PDE3413 Project Logbook

Hamza Bari M00668318

Contents

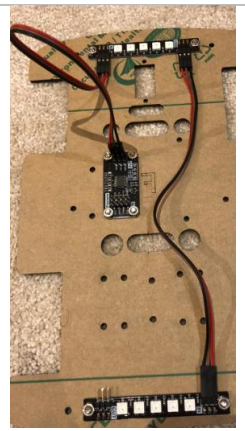
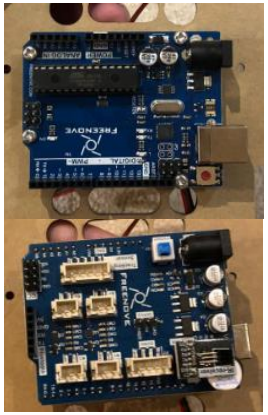
Introduction

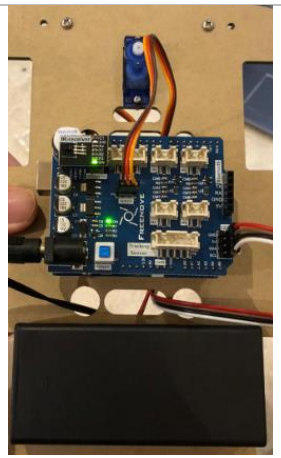
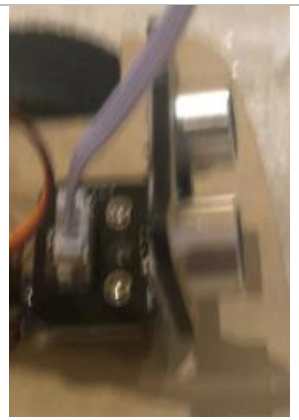
This logbook shows all the record of what I went along in duration of the project. Whatever day I worked on the project, then at the end of the day I wrote down what I did and why, did it go well or not and if not then why. These are the thoughts on the process. The appendix is used for referring to any diagrams during any of the logs.

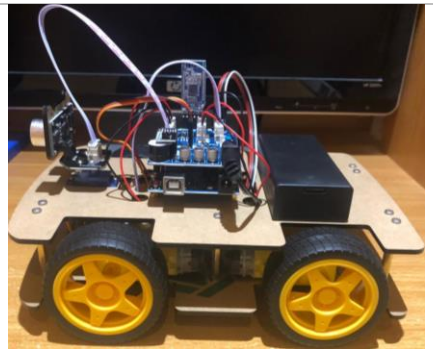
Logbook

Date & Time	***Tasks Carried Out & Why***	***What went well which works***	***What didn't go well & why***
29th December 2020 at 1PM – 3:30PM	The project task was decided such as what the robot must do. So I, decided a project which will detect the temperature in a room and then navigate to another area. This way we will find out in which are the temperate is high, low, or equal. In addition to that use an ultrasonic sensor to detect and avoid objects and obstacles in a room while the robot is navigating. The schematic diagram was also developed which is seen in appendix A1.	In appendix A1, you can witness the schematic diagram which makes it clear and gives and overview of how the project will look like once its completed. Its shows a very detailed plan in terms of how the sensors and actuators will be connected together.	In today day according to the task carried out there was nothing which didn't go well.
1st January 2021 at 1PM – 6PM	Researching on which types of sensors, were needed to be used in the project by looking at the requirements. It was decided to use the DHT11 Temperature sensor for measing the temperature and, ultrasonic sensor to avoid obstacles, and the servo motor to be used for rotating the ultrasonic sensor +90 and -90 for capturing objects in a different angle. All of the sensors which we were deciding to use for this project, were also reviewed by looking at their data sheet for looking at all the requirements and seeing if their requirements disturb anything in our project or not. Every sensor researched using data sheet seemed to be meeting the requirements. Also, have a look at the schematic diagram as well and seeing which sensors and actuators will be appropriate according to the schematic diagram.	After doing all the research we discovered which types of sensor were the best for being used in this project and those were the following: 2x Arduino Uno, 4x gear motors, 4x wheels, 2x DHT11 Temperature & Humidity sensor, 1x 4WD Arduino Uno Extension board, 1x Servo motor, 1x Ultrasonic module connector, 1x Ultrasonic sensor, 1x WS2812B_LED_controller, 2x WS2812B_LED, Double-sided and Single-Sided Jump, wire pins for sensor and actuators connections, 2x 18650 3.7V batteries, Batter holder with power jack connection to the Arduino Uno. Therefore, now we have a plan in terms of what to do.	In today day according to the task carried out there was nothing which didn't go well.

1st February at 2PM – 4:30PM	Today I was working on the bottom Acrylic sheet of the robot fixing the sensors and actuators which were required to be fixed in their current positions. Developing the robot design. Adding geared motors into the Acrylic sheet using the screws and nuts for tightening them and keeping them fixed and strong. Adding the wheels into the geared motors which would later be used for the robot navigation. While inserting the wheels the hammer was used to slightly push the wheel into the motor therefore, it stays tight. Although tracking sensor was not part of this project but, it came with the package so it was sensible to fix that in as well because, the holes left in the Acrylic sheet can cause issues. Therefore, the tracking module was fixed on its point tightly using the screws.	 As, you can witness the geared motor is tightened to the Acrylic sheet thuglty using the screws and nuts. The wheel is strongly inputted into the geared motor.  Tracking module fixed tightly into the Acrylic sheet. All sensors in the bottom acrylic sheet were fixed to their expected points.	There were no major issues which led to something going completely wrong but, in duration there were slightly some small problems which were fixed such as, the nuts being fixed into the geared motor were in the opposite direction which stopped the geared motor from being tightened any further as, a result when this issue was checked, then it was solved instantly and the nuts could tighten the geared motors. Also another small issue was that some of the screws which were being used to tighten were too small which was not tightening the geared motors but that issue was instantly found in a matter of seconds and, it was instantly fixed.

2nd February at 2PM – 6PM	Today I was working on the top Acrylic sheet of the robot fixing the sensors and actuators which were required to be fixed in their current positions. The LED controller and LED stipes were fixed into the location point and then fixed using screws. Using the robot schematic diagram from the design the LED stipes were connected to the LED controller using the jump wires. Therefore, when the Arduino gets powered, the LED will be available to flash from their right position where it can be seen from the users. The Acrylic sheet is flipped and the Arduino Uno controller has been fixed into its location. After that, the 4WD extension board has been connected on top of the Arduino controller to input the sensors into the port which correspond to the Arduino pins. In simple words, for the connections of the sensors. Same way the battery holder, servo motor has been fixed into their points using screws and, using the schematic diagram they have been inputted into the extension board ports using the jump wires. The ultrasonic module is connected to the servo motor. The ultrasonic sensor pins have been connected to the ultrasonic module connector pins. In today's date the sensors going onto the acrylic sheet two were all installed and the Arduino power was tested which as well you can see its working.	 As, you can witness that the LED controller and LED strips are tightly fixed using screws to the acrylic sheet.  The Arduino has accurately been fixed into its location using the screws, and the 4WD extension board has been connected perfectly to the Arduino Uno.	The fixing the servo motor was to lose as a result, the screws came out therefore, they were fixed using black tape which also has disadvantages as it could melt in too much heat.  There was a slight issue as, the screw wasn't going through when fixing the ultrasonic module connector. Therefore, we had to use a hammer to push it in hard which is not good as, that could have damaged the ultrasonic connector. In the image below it can be witnessed that the ultrasonic module has been fixed but, still its very lose and that's a danger as it could potentially come out when the robot is navigating and especially the twisting using servo motor constantly can make the screws even weaker.

		 As, you can see from the image above that battery power jack has been inserted into the extension board power jack, and the servo motor has been fixed and they look strongly tight. Another important thing you can witness from this image above is that after all the sensor connections the Arduino power is turned on using which shows us the green light meaning, that the Arduino is getting power.  The LED controller is working after the Arduino is getting power which shows the sign that the	

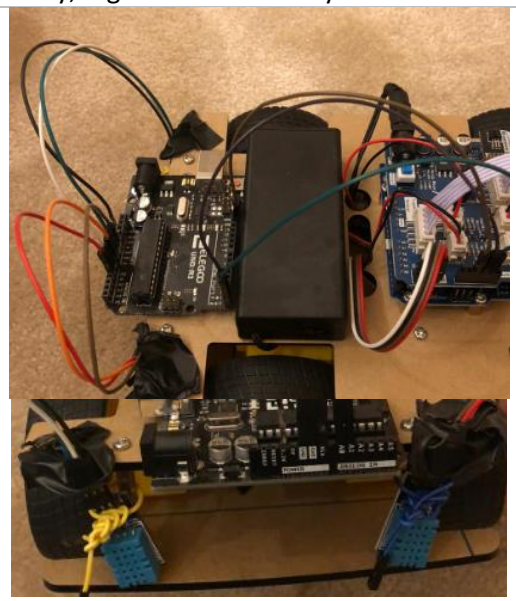
		sensors are working after they are connected to the Arduino that is powered.	
3rd February at 2PM – 3:30PM	In today day the first task was to fix both acrylic sheets together using the nuts. This way the hardware design is completed in terms of where all sensors are going to be fixed. You can witness the image below and see what I mean by that.  	 In the image above you can see that the hardware design is completed as both acrylic sheets have been fixed together using the nuts. In addition to that all the connections have been made such as from sensor to Arduino using jump wires. When the Arduino power is turned on the sensor lights were working. In today's date the hardware implementation was successfully completed.	In today day according to the task carried out there was nothing which didn't go well.

	In the image above it can be seen that the nuts were used for connecting both acrylic sheets together. Once, the hardware design has been completed, then using the jump wires the sensors were connected, from sensor to the port in the extension board. In appendix A1 you can witness that a robotic system schematic diagram has been developed, referring to that diagram is how all the connection were made in the robot using the jump wires but, excluding the second Arduino and the DHT11 sensors are they were yet to be installed in a later date. After that, the power was turned on to see if all the sensor lights were working or not, this means here I was identifying a mistake which can also be known as a small pre-testing approach.		
7th February 2021 at 1PM – 12AM	In today's date the task carried out were the following: connecting the 2x DHT11 temperature sensors to the second Arduino as well as 1x Bluetooth module. In addition to that, connect the second Bluetooth module to the main robot extension board. The reason for this is because, the using the Bluetooth module a wireless communication can be created between both Arduino therefore, the temperature values are sent from the Arduino to the main robot Arduino.	In the second Arduino both DHT11 temperature sensors were connected to the analogue pins and, then the code was written in the Arduino to test if both DHT11 temperature sensors were working or not therefore, when the code runs both DHT11 sensors print out the temperature sensor values in the serial monitor which means, that the temperature sensors are working fine and the code is working fine as well and, the connection is correct such as the pins in which the DHT11 temperature sensors have been added inserted into the analogue pins.	Another task today was to connect HC05 and HC06 Bluetooth modules together to develop a serial communication. Therefore, the HC06 was connected to the extension board Bluetooth port and the HC05 was connected to the second Arduino using the analogue pin. After that, we found out that both, Bluetooth modules weren't connection together. I tried various different techniques by researching online, looking at similar schematic diagram to apply the same solution but that didn't seem to work.

			The main technique was to use the BT serial to create a Bluetooth communication in the temperature code, which was working alright up to this point but, when the step came to configure in the main robot code where the AT+Address command needed to be used which gets the address of the Bluetooth module from the temperature sensor Arduino and, then that address is used to connect both Arduinos together for making a wireless serial communication. The problem here was that the AT+Address command wasn't responding as, a result this Bluetooth process couldn't move any further. The other alternative method which I tried was starting both Arduino serials in in one code therefore, it can get sensor values from both Arduino but, the issue here was that Arduino Uno doesn't allow double serials in their code. Therefore, wouldn't work as well. We decided to use a database as well to exchange data but, that issue was immediately scrapped as, database is in you local machine as, result it

			wouldn't work without the USB cable connection. This took the whole day as I used a various different techniques but, no solution was found today.
8th February 2021 at 1PM – 7PM	As, we had known from yesterday that the wireless communication didn't work between 2 Arduinos. Therefore, further research on how to connect both Arduinos in any sort of way will be taken place today.	After reading many online articles are watching videos, we discovered a point that if you have RX and TX pins in both Arduino, then you can connect those together to create a serial communication. Therefore, we discovered that on the extension board there are RX and TX pins on the Bluetooth module pins. Therefore, if we remove the Bluetooth module since, we don't need it and we use its TX and RX pins, then we can be in a situation where both Arduino can exchange data to each other. In the image below you can witness the RX and TX pins is the extension board.  Once, they were connected together using the using the RX -> TX, TX -> RX, in addition the ground. You can see the image below of how they were connected. 	The issue today was that the string was being sent fine but, there was a problem with sending the numbers using the serial communication as, the serial write doesn't work for integers and read bytes doesn't allow serial print, as its only for serial write. Therefore, I also tried string to number conversions such as converting the number to string and send it using serial write in the temperature sensor code and then, when receiving it in the main Arduino code then, convert back to a number. The issue here was that it would still work as it was sensing black values and not sure what the reason for that was, also tried adding delays to see maybe it would but it didn't work. Therefore, this is the issue I was stuck at till today date.

		After that, using the temperature sensor code a string message was using the serial.write("Hello") was sent to the main robot Arduino. After that, in the main robot Arduino code using the read byte we were able to receive the message "Hello" into the main robot Arduino and it was printed using the serial monitor which means, there was now a way to exchange data between the two Arduinos.	
9th February 2021 at 1PM – 6PM	Today more reasons were done in serial communications using the online sources. Especially in the areas how to exchange numbers in the serial communication.	After researching we came to a solution that if we use: if **(Serial.available() > 0) {};** function in the main robot code then, it would also print the integer values from the temperature sensor code, and serial print can be used as, in serial available there is no requirement for serial write. `if (Serial.available() > 0) {` `  tempSensorOne = Serial.read();` `  Serial.println(tempSensorOne);` `  Serial.println("");` `  delay(2000);` `  tempSensorTwo = Serial.read();` `  Serial.println(tempSensorTwo);` `  Serial.println("");` `  delay(2000);` `}` In the code screenshot above, you can witness that using the serial available it would receive data from the serial communication, which is the second temperature sensor Arduino, then it uses	In today day according to the task carried out there was nothing which didn't go well.

		the read byte to get the first value coming, which is the temperature value of the DHT11 sensor coming from the second Arduino, that value is passed into the integer variable, when the value is read using serial read. After that, the integer where the value is stored will be printed to the serial monitor and that variable will be used in the robotic algorithms. The same approach applied for the second temperature sensor value as well. So, it goes first value and then the second value and then first back again. Basically, it goes around in a cycle.	
12th February 2021 at 1PM – 2:30PM	In today task the configured second Arduino with the DHT11 temperature sensor will be fixed onto the main robot therefore, its one system all together when you look at it.	 In this image above, you can see that the Arduino is sticked to the acrylic sheet and, the DHT11 Sensors are tied to the back nuts therefore, it puts everything together and it	In today day according to the task carried out there was nothing which didn't go well.

		terms of design this is one completed robot now in terms of the hardware.	
13th February 2021 at 1PM – 3PM	In today's day we tested all the sensors, actuators, and the wheels by uploading code to check if they are all working fine or not. Basically testing small functionalities to check.	When the code was uplaoded onto each sensor they all were responding meaning there was no erros the connections or the sensors, actuators, and the wheels. As, the servo motor was rotating when the code was uploaded, the wheels were able to take a left and a right turn as well as going straight, the temperature values were being received, the ultrasonic sensor was able to detect an object and based on that the robot was able to move away using its wheels. This shows us that our sensors, actuators, and the wheels are working fine meaning that we can start working on our main algorithms to perform different tasks according to the temperature values in a room. The reason why this test was useful is because, if there are any problems later then, it wouldn't be with the hardware components as, it would be with the software algorithms.	In today day according to the task carried out there was nothing which didn't go well.
20th February 2021 at 12PM – 9PM	In today's day the task we carried out was the programming and we started writing the algorithm for the motor run. This function will be used for controlling the robotic geared motor which are the wheels. This algorithm can be used in other algorithms when setting out a navigation path for the robot. This algorithm can be used for moving the robot using it wheels forwards, backwards, left, and right.	Once, the motor run algoruthm has been written, then it called in the loop setup function onto the Arduino code. In addition to that the number for direction and speed were set, and when the program started, the the robot was able to move into the different directions which, means that using this method the robot is able to move. This method can now be used for developing a naviagtion path for the robot to change area in a room such as move back, turn left, move forward, and then turn right.	The issues we had in this algorithm was that when, the motor run method was written we were missing the digital write for setting the direction and speed for each motor. There was a confusion as, I thought I have defined them above therefore, you don't need it but, after doing a trial and error for 2 hours I discovered that, they are needed to be here as well as being set with the

	`` #define PIN_DIRECTION_RIGHT 3 #define PIN_DIRECTION_LEFT 4 #define PIN_MOTOR_PWM_RIGHT 5 #define PIN_MOTOR_PWM_LEFT 6 ``` In the image above you see that the pins conencted to the motor gears have been defined. They have been given a name for exmaple PIN_DIRECTION_RIGHT is a varible name for the right motor wheel. ``` void motorRun(int speedl, int speedr) { int dirL = 0, dirR = 0; if (speedl > 0) { dirL = 0; } else { dirL = 1; speedl = -speedl; } if (speedr > 0) { dirR = 1; } else { dirR = 0; speedr = -speedr; } digitalWrite(PIN_DIRECTION_LEFT, dirL); digitalWrite(PIN_DIRECTION_RIGHT, dirR); analogWrite(PIN_MOTOR_PWM_LEFT, speedl); analogWrite(PIN_MOTOR_PWM_RIGHT, speedr); } ``` The algorithm has been written for the motor run. Therefore, at this stage the algorithm has been written. The two parameter will be passed, for the speed and direction of movement.		direction and speed calculated earlier in the if statements. ``` digitalWrite(PIN_DIRECTION_LEFT, dirL); digitalWrite(PIN_DIRECTION_RIGHT, dirR); analogWrite(PIN_MOTOR_PWM_LEFT, speedl); analogWrite(PIN_MOTOR_PWM_RIGHT, speedr); } ```
21st February 2021 at 12PM – 1:30PM	Today the code for the LED controller and the Arduino buzzer was written. ``` #include "Freenove_WS2812B_RGBLED_Controller.h" ``` The library for the LED has been included.	Once the main method runs in the Arduino, then the LED lights are flashing red meaning, that the LED code is working alright and its ready to be used for alerting users is the temperature is high, low, or equal.	The only issue was with the buzzer as the sound was going on for a very long time, as a result the delays were added to make an appropriate sound.

	``#define I2C_ADDRESS 0x20 #define LEDS_COUNT 10 //it defines number of lEDs.``` ```Freenove_WS2812B_Controller strip(I2C_ADDRESS, LEDS_COUNT, TYPE_GRB); //initialization``` The LED controller strip has been initialized with the number of addresses and LED counts which is used for setting the LED light colour onto the LEDs. ```strip.setAllLedsColor(0xFF0000); //Set all LED color to red``` After that, using the strip the colour code red has been added into the main method. ```#define PIN_BUZZER A0``` This buzzer has been initialized. ```digitalWrite(PIN_BUZZER, HIGH); //turn on buzzer delay(50); digitalWrite(PIN_BUZZER, LOW); //turn off buzzer``` The buzzer defined property has been passed into the digital write and using the HIGH, the buzzer is turned on, and using the LOW the buzzer is turned off. The delay is used for starting and stopping the buzzer and making a specific sound.	When the buzzer was tested, it was able to make a sound when the code was executed which means this can be used for alerting users when, the navaigation path has started and ended.	
28th February 2021 at 12PM – 8PM	Today is the code written for one of the main task which was: how will the robot respond when the temperature is high. This is where I will create the navigation path, set the LED lights, and set the buzzer sound.	After the higher temperature method was written, then that method was tested in the main method. Therefore, the robot was able to navigate accroding to what navigation tasks were set inside the method using the motor run algorithm. The LED lights are also set to red while the robot is navigating and the buzzer does sound accroingly to the code when the navigation path ends.	The only issue we had here was when the code started running, then the robot navigation path was going on non-stop meaning the robot wasn't getting any break and this wasn't a good sign as it could affect the battery and make the robot crash as it may go out of our control. Therefore, we added short pauses using delays. This way the robot was able to stop and then, carry out the

```
void TempHigher() {
  //Move back
  motorRun(-100, -100);
  delay(1000);
  motorRun(-100, -100);
  delay(1000);
  motorRun(-100, -100);
  delay(1000);
  motorRun(-100, -100);
  delay(1000);
  motorRun(0, 0);

  //Left motors rotate to one direction
  digitalWrite(PIN_DIRECTION_LEFT, HIGH);
  analogWrite(PIN_MOTOR_PWM_LEFT, 100);
  delay(1000);
  //Stop
  analogWrite(PIN_MOTOR_PWM_LEFT, 0);
  delay(1000);

  //left motors rotate to opposite direction
  digitalWrite(PIN_DIRECTION_LEFT, LOW);
  analogWrite(PIN_MOTOR_PWM_LEFT, 255);
  delay(1000);
  //Stop
  analogWrite(PIN_MOTOR_PWM_LEFT, 0);
  delay(1000);

  //Move forward
  motorRun(80, 80);
  delay(1000);
  motorRun(80, 80);
  delay(1000);
  motorRun(0, 0);

  //Left motors rotate to one direction
  digitalWrite(PIN_DIRECTION_LEFT, HIGH);
  analogWrite(PIN_MOTOR_PWM_LEFT, 100);
  delay(1000);
  //Stop
  analogWrite(PIN_MOTOR_PWM_LEFT, 0);
  delay(1000);

  //left motors rotate to opposite direction
  digitalWrite(PIN_DIRECTION_LEFT, LOW);
  analogWrite(PIN_MOTOR_PWM_LEFT, 255);
  delay(1000);
  //Stop
  analogWrite(PIN_MOTOR_PWM_LEFT, 0);
  delay(1000);

  //Move forward
  motorRun(130, 130);
  delay(1000);
  motorRun(130, 130);
  delay(1000);
  motorRun(0, 0);

  delay(2000);

  digitalWrite(PIN_BUZZER, HIGH); //turn on busser
  delay(50);
  digitalWrite(PIN_BUZZER, LOW);  //turn off busser
}
```

next navigation moves. This way the battery is saved and the robot is in more control.

	Above, is where you can witness the code written for what happens when the temperature received is high.		
1st March 2021 at 12PM – 1PM	Today is the code written for one of the main task which was: how will the robot respond when the temperature is lower. This is where I will create the navigation path, set the LED lights, and set the buzzer sound.	After the lower temperature method was written, then that method was tested in the main method. Therefore, the robot was able to navigate accroding to what navigation tasks were set inside the method using the motor run algorithm. The LED lights are also set to green while the robot is navigating and the buzzer does sound accroingly to the code when the navigation path ends.	In today day according to the task carried out there was nothing which didn't go well.

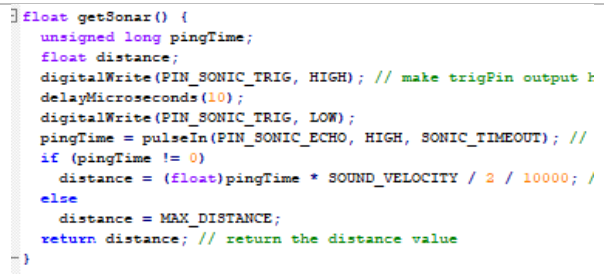
```
void TempLower() {

  //Move back
  motorRun(-100, -100);
  delay(1000);
  motorRun(-100, -100);
  delay(1000);
  motorRun(-100, -100);
  delay(1000);
  motorRun(-100, -100);
  delay(1000);
  motorRun(0, 0);
  delay(1000);
  motorRun(0, 0);
  delay(1000);
  motorRun(0, 0);

  //Right motors rotate to one direction
  digitalWrite(PIN_DIRECTION_RIGHT, HIGH);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 100);
  delay(1000);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 0);
  delay(1000);

  //Right motors rotate to opposite direction
  digitalWrite(PIN_DIRECTION_RIGHT, LOW);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 255);
  delay(1000);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 0);
  delay(1000);

  //Left motors rotate to one direction
  digitalWrite(PIN_DIRECTION_LEFT, HIGH);
  analogWrite(PIN_MOTOR_PWM_LEFT, 100);
  delay(1000);
  //Stop
  analogWrite(PIN_MOTOR_PWM_LEFT, 0);
  delay(1000);

  //left motors rotate to opposite direction
  digitalWrite(PIN_DIRECTION_LEFT, LOW);
  analogWrite(PIN_MOTOR_PWM_LEFT, 255);
  delay(1000);
  //Stop
  analogWrite(PIN_MOTOR_PWM_LEFT, 0);
  delay(1000);

  //Move forward
  motorRun(80, 80);
  delay(1000);
  motorRun(80, 80);
  delay(1000);
  motorRun(0, 0);

  //Move forward
  motorRun(130, 130);
  delay(1000);
  motorRun(130, 130);
  delay(1000);
  motorRun(0, 0);

  delay(2000);

  digitalWrite(PIN_BUZZER, HIGH); //turn on buzzer
  delay(50);
  digitalWrite(PIN_BUZZER, LOW);  //turn off buzzer
}
```

	Above, is where you can witness the code written for what happens when the temperature received is low.		
2nd March 2021 at 12PM – 2PM	Today is the code written for one of the main task which was: how will the robot respond when both temperatures are equal. This is where I will also create the circle/round navigation path around the room to alert that the temperature is equal, set the LED lights, and set the buzzer sound.	After the equal temperature method was written, then that method was tested in the main method. Therefore, the robot was able to navigate accroding to what navigation tasks were set inside the method using the motor run algorithm which was moving the robout around a cirlce using the right digital motor pins. The LED lights are also set to blue while the robot is navigating and the buzzer does sound accroingly to the code when the circle/round navigation path ends.	In today day according to the task carried out there was nothing which didn't go well.

```
void MakeACircle() {

  //Right motors rotate to one direction
  digitalWrite(PIN_DIRECTION_RIGHT, HIGH);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 100);
  delay(1000);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 0);
  delay(1000);

  //Right motors rotate to opposite direction
  digitalWrite(PIN_DIRECTION_RIGHT, LOW);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 255);
  delay(1000);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 0);
  delay(1000);

  //Right motors rotate to one direction
  digitalWrite(PIN_DIRECTION_RIGHT, HIGH);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 100);
  delay(1000);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 0);
  delay(1000);

  //Right motors rotate to opposite direction
  digitalWrite(PIN_DIRECTION_RIGHT, LOW);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 255);
  delay(1000);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 0);
  delay(1000);

  //Right motors rotate to one direction
  digitalWrite(PIN_DIRECTION_RIGHT, HIGH);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 100);
  delay(1000);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 0);
  delay(1000);

  //Right motors rotate to opposite direction
  digitalWrite(PIN_DIRECTION_RIGHT, LOW);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 255);
  delay(1000);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 0);
  delay(1000);

  //Right motors rotate to one direction
  digitalWrite(PIN_DIRECTION_RIGHT, HIGH);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 100);
  delay(1000);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 0);
  delay(1000);

  //Right motors rotate to opposite direction
  digitalWrite(PIN_DIRECTION_RIGHT, LOW);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 255);
  delay(1000);
  analogWrite(PIN_MOTOR_PWM_RIGHT, 0);
  delay(1000);

}
```

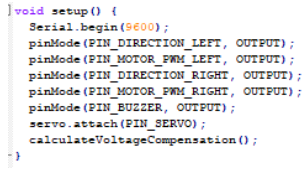
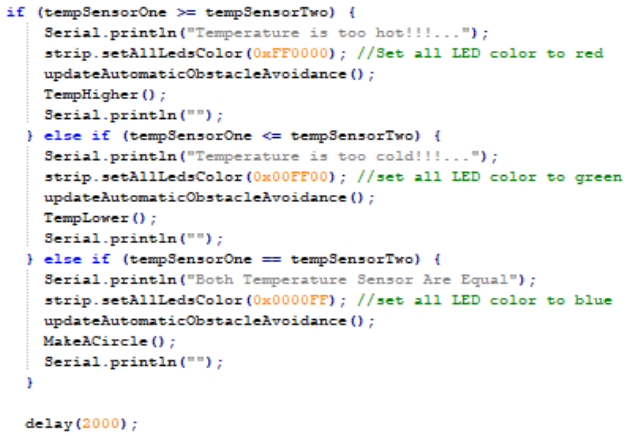
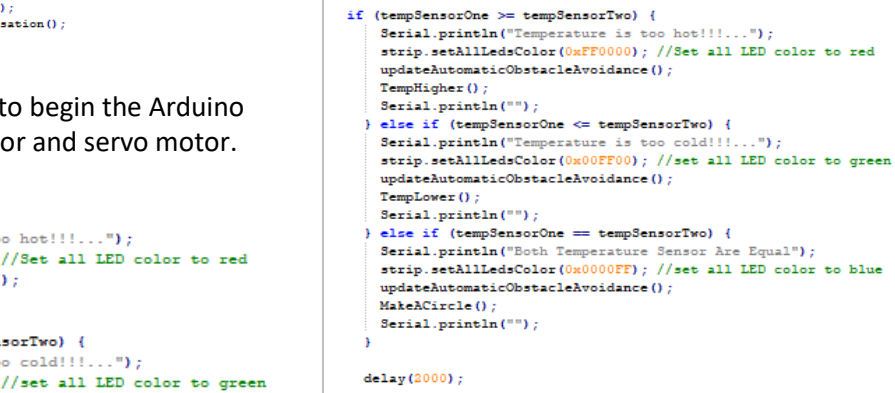
The circle method was written using the predefined digital write, that was hardcoded for the left and right

	turns of the robotic wheels. That's why the motor run method hasn't been used in the code shown in the image above. Above, is where you can witness the code written for what happens when both temperatures are equal.		
3rd March 2021 at 10AM – 3AM	In today's date three task have been completed. The first algorithm is developed for calculating the distance between the ultrasonic sensor and the obstacle/object. The other task is to develop an algorithm for obstacles avoidance during the navigation which also used the motor run method. This algorithm is used for avoiding obstacles/object during the navigation of the robot. Finally, configure the servo motor and use that in the obstacle avoidance algorithm for rotation the ultrasonic sensor +90 and -90. `#define PIN_SONIC_TRIG 7` `#define PIN_SONIC_ECHO 8` `#define OBSTACLE_DISTANCE 40` `#define OBSTACLE_DISTANCE_LOW 15` `#define MAX_DISTANCE 300 //cm` `#define SONIC_TIMEOUT (MAX_DISTANCE*60)` `#define SOUND_VELOCITY 340 //soundVel` The variables for the ultrasonic sensor pin configuration, maximum distance, obstacle distance has been declared. These values will be used in both algorithms.	The algorithm for calculating the distance between the ultrasonic sensor and the obstacle/object was working fine as, when the algorithm was tested it was returning the distance between the ultrasonic sensor and the obstacle/object. Therefore, this algorithm was capable enough to be used in the obstacles avoidance algorithm. The servo motor has been configured and tested and its returning the value when the servo motor is rotating therefore, it can be used in the obstacles avoidnce algorithm. The obstacles avoidnce algorithm has just been tested itself. It has been tested independently as, it hasn't been integrated with the other methods and algorithms. Therefore, independently itself the obstacles avoidnce algorithm is working as its gets the Get-Sonar method to get the distance between, the ultrasonic sensor and the obstacle/object of wherever the object is pointing towards. Also, the servo is being able to be rotated using the servo angle. Based on that the if statements have been created which decide where the robot will navigate if an object is seen in this certain angle. Mathematical functions are used	It was quite difficult set the speed and direction according to the distance calculated. That is what took the most time to implement this obstacles avoidnce algorithm. As, example such as if the distance is greater than the object distance then, setting the motor run such as then, where the robot should turn such as go backward or turn left or right. This was very difficult to understand. Therefore, a lot of trial and error was done to make this accurate after that, it managed to work because, at the start it wasn't perfect and was hitting some of the obstacles/objects at some points.

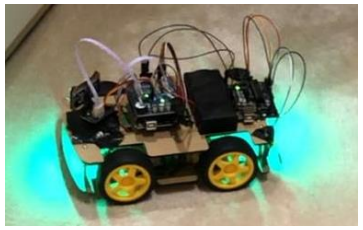
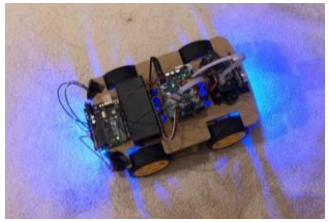
	 The algorithm for calculating the distance between the ultrasonic sensor and the obstacle/object.	for calculating the distance using distance from the Get-Sonar method for example, if the distance is greater then, turn the motor wheels backward. Therefore, this algorithm independently is able to detect objects and, avoid the robot to hit those objects and automatically change paths.	

```
void updateAutomaticObstacleAvoidance() {
  int distance[3], tempDistance[3][5], sumDisntance;
  static u8 leftToRight = 0, servoAngle = 0, lastServoAngle = 0;  //
  const u8 scanAngle[2][3] = { {150, 90, 30}, {30, 90, 150} };

  for (int i = 0; i < 3; i++)
  {
    servoAngle = scanAngle[leftToRight][i];
    servo.write(servoAngle);
    if (lastServoAngle != servoAngle) {
      delay(130);
    }
    lastServoAngle = servoAngle;
    for (int j = 0; j < 5; j++) {
      tempDistance[i][j] = getSonar();
      delayMicroseconds(2 * SONIC_TIMEOUT);
      sumDisntance += tempDistance[i][j];
    }
    if (leftToRight == 0) {
      distance[i] = sumDisntance / 5;
    }
    else {
      distance[2 - i] = sumDisntance / 5;
    }
    sumDisntance = 0;
  }
  leftToRight = (leftToRight + 1) % 2;

  if (distance[1] < OBSTACLE_DISTANCE) {        //Too little distance ahead
    if (distance[0] > distance[2] && distance[0] > OBSTACLE_DISTANCE) {     //L
      motorRun(-(150 + speedOffset), -(150 + speedOffset)); //Move back
      delay(100);
      motorRun(-(150 + speedOffset), (150 + speedOffset));
    }
    else if (distance[0] < distance[2] && distance[2] > OBSTACLE_DISTANCE) {
      motorRun(-(150 + speedOffset), -(150 + speedOffset)); //Move back
      delay(100);
      motorRun((150 + speedOffset), -(150 + speedOffset));
    }
    else {                        //Get into the dead corner, move back, then tur
      motorRun(-(150 + speedOffset), -(150 + speedOffset));
      delay(100);
      motorRun(-(150 + speedOffset), (150 + speedOffset));
    }
  }
  else {                          //No obstacles ahead
    if (distance[0] <  OBSTACLE_DISTANCE_LOW) {      //Obstacles on the left fr
      motorRun(-(150 + speedOffset), -(150 + speedOffset)); //Move back
      delay(100);
      motorRun((180 + speedOffset), (50 + speedOffset));
    }
    else if (distance[2] <  OBSTACLE_DISTANCE_LOW) {     //Obstacles on the rig
      motorRun(-(150 + speedOffset), -(150 + speedOffset)); //Move back
      delay(100);
      motorRun((50 + speedOffset), (180 + speedOffset));
    }
    else {                          //Cruising
      motorRun((80 + speedOffset), (80 + speedOffset));
    }
  }
}
```

3rd March 2021 at 10AM – 12PM	In today's date the tasks which were carried out were the following: calculate voltage method and calculate battery voltage. `int speedOffset;//batteryVoltageCompensationToSpeed` The variable has been defined here which calculates the battery voltage to the speed being used by the robotic wheels. `float getBatteryVoltage() { pinMode(PIN_BATTERY, INPUT); int batteryADC = analogRead(PIN_BATTERY); float batteryVoltage = batteryADC / 1023.0 * 5.0 * 4; return batteryVoltage; }` This method gets the battery voltage, the pin mode has been set where the battery is inputted. `void calculateVoltageCompensation() { float voltageOffset = 8.4 - getBatteryVoltage(); speedOffset = voltageOffset * 20; }` This method calculates the voltage used by the robot in terms of the speed.	The get battery volatge method works meaning, it gets the battery value from the pin connected into the arduino and then, it calculates the volatge. The calculate volatge compensation method gets the battery voltage. After that, using a function the calculate volatge compensation value will be calcuted which shows us the speed off set.	In today day according to the task carried out there was nothing which didn't go well.
7th March 2021 at 10PM – 3PM	Today is where the code for the main algorithm is going to be written. Already you've seen that the values of the temperature come through the serial communication. After that, those values will be used in an if statement to make a decision if the temperature is hot, cold, or equal. These decisions will be made in a room to see if the temperature is too hot, cold, or equal therefore, using the navigation path can change the area in a room, and	`void setup() { Serial.begin(9600); pinMode(PIN_DIRECTION_LEFT, OUTPUT); pinMode(PIN_MOTOR_PWM_LEFT, OUTPUT); pinMode(PIN_DIRECTION_RIGHT, OUTPUT); pinMode(PIN_MOTOR_PWM_RIGHT, OUTPUT); pinMode(PIN_BUZZER, OUTPUT); servo.attach(PIN_SERVO); calculateVoltageCompensation(); }`	The problem with this was the obstacle avoidance algorithm which wasn't working properly in any of the if statements block. As, when the robot was navigating, then the robot was still hitting the obstacles/object on the way the ultrasonic sensor was still rotating +90 and -90 but, the

	use obstacle avoidance so the robot doesn't hit any objects in the room.  The setup method is expected to begin the Arduino serial and start the geared motor and servo motor.  This is how the decisions are made which make the robot navigate according to the temperature decision made. Temp higher, Temp lower, Circle method has all the different types of navigation methods which will execute based on the true statement as, well as the LED colour will be set according to the decision that is true. In each if statement block there is a avoid obstacle algorithm which is used for avoiding any	The setup is working as the ardiono beighs and the geared motors are started as well as the servo motor attached is working as well as the servo moves using the obstacle avoidance algorithm.  When the temperature one is greater or equal to temperature two, then the LEDs are chnaged to red. As well as the temp higher method is executing therefore, the robot is navigating to another area in the room. When the temperature one is less than or equal to temperature two, then the LEDs are chnaged to green. As well as the the temp lower method is executing therefore, the robot is navigating to another area in the room. When the temperature are both equal, then the LEDs are chnaged to blue. As well as the cirlce method is executing therefore, the robot is	robot kept on hitting the objects. Therefore, this was a broken functionality in this robotic system. Therefore, the obstacle avoidance algorithm functionality would require improvements in the second version of the development. This is where the problem is when the ultrasonic sensor captures the object it still hits the object. The obstacle avoidance algorithm still executes are tries to move the robot away from the object but, its not fast enough or the algorithm doesn't seem to be accurate as, a result the robot does crash into the object. That is why for the current robotic version its recommended to use in rooms and areas where there are less objects.

	objects/obstacles which will be in the during any of the navigation path for example, when the Temp Higher navigation path is executing with the red LEDs, then if any object comes in the way, then use the obstacle avoidance algorithm to change path so it doesn't hit the object. At the end of this day the robotic system was implemented with a broken functionality which we research later a lot but, due to time constraints that, couldn't be fixed.	navigating around in a circle within the current area of the room. Everyting what I've metioned here is working in this algorithm.	
2nd April 2021 1PM-7PM	Today I had completed the project report explaining about the project, documenting, and justifying which sensor were used, annotating the code, testing requirements, and recommending improvements.	The reports clearly discusses about the project and highlights imortant areas such as justification of why those sensors were used, in the code hat methods are used for what reason, what is the purpose of the project therefore other people can undertand the project as well and, the testing clearly shows which areas have failed and why therefore, more reaserch can be done to make those improvments for the second version and, finally reccomending furthur improvements that can also be considered for updating the robotic system.	In today day according to the task carried out there was nothing which didn't go well.
3rd April 2021 1PM-5PM	Today I had completed the demonstration of the robotic system by carrying out tasks. The tasks in terms of the temperature sensor were working very well and fine as, it detected different temperatures and executed the algorithms to perform the different navigation paths.		As you know previously that there is a problem with the obstacle avoidance as, a result that's having problems to work, as the ultrasonic sensor rotates using the servo motor but, it doesn't stop the robot from crashing.

		The image above shows that the temperatures from both DHT11 sensnors have been calcualted and the decision is made that the temperature is hot as, a reuslt its executing higher temp algorithm navigation path with red LEDs on.  The image above shows that the temperatures from both DHT11 sensnors have been calcualted and the decision is made that the temperature is cold as, a reuslt its executing lower temp algorithm navigation path with green LEDs on.  The image above shows that the temperatures from both DHT11 sensnors have been calcualted and the decision is made that the temperatures are equal as, a reuslt its executing equal temp algorithm navigation path which is making the robot navigate in a circle with the blue LEDs on.	

Appendix

A1 - Robot Schematic Diagram