UAB COLLEGE OF ARTS AND SCIENCES

Department of Computer and Information Sciences

Please include the following declaration together with your signature on the cover page of your submissions/reports:

(1) If it's an individual assignment, use the following text for the declaration of independent completion:

I Tyler Woodby declare that I have completed this assignment completely and entirely on my own, without any consultation with others. I understand that any breach of the UAB Academic Honor Code may result in severe penalties.

(2) If it's a group assignment, use the following text for the declaration of independent completion:

We declare that we have completed this assignment completely and entirely on our own, without any consultation with others. We understand that any breach of the UAB Academic Honor Code may result in severe penalties.

We also declare that the following percentage distribution ***faithfully*** represents individual group members' contributions to the completion of the assignment:

Name	Contribution (%)	Signature	Date
Tyler Woodby	login/signup 25%		8/10/2020
Wesley Hatavany	Data visualization CSS 25%		8/11/2020
Jude Agan	login/signup HTML/CSS 25%		8/11/2020
Andrew Bertella	microcontrolled Database 25%		8/11/2020

Project Report

Arduino Tracker

Team members:

Andrew Bertella

Tyler Woodby

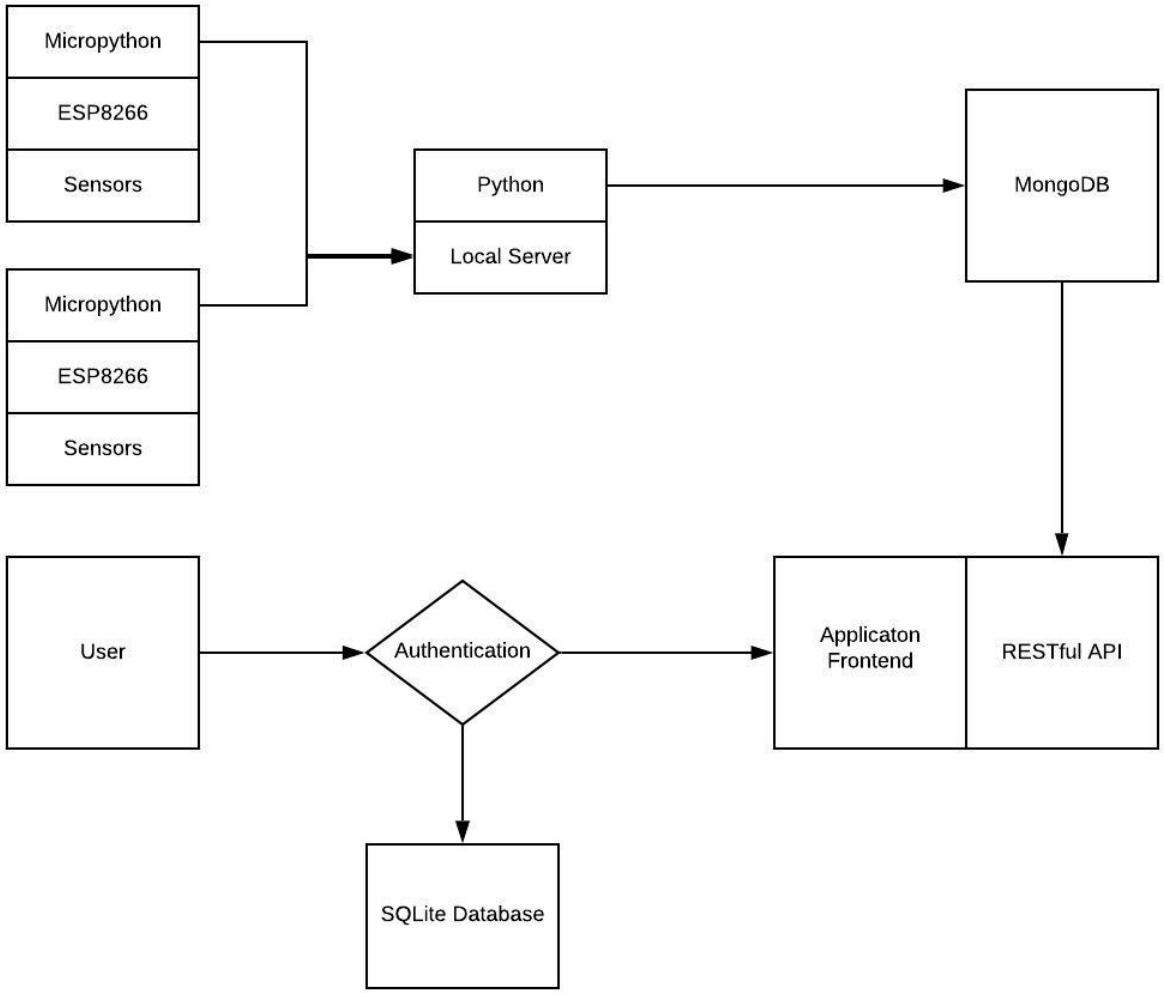
Jude Agan

Wesley Hataway

Project Title: Arduino Tracker

Summary
This project implements a simple IOT event tracking device that could be useful for Nest-like applications.
We used an ESP8266 Node MCU and a humidity sensor to prove the application concept.
The events from the microcontroller are posted to a MongoDB instance which can then be accessed and visualized through the application frontend.

Block Diagram



Technology

Frontend	Backend	Microcontroller
Flask	MongoDB/Flask-Pymongo	Micropython
CSS	Python	JSON
HTML		Python - sockets
Bootstrap		
Chart.js		
jquery		

Set-up

See requirements.txt for the required dependencies. This application relies on the mongo daemon running on the database in the db folder. Run mongod dbpath=./db in the application directory.

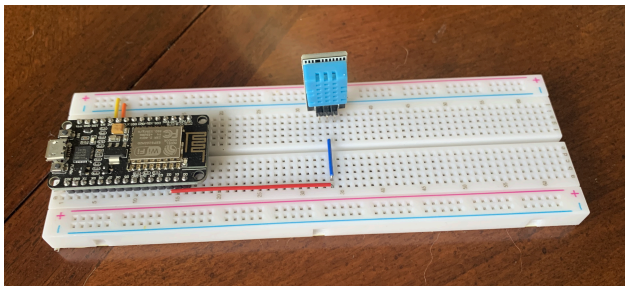
Challenges

This application is a proof of concept on the implementation of and IOT based Nest-like data driven internet application. Some areas that need improvement are the use of more than one device per user. As the application stands, no user may have more than on microcontroller on their application. The server.py file is only able to connect by one socket. Implementing a socket assigner that assigns a socket to a microcontroller and notifies the local server would be useful to this end.

A second challenge is authentication. There is nothing in the application that would prevent a user form reading database data that does not belong to them. The use of sandboxed collections in Mongo or some type of end to end encryption or a third-party authentication service may be useful.

This project also suffers from the lack of support for other sensor types other than a humidity/temperature sensor. Perhaps an area for future development would be the implementation of a library that allows the transmission of JSON event data for several sensor types (eg movement, vibration, light intensity, soil moisture etc).

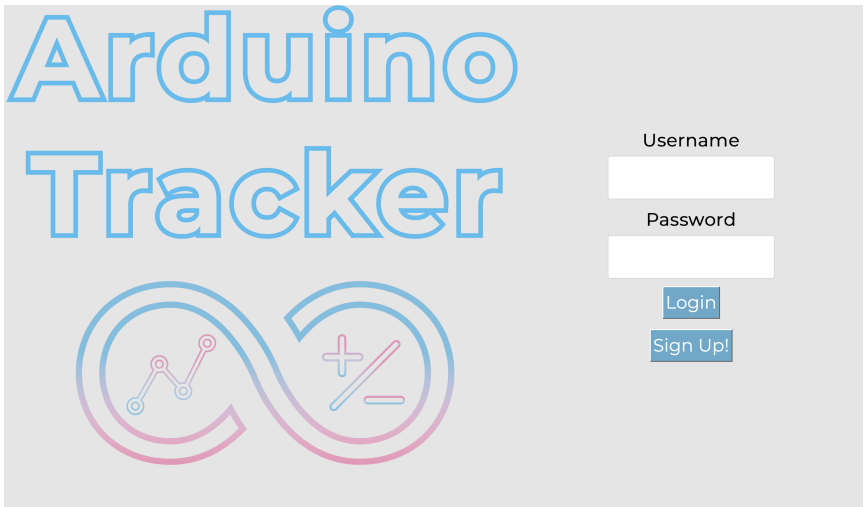
Screenshots/Photos



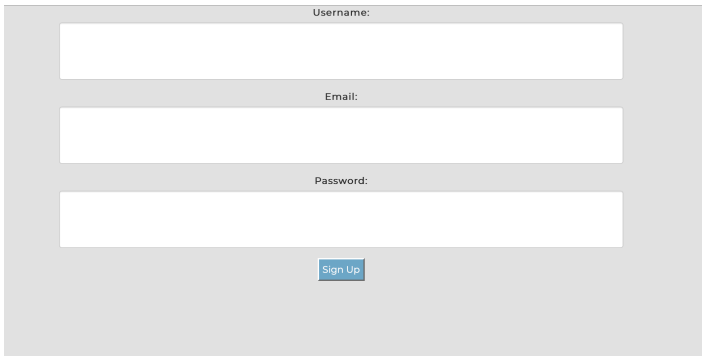
The microcontroller/sensor setup

```
Brians-MBP:CS421_Project brianbertella$ python3 server.py
Connected
b'{"temperature": 84.2, "id": "1", "humdity": 95.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:42Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cefc8>
b'{"temperature": 78.8, "id": "1", "humdity": 81.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:43Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cef08>
b'{"temperature": 78.8, "id": "1", "humdity": 85.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:45Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428ced08>
b'{"temperature": 78.8, "id": "1", "humdity": 84.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:46Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cee48>
b'{"temperature": 78.8, "id": "1", "humdity": 84.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:47Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cef08>
b'{"temperature": 78.8, "id": "1", "humdity": 83.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:48Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8419f2988>
b'{"temperature": 78.8, "id": "1", "humdity": 83.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:50Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cee48>
b'{"temperature": 78.8, "id": "1", "humdity": 83.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:51Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cef08>
b'{"temperature": 78.8, "id": "1", "humdity": 83.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:52Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cec08>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:54Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cee48>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:55Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428ced08>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:56Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cef48>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:57Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cee48>
b'{"temperature": 78.8, "id": "1", "humdity": 81.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:05:59Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428ced08>
b'{"temperature": 78.8, "id": "1", "humdity": 81.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:00Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cef48>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:01Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cee48>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:03Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428ced08>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:04Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cee48>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:05Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cec08>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:06Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cecc8>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:08Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cee48>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:09Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cec08>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:10Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cecc8>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:12Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cee48>
b'{"temperature": 78.8, "id": "1", "humdity": 82.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:13Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cec08>
b'{"temperature": 78.8, "id": "1", "humdity": 81.0, "hostname": "04:83:d0:00", "timestamp": "2020-08-11T14:06:14Z"}'
<pymongo.results.InsertOneResult object at 0x7fb8428cee48>
```

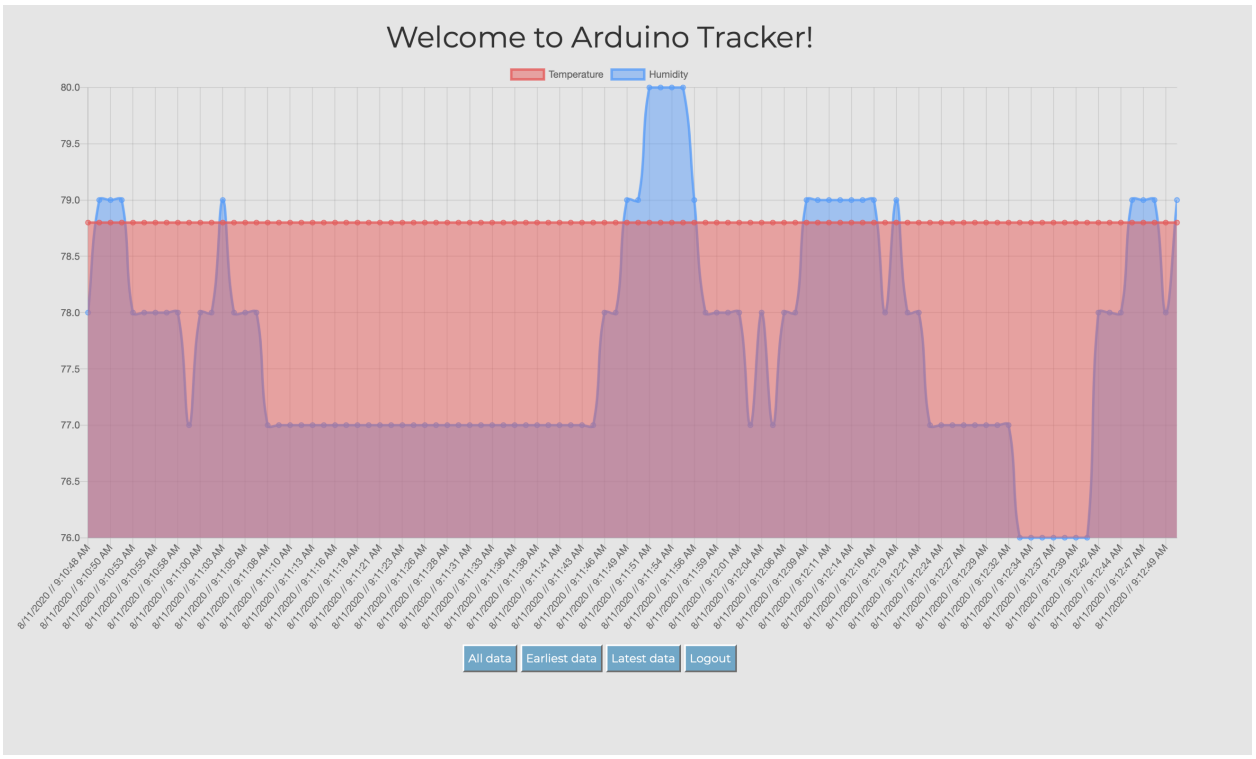
The local server receiving sensor data and storing it in MongoDB



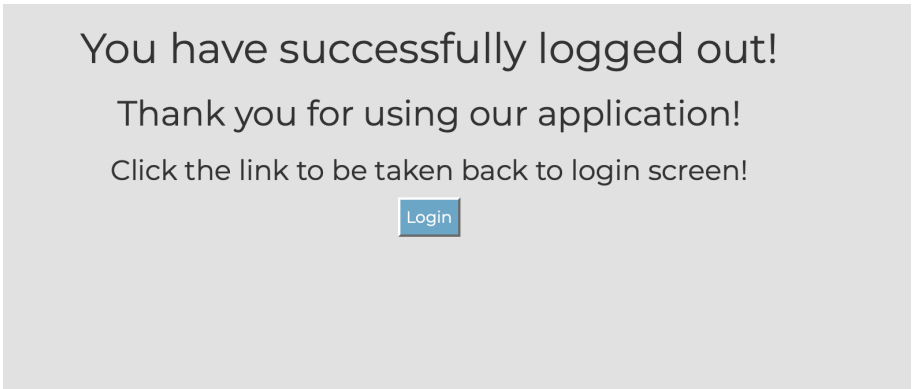
Login Page



Signup Page



Home Page with real-time data visualization



References

The gettime function in Microcontroller/main.py was adapted from https://github.com/micropython/micropython/blob/master/ports/esp8266/modules/ntptime.py

Adapted for retrieving MongoDB data for the visualization: https://www.youtube.com/watch?v=oVVpF7bIDAE&list=LLlagrUzulbKwIBOAuSepQzQ&index=2&t=330s