ECE413/599 Final project (10 pts)

This project is designed to give individuals hands-on experience with sensor conditioning and measurement using a commercial sensor, a microcontroller, and data visualization on a smartphone via Bluetooth. Here I provide several online sources that offer codes for the sensors and Bluetooth connection. These can be used as provided or modified to suit the needs of this project. Additional resources can also be found through internet searches

The project will be graded based on the following criteria:

1. (2 pts) Explain the operating principle of the sensor, device structure, and conditioning circuit, including how it detects the measurement quantity and generates output electrical signals. This information can be obtained from the sensor's datasheet or other online sources.
2. (2 pts) Demonstrate the sensor's operation by using a sketch and display the sensing readout on Arduino IDE through a USB connection to a computer.
3. (1 pts) Demonstrate real-time visualization of the sensing data on a mobile phone via Bluetooth.
4. (1 pt) Suggeste potential new applications of the sensor.
5. (1 pt) Briefly describe the contribution made by each member of your team.
6. (1.5 pts) One of your team members should upload your powerpoint slides (they don't need to be the final version) along with 4 technical multiple-choice questions and their answers about your sensor for in-class quizzes, **2 days before the presentation**. Ensure the questions delve into the operating principle of the sensor, with a focus on 'why' and 'how' questions rather than 'what' questions. Please do not include these questions in your presentation, as I may need to modify them.
7. (0.5 pt) One of your team members should upload your finalized slide and your codes with instruction after presentation.
8. (1 pt) Return the kit or sensor provided by the instructor after the presentation.

The individual's grade for the final project is determined by the sum of scores from items 1 to 8, multiplied by the individual's contribution, which can be either 1 or 0. Participants in the project and presentation will receive a score of 1, whereas those who do not attend or participate will receive a score of 0. Please let me know in advance if you have any questions or concerns.

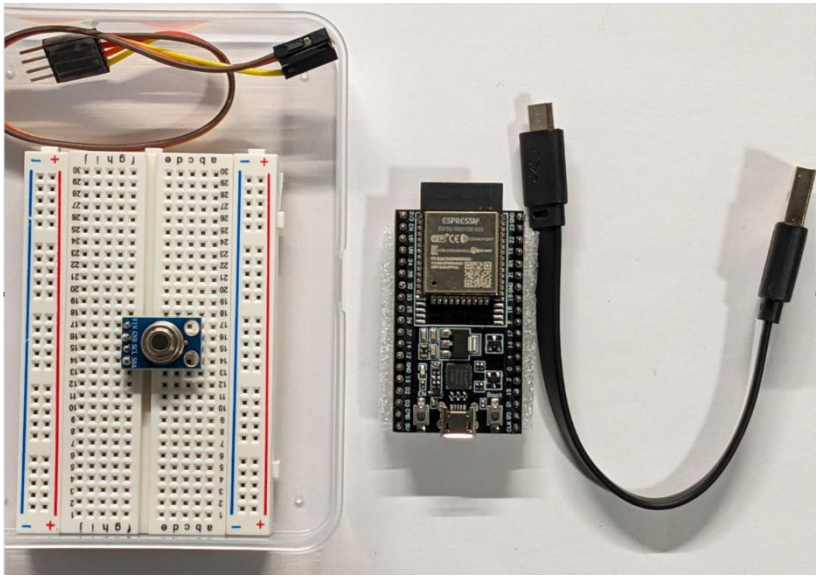
Contact TA Ahasan Ullah (ullaha@oregonstate.edu) for technical assistance if you have any questions about the project.

Group 6 members

Waxter, Silas	waxters@oregonstate.edu
Sisneros, Taylor	sisnerot@oregonstate.edu
Yakovich, Nicholas	yakovicn@oregonstate.edu
Wiese, Asa	wiesea@oregonstate.edu
Lefevre, Noel	lefevrno@oregonstate.edu

ESP32, Sensors and Phyphox

1. ESP32 Devkitc V4 microcontroller
2. GY-906 MLX90614 Non-Contact IR Infrared Temperature Sensor
3. Micro-USB connector
4. Dupont wire
5. Breadboard
6. Box



To begin programming your sensor with the ESP32, it is crucial to understand how the sensor communicates its data. Below, you will find useful resources that explain how your sensor functions, along with some example code to help you start writing your program.:

- https://www.melexis.com/-/media/files/documents/datasheets/mlx90614-datasheet-melexis.pdf
- https://components101.com/sensors/melexis-mlx90614-contact-less-ir-temperature-sensor

For extra information or sensor specifications, it is helpful to check your sensor's datasheet:

- ttps://wiki.dfrobot.com/IR_Thermometer_Sensor_MLX90614_SKU__SEN0206

ESP32 microcontroller

Programming the ESP32 is quite similar to programming an Arduino, with most basic Arduino functions (e.g., analogWrite(), pinMode, etc.) operating in a similar manner. However, it is important to note that the hardware, especially the pinout, differs significantly from the Arduino. Make sure to select ESP32 pins that correspond to the requirements of your programs. For accurate pin mapping, refer to the ESP32 pinout diagram.

https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/esp32/get-started-devkitc.html

To flash onto the ESP32, you need to install the board library in your Arduino IDE. Follow the instructions in this link to learn how to install the Arduino IDE.

https://support.arduino.cc/hc/en-us/articles/360019833020-Download-and-install-Arduino-IDE

This link has helpful instructions on programming the ESP32:

https://randomnerdtutorials.com/getting-started-with-esp32/

After testing the example code on your ESP32, you're ready to start setting up your data transmission via Bluetooth. First, make sure you can establish a Bluetooth connection with your phone. Since not all devices support classic Bluetooth, we will set up a Bluetooth Low Energy (BLE) connection, which is also the protocol used for visualizing your data. Follow the instructions at the link below to set up an example BLE server. There's no need to follow the scanner instructions.

https://randomnerdtutorials.com/esp32-bluetooth-low-energy-ble-arduino-ide/

Once you have set up the server, use any BLE scanning mobile app (e.g., LightBlue, BLE Scanner, nRF Connect) to locate and establish a connection with your ESP32.

NOTE: There may be a conflict between ESP32 BLE and Arduino BLE. If you have the ArduinoBle library installed in your Arduino IDE, uninstall it to ensure a smooth Bluetooth connection with ESP32.

Phyphox

After confirming that you can connect to your board, the next step is to visualize your data. For this project, we'll be using Phyphox. You will need to (1) download the Phyphox mobile app on your intended Bluetooth client device and (2) install the Phyphox Arduino library in your Arduino IDE. Importantly, the Phyphox library simplifies the process by setting up the BLE server for you, meaning you won't need to use any code from the BLE example. For instructions on installing and configuring the Phyphox library, please refer to the GitHub page, though it should also be searchable in the Arduino IDE library manager. This library facilitates the establishment of a BLE server and the connection to the Phyphox mobile app.

Below is the step-by-step procedure for sending data from an ESP32 to the Phyphox app:

Step 1: Include the header `#include <phyphoxBle.h>` into your code

Step 2: Create a bluetooth server by adding the following line into your **void setup() {}** part of the code

```
PhyphoxBLE::start("Device Name");
```

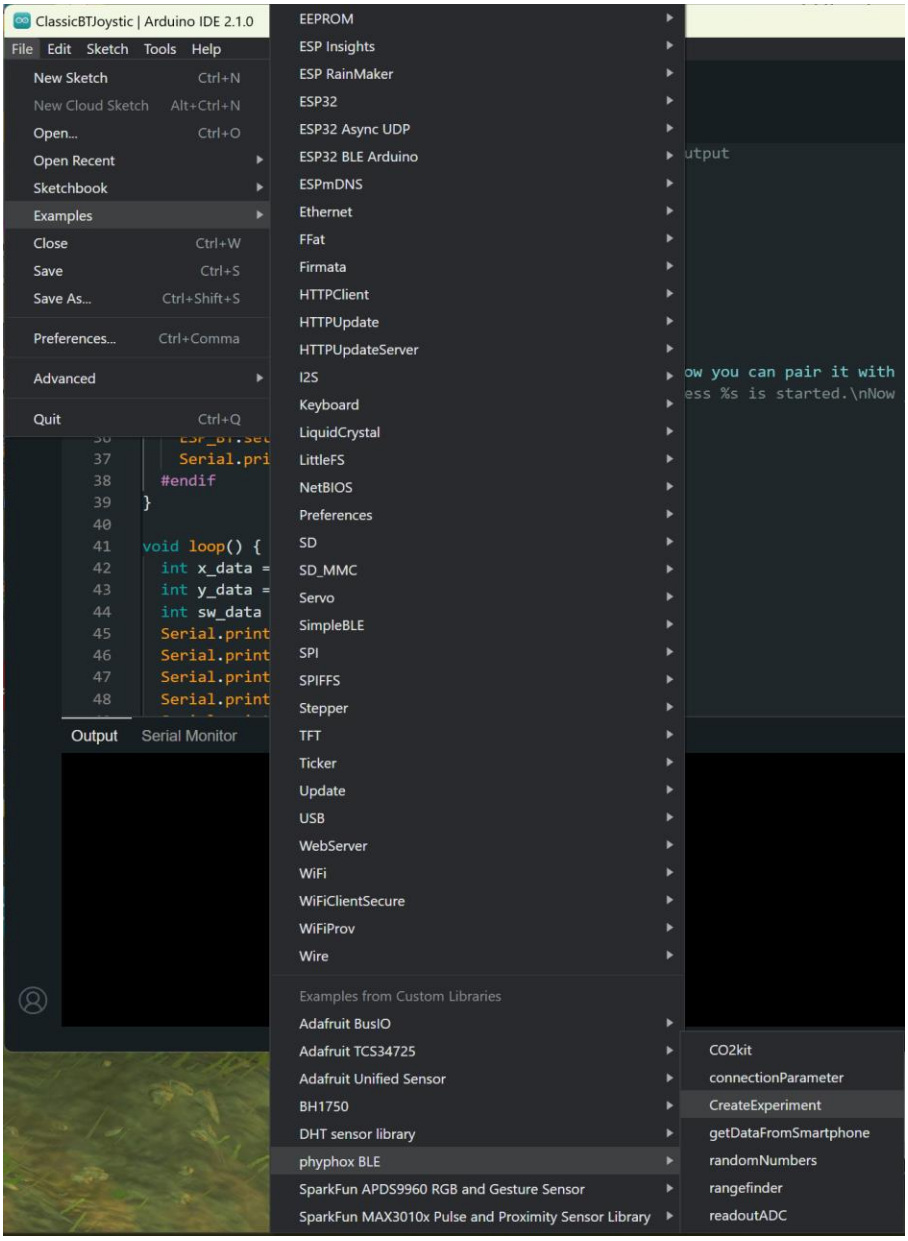
Step 3: Send the data to Phyphox app by adding the following two lines at the end of your **void loop () {}**

```
float variable_name = "data read command";
PhyphoxBLE::write(variable_name);
```

Step 4: Open Phyphox app from your smart phone. An experiment name will be displayed under "Arduino Experiments" tab. Click on the experiment name and a Bluetooth connection named "Device Name" will show up, click on it and the data will be displayed.

For a quick setup, it is recommended to reference the "Create Experiment" example (screenshot below) to get familiar with properly setting up a program to be read by Phyphox. Here is a quick video to also help familiarize you with the process:

https://www.youtube.com/watch?v=RDGyMiHUAPI



Once you have familiarized yourself with these tools, you'll be ready to start writing your program and visualizing your data in Phyphox!

Please visit TA’s office hours or contact Ahasan via email ullaha@oregonstate.edu if you have any questions.