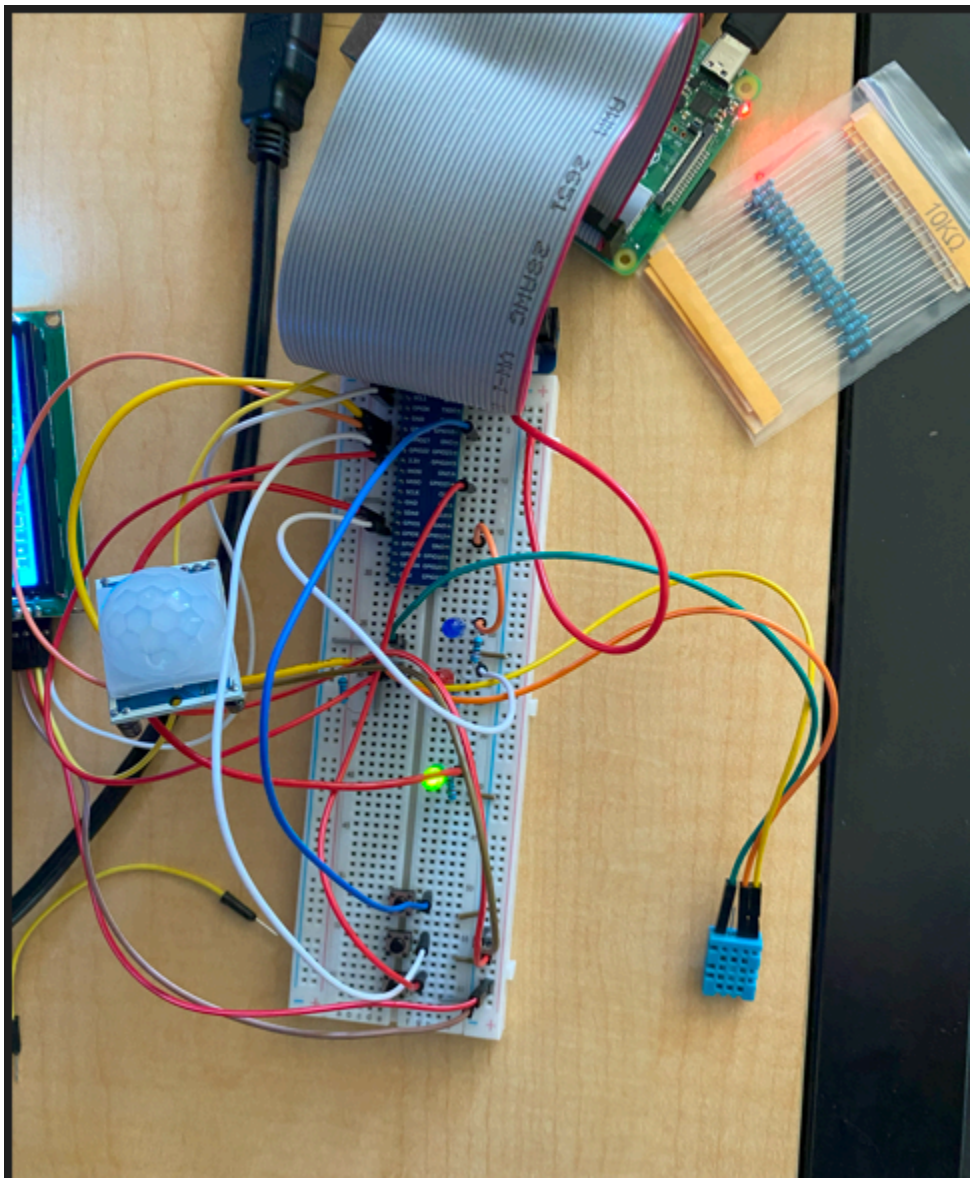
Board hardware explanation starts this page (For code explanations GOTO page 5.)
Board connection overview



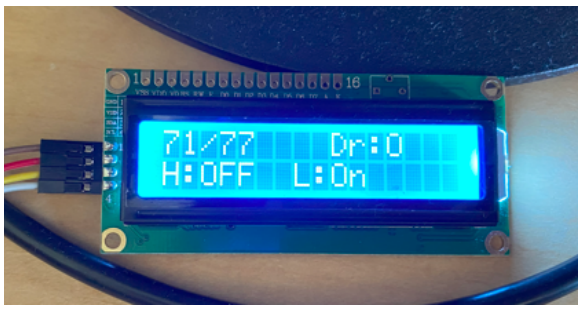
```
buttonPin = 22
button2Pin = 15
button3Pin = 12
ledPin = 29
ledRedPin = 31
ledBluePin =  32
DHTPin = 11
sensorPin = 13
```

```
mcp.output(3,1)
lcd.begin(16,2)
```

HVAC status: AC on (because desired temperature (71) is lower than current temperature (77), and door is closed)

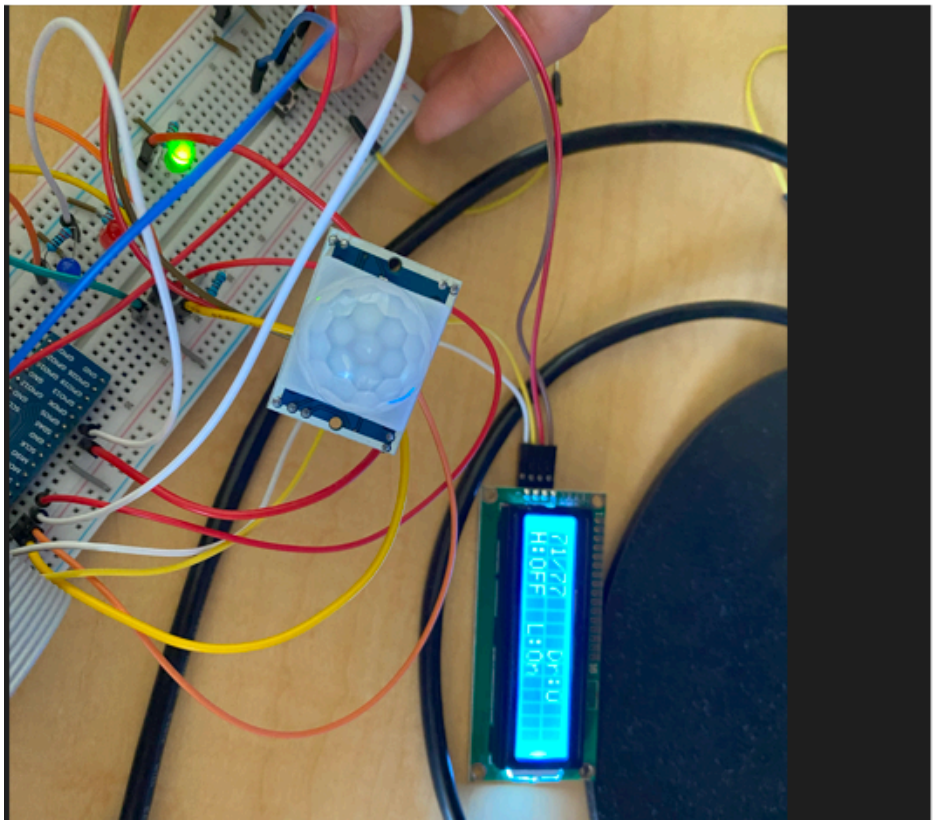
Open door

AC on (because door is open)

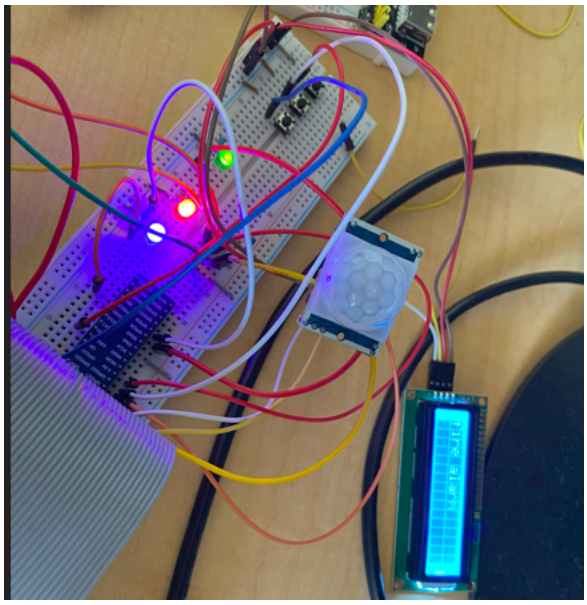


Motion sensor (green light indicator and on-screen indicator)

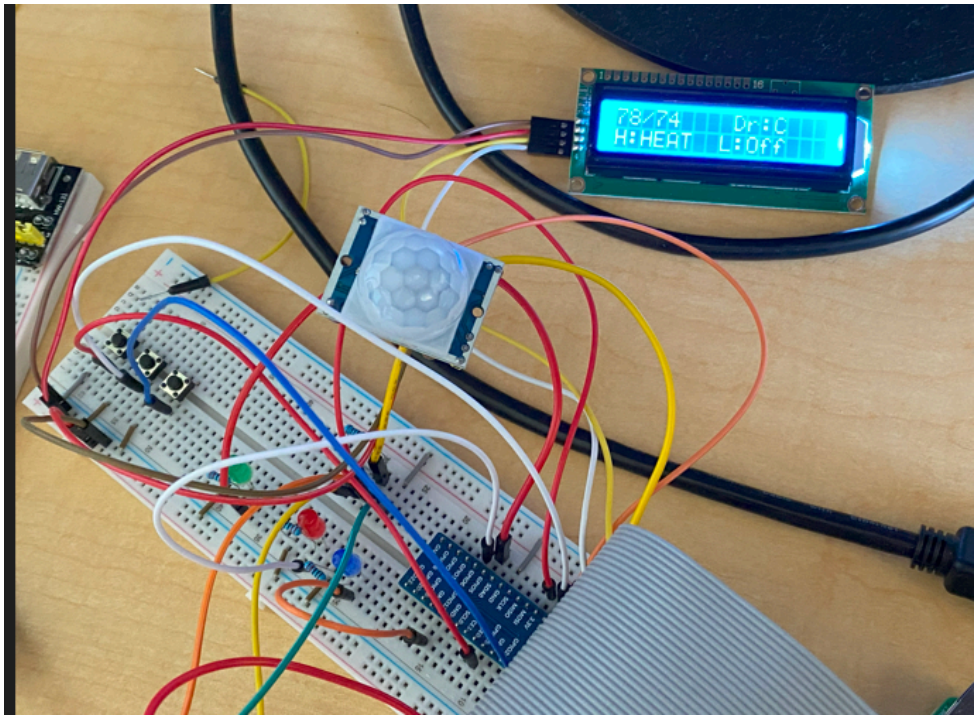
.



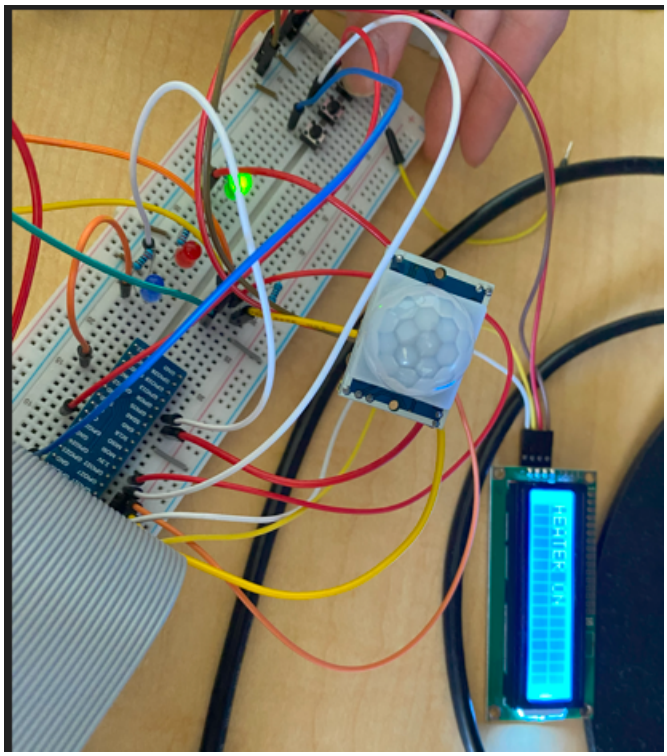
Fire alarm situation



Increasing the desired temperature



Heater turns on



Code explanation (done in more detail in the demo video)

1. Obtaining humidity data from CIMIS
    - Referenced the website API developer manual and researched some Python code on using the Requests library.

```
url = 'http://et.water.ca.gov/api/data?appKey=ab715ae9-b712-4d75-aab9-2ce3542108ed&targets=
75&startDate=2024-06-12&endDate=2024-06-12&dataItems=day-rel-hum-avg'

respons = requests.get(url)
print(response.status_code)

humidity = response.json()['Data']['Providers'][0]['Records'][0]['DayRelHumAvg']['Value']
```

2. pin set ups (GPIO setup, input / output setups)
3. control variables set ups for status, lcd print out messages, and counts

```
mcp.output(3,1)
lcd.begin(16,2)
global okTemp
global custom_message
global desired_temp
global door_is_open
global status_changed
global ac_status_change
global hvac
previous_counts = 0
counts = 0
```

4. Obtaining the temperature measurements from dht11 device
    - Referenced the Freenove tutorial codes on interfacing the dht11 and included their relevant modules in my code
    - Also factors in the humidity information.

```
while(True):
    temp_sum = 0
            temp_count = 0

    dht = DHT.DHT(DHTPin)
    counts += 1
    print('Measurement counts: ', counts)
    for i in range(0,15):
        chk = dht.readDHT11()
        if (chk == dht.DHTLIB_OK):
            temp_sum += dht.temperature
            temp_count += 1
            if temp_count == 3:
                                    break
        sleep(0.1)
    okTemp = round((temp_sum / temp_count) * (9/5) + 32) + 0.05 * humidity
```

5. fire alarm situation when the temperature goes above 95

```
if okTemp >= 95:
    door_is_open == 'O'
    GPIO.output(ledPin, GPIO.HIGH)
    GPIO.output(ledRedPin, GPIO.HIGH)
    GPIO.output(ledBluePin, GPIO.HIGH)
    sleep(1)
    GPIO.output(ledPin, GPIO.LOW)
    GPIO.output(ledRedPin, GPIO.LOW)
    GPIO.output(ledBluePin, GPIO.LOW)
    sleep(0.1)
    lcd.clear()
    sleep(0.1)
    lcd.message('fire alarm')
```

6. control logics for the interaction between door status and hvac status. As well as checking if the current temperature is within 3 degrees of the desired temperature.

```
if (door_is_open == 'O'):
    hvac = 'OFF'
elif (abs(desired_temp - okTemp) < 3) and hvac != 'OFF':
    hvac = 'OFF'
    ac_status_change = True
elif (okTemp - desired_temp > 3) and hvac != 'AC':
    hvac = 'AC'
    ac_status_change = True
elif (desired_temp - okTemp > 3) and hvac != 'HEAT':
    hvac = 'HEAT'
    ac_status_change = True
```

7. display message on LCD when status changes in door
    - Referenced the Freenove tutorial on interfacing the LCD and included the example codes in my own project.

```
if status_changed:
    sleep(0.1)
    lcd.clear()
    sleep(0.1)
    lcd.setCursor(0,0)
    if door_is_open == 'O':
        lcd.message('door open!')
        #door_is_open = 'C'
    else:
        lcd.message('door closed!')
        #door_is_open = 'O'
    sleep(3)
    status_changed = False
```

8. display message on LCD when HVAC status changes

```
  lcd.setCursor(0,0)
  if hvac == 'OFF':
      lcd.message('HVAC OFF')
  elif hvac == 'HEAT':
      lcd.message('HEATER ON')
  else:
      lcd.message('AC ON')
  sleep(3)
  ac_status_change = False
```

9. Motion sensor code (copied from the Freenove tutorial sample code on interfacing the PIR motion sensor)

```
    if GPIO.input(sensorPin) == GPIO.HIGH:
        ambient_light = 'On '
        GPIO.output(ledPin, GPIO.HIGH)
        previous_counts = counts
```

10. message to be displayed on the LCD

```
                custom_message =  '' + str(desired_temp) + '/' + str(okTemp) + '    Dr:' +
str(door_is_open) + '\nH:' + str(hvac) + '  L:' + str(ambient_light)
```

11. Example interrupt callbacks from user interactions

```
def toggle_blue_led(channel):
    global desired_temp
    global hvac
    global ac_status_change
    GPIO.output(ledBluePin, GPIO.HIGH)
    sleep(0.5)
    GPIO.output(ledBluePin, GPIO.LOW)
    desired_temp -= 1
    if (okTemp - desired_temp < 3):
        hvac = 'AC'
        ac_status_change = True

def toggle_door(channel):
    global door_is_open
    global status_changed
    if(door_is_open == 'C'):
        door_is_open = 'O'
    else:
        door_is_open = 'C'
    status_changed = True
```