EMBEDDED SYSTEMS

PROJECT DESIGN

Desk Stand Up Reminder

Member’s Name: NAVARRO, ROD GERYK C.

Course/Section: CPE160P-4/A1

Group No.: N/A

Progress Report Number: 2
Date of Submission: 10/27/2024

CYREL O. MANLISES, PH.D.
Instructor

Part I. Discuss in detail your progress here.

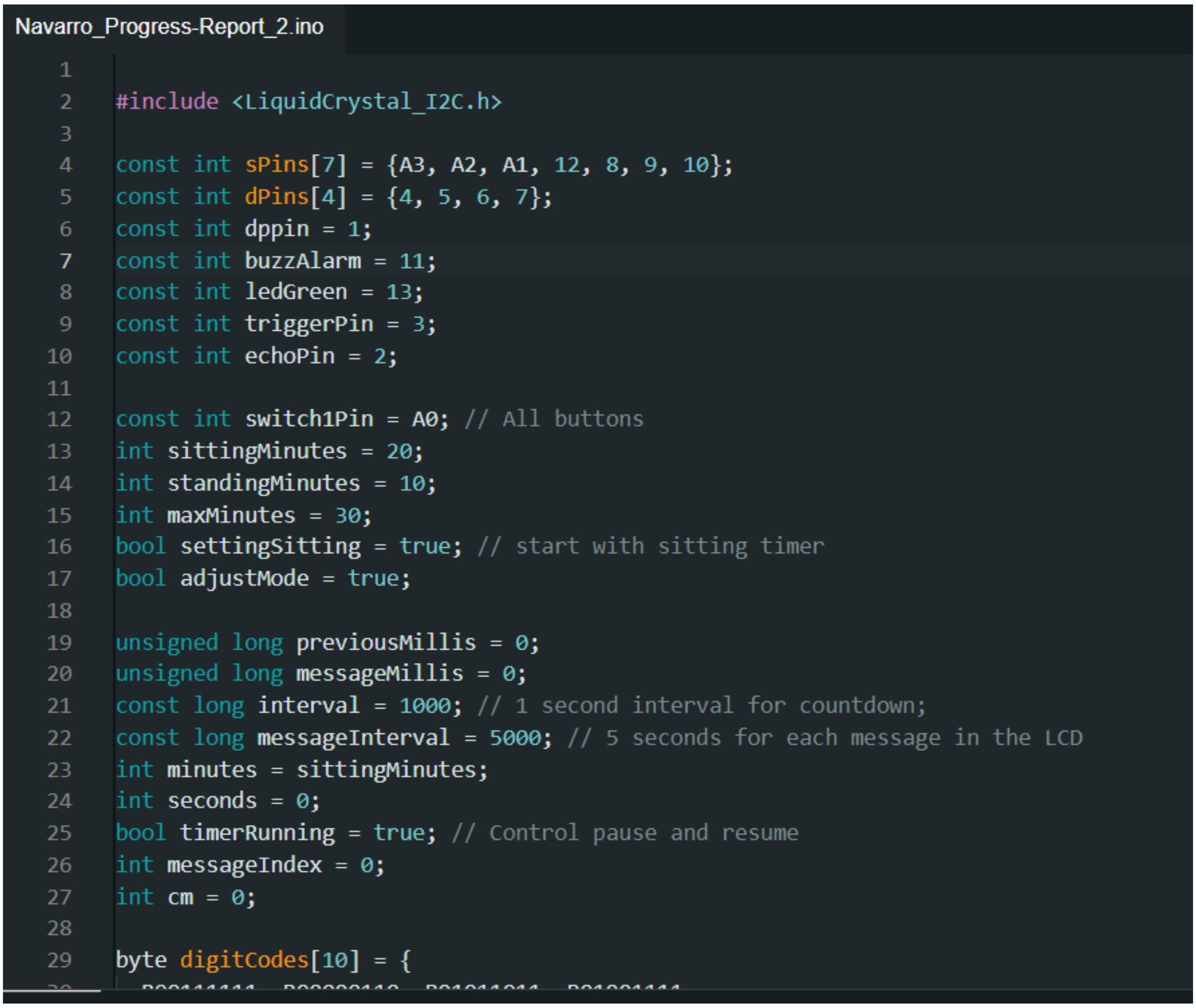
For my second progress report, I focused on refining the button functionality to ensure each press performed its designated task accurately. Initially, I encountered issues with accuracy when using different resistors for each button connected to a single analog input. To resolve this, I switched to a consistent setup using identical 220-ohm resistors for each button. This change allowed the buttons to function more accurately, ensuring each one triggered the intended response. In terms of coding, I developed a timer system with two modes: a sitting timer and a standing timer. The device begins with the sitting timer, counting down from a default of 20 minutes, which users can adjust within a 30-minute limit. Once the sitting timer completes, the program automatically shifts to the standing timer and vice versa. A green LED and a buzzer provide alerts when the timer finishes, signaling users to switch between sitting and standing.

The display component includes a 16x2 LCD screen and a four-digit seven-segment display. The LCD shows motivational messages that update periodically, encouraging the user to stay focused or take a break depending on the active timer. The seven-segment display shows the countdown time in minutes and seconds, refreshing every second to ensure accuracy.

Finally, I integrated an ultrasonic sensor to detect the user's presence. When the user moves close to the device, the timer switches between sitting and standing modes automatically. This sensor adds an interactive aspect to the system, making the device more user-friendly by adapting to the user's movements without needing manual button presses.

This progress involved both hardware adjustments and coding to ensure each component s accurate function. With all elements working correctly, the project successfully meets its objectives, reminding the user to alternate between sitting and standing at regular intervals with timely alerts and motivational messages.The setup is now reliable, making it a useful tool for promoting better posture and health.

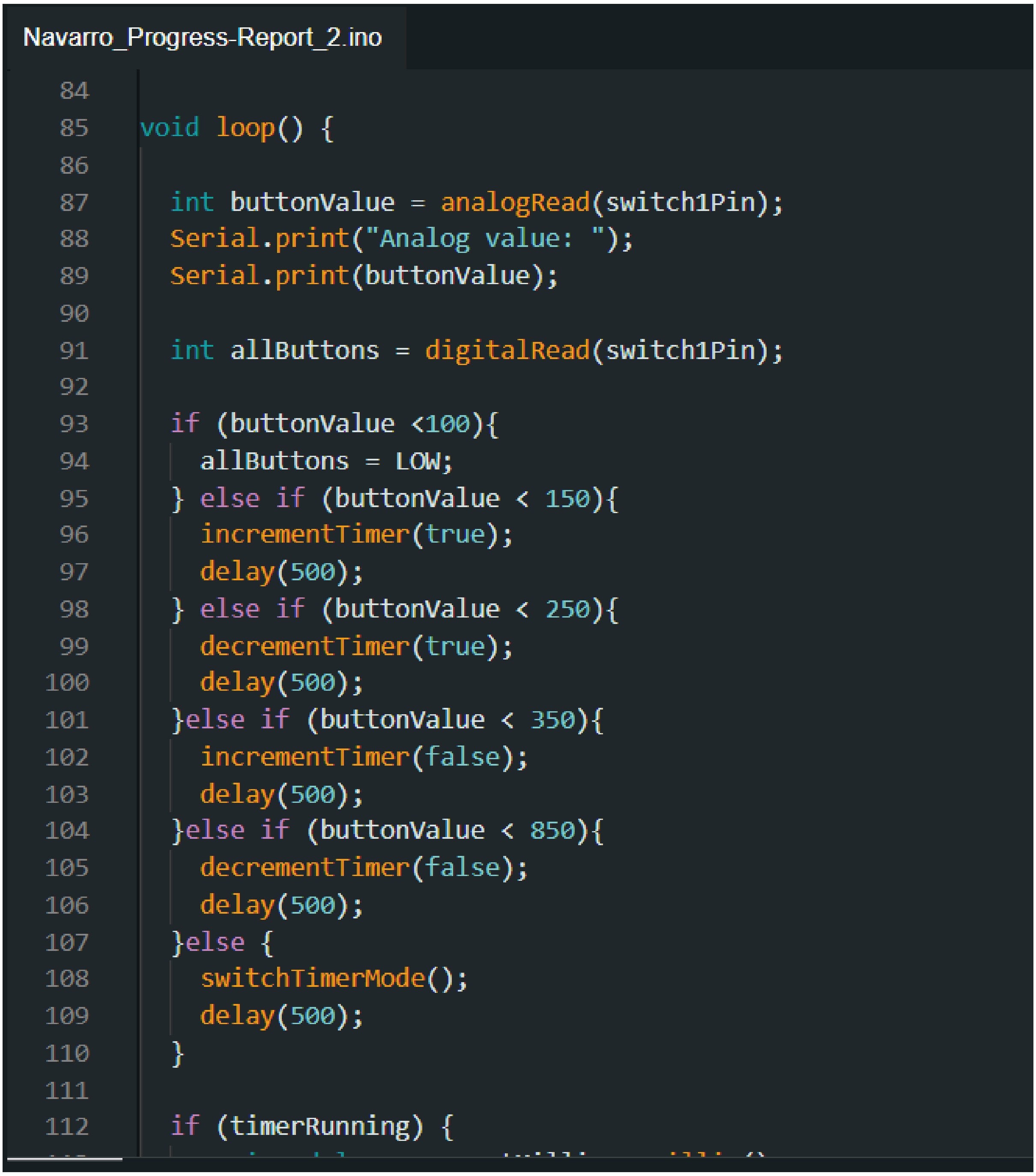
Part II. Provide screenshots as proof of your progress.



Navarro_Progress-Report_2.ino

```

#include <LiquidCrystal_I2C.h>

const int sPins[7] = {A3, A2, A1, 12, 8, 9, 10};
const int dPins[4] = {4, 5, 6, 7};
const int dppin = 1;
const int buzzAlarm = 11;
const int ledGreen = 13;
const int triggerPin = 3;
const int echoPin = 2;

const int switch1Pin = A0; // All buttons
int sittingMinutes = 20;
int standingMinutes = 10;
int maxMinutes = 30;
bool settingSitting = true; // start with sitting timer
bool adjustMode = true;

unsigned long previousMillis = 0;
unsigned long messageMillis = 0;
const long interval = 1000; // 1 second interval for countdown;
const long messageInterval = 5000; // 5 seconds for each message in the LCD
int minutes = sittingMinutes;
int seconds = 0;
bool timerRunning = true; // Control pause and resume
int messageIndex = 0;
int cm = 0;

byte digitCodes[10] = {
```

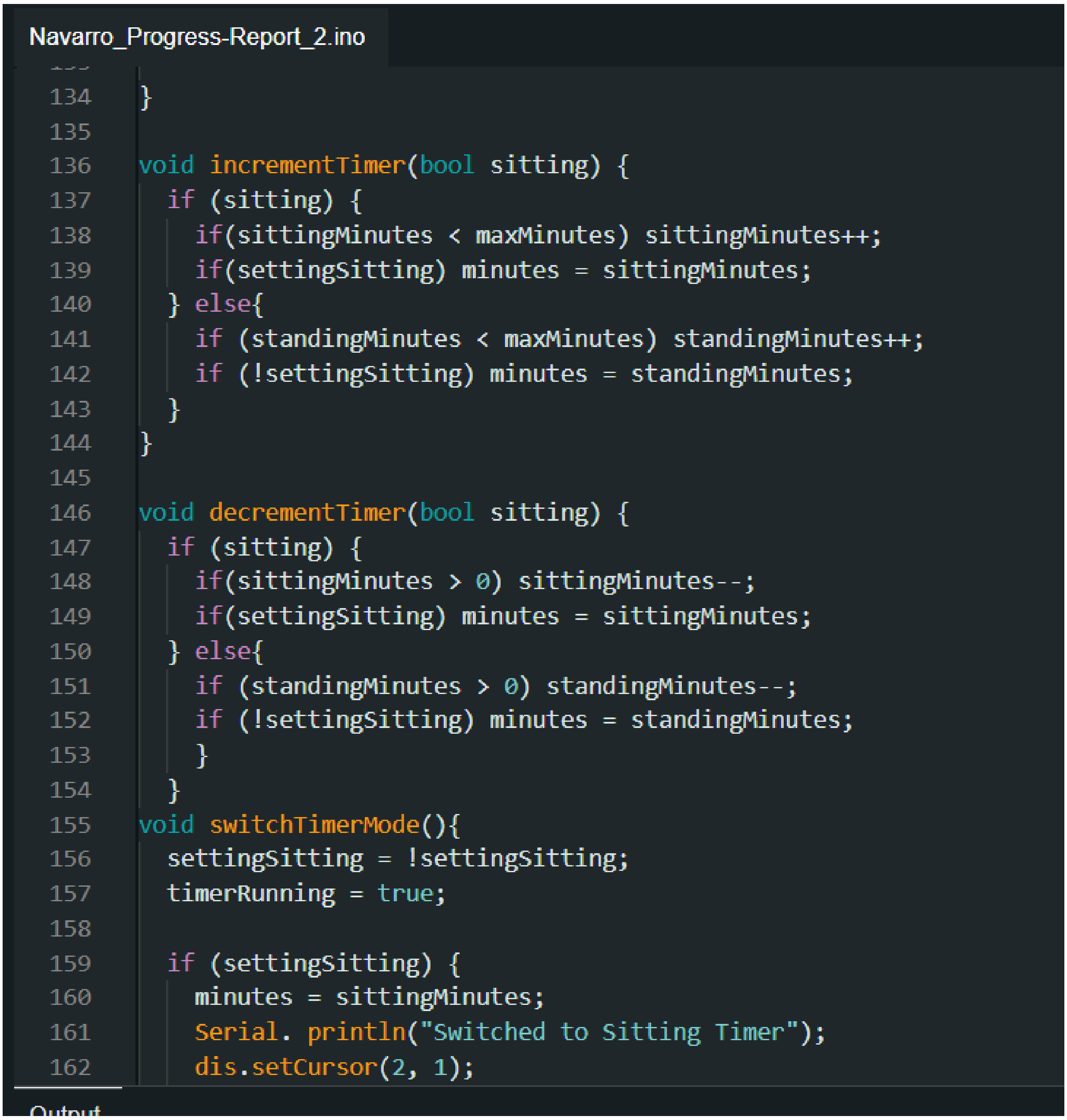
Figure1: *Initialized variables as well as set up the two primary timers: one for sitting and another for standing, with adjustable intervals displayed on a 4-digit seven-segment display.*



Navarro_Progress-Report_2.ino

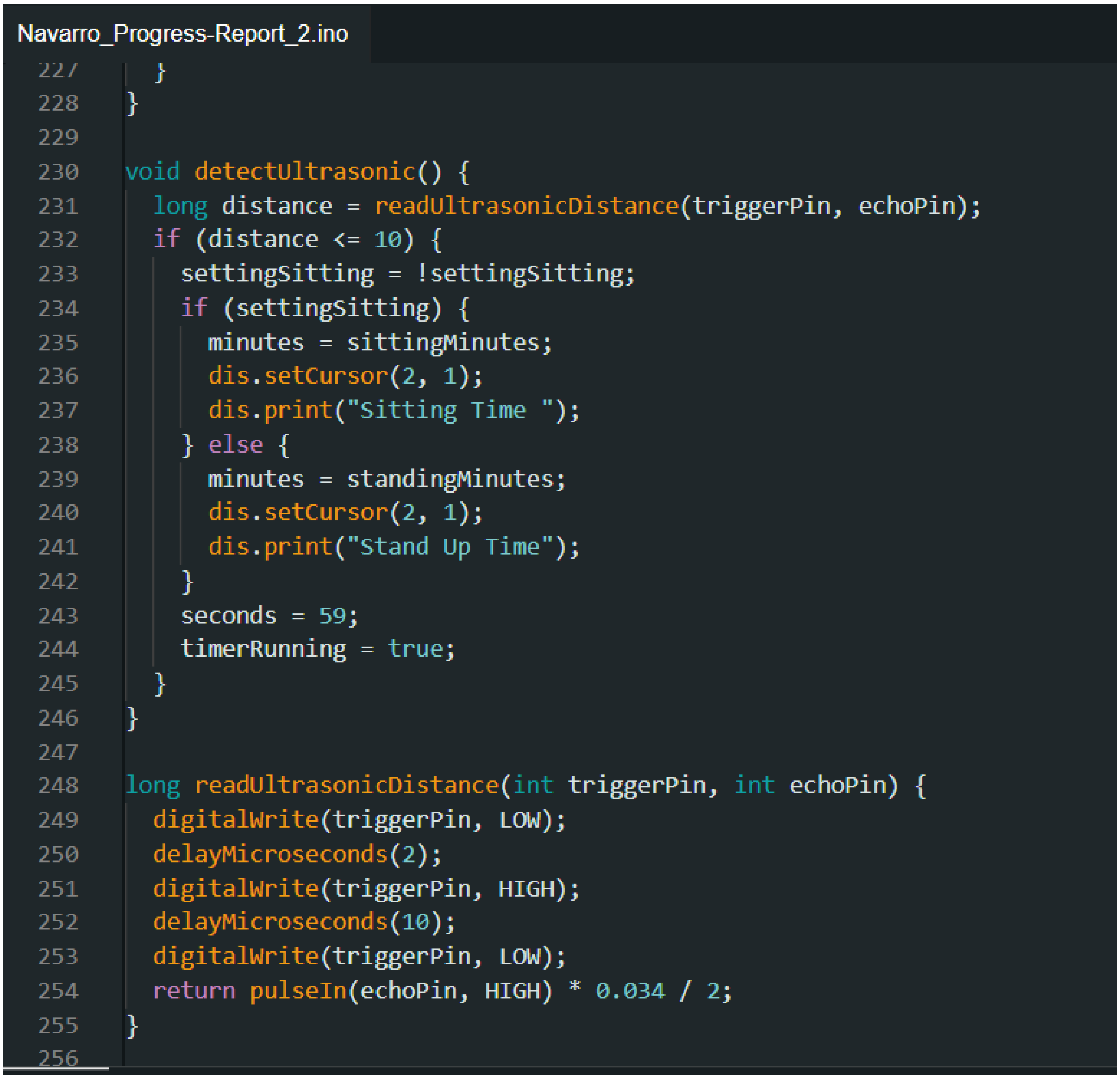
```

void loop() {

  int buttonValue = analogRead(switch1Pin);
  Serial.print("Analog value: ");
  Serial.print(buttonValue);

  int allButtons = digitalRead(switch1Pin);

  if (buttonValue <100){
    allButtons = LOW;
  } else if (buttonValue < 150){
    incrementTimer(true);
    delay(500);
  } else if (buttonValue < 250){
    decrementTimer(true);
    delay(500);
  }else if (buttonValue < 350){
    incrementTimer(false);
    delay(500);
  }else if (buttonValue < 850){
    decrementTimer(false);
    delay(500);
  }else {
    switchTimerMode();
    delay(500);
  }

  if (timerRunning) {
```

Figure2: *Assigning the five buttons to their respective tasks ensuring that each button triggered with the intended response.*



```
}

void incrementTimer(bool sitting) {
  if (sitting) {
    if(sittingMinutes < maxMinutes) sittingMinutes++;
    if(settingSitting) minutes = sittingMinutes;
  } else{
    if (standingMinutes < maxMinutes) standingMinutes++;
    if (!settingSitting) minutes = standingMinutes;
  }
}

void decrementTimer(bool sitting) {
  if (sitting) {
    if(sittingMinutes > 0) sittingMinutes--;
    if(settingSitting) minutes = sittingMinutes;
  } else{
    if (standingMinutes > 0) standingMinutes--;
    if (!settingSitting) minutes = standingMinutes;
    }
  }
void switchTimerMode(){
  settingSitting = !settingSitting;
  timerRunning = true;

  if (settingSitting) {
    minutes = sittingMinutes;
    Serial. println("Switched to Sitting Timer");
    dis.setCursor(2, 1);
```

Figure3: *Complete all the functions needed to achieve all the required outputs of the project.*



Navarro_Progress-Report_2.ino

```
  }
}

void detectUltrasonic() {
  long distance = readUltrasonicDistance(triggerPin, echoPin);
  if (distance <= 10) {
    settingSitting = !settingSitting;
    if (settingSitting) {
      minutes = sittingMinutes;
      dis.setCursor(2, 1);
      dis.print("Sitting Time ");
    } else {
      minutes = standingMinutes;
      dis.setCursor(2, 1);
      dis.print("Stand Up Time");
    }
    seconds = 59;
    timerRunning = true;
  }
}

long readUltrasonicDistance(int triggerPin, int echoPin) {
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  return pulseIn(echoPin, HIGH) * 0.034 / 2;
}
```

Figure4: The creation of the code for the ultrasonic sensor to detect if the user moves away, automatically pausing the timer until they return within range.

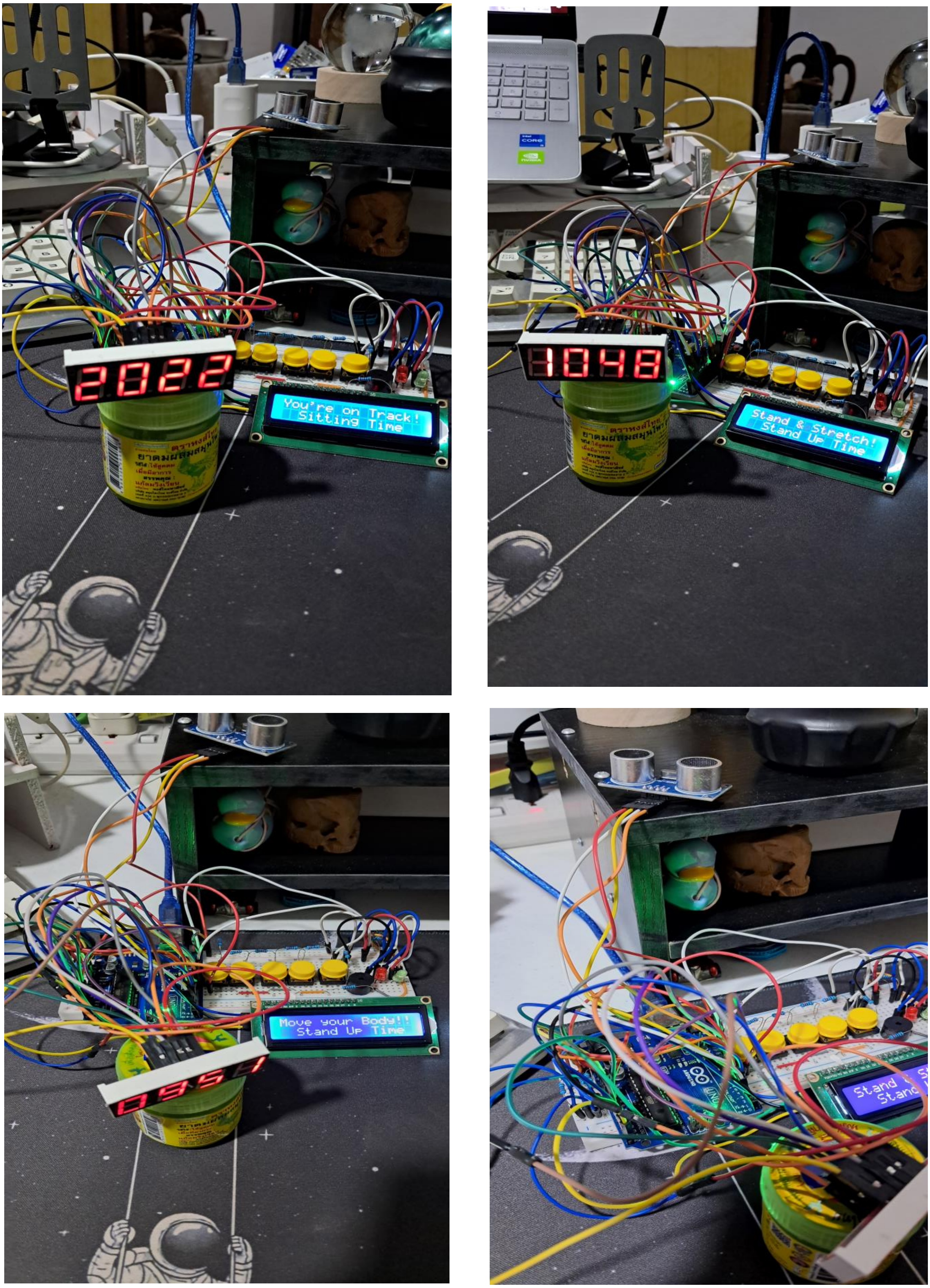
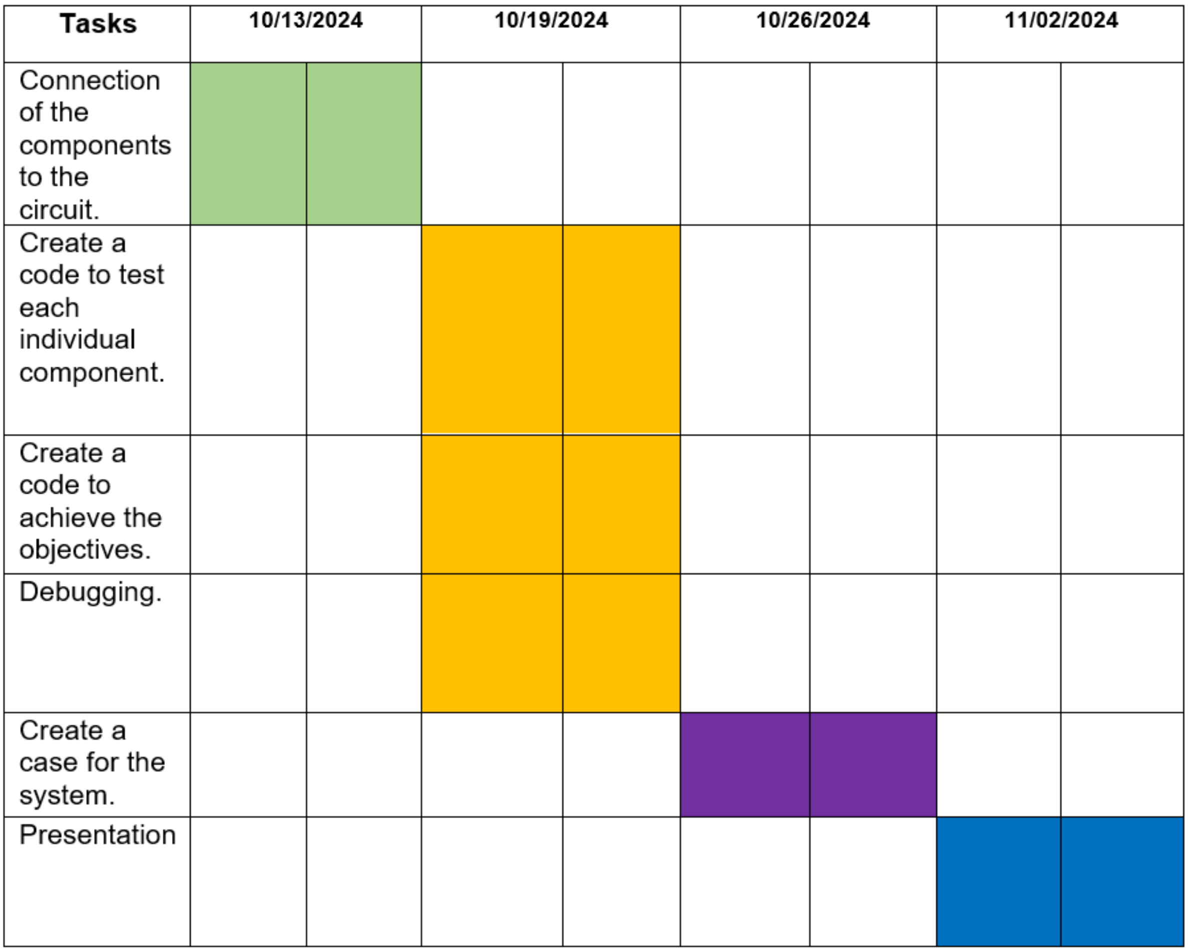


Figure 5: Working prototype of the project.

Part III. List down your work load

- Make sure all the components are connected correctly.
- Test and ensure that each button is triggered with the intended response.
- Fix any issues in the circuit.
- Complete the code for the project.
- Complete the working prototype.

Part IV. Gantt chart



Tasks	10/13/2024		10/19/2024		10/26/2024		11/02/2024	
Connection of the components to the circuit.	🟩	🟩						
Create a code to test each individual component.			🟨	🟨				
Create a code to achieve the objectives.			🟨	🟨				
Debugging.			🟨	🟨				
Create a case for the system.					🟪	🟪		
Presentation							🟦	🟦