Trac Duc Anh Luong - 103488117

Assignment 2: Individual Assignment (Practical)

1. Link to video

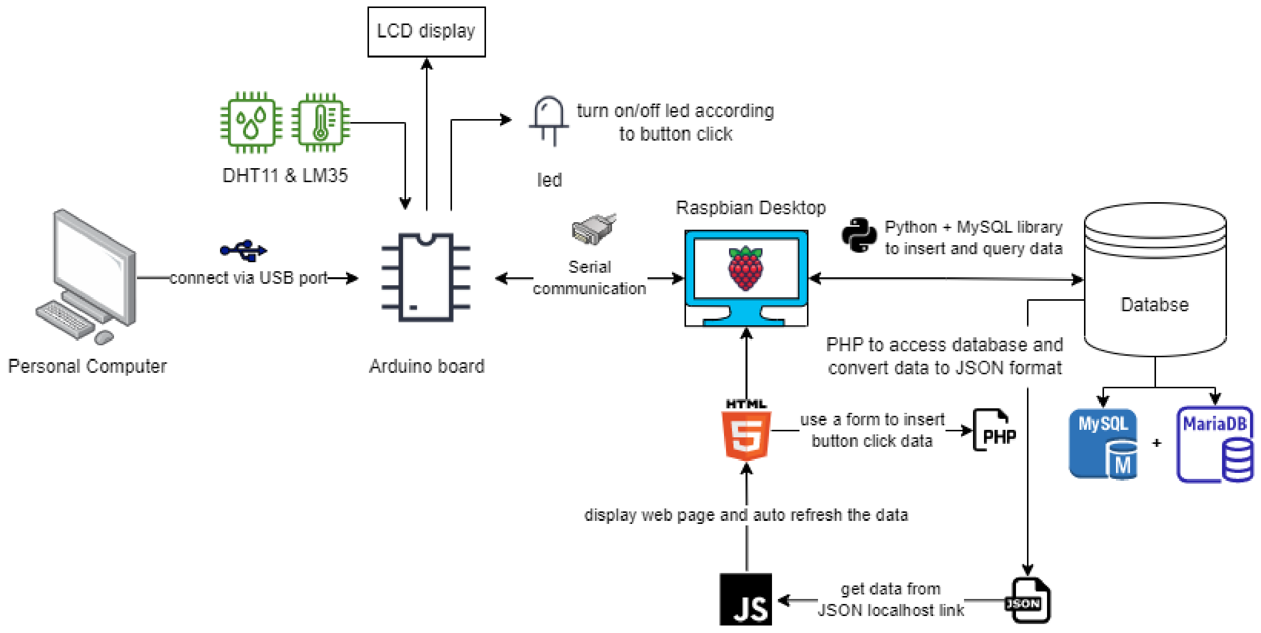
https://drive.google.com/file/d/1_KIERrIgEPMbg1Rp-8aBiZn-rDw0aSpY/view?usp=sharing

2. Summary

The chosen system for this assignment is a Weather Station system, using DHT11 and LM35 as the sensors to record the temperature and humidity. The DHT11 and LM35 module is widely available and affordable, and the entire system can be developed for a broader range of more complex and practical applications for weather monitoring, agriculture, exploitation, etc. Modern-day industries and their product storages heavily rely on a certain range of temperatures and humidity. DHT11's flexibility of usage can be proven by its capability of communicating with Arduino using both digital and analog signals. The accuracy is decent for its price, with an error range of 2 degrees Celsius and 5% humidity. On the other hand, LM35 has a better temperature range of minus 55 degrees to 150 degrees, making it durable and fit for a variety of applications, such as power supplies, battery management, HVAC (Heating, Ventilation, and Air Conditioning), and other household electronic appliances. As LM35 is an analog sensor, I will be using the DHT11 as a digital sensor to satisfy the requirement of this assignment.

The two sensors are connected to an Arduino board to read and print data. The temperature and humidity are printed on an LCD 16*2 screen as an actuator. The information and warnings will be displayed on the LCD screen along with another actuator: a led to flash at a frequency of 1 tick per second. A Raspbian Desktop will read the data from the Serial port and insert it into a MySQL database, which then will be displayed in a web UI hosted on Apache localhost. The UI also serves as a form, which can send data to the database and control the actuators connected to the Arduino board.

3. Conceptual Design


LCD display
DHT11 & LM35
turn on/off led according to button click
led
Raspbian Desktop
Python + MySQL library to insert and query data
Databse
Personal Computer
connect via USB port
Arduino board
Serial communication
PHP to access database and convert data to JSON format
HTML
use a form to insert button click data
PHP
MySQL
+
MariaDB
display web page and auto refresh the data
JS
get data from JSON localhost link
JSON

The Arduino board receives the temperature from DHT11 and prints it to a 16x2 LCD. The system then sends the temperature and humidity to Raspberry Pi using Serial communication. The Raspian desktop inserts the received data into a MySQL database using Python and available libraries, including serial and MySQLdb. To display the data to a localhost web page, we need to use PHP to query the data, and JavaScript libraries like JQuery and Chart.js to format the data as charts to a scripting language like HTML.

We can interact with the IoT node connected to Arduino from the user interface by changing the conditional rule. Under normal conditions, when the temperature exceeds 30 degrees Celsius and/or the humidity exceeds 80%, the LCD will print out an alert according to which attribute is being triggered (HIGH TEMPERATURE and HIGH HUMIDITY). Also, the led will blink with a 1-second delay. The user interface on the Raspian desktop has 2 radio buttons and a “Submit” button. The UI now serves as a form, and you can send the command option to a new table in the database. The Python program continuously loops and reads the last value inserted into the table. If you submit “Turn off”, the database will add a new “0” record. This value will be sent to Arduino through Serial communication and turn the led off. Vice versa, you can turn the led on while a warning is displayed on the LCD by submitting “Turn on”.

Please note that the Weather Dashboard automatically reloads every 30 seconds to fetch new data from the database.

4. Implementation

a. Sensors

- LM35 (analog)
- DHT11 (digital)

b. Actuators

- LCD 16*2
- led

c. Software and Libraries

- Softwares:
    - Proteus: for Arduino, sensors, and actuators simulation.
    - Arduino IDE: to write C++ instruction code, compile it and send it to the Arduino board.
    - Oracle Virtualbox: Host the Raspbian Desktop virtual machine.
    - Visual Studio Code: To write frontend (UI) and backend (database) related code.
- Libraries
    - Python
        - MySQLdb: to establish a connection to the MySQL database, insert and query data from the database.
        - Serial: receive and send Serial signal (bidirectional).
        - sys: catch exceptions from the inputted data (empty strings, value errors, keyboard interrupts, etc.).
        - time: delay the program so that it does not insert thousands of records every minute.
    - JavaScript:
        - Chart.js: To interpret structured data into charts (in my project, a line graph).
        - JQuery: To use the $.ajax() function and return XMLHttpRequest object (in this case, PHP queries after they have been formatted into JSON).
    - PHP: to interact with the database (insert and query data), print the data in JSON format so that JQuery can read from a localhost link.
    - HTML: to display the UI.

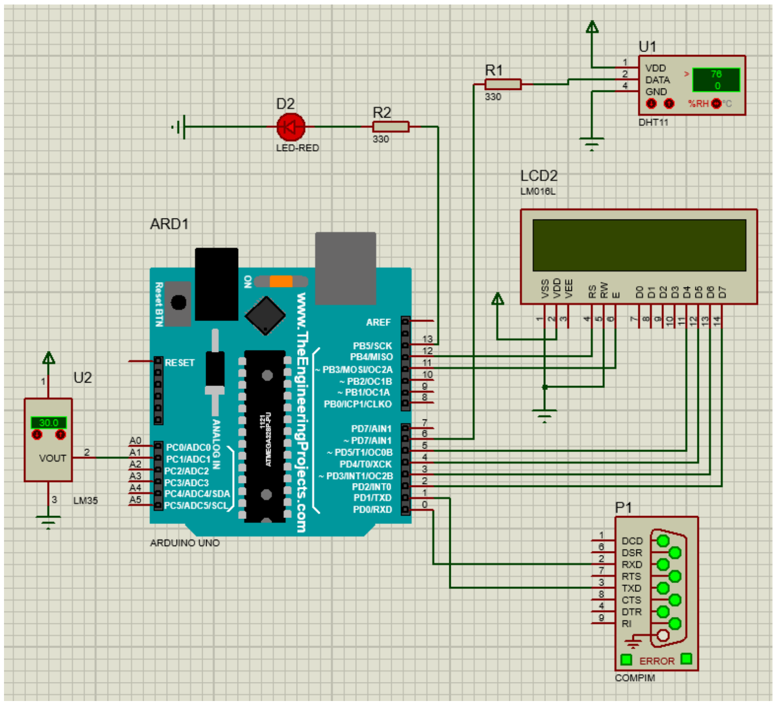
5. Resources

This project was created with the help of materials from the Canvas module, Mr Khoa's slides, and the following documentation and tutorials:

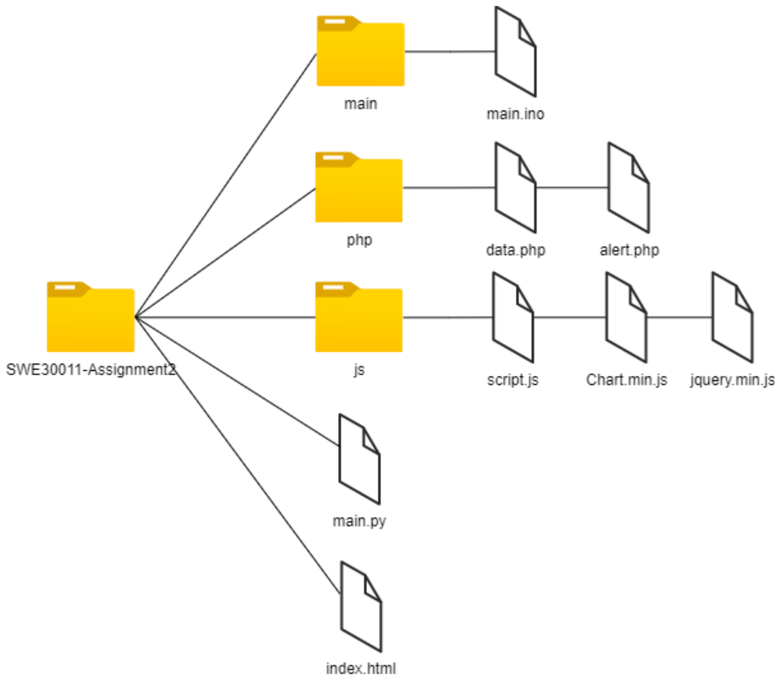
- https://projecthub.arduino.cc/arcaegecengiz/12f621d5-055f-41fe-965d-a596fcc594f6
- https://pythonhosted.org/pyserial/shortintro.html
- https://mysqlclient.readthedocs.io/user_guide.html
- https://www.mysqltutorial.org/mysql-basics/
- https://www.w3schools.com/php/func_mysqli_query.asp
- https://www.chartjs.org/docs/latest/charts/line.html
- https://www.geeksforgeeks.org/how-to-insert-form-data-into-database-using-php/
- https://stackoverflow.com/questions/32913226/auto-refresh-page-every-30-seconds

6. Appendix

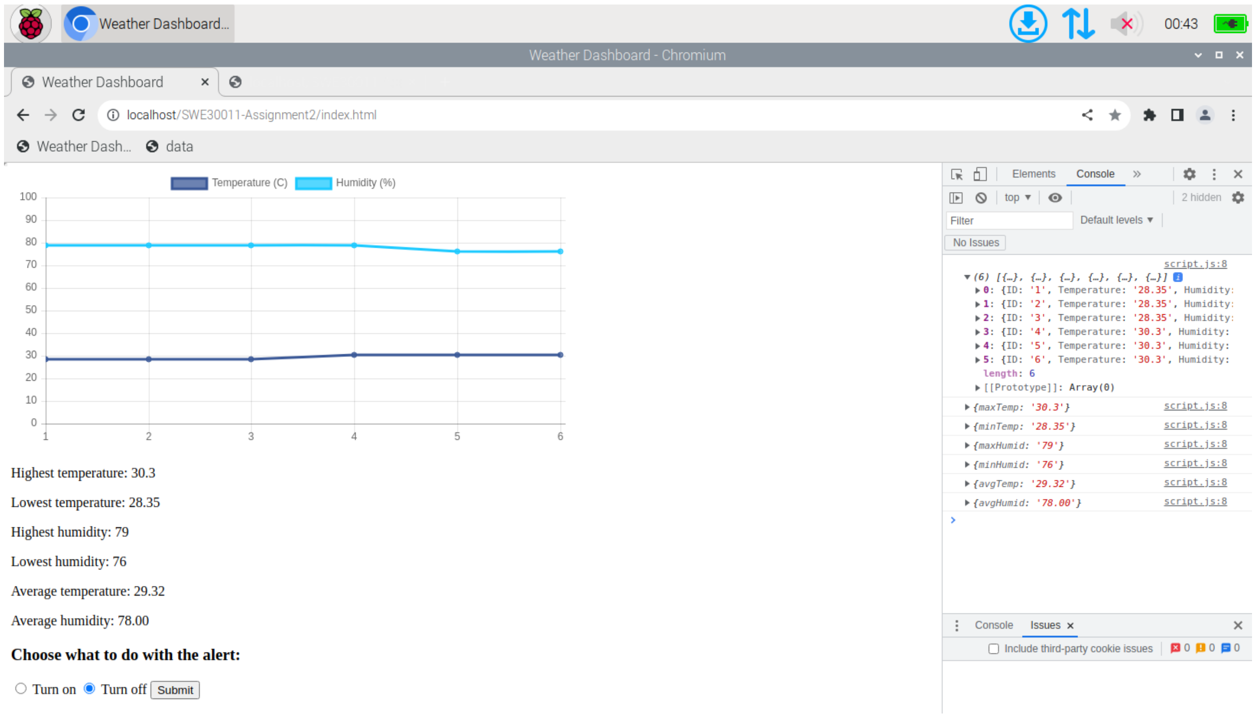
Sketch of the IoT node system on Arduino (Proteus simulation)



File system (on Raspbian Desktop)



Weather dashboard that displays the data as a line chart, other analysis (highest, lowest, average), and a button to trigger an actuator (led) connected to the IoT node:



Highest temperature: 30.3

Lowest temperature: 28.35

Highest humidity: 79

Lowest humidity: 76

Average temperature: 29.32

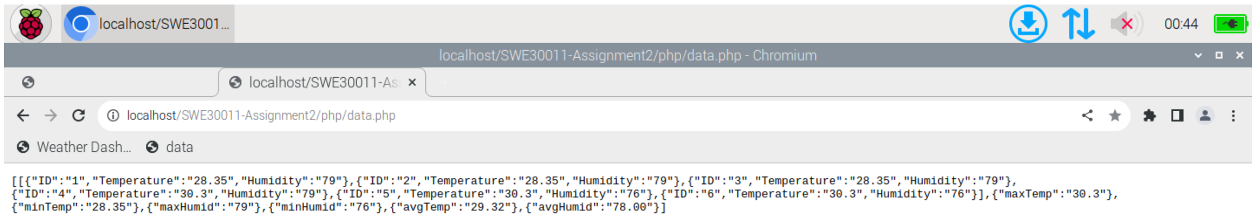
Average humidity: 78.00

Choose what to do with the alert:

○ Turn on [✓ Turn off] Submit

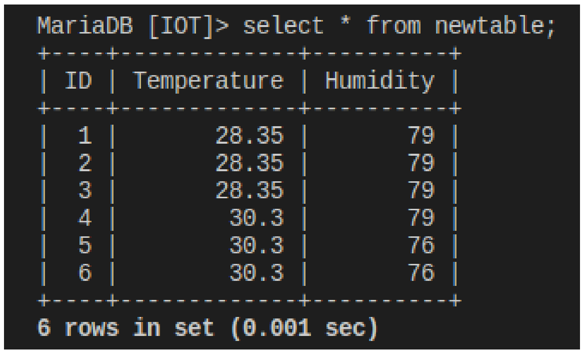
```
                                                            script.js:8
▼(6) [{…}, {…}, {…}, {…}, {…}, {…}]
  ▶0: {ID: '1', Temperature: '28.35', Humidity:
  ▶1: {ID: '2', Temperature: '28.35', Humidity:
  ▶2: {ID: '3', Temperature: '28.35', Humidity:
  ▶3: {ID: '4', Temperature: '30.3', Humidity:
  ▶4: {ID: '5', Temperature: '30.3', Humidity:
  ▶5: {ID: '6', Temperature: '30.3', Humidity:
   length: 6
  ▶[[Prototype]]: Array(0)
▶{maxTemp: '30.3'}                                          script.js:8
▶{minTemp: '28.35'}                                         script.js:8
▶{maxHumid: '79'}                                           script.js:8
▶{minHumid: '76'}                                           script.js:8
▶{avgTemp: '29.32'}                                         script.js:8
▶{avgHumid: '78.00'}                                        script.js:8
>
```

JSON data returned from data.php file:

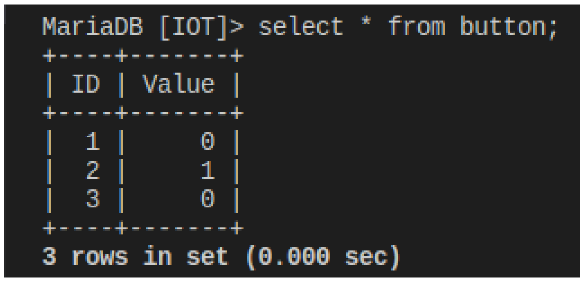

localhost/SWE3001...
00:44
localhost/SWE30011-Assignment2/php/data.php - Chromium
localhost/SWE30011-As
localhost/SWE30011-Assignment2/php/data.php
Weather Dash...
data

[[{"ID":"1","Temperature":"28.35","Humidity":"79"},{"ID":"2","Temperature":"28.35","Humidity":"79"},{"ID":"3","Temperature":"28.35","Humidity":"79"},
{"ID":"4","Temperature":"30.3","Humidity":"79"},{"ID":"5","Temperature":"30.3","Humidity":"76"},{"ID":"6","Temperature":"30.3","Humidity":"76"}],{"maxTemp":"30.3"},
{"minTemp":"28.35"},{"maxHumid":"79"},{"minHumid":"76"},{"avgTemp":"29.32"},{"avgHumid":"78.00"}]

Queries of tables in MySQL database:



```
MariaDB [IOT]> select * from newtable;
+----+-------------+----------+
| ID | Temperature | Humidity |
+----+-------------+----------+
|  1 |       28.35 |       79 |
|  2 |       28.35 |       79 |
|  3 |       28.35 |       79 |
|  4 |        30.3 |       79 |
|  5 |        30.3 |       76 |
|  6 |        30.3 |       76 |
+----+-------------+----------+
6 rows in set (0.001 sec)
```



```
MariaDB [IOT]> select * from button;
+----+-------+
| ID | Value |
+----+-------+
|  1 |     0 |
|  2 |     1 |
|  3 |     0 |
+----+-------+
3 rows in set (0.000 sec)
```