Name: Mohamed Ads

Student ID: 900142564

Project #4

Simple IoT Application

**Description**:

In this project we are going to utilize the ESP8266 module to create a small IoT

application that can enable the user to perform I/O operations with the STM32 module

through a web interface.

**Deliverables**:

The I/O operations include:

* Retrieving the date/time from RTC module connected to the STM32 module
* Control the STM32 module LEDs status

**To build this application:**

• The ESP8266 module will be programed to act as a WiFi Access Point and run a simple HTTP server.

• The ESP8266 module communicates with the STM32 module using Asynchronous Serial link (using UART)

• The STM32 module runs code that receives commands from the ESP8266 module to control the LEDs and to retrieve the current time/date from the i2c RTC module attached to it.

**Project Plan:**

What I figured so far:

Hardware components to be used:

* STM32L432KC
* Wifi Module: ESP8266MOD
* RTC Module: DS3231

Software components to be used:

* Arduino IDE
* Programming Language: Python

* CH340 Driver
* STM32 Nucleo Board: STM32CubeMX

Time plan:

What I figured so far:

**Tuesday 5/5:**

* Figure the needed software and hardware components

**Thursday 5/7:**

* Start working on establishing a connection between the http server and the wifi module

Architecture:

A screenshot of a social media post

Description automatically generated

Implementation:

## Web Server

Making a web server using Arduino C++ which uses listener functions on port 80 for HTTP web server communications and the second on the UART.

Sample code that handles the ESP LED

void handleLED() { // If a POST request is made to URI /LED

digitalWrite(led,!digitalRead(led)); // Change the state of the LED

server.sendHeader("Location","/"); // Add a header to respond with a new location for the browser to go to the home page again

server.send(303); // Send it back to the browser with an HTTP status 303 (See Other) to redirect

}

Using Keil to Control the Green LED Using Uart