader and Project Manager
  - Technology Platform Research Lead
  - Lead UI Designer and User Testing

5.2. Omer Mohammed Mirza

- StrengthsFinder Skills: N/A
- Roles:
  - Lead Device/Platform Tester
  - Programming and Repository Lead

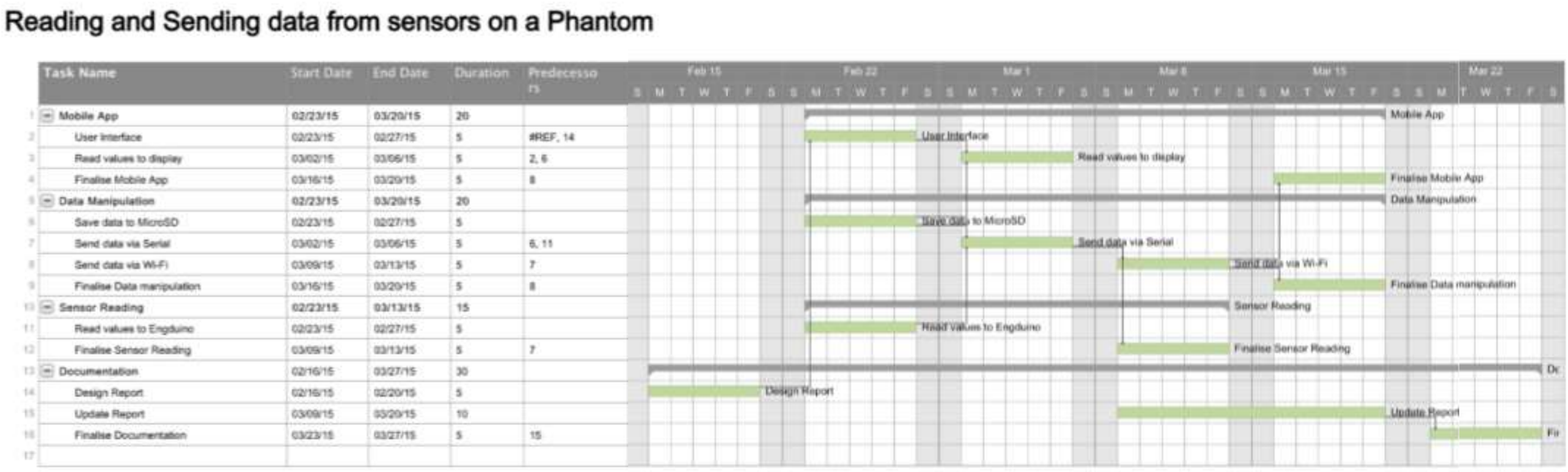
5.3. Iacovos Pantekhis

- StrengthsFinder Skills:
  - Self-Assurance
  - Includer
  - Achiever
  - Adaptability
  - Responsibility
- Roles:
  - Documentation Lead
  - Lead Interviewer for Requirements

6. Work Plan

To better organise our time, a Gannt Chart had been opted to be used.

However, later on in the process, we found out that the deadline for the project had been extended. This meant that there was extra days to allocate work. We agreed that we should aim to follow the Gannt chart as precisely as possible, and any extra time should be delegated to improving the application, testing it on multiple devices to ensure compatibility, and writing the documentation.



Reading and Sending data from sensors on a Phantom

	Task Name	Start Date	End Date	Duration	Predecessors
1	⊟ Mobile App	02/23/15	03/20/15	20	
2	User Interface	02/23/15	02/27/15	5	#REF, 14
3	Read values to display	03/02/15	03/06/15	5	2, 6
4	Finalise Mobile App	03/16/15	03/20/15	5	8
5	⊟ Data Manipulation	02/23/15	03/20/15	20	
6	Save data to MicroSD	02/23/15	02/27/15	5	
7	Send data via Serial	03/02/15	03/06/15	5	6, 11
8	Send data via Wi-Fi	03/09/15	03/13/15	5	7
9	Finalise Data manipulation	03/16/15	03/20/15	5	8
10	⊟ Sensor Reading	02/23/15	03/13/15	15	
11	Read values to Engduino	02/23/15	02/27/15	5	
12	Finalise Sensor Reading	03/09/15	03/13/15	5	7
13	⊟ Documentation	02/16/15	03/27/15	30	
14	Design Report	02/16/15	02/20/15	5	
15	Update Report	03/09/15	03/20/15	10	
16	Finalise Documentation	03/23/15	03/27/15	5	15
17					

UCL
Object Oriented Programming COMP103P
Team 5 - Phantom Drone

7. Requirements:

7.1. Must have requirements:

- A mobile application coded in TouchDevelop with a suitable UI
    - Completed: A mobile application has been created, with a user interface that optimises the functionality of the program, as well as including design features, all coded in the TouchDevelop language
- A user interface (UI) that is able to display data
    - Completed: The user interface contains a line graph, the method opted for displaying data
- Save data via the MicroSD
    - Completed: The Engduino is able to save the data it receives from the sensors to a text file on a MicroSD it has on board

7.2. Should have requirements:

- Also send data via Wi-Fi
    - Completed: The data received by the Raspberry PI 2 from the Engduino is transmitted over a Wi-Fi network to a website. This is then requested by the TouchDevelop application
- Read data from the MicroSD or via Wi-Fi
    - Completed: The Engduino is able to read data from the MicroSD, and, with changes, will be able to send this data via Wi-Fi to the website (however, given the data is the same, there is no need at this current time). The mobile application would also be able to request the readings from the website.
- Also save on the mobile phone
    - Incomplete: The data is stored in the mobile phone as a Linked List. This will need to be saved in some memory on the phone, for example a text file

7.3. Could have requirements:

- Easily switch between MicroSD saving and Wi-Fi streaming
    - Discarded: There is no need to switch between saving data onto the MicroSD and streaming via Wi-Fi. Instead, we do both simultaneously
- Allow easy switch between different data
    - Completed: The application is able to switch between different pieces of data from a drop down button
- Different modes of displaying data
    - Incomplete: There is only one mode of displaying data, a line graph. Further programming may have led to multiple graphs on the page, or multiple data types on the same graph, or entirely different ways of displaying data
- Drone control
    - Incomplete: It was not necessary to control the drone. The drone had a remote control controller that allowed flight. Changing this would require breaking the drone down and analysing how it fully worked

7.4. Would have requirements:

- Compatibility with other devices
    - Completed: The application works on all types of mobile device, laptops, and computers, and possibly many others. This is due to the fact that the application is represented as a website, and supports different size screens

7.5. Glossary of Terms:

- Components: Important items to be used, with the intention of having a specific function
- Sensors: A sensor is a component which is able to produce a specific type of data, gathered from its surroundings (for example, temperature)
- Engduino: A component which controls what readings are taken, and when. It is able to manipulate these readings, if necessary, and save them via a MicroSD chip, or send them via the USB serial port
- Serial port: In this case, it refers to a USB (Universal Serial Bus). It will allow data transfer between the Engduino and the Raspberry Pi 2
- MicroSD: This is a component which is able to hold files which contain data. Effectively, this component allows us to save data locally.
- Raspberry PI 2: A component which will take any data received by the Engduino and send it to the mobile phone via Wi-Fi and using a Wi-Fi Adaptor
- Wi-Fi Adaptor: A component which is able to receive data through its USB and send data via Wi-Fi. Effectively, it allows the Raspberry PI 2 to send data via Wi-Fi
- Mobile Phone: A component which can receive data via Wi-Fi and start our app
- App: A mobile phone application (app) that will take data and display it in an aesthetic form

8. Design

8.1. Proposed Design

Our aim is to have a simple and efficient user interface (UI) which will provide the user with all the information they require in one display. There will be no need to move from one screen to the other.

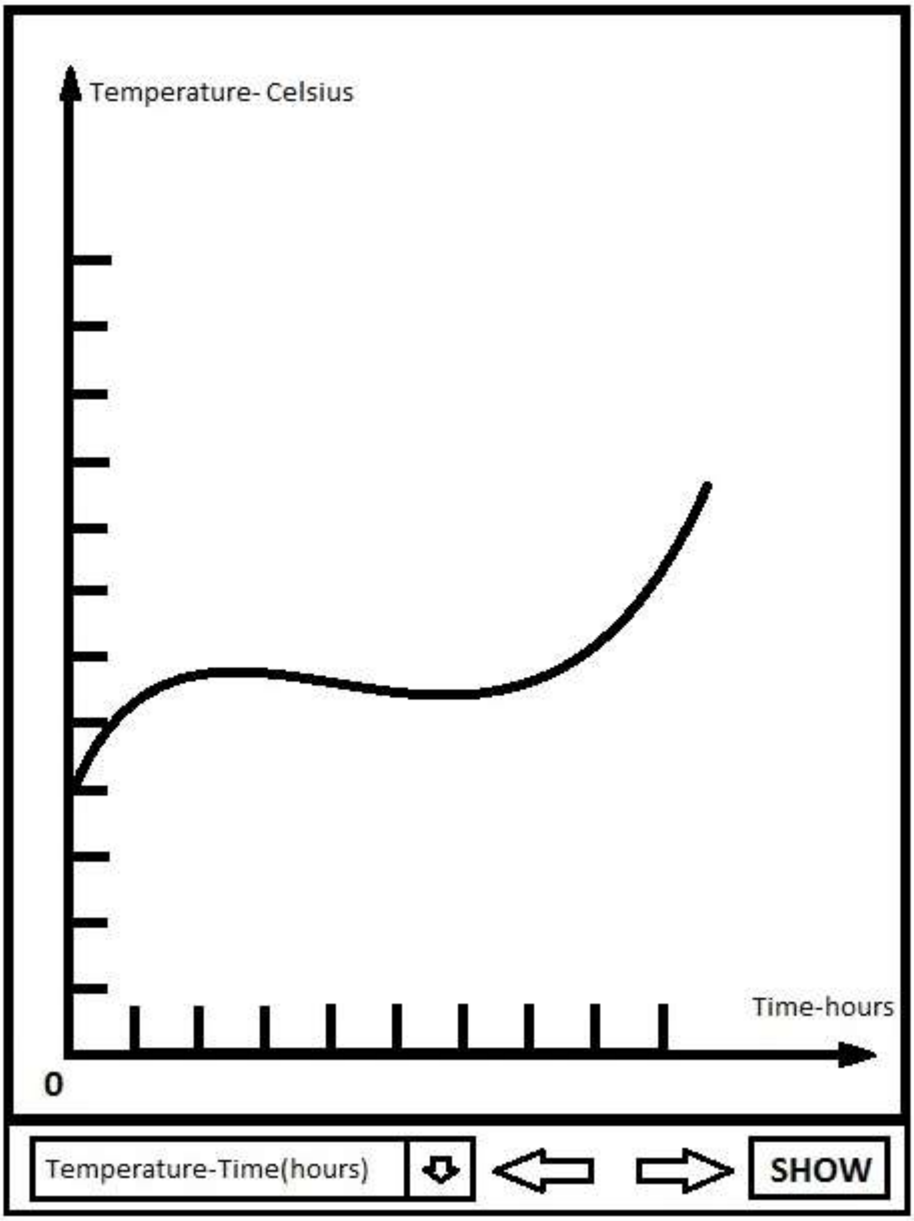


***Figure UI-1**: Mock-up of the user interface, which will be used in the application.*

As seen in the mock up above, the majority of the UI consists of a graph which displays the information that the user has requested. The user can change the variables displayed on the graph with the drop down menu, located beneath the graph and then pressing the "show" button. There is also an option of moving through the different graph options with the back and forward buttons. Our goal is to have a design which will keep everything neat and tidy by providing the desired result from the off. Moreover we want to avoid overloading the platform that the application is being run on, as well as the application itself, with unnecessary information and processing.

8.2. Design Patterns

8.2.1 The Singleton

The Singleton is a design pattern whereby you create only one object of a certain class, and you provide a method for getting an instance of it.

In our application, we create a single instance of the Web Socket class, an object called sock, in our Web Socket library; we create a new Web Socket if sock has not yet been initialised, and we return sock if it already has.

8.2.2 The Adaptor

The Adaptor design patterns involves placing an already defined class into a new one, allowing new methods and functions to be applied to it. In our project, we put the Web Socket class into a new library we made, called Web Socket library. We added several functions to it for the benefit of our application.

8.2.3 The Interpreter

The Interpreter design pattern produces a grammar for a given language in order to better interpret its meaning.

The Interpreter design pattern is present in the connection side of the project because the commutation code uses a domain specific language. It is a request-response based system with a request sent to the Raspberry PI and a response received.

8.2.4 The Null Object

The Null Object design pattern involves creating an object that is supposed to represent null (the lack of anything). This pattern is particularly useful in avoiding a possible NullPointerException.

In our situation, it is used to represent a Null Object in our Linked List library. This provides a better and safer way of making an object invalid so that the Linked List can function properly.

8.2.5 The Iterator

The Iterator is a design pattern for being able to access each element in a collection sequentially without exposing the representation of the elements. Since TouchDevelop does not have a version of implementation, the fundamental concepts of Iterator lie in certain functions in our Linked List library.

Most importantly, we have produced an object type Node which is able to take in any other object (represented as a Json Object), and the next Node. This allows us to make and use certain functions such as ‘has next’, which will return true if there something not equal to the Null Object. We also provide an ‘iterate through’ method, which will sequentially traverse through a collection, and is the primary method for going through the elements in our Linked List.

8.3. The Engduino

The Engduino program is coded in a C/C++ environment. Many of the functions that we used were essentially C code, but we used some C++ where it was necessary and the best method for the project.

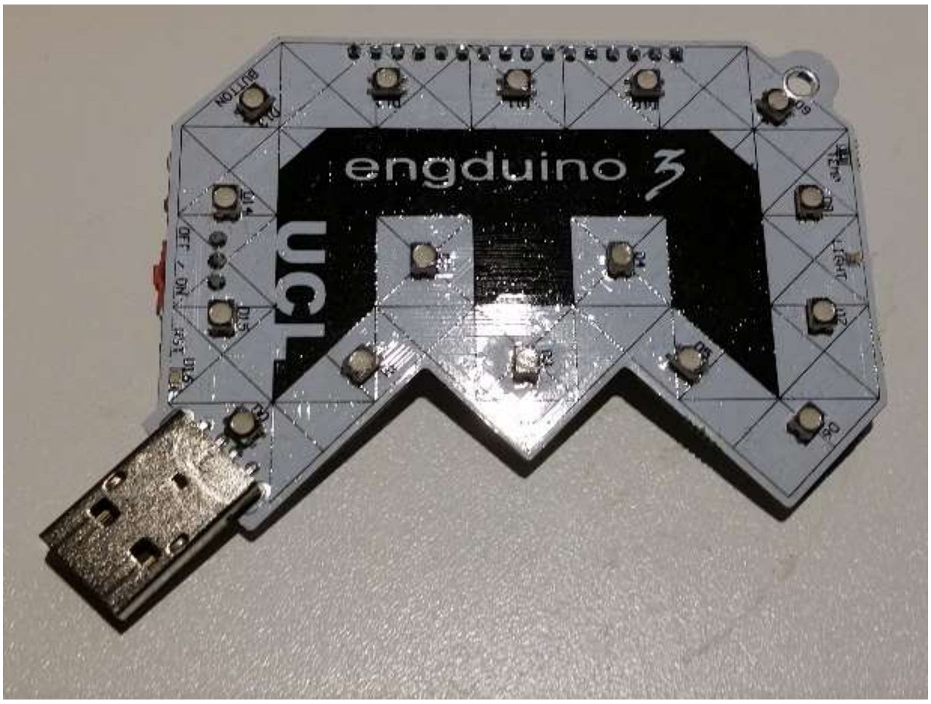


Figure Eng-1*. An Engduino, UCL's developed version of the Arduino. An Engduino Version 3 is displayed here.*

The Engduino has several functions. These functions are quite basic in the sense that they each do one single action. Namely, we made a read data function, a save data function, and a send data function. This outlines the basic requirements for the Engduino side of the project.

We also made a function to compress data. For example, it took in a String, say, "Temperature", and a number, say, "26.95", and returned the String "T26.95". This would help in sending over the minimal amount of information that we needed, as well as improve the parsing later on in the TouchDevelop application.

We then made helper functions that increased the abstraction level. Namely, we made a single function that took as parameters the name of a sensor, and its current piece of data: it compressed both, saved this new piece of data in the MicroSD, and then send the data to the PI via the serial port. It does this simply by using the basic functions described above.

In the end, the Engduino acted simply as a logger, by read sensors, saving the data, and then sending it over to somewhere else (in this case, the PI).

8.4. Connections between the Raspberry Pi 2 and the Azure Website

It was necessary for the Web Application to establish a secure two way communication with the Engduino on the drone. This allows for the sensors to be read and commands to be sent to configure sensors for example. This was achieved by the use of a secure Web Socket connection. There are three components in the set up; a Raspberry PI 2 (PI) connected to an Engduino situated on a drone, a web server in the cloud, and the Web App for the end user.

The central component of the set-up is the web server in the cloud which acts as an intermediary relay agent between the Web App and the Raspberry PI. Both the PI and the client connect to this server via a secure Web Socket to send requests and receive responses. The server also supports multiple users. For example the app can send a “T” request to the Azure cloud server which then forwards it to the PI. The PI sends the latest temperature reading from the Engduino sensors back to the cloud server which then forwards it to the client that made the request. The server does not need to do any parsing of the language and thus does not need be changed if more commands are added in the future.

Figure PI-1: *A Raspberry Pi 2, with the USB Wi-Fi module attached on.*

The advantage of this method is that only an internet connection is required by the client and the drone to communicate. This is not limited by Wi-Fi range on the drone as 3G Dongle can be used and the readings be read from the Web App in a remote location.

The code is written in JavaScript and executed by a Node.js web server. It is all event based. The script running on the Raspberry PI keeps an array of the latest readings which is updated by a serial event that fires each time data is received from the Engduino. When a request for some data is received from the cloud it must be parsed and the appropriate item in the array be sent. Other commands can be added as long as there is some code to recognise it and carry out the appropriate action. The system is therefore easily extendable.

The cloud server code is also simple. The Raspberry PI must send a special key to tell the server it has connected. The server stores this connection and will forward any requests it receives from the clients to it. When an event signals a response from the PI, it is forwarded to the client.

In the case of multiple clients each request is forwarded to the Raspberry PI, and the connection of the source is then pushed into queue. When an event signals a response from the PI, a connection from the queue is popped and then the response forwarded to that connection.

8.5. User Interface

As a result after getting some work done, and completing the first steps of our application in TouchDevelop, we achieved a functional User Interface which other than having a basic graph, which is the main aspect of the design, was far off from what we set off to achieve as far as aesthesis and ease of use are concerned. We had implemented a variant of the design shown in Figure UI-1, which left us bemoaning the fact that we didn’t put sufficient work into it.

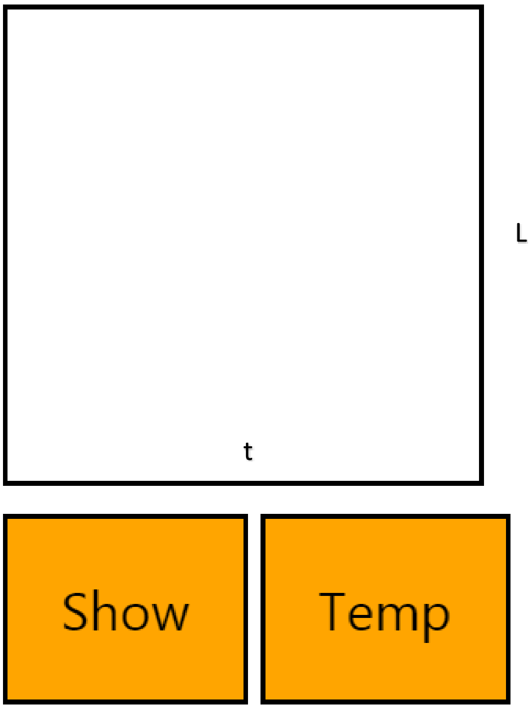


__Figure UI-2__. One of the first UI designs used.

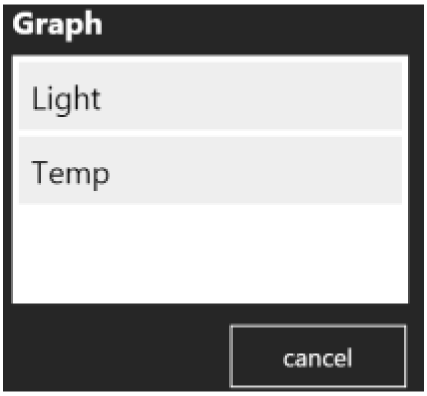
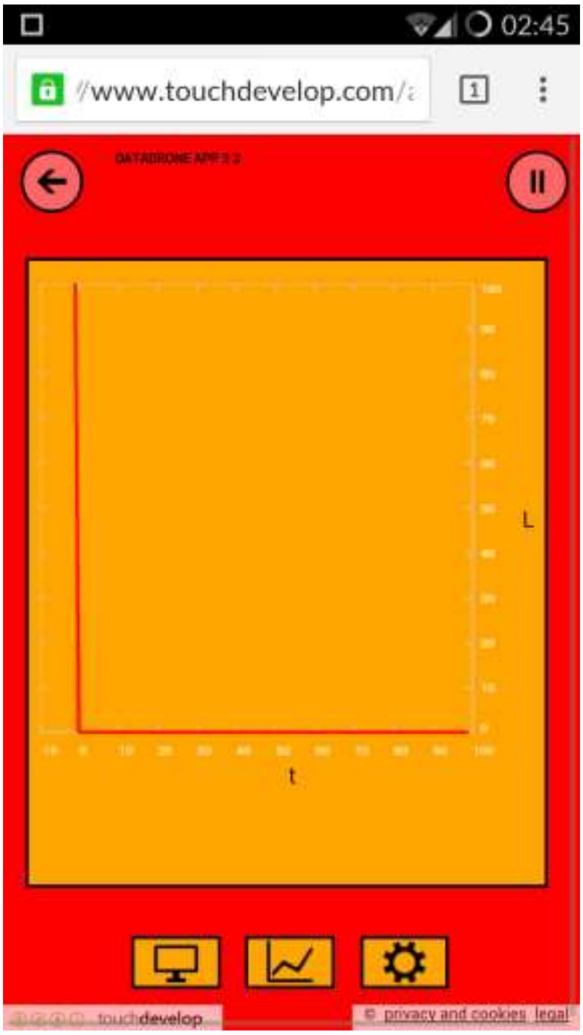


__Figure UI-3__. Dropdown menu, shown by pressing “Temp”

As shown in Figure UI-2, our design wasn’t even close to being acceptable, not by the future user, but by us. We were developing an application and we wanted to be as professional as possible in every aspect of our work. After all we had already imagined how we wanted our UI to look like and we were not every close in implementing it.

After pointing out to each other time and again, that our interface wasn’t up to par with the standards we set out to achieve, we decided it was time to make it look good, adding more slick and modern icons, instead of the buttons we had as shown in Figure UI-2, that can be understood by anyone due their commonality in smartphones and tablets. Our final design is as shown below in Figure UI-4.



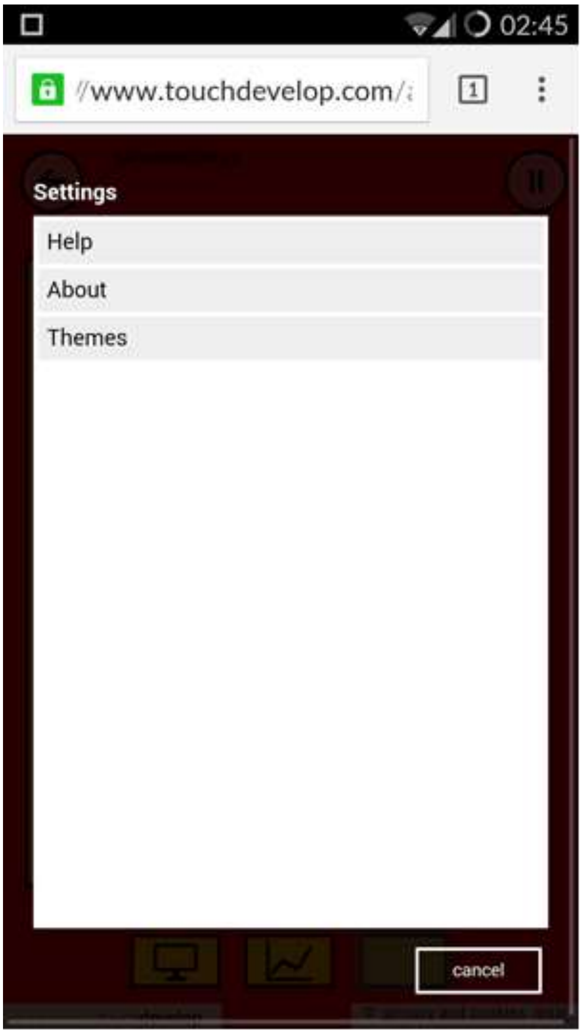


Figure UI-4. *Our final UI design.* ***Figure UI-5****. The settings menu.*

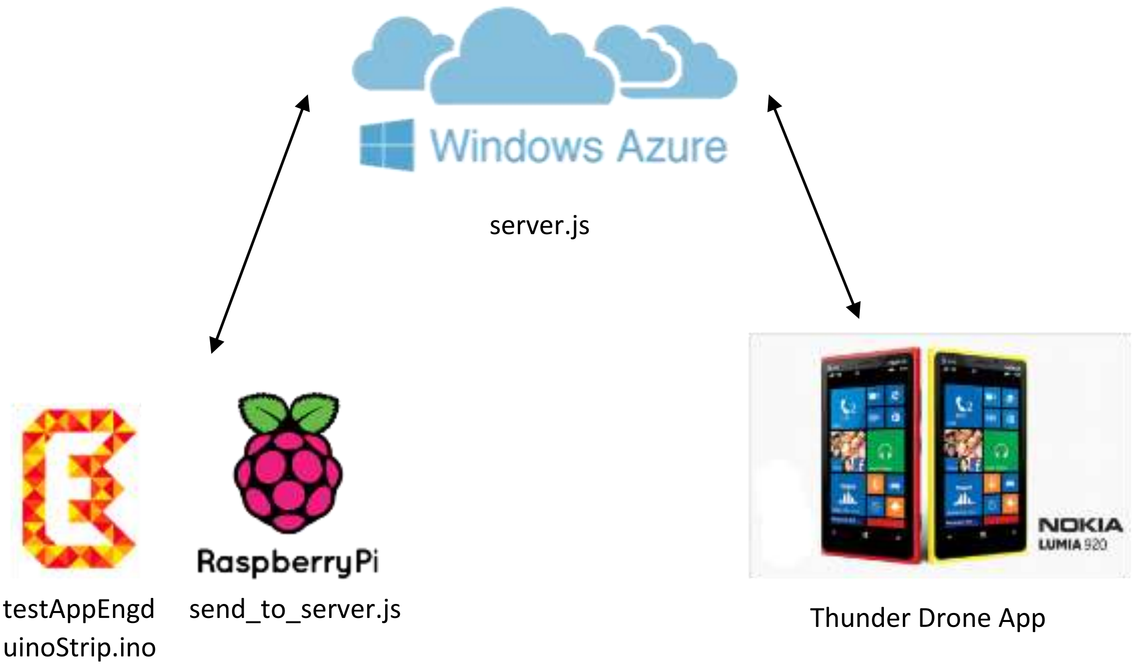
In our final design, as shown above, we have 3 buttons, represented by icons. The first "Monitor" icon on the left acts as what was previously the "Show" button. The middle one acts as the dropdown menu which will produce the same menu as shown in Figure UI-3. Finally the button on the right is the settings button which will produce the menu shown in Figure UI-5. We chose to add a menu instead of another page, so that the user can finish setting up whatever is required as fast as possible with no delays.

TouchDevelop has a very friendly environment which after getting used to, enables programmers to easily develop their User Interface, with simple features called pages and boxes. A page is as one would expect the whole screen and on top of that the user adds the boxes in order to craft the interface as they see fit. After that boxes and pages can be manipulated by adding borders, margins, changing background colours and pictures and generally various options that allow creativity to a big degree. We tried to make use of all the tools that were given to us by TouchDevelop in order to create an interface which would satisfy both us and potential users.

8.6. Project Schematic

Because our code is spread out across different platforms, it would not be ideal to create a code hierarchy, as the JavaScript code does not even make use of any functions.

Instead, we have provided a Project Schematic to simply explain the connections between which programs and which platforms.



9. Work Distribution

9.1. Garrett May

- Creating the code for the Engduino to do the following tasks:
  - Read data from sensors
  - Save data to a MicroSD
  - Send data over to the Raspberry PI via the Serial port
- Creating the following TouchDevelop libraries:
  - Linked List library (for holding current data)
  - Graphing library (for displaying data)
- Editor for the TouchDevelop application

9.2. Omer Mohammed Mirza

- Creating the C program for the Raspberry PI to read from the Serial port and send via Wi-Fi data to the Azure website
- Setting up and configuring the Azure website
- Creating the following TouchDevelop libraries:
  - Web Socket (to make a connection with the Azure website)

9.3. Iacovos Pantekhis

- Creating the TouchDevelop User Interface
- Creating the following TouchDevelop libraries:
  - Button library (to produce buttons for user input)

10. Testing

Testing was done in both the development of the project and after we had finished our mobile application.

Importantly, we tested at the following moments:

- End of every week
- After our discussion with UCL Advances
- After our discussion with Dean

The discussions are noted in the meetings, in the appendices.

We also made a questionnaire to analyse our work. This can also be found in the appendices section below.

11. Installation

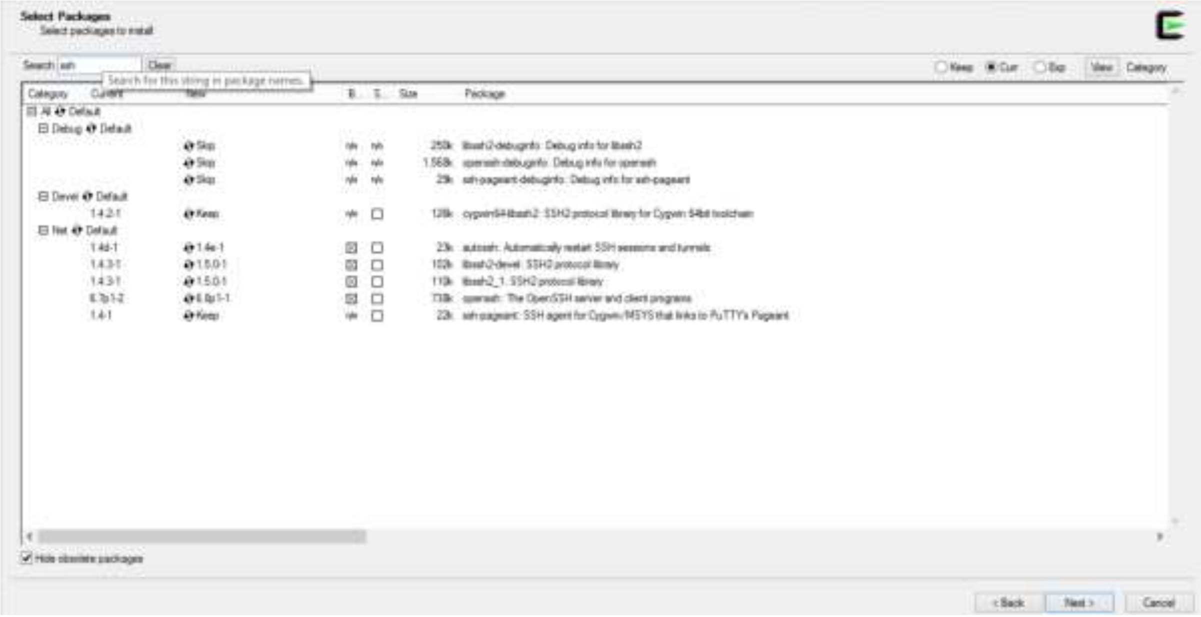
11.1 Setup

For our project, you will need the following:

- An Engduino with the uploaded code
- A PI with the program
- An Azure website
- A laptop or computer that has SSH installed
- A monitor and keyboard

The Engduino, PI and Azure website should be setup accordingly.

To install SSH onto your laptop, I recommend downloading Cygwin from the website (https://cygwin.com/install.html) and install the SSH packages when it appears on the screen.



Select Packages
Select packages to install

Search: ssh

○ Keep ● Curr ○ Exp

Category	Current	New	B	S	Size	Package
⊟ All Default						
⊟ Debug Default						
		Skip	n/a	n/a	250k	libssh2-debuginfo: Debug info for libssh2
		Skip	n/a	n/a	1,568k	openssh-debuginfo: Debug info for openssh
		Skip	n/a	n/a	29k	ssh-pageant-debuginfo: Debug info for ssh-pageant
⊟ Devel Default						
	1.4.2-1	Keep	n/a	☐	128k	cygwin64-libssh2: SSH2 protocol library for Cygwin 64bit toolchain
⊟ Net Default						
	1.4d-1	1.4e-1	☒	☐	23k	autossh: Automatically restart SSH sessions and tunnels
	1.4.3-1	1.5.0-1	☒	☐	102k	libssh2-devel: SSH2 protocol library
	1.4.3-1	1.5.0-1	☒	☐	110k	libssh2_1: SSH2 protocol library
	6.7p1-2	6.8p1-1	☒	☐	738k	openssh: The OpenSSH server and client programs
	1.4-1	Keep	n/a	☐	22k	ssh-pageant: SSH agent for Cygwin/MSYS that links to PuTTY's Pageant

☑ Hide obsolete packages

11.2. Instructions

If you are going to be using a different Wi-Fi network, you will need to turn the PI on with the monitor and keyboard present.

```
Login user: root
Password: raspberry
```

You will need to type in the command:

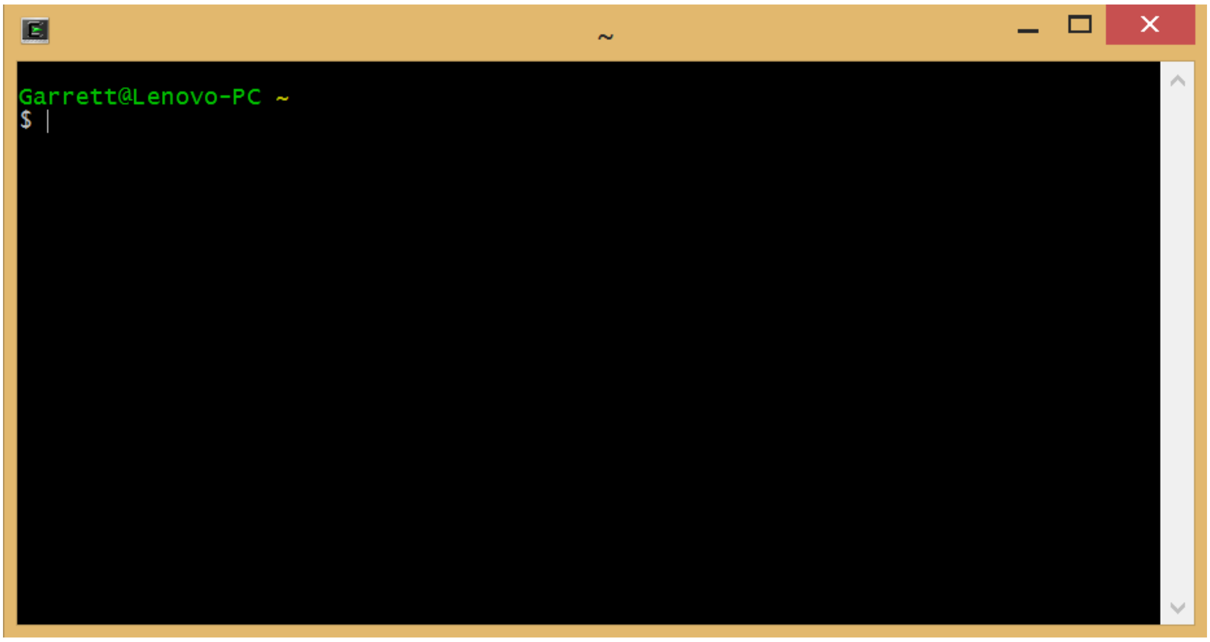
```
iwlist wlan0 scan
```

and change the Wi-Fi network there. This is to ensure that the PI connects to the network on startup. You will need to reboot.

For further reference, please see this website: https://www.raspberrypi.org/documentation/configuration/wireless/wireless-cli.md

If you have already setup the PI's Wi-Fi network configuration, you can SSH through Cygwin.

First, open up Cygwin.



```
Garrett@Lenovo-PC ~
$ |
```

Type in the command:

```
ssh <IP address> -l root
```

Now type in the password: `raspberry`. Afterwards, move into the server3 folder by typing in the command:

```
cd server3
```

Next, you will want to make a connection with the Azure website, to send data to it. Type in the command:

```
node send_to_server.js /dev/ttyACM0 wss://relay-
server.azurewebsites.net
```

This will begin the connection. It will successfully connect if it says that a connection has been opened.

To visit our application, you should go to the Azure website: *data-drone.azurewebsites.net*

12. References

12.1. References for the Code

- Engduino
  - String reference
    - URL: http://www.cplusplus.com/reference/string/string/
  - C++ reference
    - URL: www.cppreference.com
- PI
  - JavaScript Tutorial
    - URL: http://www.w3schools.com/js/
  - Node.js Guide
    - URL: https://nodejs.org/
  - Node.js Web Socket library
    - URL: https://www.npmjs.com/package/nodejs-websocket
  - Serial port library
    - URL: https://github.com/voodootikigod/node-serialport
- TouchDevelop
  - Programming on the Go book
    - Published: January 1st, 2013
    - Author: R. Niegel Horspool
  - drawing library
    - URL: https://www.touchdevelop.com/jwlya
    - Author: Peli de Halleux

12.2. Gratitudes

We would like to express our thanks to the following people:

Dean Mohamedally (Senior Teaching Fellow)

- For proposing the project to us as an interesting and unusual experimentation on utilising TouchDevelop

Peli de Halleux (Principal RSDE for Microsoft Research)

- For providing technological advice

Dr Yun Fu (Teaching Assistant for COMP103P)

- For communicating to Dean, Graeme, and others on our behalf
- For spending the time to learn the project installation, and for demonstrating the project in Paris on our behalf

Graeme McPhillips (UCL Electrical Engineering Technician)

- For ordering our hardware parts

David Xiao (First Year Computer Science Student)

- For referencing us to the drawing library in TouchDevelop
- For explaining how to update a graph
- For inventing the name “Thunder Drone”

13. Appendices

13.1. Client Feedback

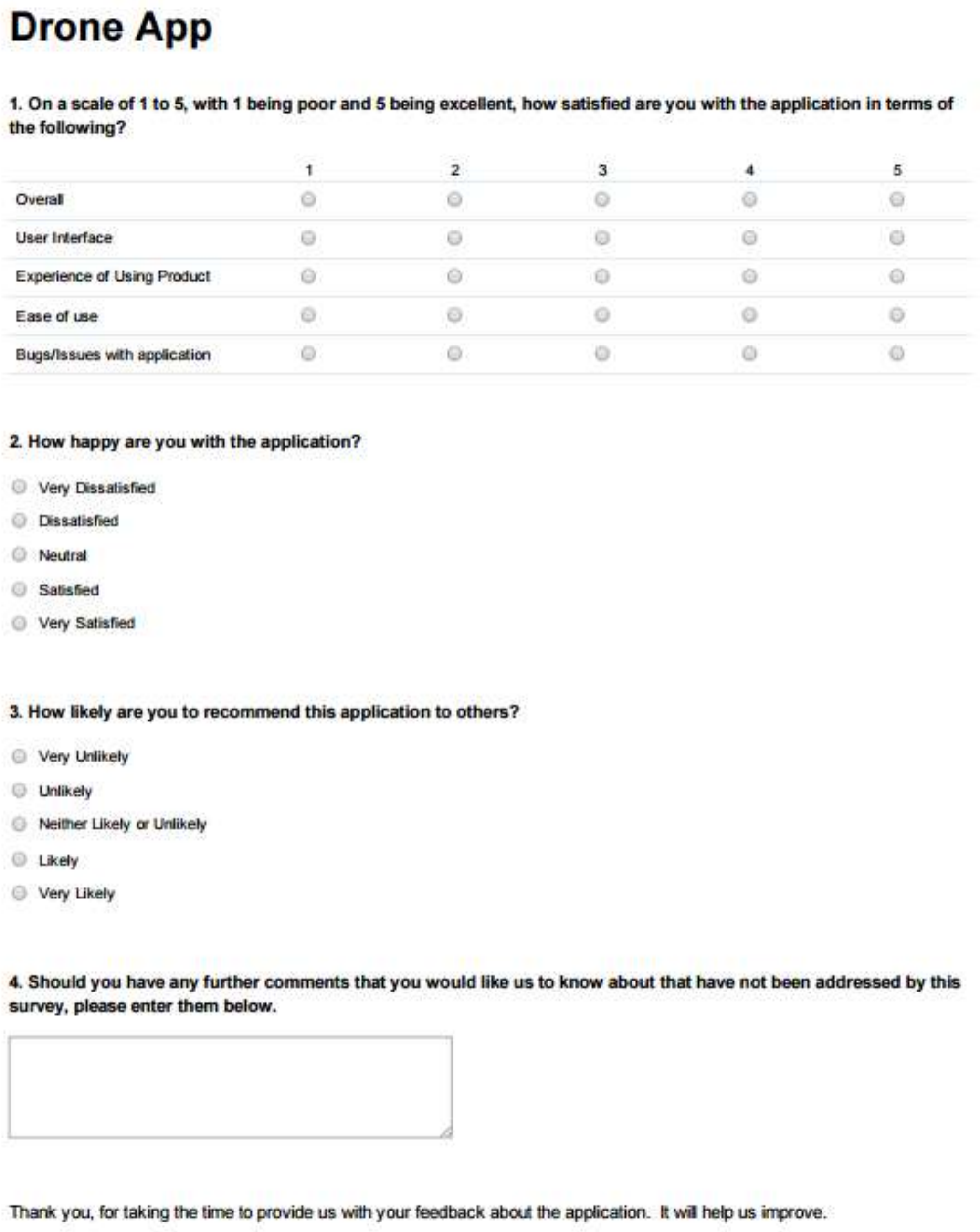
Unfortunately, we have no written documentation of Dean, our effective client (since it was he who proposed the project, and whom we discussed about the project in the beginning) making a final evaluation of our project. However, when it was shown to Dean on the 9th April, he expressed good comments about our work, especially the bidirectional secure communication present in the system.

13.2. Testing and Survey Feedback

After reaching the final stages of our application. We realised that it was time to create a questionnaire to see what people would think of our application and whether the User Interface and User Experience would be as good as we thought they would be and also help us improve it with their feedback.

However we faced an obstacle in our efforts to get people to use the application, and that was the fact that our application required a certain set-up of the Raspberry PI, the Engduino, and the Azure website to be up and running before you can get the application working properly. For that reason we thought that it would be best that we ask some of our peers, in the computer science department of UCL to firstly use the application, after we had set up our system, and then secondly to answer our questionnaire over the internet. We fully acknowledge and know very well that our fellow students might be a bit biased, but alas, we didn't have a choice. Thus we asked them to answer with honesty and made sure they understood that it would only harm us if they didn't take it seriously.

The questionnaire we presented to them is the one below:



Drone App

1. On a scale of 1 to 5, with 1 being poor and 5 being excellent, how satisfied are you with the application in terms of the following?

	1	2	3	4	5
Overall	○	○	○	○	○
User Interface	○	○	○	○	○
Experience of Using Product	○	○	○	○	○
Ease of use	○	○	○	○	○
Bugs/Issues with application	○	○	○	○	○

2. How happy are you with the application?

○ Very Dissatisfied
○ Dissatisfied
○ Neutral
○ Satisfied
○ Very Satisfied

3. How likely are you to recommend this application to others?

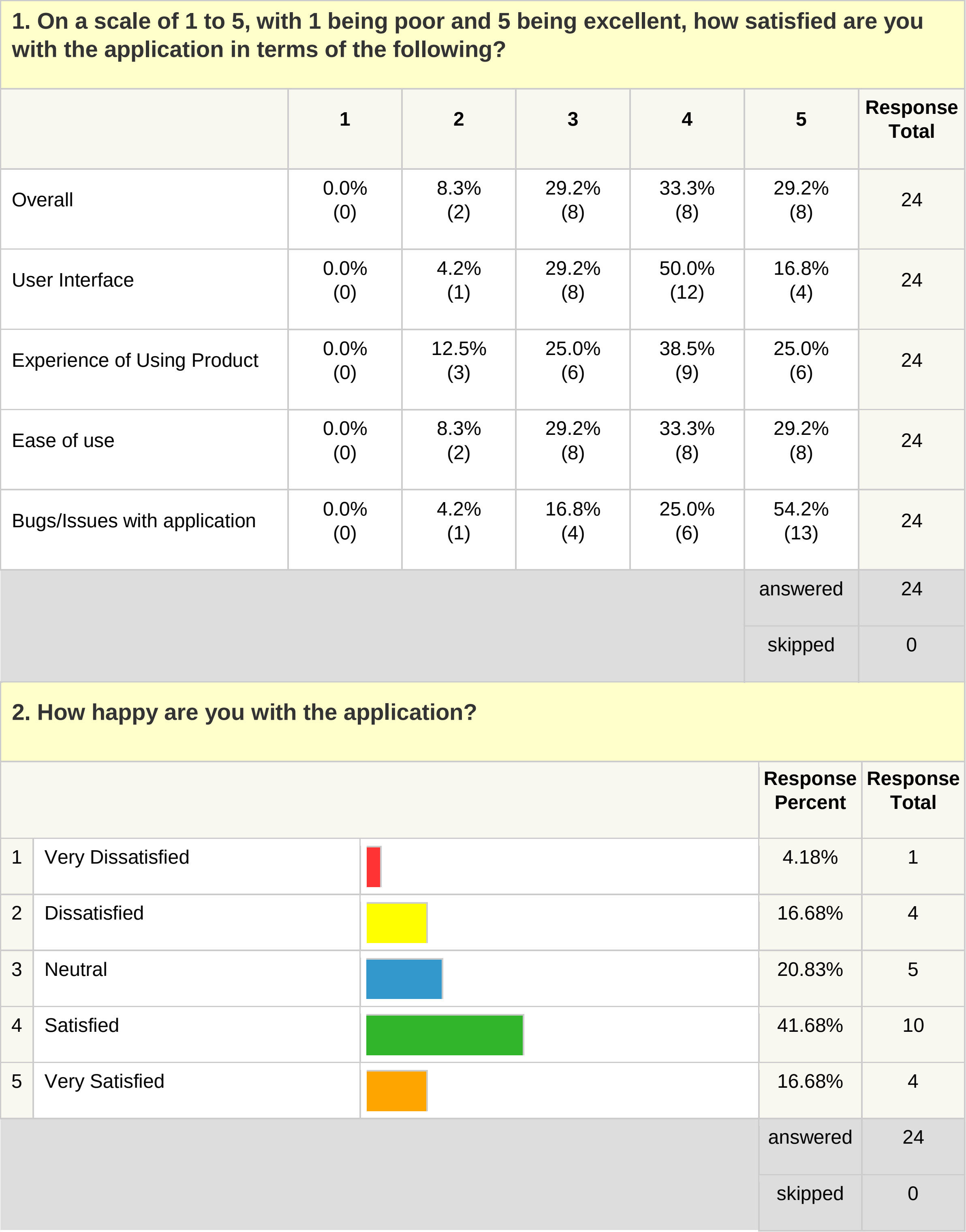
○ Very Unlikely
○ Unlikely
○ Neither Likely or Unlikely
○ Likely
○ Very Likely

4. Should you have any further comments that you would like us to know about that have not been addressed by this survey, please enter them below.

Thank you, for taking the time to provide us with your feedback about the application. It will help us improve.

And the results we got from 24 students that took part in answering are questionnaire are the ones below:

How Happy Are You With The Application?

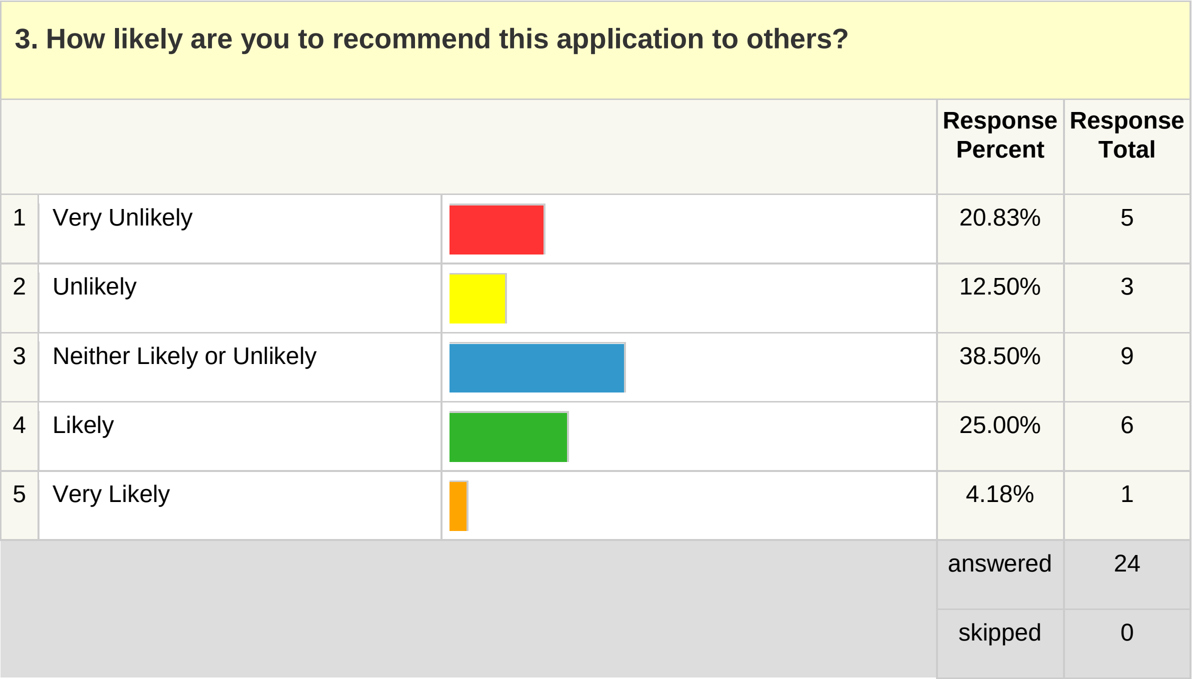


1. On a scale of 1 to 5, with 1 being poor and 5 being excellent, how satisfied are you with the application in terms of the following?

	1	2	3	4	5	Response Total
Overall	0.0% (0)	8.3% (2)	29.2% (8)	33.3% (8)	29.2% (8)	24
User Interface	0.0% (0)	4.2% (1)	29.2% (8)	50.0% (12)	16.8% (4)	24
Experience of Using Product	0.0% (0)	12.5% (3)	25.0% (6)	38.5% (9)	25.0% (6)	24
Ease of use	0.0% (0)	8.3% (2)	29.2% (8)	33.3% (8)	29.2% (8)	24
Bugs/Issues with application	0.0% (0)	4.2% (1)	16.8% (4)	25.0% (6)	54.2% (13)	24
					answered	24
					skipped	0

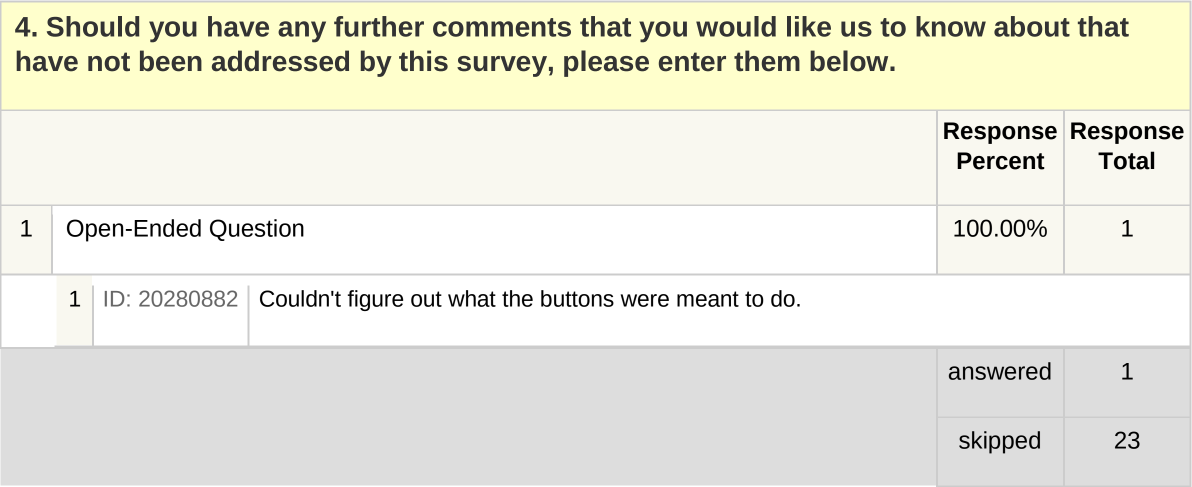
2. How happy are you with the application?

			Response Percent	Response Total
1	Very Dissatisfied	🟥	4.18%	1
2	Dissatisfied	🟨	16.68%	4
3	Neutral	🟦	20.83%	5
4	Satisfied	🟩	41.68%	10
5	Very Satisfied	🟧	16.68%	4
			answered	24
			skipped	0



3. How likely are you to recommend this application to others?

			Response Percent	Response Total
1	Very Unlikely		20.83%	5
2	Unlikely		12.50%	3
3	Neither Likely or Unlikely		38.50%	9
4	Likely		25.00%	6
5	Very Likely		4.18%	1
			answered	24
			skipped	0



4. Should you have any further comments that you would like us to know about that have not been addressed by this survey, please enter them below.

		Response Percent	Response Total
1	Open-Ended Question	100.00%	1

1	ID: 20280882	Couldn't figure out what the buttons were meant to do.

	Response Percent	Response Total
	answered	1
	skipped	23

13.3. Meetings minutes

In general we arranged one main meeting a week, usually every Friday; however, we met for smaller meetings a few times more each week, normally before lectures and whenever we had an opportunity. Throughout the week we would engage in discussions about our progress; how we should approach every challenge that came up, and if anything was troubling us or if anything at that moment in time was causing a struggle. Moreover, every Tuesday when we had our lab session we spent a good 5-10 minutes before we started working on the application so that we could again inform one another of how we had progressed after our last meetup and also at the end of the lab session we again talked about what we should be doing in the following days, until our next meeting. Fully fledged meetings began once Dean had described the project in full to us.

Meeting #1: 5th February, 2015; 16:00 - 16:30

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

Dean Mohamedally

The interview with the client (Dean) was our primary source of what the intention of the project is. We had many questions to ask, such as: “What languages will we be using?”; “What is the purpose of the drone?”; “What are the requirements that we must achieve?” and many more.

Our project requirements and intention of the drone seem quite lax, as the project is supposed to be creative such that it allows us to imagine an interesting function or benefit for society. The idea is that there will be sensors connected to an Engduino on a Phantom Drone, and we must read values from it, save the data, and send it to a phone for displaying. The Engduino is to be coded in C, and should be connected to a Raspberry Pi 2 alongside a Wi-Fi adapter. This setup allows data to be streamed to the phone via Wi-Fi to our app, which is to be programmed in TouchDevelop, and then displayed in a graph. Data could also be stored in the SD card as well.

The project is open ended; we can choose what sensors we like, so long as they are light and privacy is to be taken into account, with the hope that we will come up with a unique way of helping people using this concept. Recommended companies to look at were RS, Farnell, and Maplin.

TouchDevelop is to be used for the app, as the programming language is intended to be easy for children to use. This most likely means that the app should be able to be modified or changed easily, depending on what is necessary. It also should be useful considering we will be using a phone with a touchscreen, and so we may even be able to change any code we have on the spot.

Although much was cleared up in the interview, more problems have arisen from the answers given. One such is how we will code the Raspberry Pi 2, and in what language, as it needs to link the Engduino to the Wi-Fi adapter. Another is how the User Interface (UI) will look, as it is likely that we have multiple types of data (for example, temperature, light sensing, etc.), and so whether we have one graph and many buttons, or all graphs on one page, should be considered.

Meeting #2: 6th February, 2015; 10:00 - 11:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

This was the day after our first interview with the client (Dean), so we talked about the general purpose of our application and started thinking of possible real life problems that our Drone App would tackle and also what requirements we must achieve.

We started fleshing out how the User Interface should look. We all agreed that it be relatively simple, seeing as TouchDevelop was a language even children should be able to use, and so the design should be easily able to be replicated.

After discussing over several different options (such as multiple pages), we decided to go with the following: a set of buttons at the bottom, and the rest should contain the graph that would be able to display the data. A different sensor type would be able to be chosen by one of the buttons, and there would also be a settings button to change certain aspects of the app, such as the theme.

Meeting #3: 9th February, 2015; 13:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

For this meeting, we began discussing how we would implement the current concept of our application design in the language set to us, called TouchDevelop. TouchDevelop is supposed to be 'touch-friendly', and so we believed that it would have many features that would be suitable both for a mobile application and a computer application. This meant that our app would be very versatile on a device.

We started to begin learning the TouchDevelop language. This proved to be somewhat difficult, being unknown to us, and very different to the other languages we had more experience in (such as C and Java). Nevertheless, we decided on using a Page and utilising boxes, which were UI widgets.

Meeting #4: 10th February, 2015; 9:00 - 9:10

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We met up in the labs in order to discuss to procedure for today. We agreed with the following:

- Garrett: Working on learning C/C++ for coding the Engduino
- Omer: Working on learning C for the Raspberry PI
- Iacovos: Working on learning TouchDevelop for the mobile app

Of course, all of us learned some C/C++/TouchDevelop, but we each mainly focused on the language(s) above, in order to optimise our work

Meeting #5: 12th February, 2015; 11:00 - 12:00

Having made the some progress in the labs previously, we discussed what should be done next. Dr Yun Fu recommended to us that we choose a suitable application or use for our project; how might it benefit the society? We posed this question to ourselves, and we decided the make our answer in the next meeting.

In terms of the project, everything was moving smoothly. We also decided to start combining our views in order to produce our Design Report for Milestone 1

Meeting #6: 13th February, 2015; 10:00 - 11:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

After going over the various candidates for which problem our Drone App would tackle, we figured we would focus on a form of Hurricane Hunting Drone, which is basically the idea of using a drone to charge into the heart of a storm to gather and collect data, without risking human life, and to help improve forecasts, save money, and lives. We decided that the name of our application would be "Thunder Drone", as a reference to stormy weather. We also marked out the Must-Should-Coloud-Would (MoSCoW) requirements, which were most of the requirements that our application must satisfy.

Meeting #7: 16th February, 2015; 13:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We began to sum up the Design Report in preparation for handing it in for Milestone 1. We finalised each team member's roles in the project. We attempted to delegate the seven roles to the person who either had the best experience in that role and/or who might be the best suited to that task.

Meeting #8: 18th February, 2015; 9:00 - 9:10

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We all discussed with each other what we had learnt about the respective languages. This was extremely important, because, as a team, we were able to better convey a topic of a certain language to all the other members in the team, such that we improved our understanding of both the topic and language, by not only listening to our teammates but also teaching each other.

Meeting #9: 19th February, 2015; 11:00 - 12:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

In this meeting we finished the Design Report. Importantly, we reminded ourselves of the project, and began discussing the Gannt chart we had opted to use as our primary way of displaying our work plan.

Meeting #10: 20th February, 2015; 10:00 - 11:00

We began to start our programs for the Engduino, Raspberry PI and TouchDevelop. We also received an Engduino, and we made an order for the following:

- Raspberry PI 2
- USB W-Fi Adaptor
- Pressure Sensor
- Humidity Sensor

The PI was necessary for taking data from the Engduino and trasmittin it via Wi-Fi, which was done through the USB Wi-Fi Adaptor. The pressure sensor and humidity sensor were ordered as example sensors, considering our project has the possible application of data gathering in stormy weather. However, this was not needed to prototype, test, and show off the application, as the Engduino had on board sensors (temperature and light sensor), which proved to be more useful and reliable when testing.

Meeting #11: 23rd February, 2015; 13:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We continued to discuss the code we were writing for our respective subsections. Importantly, there were a number of problems:

- It seemed the Engduino had problems saving data to a MicroSD. This was rectified by not utilising the Engduino’s LEDs. Apparently, there is a problem with SPI which conflicts, causing saving to the MicroSD and using LEDs to effectively be mutually exclusive
- We thought that we should parse the data in the Raspberry PI; however, on further retrospect, parsing in the mobile application seems better, as that is where the data is being displayed. The Raspberry PI simply acts as a relay agent, transferring the data from one place to another
- TouchDevelop had major problems involving Page, boxed, and the Number Map which we had been using to represent a graph. Many of the problems were the lack of functionality, the difficulty at performing a task, and a lack of customising the UI

Meeting #12: 24th February, 2015; 9:00 - 9:05

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

After solving the problem with the Engduino, much of the programming of the Engduino seemed to be complete. Making functions to read data, save data, and send to the Serial port was very easy, after enough reading of the documentation. We then wrapped these into more suitable, user-friendly functions, to better represent what we were doing.

Since there was more time due to the Engduino being nearly finished, more time was delegated on the TouchDevelop application, including more team members.

Meeting #13: 26th February, 2015; 11:00 - 12:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

More emphasis was placed on the TouchDevelop side of the project. We were able to produce a crude graph, and a couple of buttons in order to start testing.

It was at this point that we received the Raspberry PI, as the hardware had arrived late. We also received the Wi-Fi module, but not the sensors. We attempted to write an operating system to the PI, but failed, due to the fact that the MicroSD did not have sufficient storage to contain it fully.

Meeting #14: 28th February, 2015; 10:40 - 11:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We improved the user interface of the TouchDevelop application to visualise the data that is to be received from the Drone. Having received the PI and WiFi adapter hardware previously, we now needed to set-up the software which would send data from the sensors connected to the Engduino over WiFi. A program will be written for this purpose.

The user interface needs to be polished and we are working on a solution to allow the touch develop to communicate with the Raspberry Pie. We aim to do this by the 20th March.

Meeting #15: 2nd March, 2015; 13:00 - 13:30

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

Today was quite a short meeting. We discussed about what operating system we should use for the PI. We opted to go for Debian, a kind of Linux distribution, common, well-known, and favourable.

Meeting #16: 3rd March, 2015; 9:00 - 10:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

Our application has now started to take some form, after having started working on the User Interface using TouchDevelop, sending and receiving data from the Engduino to the Raspberry PI. We established internal deadlines, including deadlines set before Milestone 2 and Milestone 3. We also talked about improving our familiarity with TouchDevelop, which is still a bit of a struggle. Moreover we talked about reviewing the order for the sensors that we require, and what we would do if we receive our order.

Meeting #18: 5th March, 2015; 11:00 - 12:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We discussed how to begin combining the connection between the TouchDevelop mobile app and the PI. Our initial thoughts were to send the data received from the Engduino via a free Wi-Fi network and for TouchDevelop to read from this Wi-Fi hotspot, in order to then parse it to receive the data. We started to look for a way in TouchDevelop to make a Wi-Fi connection.

Meeting # 18: 6th March, 2015; 10:15 - 10:45

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We began to implement the parsing and the connection to a Wi-Fi hotspot in TouchDevelop. One important aspect was that we had to go back to the Engduino to make some adjustsments. We decided to send over a piece of data in this compressed form: “T23.95”, where the first character would represent the sensor type (in this case, temperature), and the rest would represent a number we would parse. This seemed to be difficult on the TouchDevelop side, in the way the PI would send data over. We agreed to discuss this in the next meeting.

Meeting #19: 9th March, 2015; 13:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We ran into a problem with the PI. Unfortunately, the Raspberry PI we had was a PI, which used a MicroSD for storing the operating system, rather than an SD like the previous Raspberry Pi. Since we only had a 1GB MicroSD, and Debian required around 4GB, we were quite stuck at the problem. However, we researched into this, and found out another operating systenone called Minibian. Minibian is an operating system similar to Debian, a Linux distribution. However, it does not contain any UI features. It only needs around 1GB in order to run. Not only was this suitable, given our constraint, but it may even be more optimal, as it instantly goes to the command line, rather than opening up a terminal in Debian.

Meeting #20: 10th March, 2015; 9:00 - 9:10

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

In this meeting we successfully rectified our problems.

The Engduino problem was solved by returning using the String() function in the C++ Engduino environment. This function was able to take a range of different parameters, and turn them into a String. This successfully meant that no matter the data type, we were able to convert it into a String.

The TouchDevelop problem was caused by not actually reading the first character sent. When reading the data over Wi-Fi, TouchDevelop seemed to be quite slow, and actually read 3 or 4 pieces of data. This caused invalid values, especially by the introduction of the newline character, "\n". We aimed to create a solution soon.

Meeting #21: 12th March, 2015; 11:00 - 12:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We had quite a difficult time working out wherein lied the problem in our code. It seemed that the PI worked perfectly fine when sending over data to the Wi-Fi network, yet the TouchDevelop app was unable to read the first character and parse the rest. We continued to struggle over this, but unfortunately, to no avail.

There was also a problem with the way we handled data in the Engduino. When incorporating the light sensor into the project, we seemed to have a difference in opinion over what data type it actually was. This caused several problems, including reverting to using a char array instead of a String.

Meeting #22: 13th March, 2015; 10:30 - 11:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

Progress was made on both the UI and the transmitting data between Engduino and PI fronts. We were led to believe that we have had to use Cordova to create an app, but it is not necessary, so long as we can find a way to use the internet.

Meeting #23: 16th March, 2015; 13:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We discussed what has been met so far.

Currently, the Engduino is working, and now, so is the PI. That subproject that contained the Engduino and the PI were now completed, and correctly sending messages; we just now needed to fix the TouchDevelop application. We agreed to all put more time into it.

Meeting #24: 18th March, 2015; 9:00 - 9:10

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We were able to successfully finish the project, with the application running on a laptop. We are currently using the Engduino's on-board sensors, as the sensors have not arrived yet. We did numerous testing today, including how the application might react in problematic and different situations.

Meeting #25: 19th March, 2015; 11:00 - 12:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We met to attempt to try and get the TouchDevelop application working on a mobile device. Unfortunately, due to the restraint that it would only work on TouchDevelop local, we were unable to. We attempted to run a TouchDevelop local server on the PI, but were unsuccessful in our mission. We shoul endeavour to solve the problem as soon as possible.

Meeting #26: 20th March, 2015; 10:30 - 11:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We continued to find a method of running the app on a mobile device. The constraint on being on TouchDevelop local still existed.

Meeting #28: 23rd March, 2015; 13:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We sat down and discussed the comments presented to us previously by UCL Advances:

- They saw that we had placed most of our effort into the hard graft of getting the connections sorted. Indeed, this was the most difficult part of the project.
- They suggested that we focus on the User Interface; for example, perhaps not having a white background; themes might be preferable.
- They also would have liked to have seen it work as a mobile application. We were unable to show them this; only via a laptop.

We agreed to focus on tidying up the User Interface and to make it work on a mobile phone.

Meeting #28: 24th March, 2015; 9:00 - 9:10

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We decided that we should make improvements to the UI, replace the bars that act as buttons with more slick and modern icons that can be understood by anyone due their commonality in smartphones and tablets. The UI changes should be completed by next week since some work has already been done on it.

Meeting #28: 26th March, 2015; 11:00 - 12:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We worked on the UI of TouchDevelop. Importantly, we were able to add a Settings button, which included another dropdown, with a Them button. This would allow us to choose from different Themes, several different ones if we decided to add them. Currently, there are three: Basic (red and orange), Light (white and black), and Dark (black and white; reverse of Light). We also deliberated on how to choose the theme, as we were able to either choose different Strings that represented the themes, or different themes, that gave out each information about the theme. We decided to stick with Strings, as this seemed the least complicated, and would actually provide a better User Experience than the latter.

Meeting #29: 28th March, 2015; 10:15 - 10:45

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We decided that we need to order the Drone which will allow us to take one more step closer to the finalisation of the project, since we needed to test it in close to real life conditions with all the major parts that it’s comprised of.

Meeting #30: 30th March, 2015; 13:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We attempted to launch the Node.js server such that it would run a local version of TouchDevelop. This was because our application made use of the Shell Socket class, which was constrained by the local version. We needed to work in this way so that we could use a mobile phone to launch the TouchDevelop app.

However, after many tries, we were unsuccessful in our plan. Unfortunately, the TouchDevelop application kept on timing out, and eventually refused to let us login.

Meeting #31: 31st March, 2015; 9:00 - 9:05

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

After asking our client, Peli, what the problem might be, he replied that the scenario is not supported. This meant that we would have to reconsider the connection between the PI and the TouchDevelop app.

Meeting #32: 2nd April, 2015; 11:00 - 12:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

The drone has arrived. We wanted to pick it up; however, it was recommended to us to test it a week later, so that we could improve our application and then get any feedback or advice from Dean.

We also changed how our project worked in terms of the connection between the PI and the TouchDevelop application. Previously, the PI would send data from the Engduino to the TouchDevelop app via a Wi-Fi network. Now, instead, we have setup an Azure website; the PI will send data from the Engduino to the Azure website, which will store the data. The TouchDevelop application will make a request to the server, say, for example, “T” for temperature, and the server will return the most up-to-date value for the temperature.

Meeting #33: 3rd April, 2015; 10:15 - 10:45

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We improved the design of the application again. We included the modernised buttons, which had background images instead of words, allowing the application to be more diverse, since it would be restrained less by language barriers. However, we struggled at changing the colour of the background image, depending on the theme, even with the image being partially transparent.

Meeting #34: 6th April, 2015; 13:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We finished off the buttons; we successfully were able to change the background colour of them. This was done by placing the buttons into another box, a box which was able to have its background et to a different colour. The transparent nature of the icons used allowed the background colour to shine through.

We also turned the buttons and buffer into libraries, Button and Buffer, respectively.

Meeting #35: 8th April, 2015; 9:00 - 9:05

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We intended to switch out our old graphing method with a new one. A UCL first year student named Toby Li seemed to have built a graphing library. We asked his permission to make adaptations (for example, colour changing, and axes labelling), for the benefit of our program, and he agreed. However, when testing out the graph, it appeared to blink both randomly and consistently. It also did not update as well as our Number Map version. We will reconsider our options for this later.

Meeting #36: 9th April, 2015; 11:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

Considering that later on we had to meet up with Dean to show off our project, we went through our project to work on improving it. Most notably, we made the connections between the PI and the website and the application secure.

We also heard note that Graeme McPhillips had made a small board for us. When he receives the sensors we ordered, he will solder them on, and then give it to us. This will allow us to log Humidity, Pressure, and Temperature (the temperature sensor was an extra inclusion).

Meeting #37 10th April, 2015; 11:00 - 15:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

After receiving the phantom drone which would be used for the testing purposes of our application we set out to a field where we would be able to flight test it, so as to comply with the law on drones. We proceeded in capturing a few recordings of the drone that would be used in our video, as requested for the Microsoft presentation. However, due to complications, there was a minimal amount of footage involving the drone.

Meeting #38: 13th April, 2015; 13:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

Considering that neither of our graphing options were suitable for our application, we emailed Peli, who is affiliated with the TouchDevelop language itself, whether he could recommend a graphing library, or, at any rate, of a good way to update a graph. Peli produced a library that interacted with JavaScript, that seemed to work very well. However, on further analysis, we were not able to update the graph, and so we had to scrap this idea entirely.

Meeting #39: 14th April, 2015; 9:00 - 9:05

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We finished of the video for Microsoft that briefly explained how our project worked, how it could easily be setup, and showing off the interesting aspects of it, such as real time data displaying and choice of sensor for displaying.

Meeting #40: 16th April, 2015; 11:00 - 11:30

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We had a short meeting where we discussed how to represent the graph, as this seemed to be the most important topic currently. We believed that it should be on a canvas, and so we looked back on our previous adaptation of Toby's library, which utilised a Picture. We are now attempting to find a suitable library.

Meeting #41: 18th April, 2015; 10:15 - 10:45

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We met to discuss the requirements laid down at the beginning of the project. Importantly, we looked at the ones that had not been implemented yet, and talked about what should be done:

- Saving data to the phone. Whilst this was important for the user of our mobile application, we believed that it may be more preferable to continue to save on the Engduino's MicroSD. This would make the TouchDevelop program less complicated to code and use.
- Different methods of displaying data. We agreed that the graphing method should be changed, as the current one lacked customisation. However, we believed that different types of graph would be beyond our scope, considering the current one had problems of its own.
- Controlling the drone using the mobile application. Due to the fact that the drone had its own remote controller, we believed that controlling the drone using a phone would be unnecessary, as whoever had the drone would need the controller. It also seemed too convoluted, as, due to the fact that we were contacting a website via Wi-Fi, rather than sending data over via a network, this would allow multiple users to control the drone, and drone control over the internet, which could have been unsuitable.

Meeting #42: 20th April, 2015; 13:00 - 14:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We all agreed that some of our time should be dedicated to the documentation of the project as a whole. We decided to begin creating our Individual Reports.

Meeting #43: 21st April, 2015; 9:00 - 9:05

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

Today, we met up in order to prepare for the team leader, Garrett, to go to Yun Fu and help explain how to setup the project. This was because Yun Fu was travelling to Paris to Microsoft in order to demonstrate our projects, how they worked, and how they were important, considering they would be programmed in TouchDevelop. We laid out what exactly would someone need in order to show our project off to someone else, and began working from there.

Meeting #44: 23rd April, 2015; 11:00 - 12:00

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

The team leader, Garrett, notified to the rest of the team Yun Fu's comments on the app. They were thus:

- To remove the address option choice in one of the buttons. This would have allowed others to choose which Azure website they got the data from; however, since we were using our own default one, removing this would be better, to avoid confusion.
- To make the buttons in the drop down larger, if possible, for ease
- To move the type of sensor display at the top. Currently, we have a box next to the graph that displays a character, depending on the sensor type. He recommended moving it to the top, so that it would be easier to see

Meeting #45: 24th April, 2015; 10:15 - 10:45

Attendees:

Garrett May; Omer Mohammed Mirza; Iacovos Pantekhis

We decided to see if we could write a better graphing library. One friend recommended the drawing library, a library that involved using a canvas. We decided to create our own graphing library, which would utilise a Linked List. We would also have to create our own Linked List library, in order to store the data coming in from the website. This will be finished by the time Yun Fu travels to Paris to demonstrate the project.

We also began to gather up our work in order to create a Group Report for Milestone 3. This would hold the documentation for the bulk of our work.