**Documentation Report**

**By Mouhannad Chehab, Mohammad Shehab, Hussein Mourtada**

**Project Overview:** The goal of this project was to develop a "Temperature Monitoring System" using the Arduino Nano 33 BLE Sense Rev2. This system monitors temperature readings and transmits data via Bluetooth Low Energy (BLE) to a desktop GUI for real-time visualization. The purpose of the project is to demonstrate the use of the Arduino Nano 33 BLE Sense Rev2 in creating simple IoT-based monitoring systems.

**Steps Taken:**

**1. Arduino IDE Setup:**

* Downloaded and installed the latest version of the Arduino IDE from the Arduino website.
* Installed the necessary board package for the Arduino Nano 33 BLE Sense Rev2 via the Arduino Board Manager:
  + Opened Arduino IDE > Tools > Board > Boards Manager.
  + Searched for "Arduino Mbed OS Nano Boards" and installed the package.
* Installed the required libraries using the Arduino Library Manager:
  + ArduinoBLE for Bluetooth Low Energy communication.

**2. Deploying the Code to the Arduino:**

* Connected the Arduino Nano 33 BLE Sense Rev2 to the computer via USB.
* Uploaded the temperature monitoring code to the Arduino using the Arduino IDE:
  + Verified the code to ensure no syntax errors.
  + Uploaded the code successfully, confirming the "Done Uploading" message in the IDE.
* The Arduino was configured to transmit temperature readings as a BLE characteristic.

A screenshot of a computer

Description automatically generated

**3. Configuring the GUI:**

* Developed a Python-based GUI using the Tkinter library to display temperature readings in real-time.
  + The GUI included a thermometer visualization, displaying temperature data dynamically.
  + Added labels to show temperature values in both Celsius and Fahrenheit.
* Used the bleak Python library to connect to the Arduino Nano 33 BLE Sense Rev2 via BLE and retrieve temperature data.
* Implemented BLE scanning, device selection (based on the device name "BLE-TEMP"), and real-time data updates in the GUI.

A screen shot of a computer program

Description automatically generated

**4. BLE Data Transmission:**

* Configured the Arduino to advertise a BLE service for temperature monitoring.
* The GUI successfully connected to the Arduino and displayed live temperature readings on the thermometer visualization.

A screen shot of a thermometer

Description automatically generated**Screenshots and Images:**

* **GUI Display:**
  + Image of the Python GUI showing the thermometer

visualization and real-time temperature updates.

**Reflections on Challenges and Resolutions:**

**1. BLE Connectivity Issues:**

* **Challenge:** The Arduino Nano 33 BLE Sense Rev2 frequently changed its MAC address, causing reconnection issues.
* **Resolution:** Configured the GUI to connect based on the device name ("BLE-TEMP") rather than the MAC address. This ensured consistent connections.

**2. BLE Data Parsing:**

* **Challenge:** The temperature data transmitted by the Arduino was initially not formatted correctly, causing decoding issues in the GUI.
* **Resolution:** Adjusted the Arduino code to transmit temperature as a string, which was then properly parsed in the Python GUI.

**3. GUI Responsiveness:**

* **Challenge:** Real-time updates in the GUI were delayed due to inefficient handling of BLE notifications.
* **Resolution:** Optimized the BLE notification handling in the bleak library and updated the GUI rendering logic to refresh smoothly.

**Conclusion:** This project successfully demonstrated a simple temperature monitoring system using the Arduino Nano 33 BLE Sense Rev2. By integrating BLE communication with a Python GUI, the system provides a functional and user-friendly interface for real-time temperature monitoring. The challenges faced during the project provided valuable insights into BLE communication and GUI development, making this a strong foundation for future IoT-based projects.