ME 522 Mechatronics

Spring 2024

Final Design Report

M4

May 8, 2024

"I pledge that I have abided by the Graduate Student Code of Academic Integrity."

Luis Lara Siria, Muthathal Chinniah, Sathya Prasad Reddy Patlolla, Vignesh Pagilla

This report has been prepared by:

1. Lab Leader: Muthathal Chinniah
2. Luis Lara Siria
3. Sathya Prasad Reddy Patlolla
4. Vignesh Pagilla

Table of Contents

1. Abstract:

The objective of the project is to show an electronic device that functions based on temperature input data into the circuit. The outcome of this project is a Massage belt that can have two working modes: Hot Massage where the massage is done at a temperature set by the user and Cold Massage which is done at room temperature. This device is called the "ThermEaser". The process is done by having temperature sensors that can read temperature measurements, which are later sent to a microcontroller which decides whether to turn on the heating equipment. The user can activate the required mode, control the speed, set the required temperature of the heat massage via a Bluetooth interface in a smartphone. The results show how with the combination of automatic and manual inputs, the motors provide a massage based on the temperature and how the user can manipulate the speed.

2. Introduction:

Carrying weight on people's backs regularly can result in stress that can cause muscle injuries. These injuries can be prevented by reducing the load that a person carries every day, regular exercises to strengthen the back muscles. But after the injuries happened, applying massages to the back often can soothe the back and help the patient by reducing the pain, especially if the user uses too much weight constantly and doesn't have the proper physical strength to hold the weight. The massages can be combined with heating equipment to achieve better muscle relaxation.

The "ThermEaser" tends to provide a solution to this issue by providing a massage and a at a warm temperature. This is done by developing an electronic circuit that includes the following elements: an output display, manual data input, automatic sensor input, actuators and electromechanical, logic, processing and control unit, an audio output device. All these components will be discussed further in section 3.

The device with all the components can show the user information about the massage, have the capabilities of manipulating the speed of the massage, read temperature measurements and control the heating element based on the temperature measurements, provide a massage, and provide an audio signal once an specific temperature has been achieved and the massage is not needed any more.. In short, the device can read temperature measurements in order to decide the type of massage while the user can manipulate the speed of the massage.

For this project, one of the issues is to work with heat without having the electronics malfunctioning. Therefore, an LED is been used to represent that some temperature has been achieved that would activate the device into functioning as desired without any real heat application. Nevertheless, a heat input is still required to read a temperature measurement.

3. Design Overview:

An image of the final prototype developed is attached in Appendix B

3.1 Circuit Connection

The circuit design contains 21 components which conform to the mechanical and electronics areas. All the components are listed in Appendix A. The electronics area contains 10 components which are: Arduino Uno (logic, processing and control), LCD Screen (display), resistors (miscellaneous), TMP36 temperature sensor (automatic sensor input), Bluetooth mate silver (logic, processing and control), LED (output display), Buzzer (audio device), potentiometer (automatic sensor input), DC motor (actuators and electromechanical hardware), as shown in the schematic below.

Appendix B shows the schematic for electronic components and connections. The black lines represent ground connections, the red lines represent positive voltage connections, the green lines represent connections towards output devices, the yellow connections represent connections from input devices, and the purple connections represent miscellaneous connections. Beginning with the Arduino Uno, this has a voltage source from a USB connection which comes from a laptop. Then, it provides voltage into the circuit but directly to the potentiometer, Bluetooth module, LCD screen and temperature sensor. The Arduino also provides the ground connection to the circuit. The only analog input connection comes from the TMP36 temperature sensor in connection A5. The temperature sensor provides the temperature measurements in order to automatically decide when to turn off the massage once it reaches a specific temperature. The Bluetooth input from D2 and D3 connects the Arduino with the user interface used in a phone. The inputs include hot or cold massage and the speed of the massage.

For the outputs, there are several connections towards the LCD screen. From the Arduino, connections D4, D5, D6, D7, D11 and D12 go to the LCD screen, which signal the output text from the code, which is in Appendix C. The LED, from D13, turns on based on the input from the user when the user selects a hot massage. The buzzer, from D8, turns on when the temperature measurement is high enough to signal the user the increasement in temperature but also to automatically stop the massage. The resistors in both LED and buzzer are to protect them from receiving too much voltage and causing a malfunction. The DC motor, from D9, receives the signal to provide the massage through the CAM designs. The input signal comes manually from the user to select cold or hot massage, and the desired speed.

3.2 Bluetooth Application- GUI

Figure 1 shows the user interface for the massage, where it includes a cold massage, a hot massage, a stop button, speed control, and a text input. This is the only user interface for the user and the manual input in the circuit.

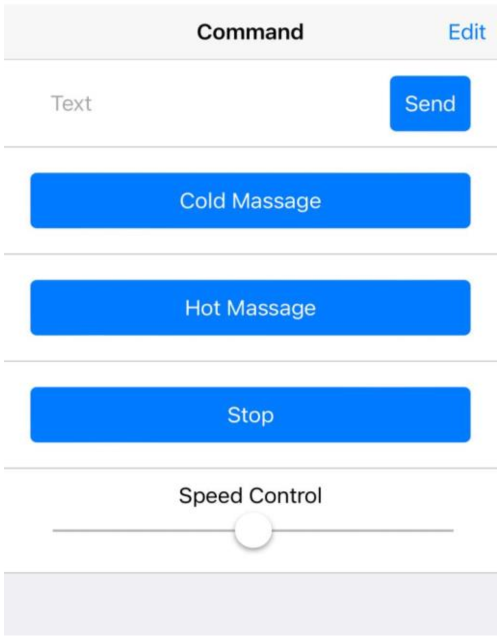


Figure 1. User Interface in Bluetooth App

3.3 Working Flow

When the user selects cold massage, a “Cold massage” text appears in the LCD screen as the massage begins. The user decides to stop the massage or change the speed according to its desire. The user can also select a hot massage, where the massage will stop if the user decides to manually stop it. The LCD screen also displays a “Hot massage” text. The user can also input a preferred temperature in the text section. With a default temperature hot massage of 85°F, (though the user can manually change it from the Bluetooth app) the heating element will turn on and off based on the temperature to ensure that the hot massage happens at that set temperature.The LCD screen also displays a “High Temp Alert” when the temperature sensor measures temperature greater than the safety threshold.

Appendix B shows the workflow that the product follows based on manual input from the user. First, there is a reading for high temperature. This works as a safety feature so if the temperature is too high, either cold or hot massage will always be off. If the temperature is below high temperature, the user can choose either hot or cold massage. Within cold massage, the user can manually stop and change the speed. If the user manually stops the massage, the circuit goes back to wait for an user input. For hot massage, the user can set the desired temperature for the massage. The heating element will be turned on or off to maintain this temperature. The user can also manually stop and change speed as long as the high temperature has not been achieved. If the high temperature is achieved, then the massage will automatically stop and go back to user input. The following table also shows the output component’s state with respect to each specific input (cold massage, hot massage with predetermined temperature, hot massage without high temperature and hot massage with high temperature).

Table 1 - Output Component's State With Respect to Input					
	Cold Massage	Hot Massage		Always	
		With Predetermined Temp		Temp < Safety Threshold	Temp > Safety Threshold
		< 85°F	>= 85°F		
LCD	On	On	On	On	On
Heating Equipment (LED)	Off	On	Off	Prev State	Off
Buzzer	Off	Off	Off	Off	On
DC Motor	On	On	On	Prev State	Off

3.4 Software Flow

Appendix B shows the software workflow for this project. Initially the variables, constant values such as temperature threshold are setup and the code enters an infinitely repeating loop. In this loop the software reads the temperature and receives input from the Bluetooth app and responds accordingly. If the temperature is greater than the safety threshold, a high temperature alert is given and the buzzer alarms. The system waits for 10 seconds to cool down. Then it is ready to receive commands. The commands from Bluetooth are received in this loop. And the appropriate pins are turned High and Low according to Table 1. And the loop repeats to receive the next command.

4. Experimental Test Results:

We have conducted several Experiments to evaluate the working of all the components which is clearly demonstrated in the video link appended in the section Appendix-E.

The following sections will explain the experiments conducted for testing the various functions that the Actual "ThermEaser" is supposed to perform.

4.1 User Interface and User Interaction:

We'll start with opening the Bluetooth app that we are using to control the device. The app shows the following screen as shown in fig 4.1 where it has the options to select between Cold Massage, Hot Massage, Stop, Speed control bar for controlling massage intensity and a text input section to select the required temperature. When the initial setup function of Arduino is done, LCD shows: "Hey! I'm Ready" informing the user that they can give the next command.

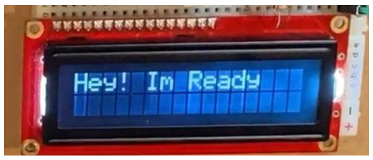


Fig 2 Ready to receive Next Command

4.2 Test for Cold Massage function:

After clicking on Cold Massage: Cold Massage ON! During this function an LED is used to represent the heat pad stays in OFF state as the cold massage mode is meant to work at room temperatures as shown in Fig 4.23. The motor starts running and can accept speed control.

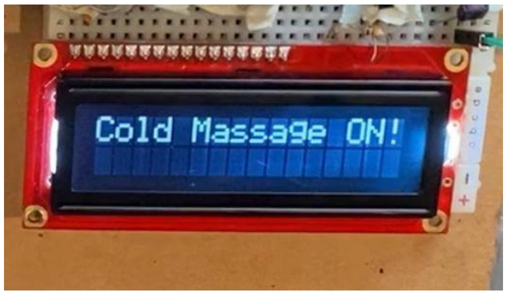


Fig 3 Cold Massage

4.3 Test for Hot Massage function:

Now we click on Hot massage button to turn the hot massage mode ON. This turns ON both the Motor and the LED which means the massage is aided with heat from the heat pad (LED in the Model) as shown in Fig 4.51. The heating pad controlled based on the user set temperature or default 85 F.

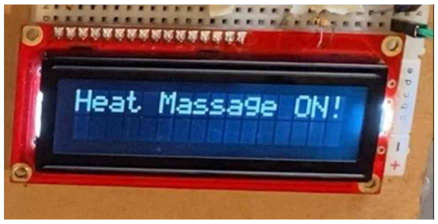


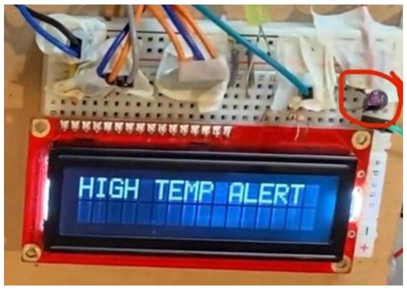
Fig 4 Hot Massage

4.4 Temperature input test:

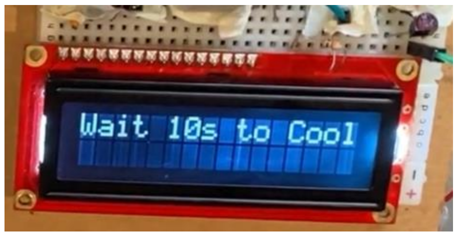
The text input function in the app will be useful for controlling the temperature of the heat pads during the hot massage function. Based on the set temperature and the current temperature reading from thermal sensor, the LED (heating element) turns on and off to regulate the set temperature.

4.5 Safety Alarm Test for Over Heating Conditions:

An overheating condition is not favorable and unsafe for both the electronic components and the user. So, a safety system is designed to alert the user during overheating conditions. We can observe that the buzzer alarms and the functioning of the massager stops if the temperature goes beyond safety threshold and waits for 10 seconds to cool down.



(a)



(b)

Figure 5: High temperature Alarm

5. Design Evaluation:

We were successful in fulfilling the requirements for including at least one component from each category(a-g) for functional elements mentioned as per the project requirements criterion. We used every component we proposed in this project.

1) Output Display:
    a) Bluetooth Application
    b) LCD Display
2) Manual Input:
    a) Bluetooth Application
    b) Potentiometer
3) Control Interface:
    a) Microcontroller unit- Arduino
4) Actuator:
    a) DC Geared motor with Cam Mechanism
5) Automatic Sensor Input:
    a) Thermal Sensor
6) Miscellaneous
    a) Bluetooth control to make the design user friendly
7) Audio Output
    a) Buzzer

6. Lessons Learned:

This project gave the team hands-on experience with the Bluetooth module of Arduino. We used a HM10 Bluetooth Module. One of the main difficulties faced was interfacing this module to the Arduino. We tried various open-source applications available that communicated well between the mobile and the Arduino. Some applications worked, some did not. Those that worked did not have the features that we required (Slider feature). After a lot of research, the ArduinoBlue App was found to be useful and met all the requirements. It also took the team some time to understand the features and working of this application.

If the project was started over, the team would work on a custom Bluetooth application particularly for Thermal Massaging Belt. This is because in the current application, all the buttons and the slider are all on the same page. It would be more user friendly if the speed control slider appeared only after the user selects a particular massage mode. Though the worst condition is handled in the code, this option would make the application more user friendly.

Moreover, the geared DC motor used for massaging has very low torque that it cannot actually massage. But since this is a prototype that was used to show the functionality of the belt. If the project was restarted, the team would do more research on motors that would be best suited for this application with a reasonably higher torque. That would require external power supply. So that also would have been added to the to do list.

7. Conclusion:

In conclusion, our "ThermEaser" project successfully demonstrated the feasibility of creating a mechatronic massage belt that provides customizable hot and cold massages. By integrating various components and overcoming initial challenges with hardware and software integration, we developed a functional prototype which demonstrates the working of components of ThermEaser . Through Experimental testing, we validated the performance of the device and highlighted its potential to offer Physical Rehabilitation benefits for managing injuries in waist region of the users. Moving forward, further refinement and optimization could enhance the user experience and broaden the applicability of the "ThermEaser" in healthcare and wellness contexts.

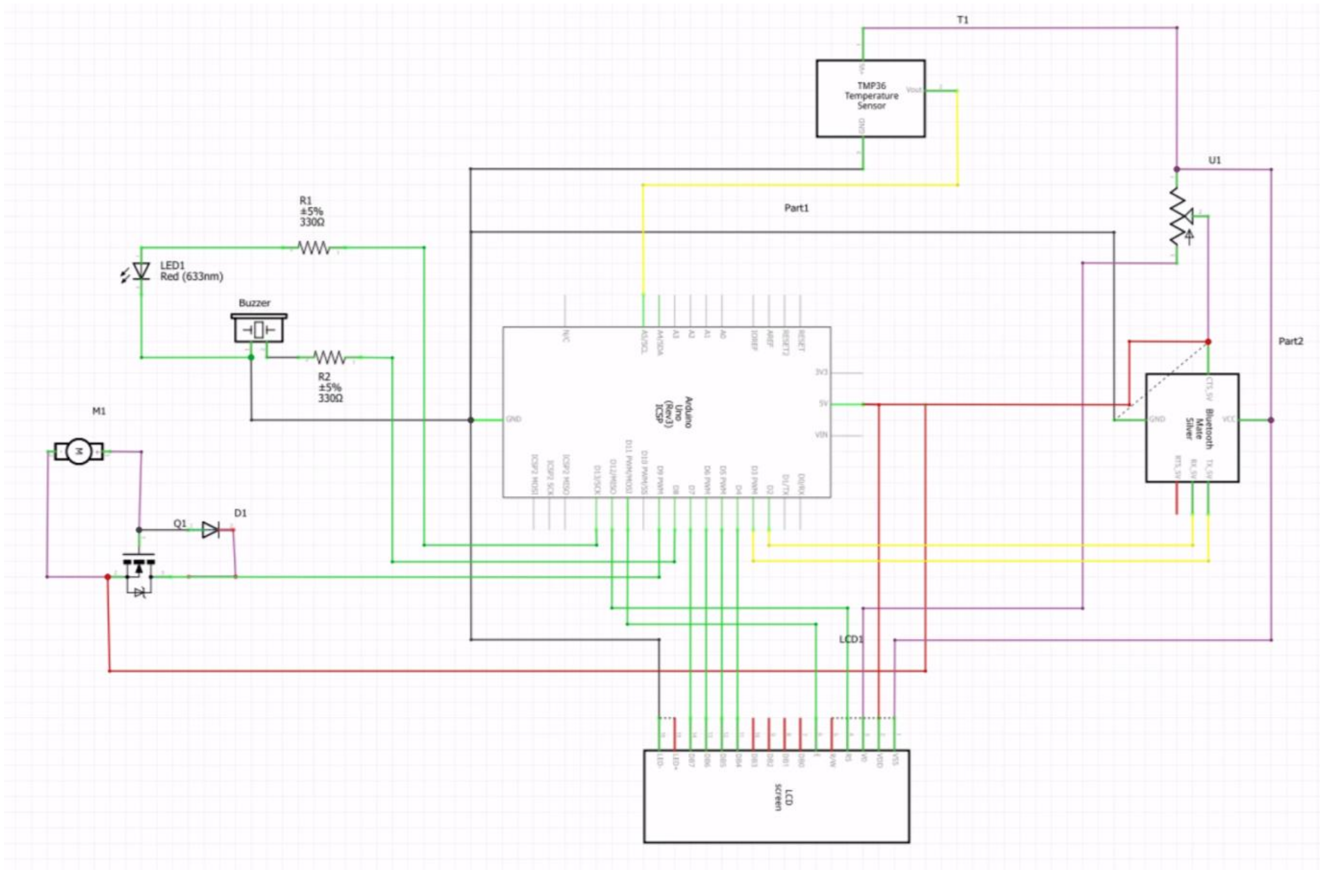
8. Appendix:

Appendix A- List of Materials:

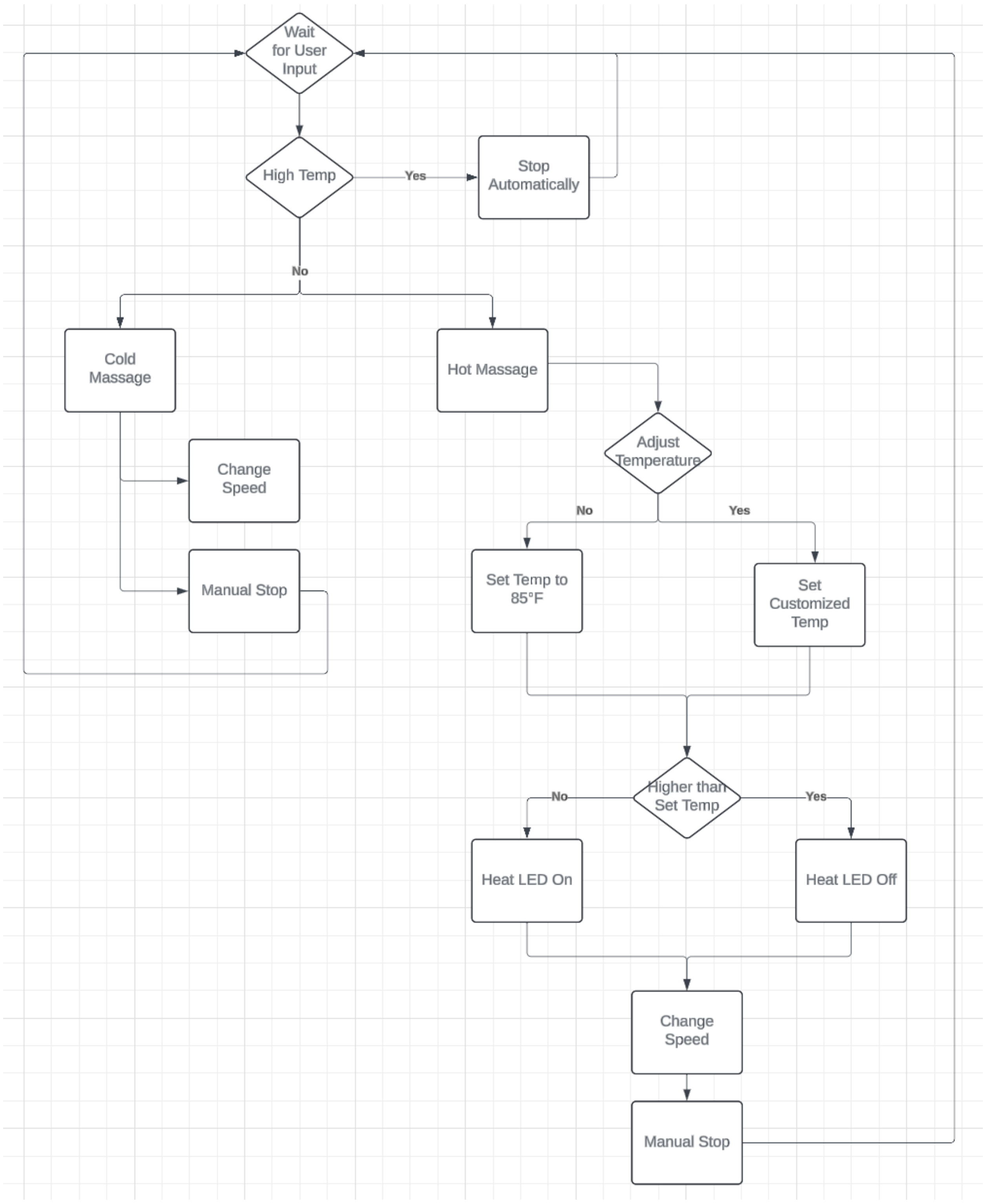
Sno	Material	Quantity
1	Arduino UNO	1
2	LCD Display	1
3	HM10 Bluetooth Module	1
4	Potentiometer	1
5	Piezzo Electric Buzzer	1
6	DC Geared motor	1
7	Yellow LED	1
8	1N4001 power diode	1
9	IRF 620 MOSFET	1
10	TMP36 Temperature Sensor	1
11	330 Ω Resistors	2
12	Jumper wires- Male to Female	7
13	Jumper wires- Male to Male	25
14	Iphone with ArduinoBlue App Installed	1
15	Arduino UNO- Laptop Connecting Wire	1
16	Adhesive tapes	1
17	Cardboard	1
18	Scissors	1
19	3D printer CAM profiles	1
20	Double sided Tapes	1
21	Laptop/PC with Arduino IDE installed	1

Appendix B- Design Drawings:

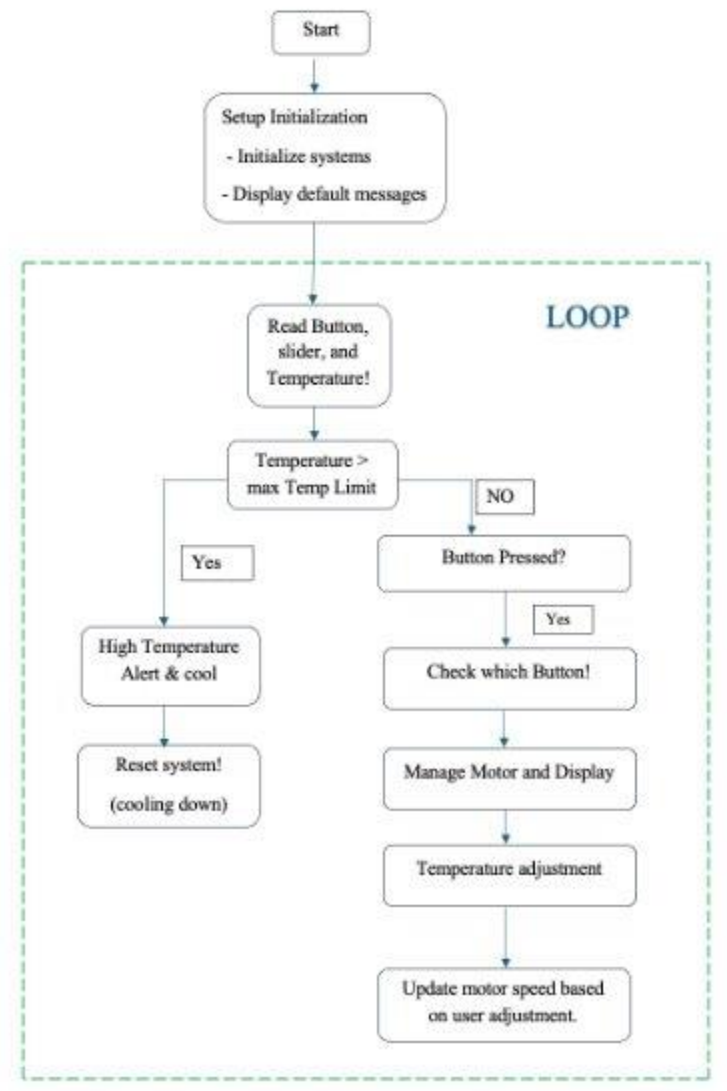
i) Circuit Diagram:



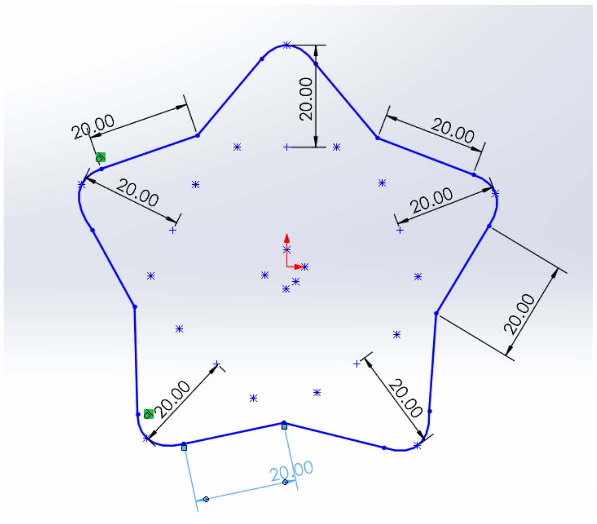
ii) Functional Block Diagram:



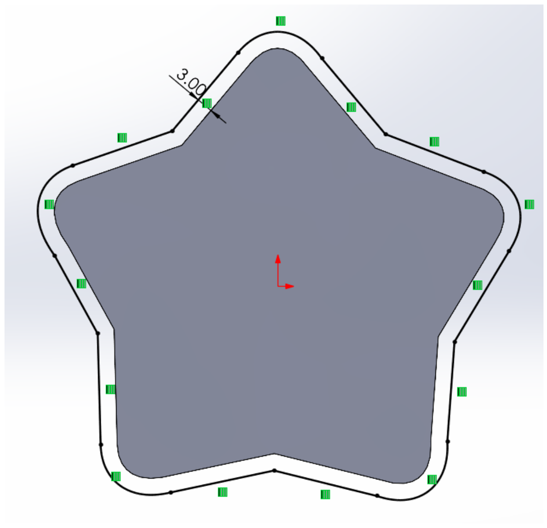
iii) Software Flow Chart:


Start
Setup Initialization
- Initialize systems
- Display default messages
LOOP
Read Button, slider, and Temperature!
Temperature > max Temp Limit
NO
Yes
Button Pressed?
Yes
High Temperature Alert & cool
Check which Button!
Reset system! (cooling down)
Manage Motor and Display
Temperature adjustment
Update motor speed based on user adjustment.

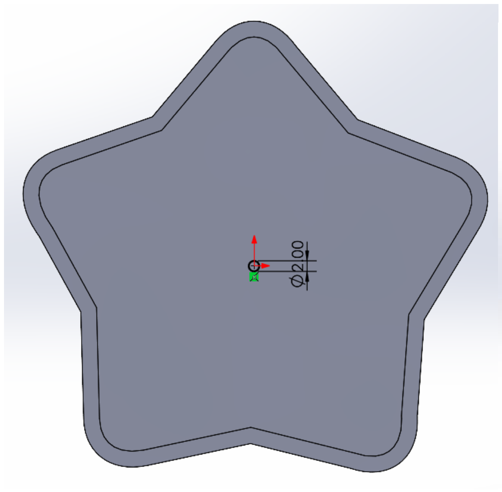
iv) SOLIDWORKS Model:



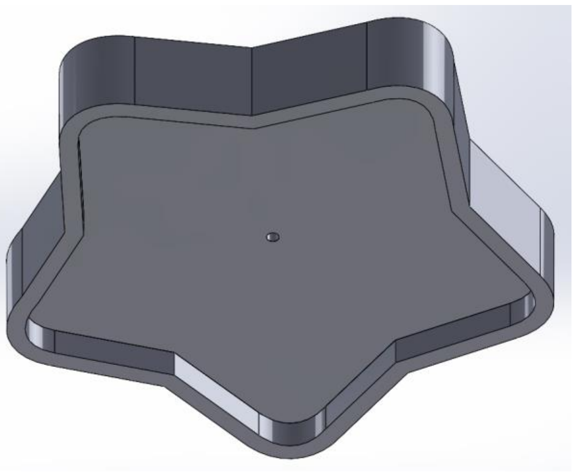
(a)



(b)



(a)



(b)

Fig CAM Model 1

Appendix C- Software:

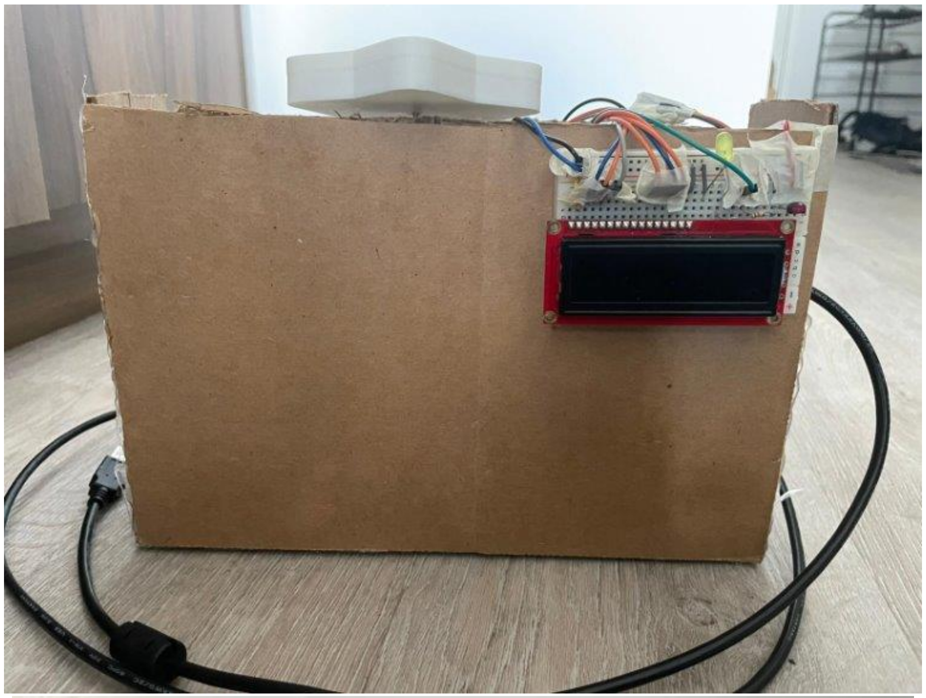
```
#include <SoftwareSerial.h>
#include <ArduinoBlue.h>
#include <LiquidCrystal.h>


#define MOTOR 9 // PWM control for motor
#define Heat_LED 13 // LED Replacement for Heating systems in the Massaging Belt
#define sensorPin A5 // Thermal Senor
#define buzzerPin 8 // Piezoelectric Buzzer
```

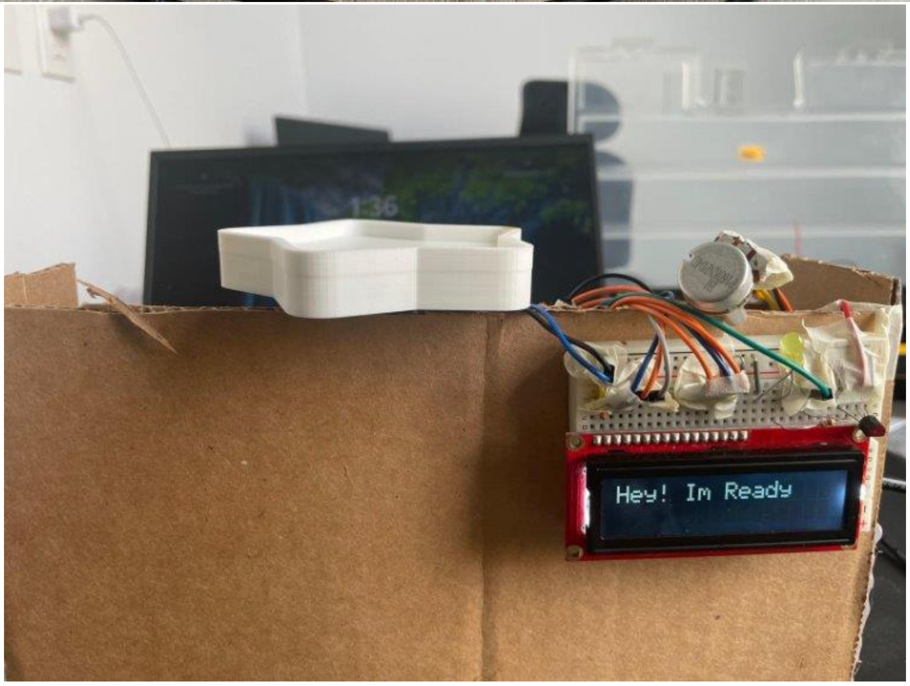
```
#define BLUETOOTH_TX 3 //Bluetooth module's TX pin connected to 3
#define BLUETOOTH_RX 2 //Bluetooth module's RX pin connected to 2

const unsigned long BAUD_RATE = 9600;
const unsigned int MIN_SPEED = 50; //Minimum Speed for CAM motor
const unsigned int MAX_SPEED = 200;  //Maximum Speed for CAM motor
const unsigned int START_SPEED = 100;  //Strting/Default Speed for CAM motor
const unsigned int MAX_TEMP_LIMIT = 105; // Temperature limit to turn on the buzzer, 100 Farenheit
const unsigned int TEMP_RESET_TIME = 10000; // Time to cool down if temperature limit is reached: 10 seconds
float HotMassageTemp = 85; // Default Temperature for Hot Massage


const int rs = 12, en = 11, d4 = 7, d5 = 6, d6 = 5, d7 = 4; //LCD communication Pins
LiquidCrystal lcd(rs, en, d4, d5, d6, d7); //Setting Up LCD

SoftwareSerial bluetooth(BLUETOOTH_TX, BLUETOOTH_RX);  // Connecting to the Bluetooth Module
ArduinoBlue phone(bluetooth); // pass reference of bluetooth object to ArduinoBlue constructor


//Variables for Code
int speed=START_SPEED;
char cold_message[]= "Cold Massage ON!";
char heat_message[]= "Heat Massage ON!";
String default_message = "Hey! Im Ready";
String my_message = "";
int current_mode = 2;
int button, sliderId, sliderVal;
int high_temp_count = 0;
int req_temp_reached = 0;

void setup() {
 // Start serial communications.
 Serial.begin(BAUD_RATE);
 bluetooth.begin(BAUD_RATE);
 delay(100);
 pinMode(Heat_LED, OUTPUT); // LED to replace thermal equipments
 pinMode(MOTOR, OUTPUT);  // CAM Motor pin
 pinMode(sensorPin, INPUT); //Thermal Sensor Pin
 pinMode(buzzerPin, OUTPUT); // Piezo Electric Buzzer Pin
 lcd.begin(16, 2);
 reset (1);
 Serial.println("setup complete");
}

// Put your main code here, to run repeatedly:
void loop() {
 // ID of the button pressed pressed.
 button = phone.getButton();

 // ID of the slider moved.
 sliderId = phone.getSliderId();

 // Slider value goes from 0 to 200.
 sliderVal = phone.getSliderVal();

 // Get the voltage reading from the TMP36
 int reading = analogRead(sensorPin);
 float voltage = reading * (5.0 / 1024.0);
 float temperatureC = (voltage - 0.5) * 100;
 float temperatureF = (temperatureC * 9.0 / 5.0) + 32.0;
 Serial.print(temperatureF);
 Serial.print("\xC2\xB0"); // shows degree symbol
 Serial.println("F");
 if (current_mode != 2){
  temperatureF = temperatureF - 5; // Temperature sensor give high values when the motor is running. Not sure why
 }
 if (temperatureF > MAX_TEMP_LIMIT){
  // The temperature sensor sometime gives a high temperature as noise when reading was taken at a high frequency.
  // So ensuring that the there is high temperature for atleast 20 times (using high_temp_count variable) before starting the cooling down
procedure
  high_temp_count +=1;
  if (high_temp_count>20){
   Serial.println("Buzzering");
   tone(buzzerPin, 300);
   delay (1000);
   noTone(buzzerPin);
   digitalWrite(Heat_LED, LOW); // switch OFF Heat_LED
   digitalWrite (MOTOR, LOW); // Turn off motor
   send_message ("HIGH TEMP ALERT", 0);
   delay (2000);
   send_message ("Wait 10s to Cool", 1);
   delay (TEMP_RESET_TIME); // Waits for 10 s till the system coold down before executing the next command
   reset (1);
  }
 }
 else{
  high_temp_count = 0;
 }

 // Display button data whenever its pressed.
 if (button != -1) {
   Serial.print("Button: ");
   Serial.println(button);
 }

 if (button == 1) { //Heat Massage Button Pressed
  Serial.println("Starting heat Massage");
  Serial.println("Heat_LED ON");
  digitalWrite(Heat_LED, HIGH); // switch ON Heat_LED
  analogWrite (MOTOR, speed); //Starting Motor with current speed (If the speed was not modified by the user, start speed will be used)
  my_message =  heat_message;
  send_message (my_message, 1);
  current_mode = 1;
 }


 if (button == 0) { // Cold Massage Button pressed
  Serial.println("Starting Cold Massage");
  Serial.println("Heat_LED OFF");
  digitalWrite(Heat_LED, LOW); // switch ON Heat_LED
  analogWrite (MOTOR, speed); //Starting Motor with current speed (If the speed was not modified by the user, start speed will be used)
  my_message = cold_message;
```

```
    send_message (my_message, 1);
    current_mode = 0;
  }

  if (button == 2){ // Stop button Pressed
    reset (0);
    my_message = default_message;
  }

  if (sliderId == 0) { //Speed slider Modified
    // Slider sends value from 0 to 200- Default value (the mid value) is 100
    if (sliderVal <MIN_SPEED){ // If slider inputs value less than Min speed, set min speed
      sliderVal = MIN_SPEED;
      send_message ("Setting MinSpeed", 0);
      delay (1000);
      send_message (my_message, 0);
    }
    if (sliderVal > MAX_SPEED){ // If slider inputs value greater than Max speed, set Max speed
      sliderVal = MAX_SPEED;
      send_message ("Setting MaxSpeed", 0);
      delay (1000);
      send_message (my_message, 0);
    }


    speed = sliderVal;
    Serial.print(speed);
    Serial.println(" is my new Speed");

    // If motor was running already, then write this speed.
    // This checkpoint is to handle situations in which user modifes the slider even before starting the massage or after stopping it
    if (current_mode!=2){
      analogWrite (MOTOR, speed);
    }
  }

  if (current_mode == 1){
    Serial.println(temperatureF > HotMassageTemp);

    if (temperatureF-5 > HotMassageTemp){
    // The temperature sensor sometime gives a high temperature when motor is running- Not sure why. So adjusting it.
    // So ensuring that the there is high temperature for atleast 20 times (using high_temp_count variable) before starting the cooling down
procedure
    req_temp_reached +=1;
    if (req_temp_reached>10){
      digitalWrite(Heat_LED, LOW); // switch OFF Heat_LED
      Serial.println ("Temp Reached");
      delay (500);
    }
  }
  else{
    digitalWrite(Heat_LED, HIGH); // switch OFF Heat_LED
    req_temp_reached = 0;
  }
}
```

```
 String str = phone.getText();
 // If a text from the phone was sent print it to the serial monitor
  if (str != "") {
    HotMassageTemp = str.toFloat();
    send_message ("Updated", 1);
    delay (1000);
    send_message (my_message, 0);
  }
}

void reset(int send_phone){ //Function to turn off motor and LED, Tell user that the system is ready to receive commands
 digitalWrite(Heat_LED, LOW); // switch OFF Heat_LED
 digitalWrite (MOTOR, LOW);
 send_message (default_message, send_phone);
 current_mode = 2;
}

void send_message(String msg, int send_phone){// Function to write a message to Serial Monitor, LCD and send to phone at the same time
 lcd.clear ();
 lcd.print(msg);
 Serial.println (msg);
 if (send_phone==1){
  phone.sendMessage(msg);
 }
}
```

Appendix D- Prototype:




Hey! Im Ready

Appendix E- Videos:

https://www.youtube.com/watch?v=-fOEsNSXCSk