Text, logo

Description automatically generated

Internet of Things Project

NaviCar

Nathan Ferry

Bachelor of Software & Electronic Engineering

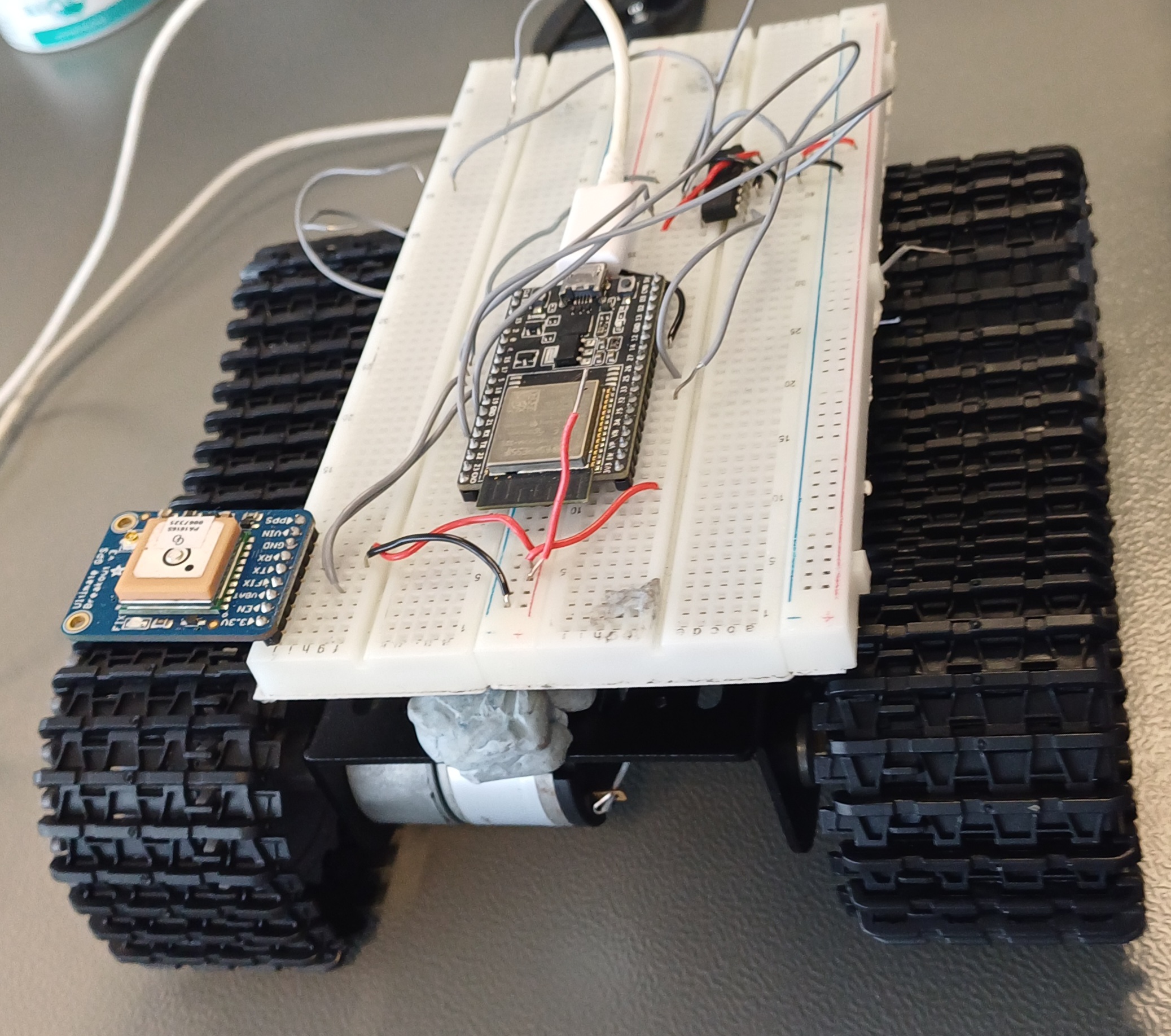
Atlantic Technical University

2023/2024

A screenshot of a computer

Description automatically generated

IoT Project Poster



NaviCar Hardware

**Declaration**

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Software & Electronic Engineering at the Atlantic Technical University, Galway campus.

This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

\_\_Nathan Ferry\_\_\_\_

**Acknowledgements**

Use this section to acknowledge anyone, if you wish to, who might have helped during your project.

**Table of Contents**

[1 Summary 8](#_Toc163318084)

[2 Project Architecture 9](#_Toc163318085)

[3 Development Platform and Tools 10](#_Toc163318086)

[4 Sensors 10](#_Toc163318087)

[4.1 Temperature Sensor 10](#_Toc163318088)

[4.1.1 Temperature Sensor Code 11](#_Toc163318089)

[4.2 SEN0017 Sensor 12](#_Toc163318090)

[4.2.1 SEN0017 Code 12](#_Toc163318091)

[4.3 HC-SR04 Ultrasonic sensor 13](#_Toc163318092)

[4.3.1 Ultrasonic sensor code 13](#_Toc163318093)

[5 GPS Ultimate V3. 14](#_Toc163318094)

[5.1 GPS Code 14](#_Toc163318095)

[6 Interrupts 15](#_Toc163318096)

[7 Motors 16](#_Toc163318097)

[7.1 DC Motors 16](#_Toc163318098)

[7.2 Servo Motors 16](#_Toc163318099)

[7.3 Motor Control Code 17](#_Toc163318100)

[8 3D printed components 17](#_Toc163318101)

[8.1 Designing the components 17](#_Toc163318102)

[9 Web Server 18](#_Toc163318103)

[9.1 Wi-Fi 18](#_Toc163318104)

[9.2 Web Server 18](#_Toc163318105)

[9.3 HTML & CSS Code 18](#_Toc163318106)

[10 Problem Solving 19](#_Toc163318107)

[11 Impact of Project on Sustainability 19](#_Toc163318108)

[12 Conclusion 20](#_Toc163318109)

[13 References 20](#_Toc163318110)

[Appendix 1: Code 21](#_Toc163318111)

[Appendix 2: Bill of Materials 22](#_Toc163318112)

[Appendix 3: Schematic 23](#_Toc163318113)

# Summary

The summary should concisely describe your whole project. Why? What? How? It should communicate:

* + The goal of the project.
  + The scope of the project.
  + The important features of the project. Include the impact on United Nations Sustainable Development Goals UN SDGs.
  + The approach to the project.
  + The main methods & technologies used.
  + What was accomplished.
  + The main conclusions.

The length of the summary should be 200-300 words or fit well on this page.

The NaviCar is a self-autonomous line following vehicle with a main goal of self-navigation while also updating the user of their location and the local temperature and humidity via a website hosted on the ESP-32. It consists of a chassis, ESP-32 microcontroller, 2 SEN0017 line sensors, a GPS ultimate V3, a Servo motor, 2 DC motors, DHT11 temperature and humidity sensor and a HC-SR04 Ultrasonic sensor. The project aims to tackle the U.N. Sustainable development goals (SDGs) 3 and 11 which focus on good health and wellbeing as well as sustainable cities and communities. Indicator 3.6 which is to road deaths and accidents by 2030 is targeted by the NaviCar as its self-navigating technology aims to provide safer transport than a human driver would as humans are prone to mistakes. Indicator 11.2 is to provide access to safe, affordable accessible transport as well as improve road safety for all. The NaviCar’s ability to follow strict routes will improve road safety in any given city. The NaviCar was made by employing the philosophy of agile development. It is modular and while at its core it was designed to follow specific routes, it has turned into much more. Once I completed several aspects of the project, I was able to use remaining time to not only polish it but add extra features. I programmed all components using the Arduino ide but also utilised past knowledge of 3d drawing to design and print 2 components that my hardware mounts to. I did all web design on the ide Visual Studio Code to test my website. I used Thingspeak to graph temperature and humidity the DHT11 read live. In total I developed a website that utilises Thingspeak to graph temperature and humidity over time as well as displays current GPS co-ordinates, have functional DC motors, line sensors, servo motor and ultrasonic that can notify if there are obstacles ahead. In conclusion, I believe the project is successful in what it aimed to do with further added features.

# Project proposal

This section is optional.

Include a project plan if you had one.

Here include your Gantt chart or project timeline.

The names of tasks should be self-explanatory.

If you had to modify your project plan – and it is expected that this happens in projects – write a brief description to explain.

# Project Architecture

This section is compulsory. Start with a description of your development platform, and the development tools you are using. For example, give a short overview of the or ESP32 board [1], and the Arduino IDE. If you are using many different tools, and/or your development platform requires a longer explanation, you could put this in a separate section (see next section).  
You could also add a high level flowchart for the project.

The project architecture of this project includes the ESP32-DevKitC, Visual studio code IDE, Arduino IDE Thingspeak and Onshape. Everything runs through the ESP32, and all hardware is connected to it. Visual studio code was used for all html/CSS programming. The Thingspeak API was used for displaying data to my webpage. The Arduino IDE was used for all other programming and Onshape was used for designing models to 3d print.

A diagram of a car

Description automatically generated

Figure 2‑ Architecture Block Diagram

# Development Platform and Tools

This section is not compulsory (see note in previous section).

You could include a description of

* development platform ESP32
* IDE used - Arduino IDE and/or whatever editor you use
* Development platform for website design Notepad++, W3Schools, Atom etc
* Web Browser Developer Tools
* Simulators – SimulIDE, MultiSim etc
* App Inventor
* IoT Platform – ThingSpeak [2], Blynk etc

The ESP32 has a host of features such as built-in Wi-Fi, Bluetooth, optimisation for prototyping on a breadboard, 3.3V and 5V pins, pins that can handle Pulse width modulation(PWM). I connected all my hardware to my EPS32 and upload code to it to bring my project to life.

The Arduino IDE contains a text area for writing code and a console for outputs to be printed among other built-in features. It has a verify button for you to check your code before uploading it to your microcontroller. I programme all of my components using Arduino making functions and a main code loop.

Thingspeak allows you to send sensor data privately to the cloud, analyse and visualise it. You can graph results against time, have a light that when a value is within a specific range its green but if it exits the range, it turns red alongside other features. You can embed the iframes of graphs onto your website.

Visual studio code allows you to write html/CSS and then test it with a browser. It has built in debugging and allows various extensions for ease of code writing. I created my website using visual studio code.

Onshape allowed me to write sketches that I would later 3D print to mount my ultrasonic to, as well as my servo motor and both SEN0017s.

# Sensors

This is an example heading for a section in a project. You choose your sections to suit your project. Here all the sensors are introduced together, briefly, then elaborated on in the subsections below.

There are three types of sensors involved in my project. There is a DHT11 temperature and humidity sensor, two SEN0017 light sensors and an ultrasonic sensor. The DHT11 is simply used for readings but the SEN0017s are used for navigating the NaviCar and following a line.

## Temperature Sensor

Description of temperature sensor and how you’re using it. Include both the hardware and software.

Figure 6‑1 Temperature sensor circuit

Photographs are not circuit diagrams. Do use photographs to enhance a report, but not as a replacement for circuit or other engineering diagrams.

A diagram of a circuit

Description automatically generatedA circuit board with wires

Description automatically generated

V

Figure 6‑2 A photograph is not a replacement for a circuit diagram

To describe your software, it is not sufficient to just paste some code in here, you should describe what your code is designed to do, in English. Write out any mathematical equations or calculations and explain them. If you decided to code using functions, libraries and/or classes, describe this. A good layout is to include a snippet of code alongside its explanation. You do not have to explain every part of your code, pick the important parts.

Include any simulations you did.

**The following is an example of a Temperature sensor description**

The temperature sensor used in this project is the LM35 Precision Centigrade Temperature Sensor. [3]

Diagram

Description automatically generated

Figure 2‑1 Temperature sensor circuit

This sensor is used to determine the heat of the water when it’s pumped to the coffee. The data acquired from the sensor will be used by the microcontroller to determine when to stop the water heater.

The DHT11 temperature and humidity sensor is used to measure both variables and by uploading them to the cloud with Thingspeak I can embed graphs on my webpage. The DHT11 works by using a thermistor and capacitive humidity sensor for measuring the temperature and humidity.

Figure 4‑ DHT11 temperature sensor circuit

### Temperature Sensor Code

Text, letter

Description automatically generated

Figure 2‑2 Temperature sensor constructors

Two constructors are used in the temperature sensor class. A parameterized constructor and a default constructor.

The default constructor sets the pinout to pin 24 which I have been using in my project.

For flexibility the parametrized constructor can be used to change the default pin to any other IO pin on the Arduino.

The **pinMode()** is also set in the constructors so that it wouldn’t need to be declared in main after creating an object .

Text

Description automatically generated

Figure 2‑3 Temperature sensor code

The temperature sensor returns an analog voltage between 0 and 5 volts. The **analogRead()** command is used to read this value sent to the Arduino in millivolts.

An analog to digital converter is used to read the voltage. Then we need to calculate the voltage using the ADC resolution (1024 bit).

* BITS is the ADC resolution
* val is the value read from the ADC
* VOLTAGE is the supply for the LM35 in millivolts

Once the voltage is determined, the exact temperature can be calculated. The LM35 reads 1 degree Celsius per 10mV so the mv value calculated earlier is divided by the temp per division.

Then finally the temperature is returned to the main program.

In my code the main loop pauses to allow the CPU to do other tasks. During this pause the values the DHT11 reads are uploaded to Thingspeak and displayed on my webpage. DHT.read(DHT11\_PIN) reads pin 33 which is attached to my DHT11 and uses two variables temp and humi to store temperature and humidity. Thingspeak.setField() tells Thingspeak where to store the values(i.e. field 1 or 2 for separate graphs) and I input my variables current reading.

A screen shot of a computer screen

Description automatically generated

Figure ‑ Temperature sensor circuit

I have also created three functions for my DHT11. Two to pass strings of the temperature and humidity onto my webpage and a temperature that I pass to the ultrasonic for a calculation I will go further into at the ultrasonic section.

A screen shot of a computer program

Description automatically generated

Figure ‑ Temperature sensor circuit

## SEN0017 Sensor

The SEN0017 sensor can detect light and returns either a 1 or 0 for the value (i.e., 1 for light and 0 for dark) This data is used to tell if the sensor is over the line or not. If not the motors are told to turn left or right depending on which sensor reads that it is not on the line. Interrupts are used to achieve this.

Figure 4‑2 SEN0017 sensor circuit

### SEN0017 Code

Below is an example function of the SEN0017 code which prints the value read for test purposes. The interrupts used will be explained further later. The integers r1 and r2 hold the readings and digitalRead() reads the two pins the SEN0017s are attached to. The values are then printed to the screen.

A screen shot of a computer code

Description automatically generated

## HC-SR04 Ultrasonic sensor

The Ultrasonic sensor is used to measure the distance of the NaviCar from obstacles. There are an echo and trigger pin that receive and send out small pulses which reflect off of objects and come back to the sensor. The sensor takes the time taken for the pulse to return and the speed of sound to get the distance travelled. This distance is halved in order to get the distance of an object from the sensor.

Figure 4‑3 Ultrasonic sensor circuit

### Ultrasonic sensor code

It is passed the live temperature from the DHT11 and uses the formula: v=331+(0.6T) to measure the speed of sound in air. We can estimate the speed, but I would prefer to pass the live temperature(T) and use the formula to attempt to get a more accurate distance. The formula: (duration/2) \* (speed of sound/10000) is used to calculate the distance from an object.

A screenshot of a computer program

Description automatically generated

# GPS Ultimate V3.

## GPS Code

# Interrupts

If you use interrupts, include a section on them, or alternatively incorporate your discussion on interrupts with another part of your report dealing with the feature that them, for example the PIR sensor section.

I use interrupts to get the NaviCar to adjust course on the line it is following. The SEN0017s work on a falling edge where if they were to go from reading a 1 to 0 then an interrupt occurs.

I use attachInterrupt() to attach the interrupt to a pin, state the interrupt service routine (i.e. turn left/right to go to the line) and the condition for the interrupt to enact (i.e. falling, rising etc)

A screen shot of a computer program

Description automatically generated

# Motors

If your project has motors, put this section in to explain. Describe both hardware and software in subsections. Include a subsection on your motor’s power requirements.

I have several types of motor on my project. I have two DC motors and a servo motor. The DC motors drive the NaviCar, and I have several functions written to control them. The Servo motor rotates the ultrasonic if need be and has multiple of its own functions.

## DC Motors

My DC motors have several functions to go forward, reverse, left or right. The motors drive the chassis and is connected to an L293D motor driver. The motors would damage the ESP32 if directly connected as they use so much current, so they require a driver to operate. The motors also have a separate power supply as they take up more than the 3.3V the ESP32 supplies all other components.

Figure 7‑ DC motors circuit

## Servo Motors

The servo motor operates by being fed a pulse of variable width. If the pulse if 20ms and it is high for 1ms and low for 19ms the servo could be held at 90 degrees, for 2ms high it could be 180 etc. The duration of the high pulse tells us how far the servo is rotated and held at that specific degree. I have code to turn the servo left and right.

Figure 7‑2 Servo motor circuit

## Motor Control Code

The servo motor control code works by using for loops. Servo1.write tells the servo a number between 0-180 degrees to put the servo to, therefore a simple for loop turns the servo left or right.

A computer screen shot of a program code

Description automatically generated

The DC motors each have three pins to control each motor. The enable pin turns them on, this has to be high for the motor to do anything, and there are two other pins which tell the motor to go forward or back (i.e., one high and the other low mean go forward).

# 3D printed components

I have had to sketch and print two components to mount my hardware to for the NaviCar to not only look better but be easier to build and carry. This problem arose when I saw how many components, I had to neatly fit on my chassis several weeks ago.

Figure 8‑ Component concept sketches

Figure 8‑2 Component 3D models

## Designing the components

I first designed a holder for the ultrasonic to be mounted to the servo motor. This was easier to design and draw. I then created a design for my SEN0017s and servo motor to fit to. These are easily placed on my chassis and worked well the first time I printed them.

# Web Server

My ESP32 hosts a webserver that my webpage is sent to as a message containing functions and raw string literals. I have an ssid and password in a separate header file.

A screen shot of a computer program

Description automatically generated

## Wi-Fi

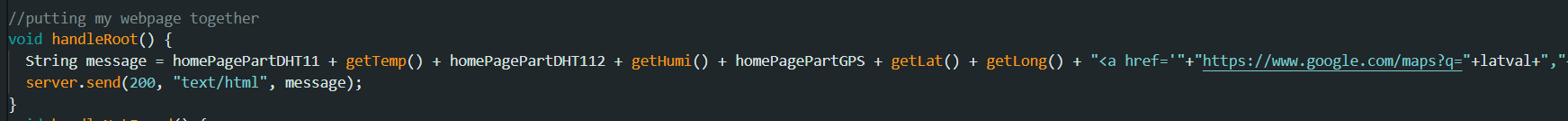
The ESP32 has built-in Wi-Fi capabilities that I use to connect and host a webserver on using a mobile hotspot.

## Web Server

A Webserver is hosted on the ESP32 using an ssid and password (the name and password of my hotspot). The ssid and password are in a separate header.

## HTML & CSS Code

I wrote the HTML/CSS code to split the website into rows of information. There are headings and data given for each row and it shrinks and grows with the webpage. A header file called homepage.h holds my html code as it is split into a message as seen below where segments of my webpage and function calls are added together.



## Web accessibility

Detail what you have included to ensure that your website is accessible.

You could break it down to the following headings:

* Perceivable - Web content is made available to the senses - sight, hearing, and/or touch
* Operable - Interface forms, controls, and navigation are operable
* Understandable - Information and the operation of user interface must be understandable.
* Robust - Content must be robust enough that it can be interpreted by a wide variety of user agents, including assistive technologies

Perceivable:  
e.g. For Screen readers to know what language to use:  
<html lang="en" >   
All links and alt tags were given meaningful names e.g.   
<p><a href="http:some big long address" >Make link text meaningful</a></p>

<p><img src="/image path.png" alt="Include meaningful description/make empty if unnecessary"></p>   
etc….

# Problem Solving

Describe the major problems / challenges encountered during your project, and how you solved them. Highlight your approach to problem solving and the steps you took to successfully solve problem(s).

I have two examples of my problem solving from within this project.

I have had to create my own 3D models on Onshape in order to house all of my components neatly and in a confined space (mounted to my chassis).

I connected my SEN0017s to a power supply before a Christmas demo. The code to follow a line was working and the interrupt code worked but I had to rework the demo to focus on my ultrasonic stopping the NaviCar if it got to close to a wall instead of my preferred line follower idea as I broke the ICs on the sensors by overloading current. It was a good lesson as it showed how costly simple mistakes can be.

# Impact of Project on Sustainability

Consider

* Project application

Impact of project on United Nations Sustainable Development Goals UN SDGs. Pinpoint the exact SDG targets impacted negatively or positively.

* Accessibility e.g. website accessibility. Include a screenshot.
* Power Budget
* Health and Safety
* Programming Style
* Component reuse and recycling
* Plagiarism/referencing

The project aims to tackle the U.N. Sustainable development goals (SDGs) 3 and 11 which focus on good health and wellbeing as well as sustainable cities and communities. Indicator 3.6 which is to road deaths and accidents by 2030 is targeted by the NaviCar as its self-navigating technology aims to provide safer transport than a human driver would as humans are prone to mistakes. Indicator 11.2 is to provide access to safe, affordable accessible transport as well as improve road safety for all. The NaviCar’s ability to follow strict routes will improve road safety in any given city. The components on the NaviCar are also reusable as the components are modular and will in time be taken off the breadboard and used for other projects.

# Conclusion

Write a short conclusion. What is the outcome of the project. Perhaps you have a product prototype, or some analysis of data from sensors. Keep it focussed on what you have done. You can mention future opportunities for the work but keep this part short.

The NaviCar can tell you where you are, how hot and humid you are, tell you if you are about to hit something and lead you along a specified line. This one project does all of that as well as host a webpage with all the information measured and allows us to see insights into the self-autonomous future. The NaviCar is the culmination of a lot of effort, but all worth it in the end thanks to the final product.

# References

|  |  |
| --- | --- |
| [1] | Espressif, "ESP32\_Datasheet," 2022. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32\_datasheet\_en.pdf. [Accessed 4 April 2022]. |
| [2] | Mathsworks, "IoT Analytics - Thingspeak Internet of Things," [Online]. Available: https://thingspeak.com/. [Accessed 28 March 2023]. |
| [3] | Texas Instruments, "www.ti.com," 1999. [Online]. Available: https://www.ti.com/lit/ds/symlink/lm35.pdf. [Accessed 4 April 2022]. |
| [4] | Arduino, "www.arduino.cc," Arduino, 6 April 2022. [Online]. Available: https://www.arduino.cc/reference/en/libraries/servo/. [Accessed r April 2022]. |

Some example references are given above. For example, the section in your report on your servo motors should have a sentence that includes [3] in it, referring to reference [3] here.

Use the References facility in MSWord as described in the Lecture. Use the IEEE style.

Include all your Datasheets.

# Appendix 1: Code

Include all the code you have written here - your top-level Arduino .ino file, and any .cpp/.h files you created. Do not include third party code.

It is worthwhile including your website as HTML/CSS code which can be easily viewed from any browser.

# Appendix 2: Bill of Materials

Include a Bill of Materials table. State whether you got your part from ATU stores by placing a ‘y’ or ‘n’ under the ATU Stores column.  
If you can include the price of all components to show the approximate cost of your project. It’s not mandatory to have all costs.  
Mandatory sections to complete are Item, Quantity and ATU Stores.  
Don’t forget to multiply the unit cost by the number of components you use in your project e.g. the HC-05 module costs 19.45 each so the total cost for 2 is 38.90.  
You don’t need to cost resistors, breadboards, stripboard and wiring.  
It’s useful to create the table in Excel and then copy and paste here.   
Some students sourced parts themselves and you can indicate this by using the ‘\*’.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Item** | **Quantity** | **Manuf** | **Manuf No** | **ATU Stores** | **Sourced from** | **Order No** | **Cost Euros** |
| Arduino Uno | 1 | Arduino | A000066 | y | Radionics | 715-4081 | 18.86 |
| 3K3 | 1 |  |  | y |  |  | 0 |
| 1K8 | 1 |  |  | y |  |  | 0 |
| ESP8266 | 1 | Sparkfun | WRL-17146 | y | Mouser | 474-WRL-17146 | 5.89 |
| HC05 Module | 2 | Velleman | 158 | y | Robot Shop | RB-Vel-158 | 19.45 |
| Fingerprint Sensor | 1 | Adafruit | 4690 | n | Mouser | 485-4690. | 16.9 |
| LCD | 1 | Displaytech | 162KBCBW | y | Radionics | 210-9031 | 11.94 |
|  |  |  |  |  |  |  |  |

Components marked with an asterisk indicate that the student sourced this component themself.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Item** | **Quantity** | **Manuf** | **Manuf No** | **ATU Stores** | **Sourced from** | **Order No** | **Cost Euros** |
| HC-SR04 | **1** |  |  | **y** |  |  |  |
| GPS Ultimate V3 | **1** |  |  | **y** |  |  |  |
| SEN0017 | **4** | **DF Robot** | **SEN0017** | **n** | **Farnell** | **2946112** | **€19.48** |
| DC Motor | **2** | **RS Pro** | **n/a** | **n** | **Radionics** | **238-9737** | **€15.72** |
| Servo | **1** |  |  | **y** |  |  |  |
| DHT11 | **1** |  |  | **y** |  |  |  |
| ESP32 | **1** |  |  | **y** |  |  |  |
| Chassis | **1** | tbc | n/a | y | n/a | n/a | **tbc** |
| L293D | **1** | STMicroelectrics | L293D | y | STMicroelectrics | 714-0622 | **€5.12** |

Components marked with an asterisk indicate that the student sourced this component themself.

# Appendix 3: Schematic

