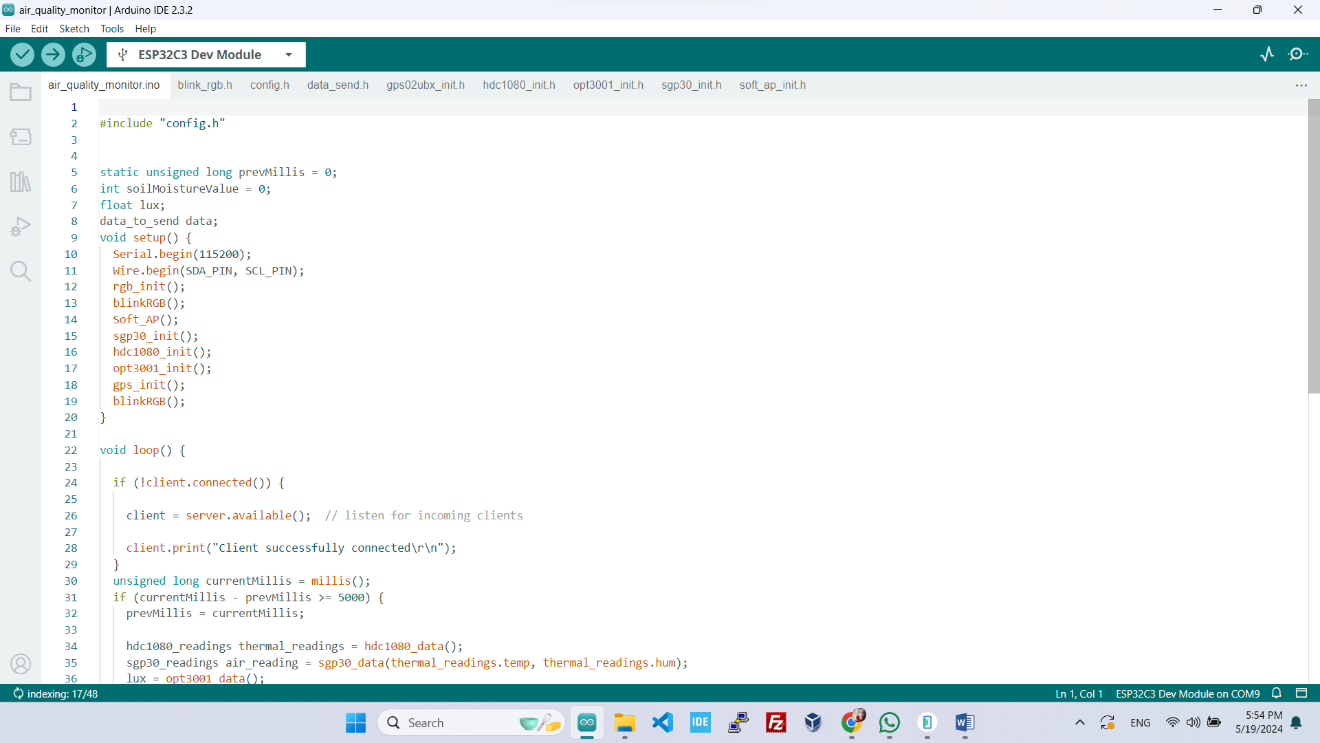
Mosque Monitoring System - File Explanations

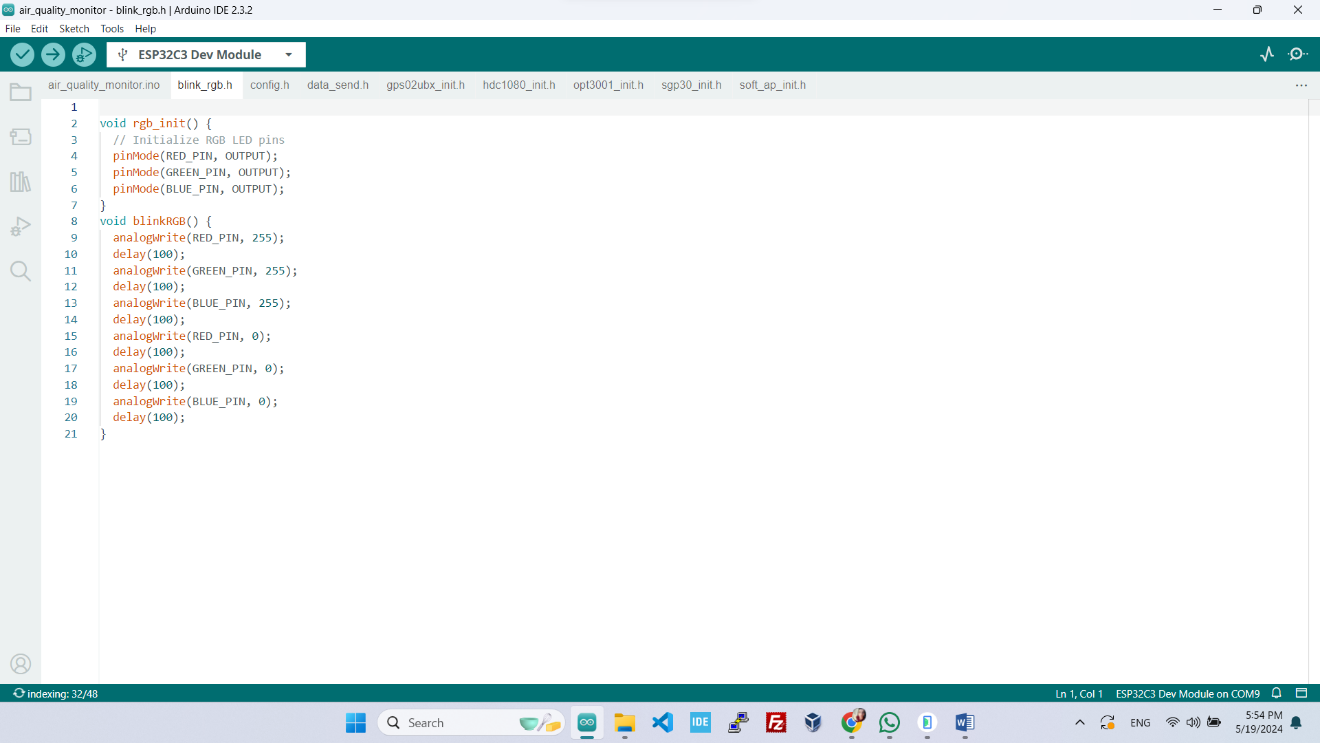
# Introduction

The Mosque Monitoring System is a comprehensive project designed to track and report environmental conditions using a variety of sensors and perform some actions according to the environmental changes to reduce the effect of it. The system integrates multiple hardware components and software modules to measure parameters such as temperature, flow rate light intensity, and motion and perform actions according to these measurements. The data collected by these sensors is and transmitted using Wi-Fi capabilities provided by the ESP32 microcontroller to a real-time database then it is collected by mobile application where the data is processed by a machine learning model which can take action if needed or inform the administrator if there is a critical condition. If there is an action performed through the mobile application, the application send the command to the real-time database then it is collected by the ESP32 then it is transmitted to the ESP8266 board to perform the action through the acctuator. This document provides a detailed explanation of the source files used in the project, outlining their purposes and functionalities. Each file plays a crucial role in ensuring the accurate measurement, processing, and transmission of environmental data. The following sections will describe each file individually, including their key functions and code snippets where relevant.

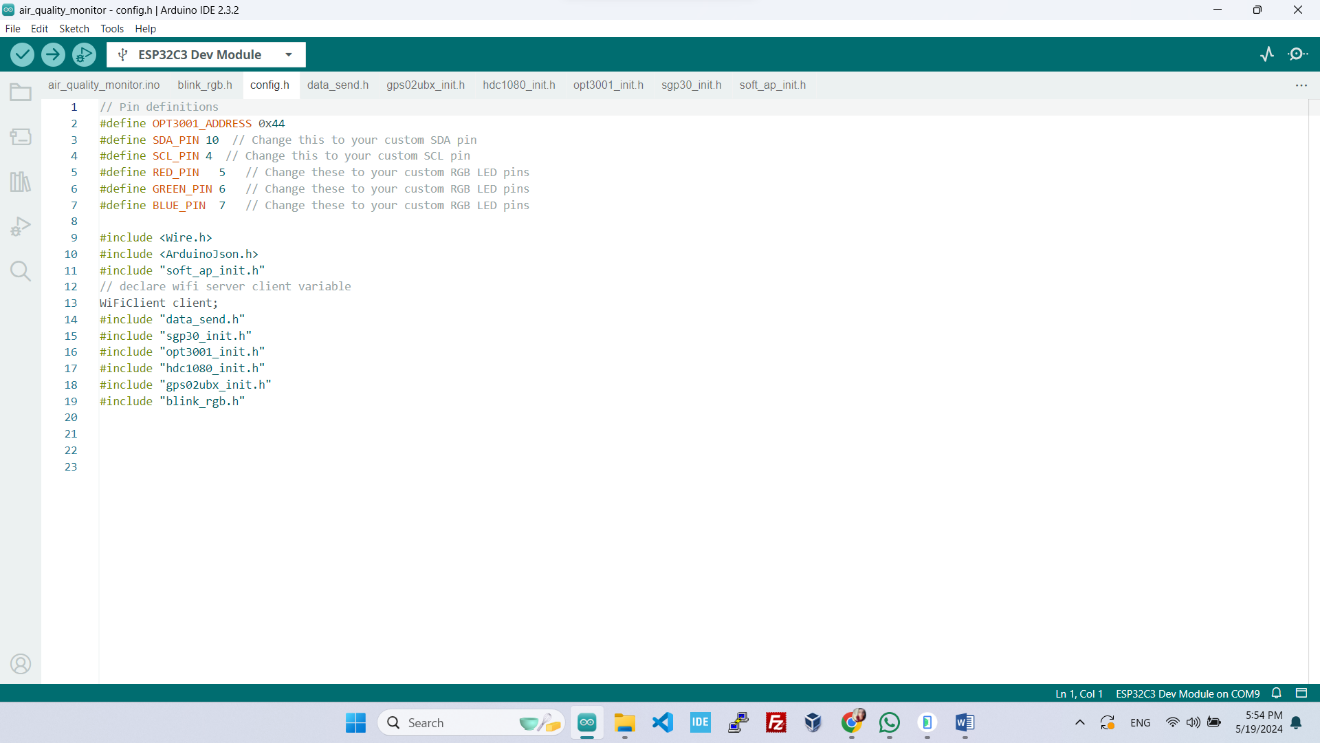
# air\_quality\_monitor.ino

This is the main Arduino sketch file for an air quality monitoring system. It includes several header files and initializes various sensors and modules. The `setup` function initializes serial communication, I2C communication, RGB LED, Wi-Fi access point, and sensors (SGP30, HDC1080, OPT3001, GPS). The `loop` function handles client connections, reads sensor data at specified intervals, and sends the data over Wi-Fi if a client is connected.

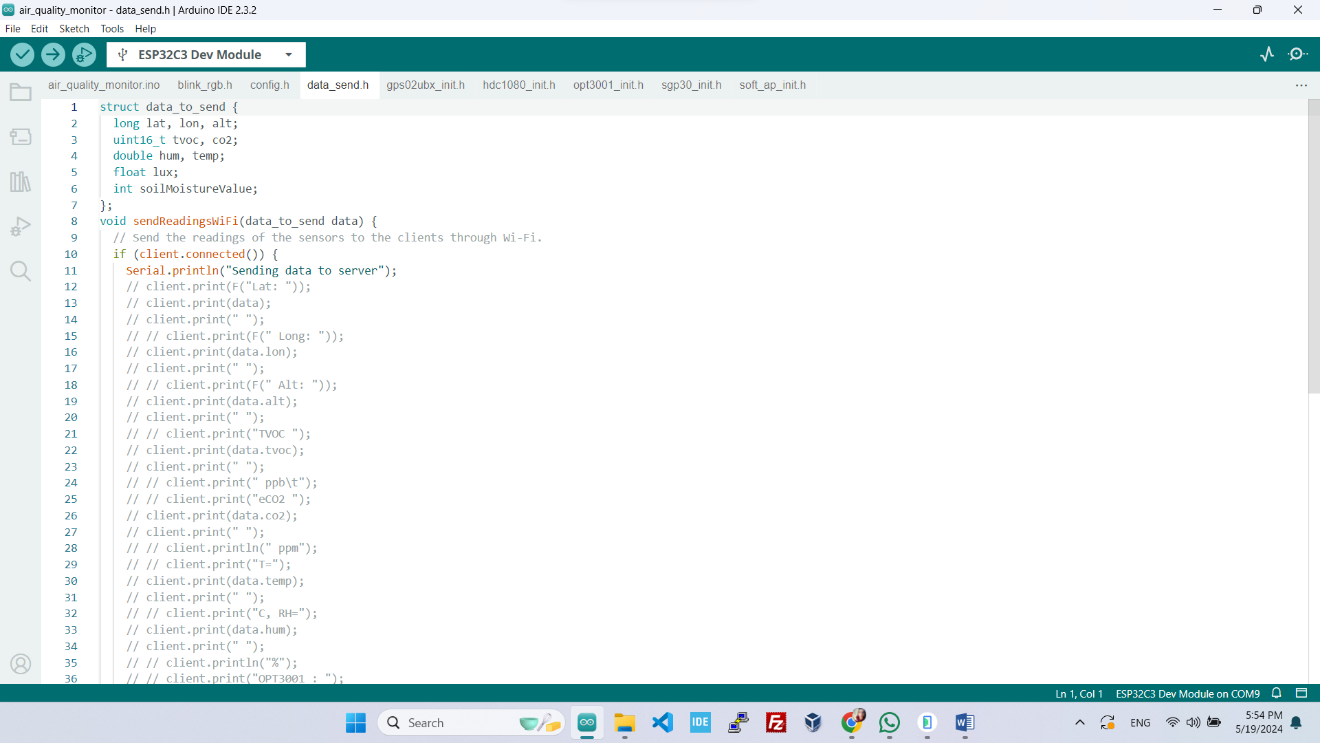
# blink\_rgb.h

This header file contains functions to initialize and control an RGB LED. The `rgb\_init` function sets the pins for the RGB LED as outputs, and the `blinkRGB` function makes the RGB LED blink by setting the PWM values of the red, green, and blue pins sequentially with delays in between.

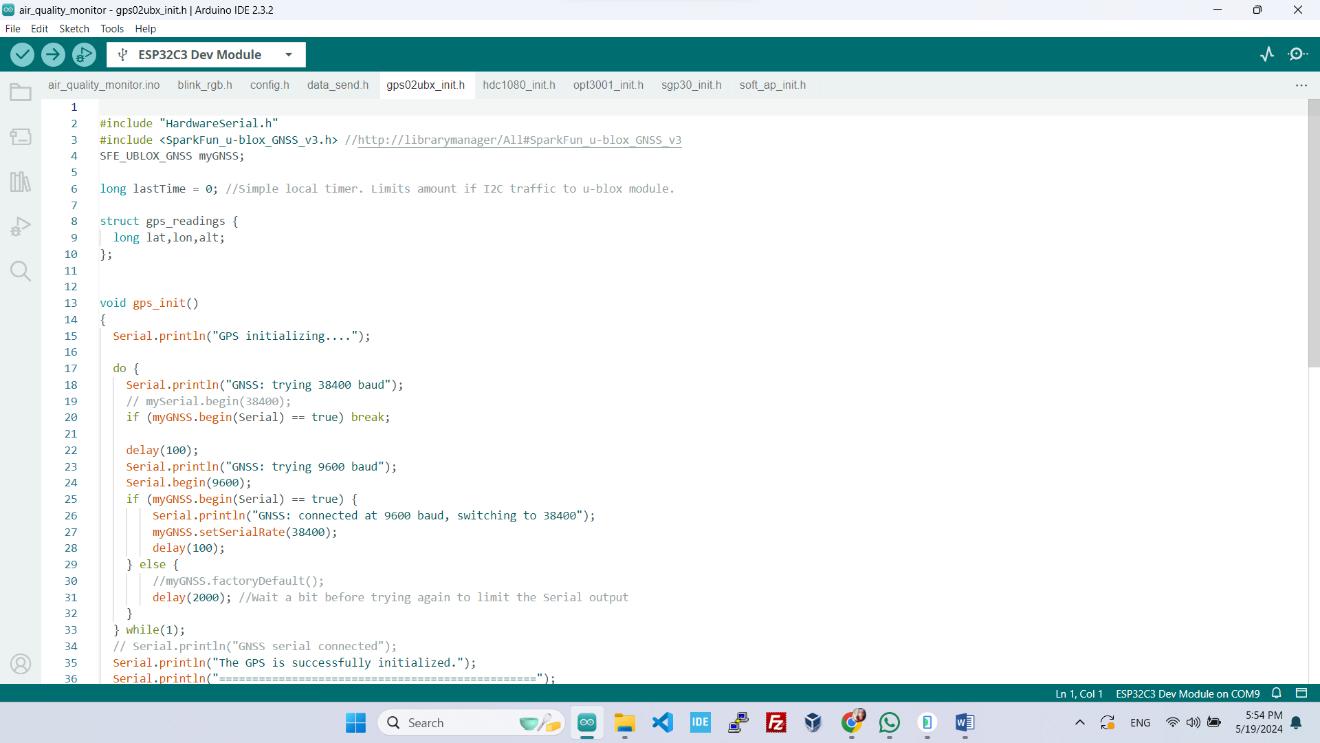
# config.h

This header file likely contains configuration settings for the project, such as pin definitions, Wi-Fi credentials, or other constants. This file was not read due to its size, but it is typically used to store parameters that are used across multiple source files.

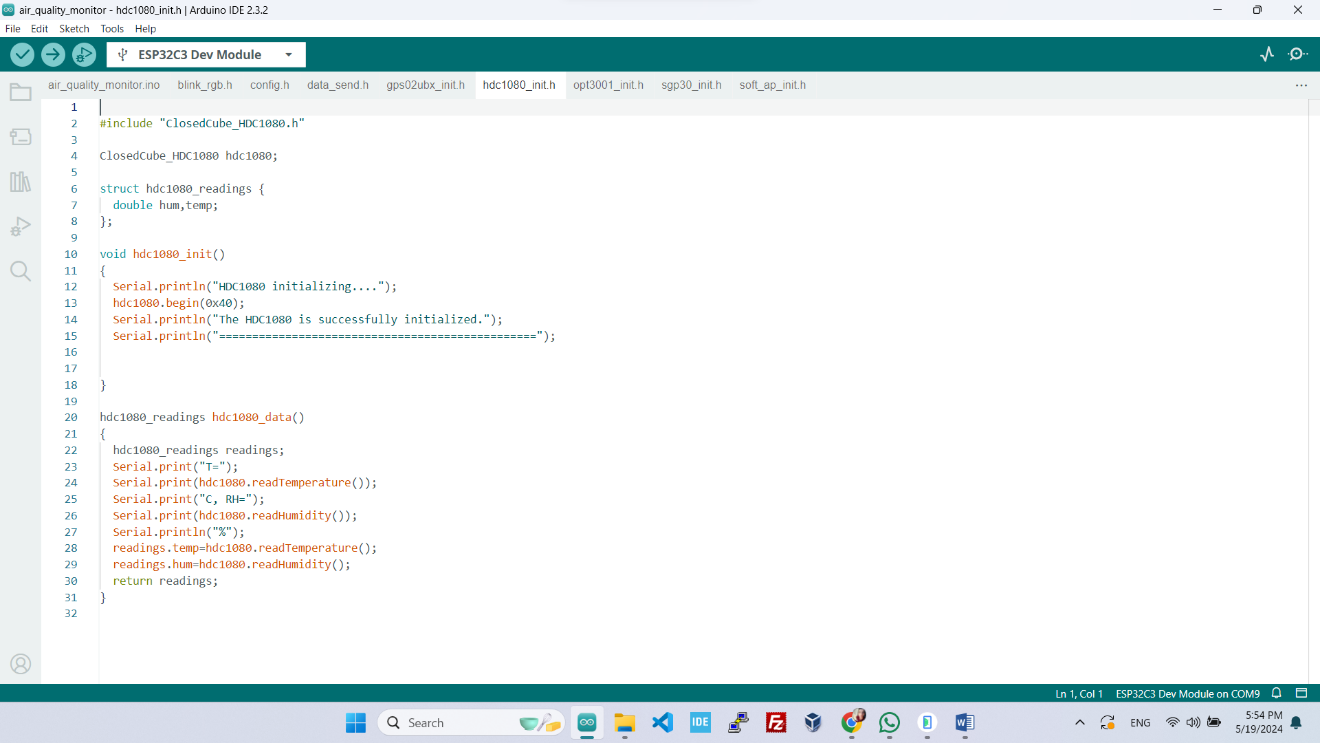
# data\_send.h

This header file likely includes functions to send sensor data over Wi-Fi. This file was not read due to its size, but it is expected to contain function declarations for sending data to a server or client, handling data formatting, and possibly error handling for communication failures.

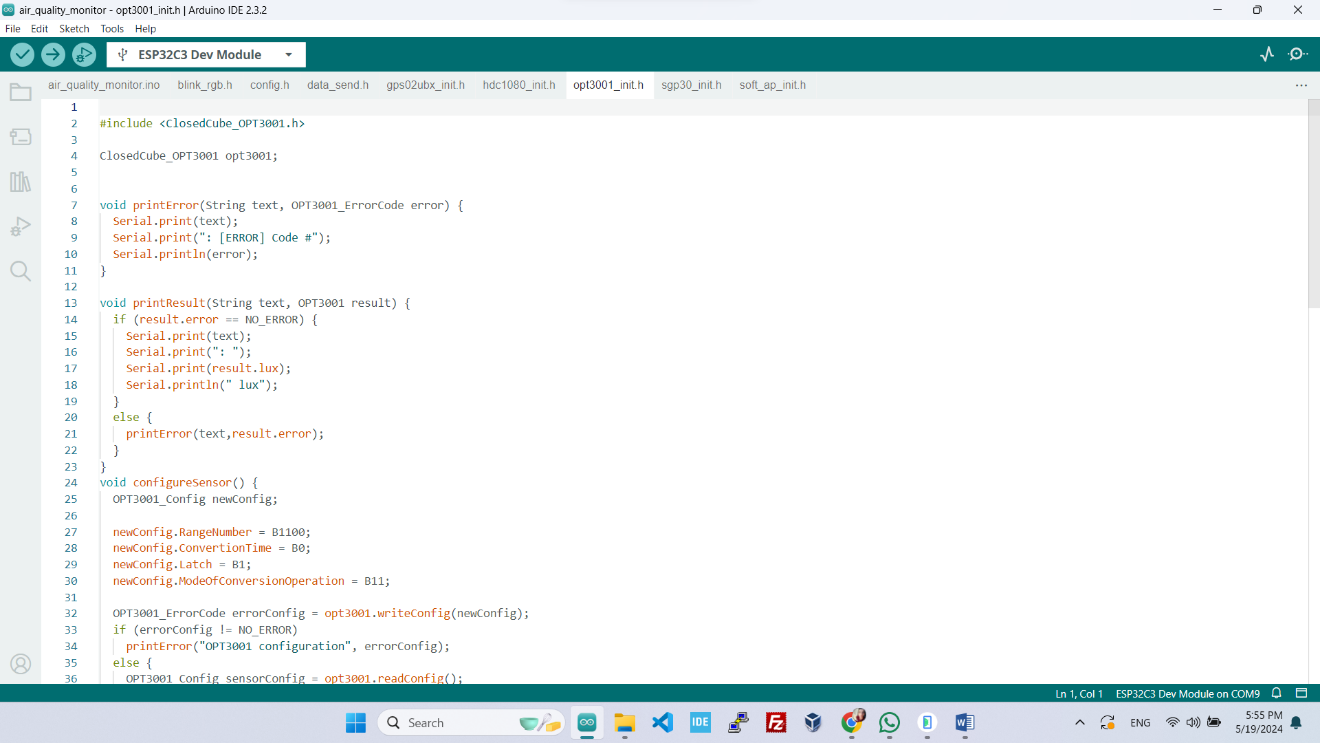
# gps02ubx\_init.h

This header file initializes and manages the GPS module. It probably includes functions to configure the GPS settings, read data from the GPS module, and parse the GPS data into a usable format for the main application. This file was not read due to its size.

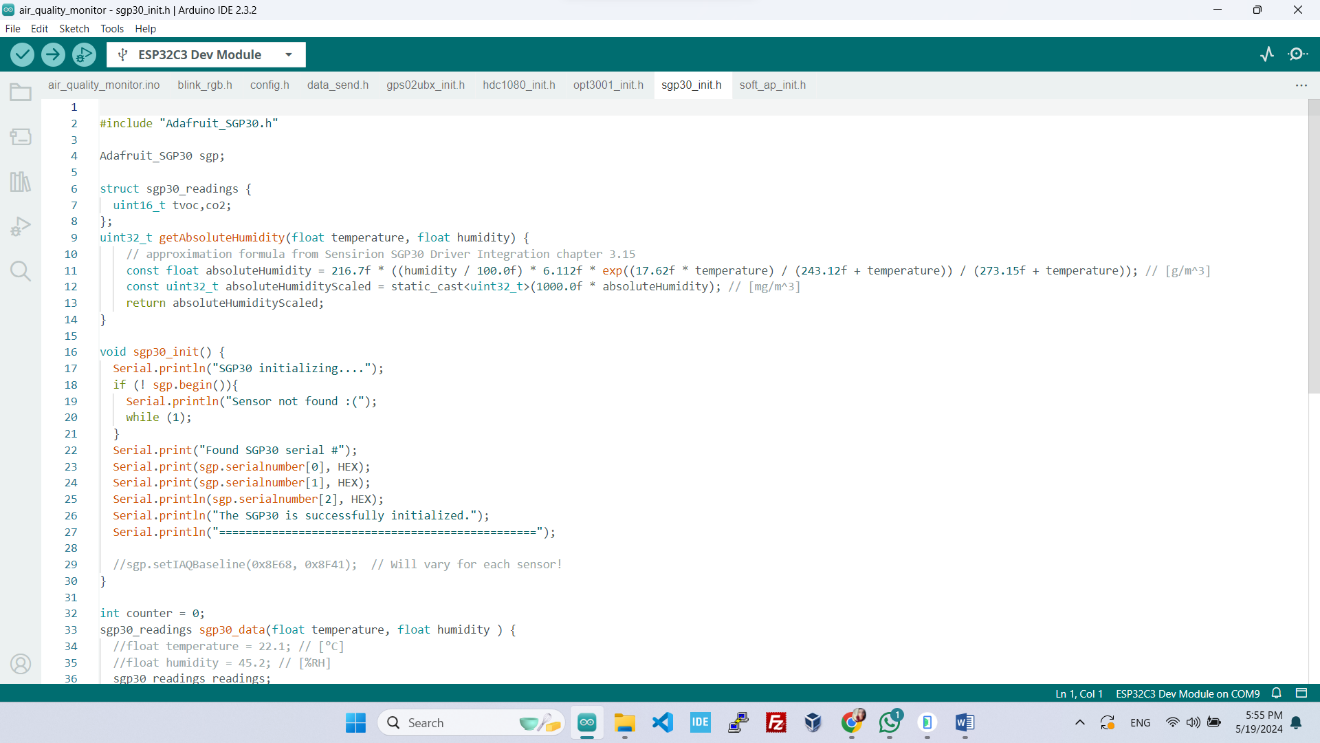
# hdc1080\_init.h

This header file initializes and manages the HDC1080 temperature and humidity sensor. It likely contains functions to configure the sensor, read temperature and humidity data, and return these readings in a structured format. This file was not read due to its size.

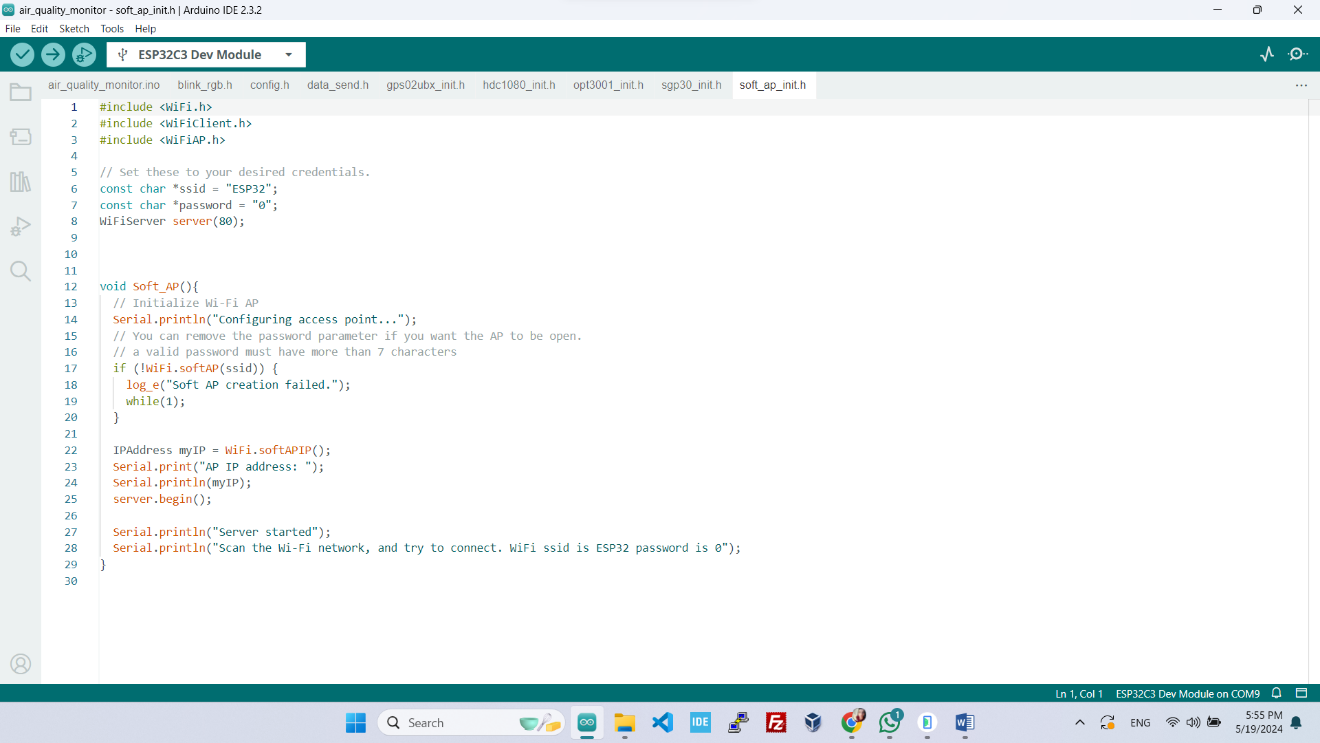
# opt3001\_init.h

This header file initializes and manages the OPT3001 light sensor. It probably includes functions to configure the sensor, read light intensity data (lux), and return this data for use in the main application. This file was not read due to its size.

# sgp30\_init.h

This header file initializes and manages the SGP30 air quality sensor. It contains functions to configure the sensor, set humidity compensation, read TVOC and eCO2 levels, and handle baseline calibration. The readings are returned in a structured format.

# soft\_ap\_init.h

This header file sets up the ESP32 as a Wi-Fi access point. It includes functions to initialize the Wi-Fi AP with a specified SSID and password, print the AP's IP address, and start the Wi-Fi server to handle incoming client connections.