Introduction

In this lab, you will install and configure a web server with a filesystem on your IoT device and creating an app to display sensor data on your phone.

Objectives

- Create and run an HTTP web sever on the IoT Device
- Display IoT sensor data and display it on your phone's browser.

Prerequisites

- COMP-10184 Lab Kit
- Visual Studio Code: https://code.visualstudio.com/
- PlatformIO extension for VS Code.
- Third-party libraries: AsyncTCP, ESPAsyncWebServer
- Webserver tutorial

Lab Setup

Wire all the kit sensors as per ** **All Kit Devices** **

in the **Course Resources** module

Creating a Basic Web Server

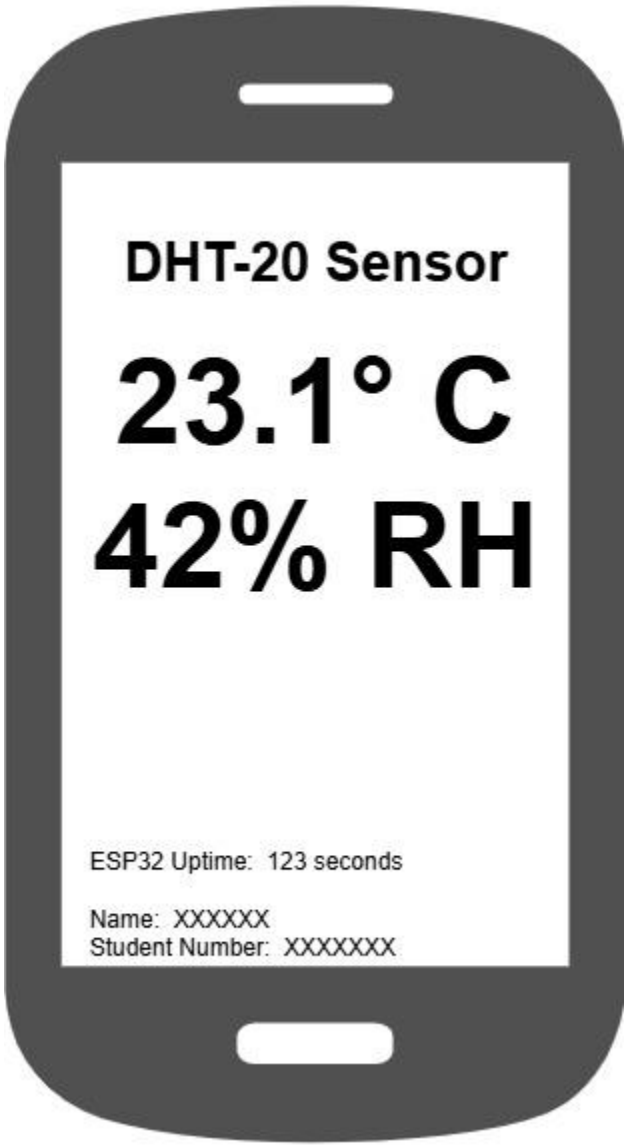
Work through the **Web Server and LittleFS Tutorial** available on Canvas. When complete you will have a webserver with a small file system serving static HTML files from your IoT device.

Extending the Web Server

1. Extend the tutorial by creating an app to dynamically update the DHT-20 sensor's temperature/humidity values and the ESP32 uptime (in seconds). Remove unnecessary HTML and JPG files from the tutorial.

2. Your temperature/humidity/uptime values should update every second. Use some JavaScript/AJAX for this – DO NOT use **meta http-equiv="refresh"** tags!

3. Add CSS in a separate file to properly format the output. Here are the minimum display requirements:



Tips

- The millis() function returns the number of milliseconds since the ESP32 was booted. You can scale that to give you uptime in seconds.
- You may need to add **DOCTYPE** and a **<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">** tag to get Safari to display HTML.

Results

1. Demonstrate your working application to your instructor. If you are unable to do that in class, submit a link to a video (1 min max!) as a comment with your code submission.

2. Include a **Statement of Authorship** in **main.cpp.**

3. Ensure your WiFi credentials are in an **#include** file (e.g. #incliude "secrets.h").

DO NOT INCLUDE WIFI CREDENTIALS IN MAIN.CPP!

4. Rename **main.cpp** to **main.cpp.txt**.

5. Submit the **main.cpp.txt** and any HTML, JS, CSS files from your project to Canvas.

6. Download and complete the Lab Report, submitting your completed file to Canvas.

Make sure to reconnect your phone to the main college network after completing this lab!

Grading:

See Canvas for a detailed grading rubric.