

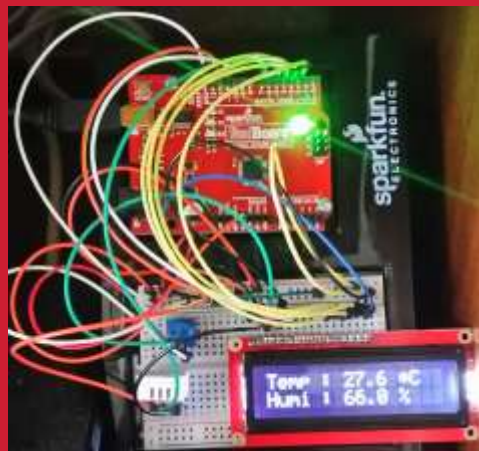


# Arduino-IoT

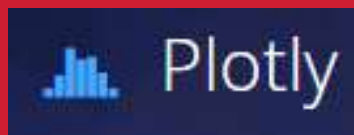
[wk12]

Arduino + Node

Data mining



Visualization of Signals using Arduino,  
Node.js & storing signals in MongoDB  
& mining data using Python



Drone-IoT-Comsi, INJE University

2<sup>nd</sup> semester, 2021

Email : chaos21c@gmail.com



# My ID

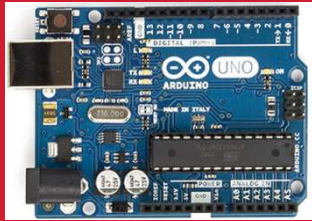
## ID를 확인하고 github에 repo 만들기

AA01	김준수	AA13	조재윤
AA02	김현서	AA14	고태승
AA03	박영훈	AA15	이한글
AA04	박윤호	AA16	장세진
AA05	성은지	AA17	장태호
AA06	손윤우	AA18	정지원
AA07	오세윤	AA19	진우태
AA08	우승철	AA20	황혁준
AA09	윤현석	AA21	장이제
AA10	이예주	AA22	박상현
AA11	강지환	AA23	정은성
AA12	성인제	AA24	김경영

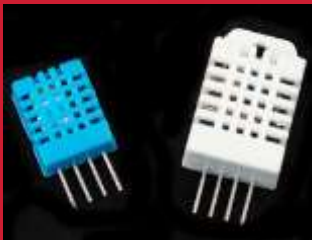
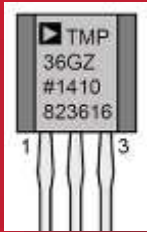
위의 id를 이용해서 github에 repo를 만드시오.

Option: 아두이노응용 실습 과제 - AAnn

Public, README.md check



# [Review]



## ◆ [wk11]

- RT Data visualization with MongoDB
- Multi-sensor circuits (cds-dht22)
- Complete your project
- Upload folder: aann-rpt11
- Use repo “aann” in github

# wk11 : Practice : aann-rpt11

## ◆ [Target of this week]

- Complete your works
- Save your outcomes and upload outputs in github

제출폴더명 : **aann-rpt11**

- 제출할 파일들

- ① **AAnn\_iot\_mongodb\_web.png**
- ② **AAnn\_iot\_json.png**
- ③ **AAnn\_iot\_client.png**
- ④ **All \*.js**
- ⑤ **public/All \*.html**
- ⑥ **public/data/All data (\*.csv)**

# Purpose of AA

주요 수업 목표는 다음과 같다.

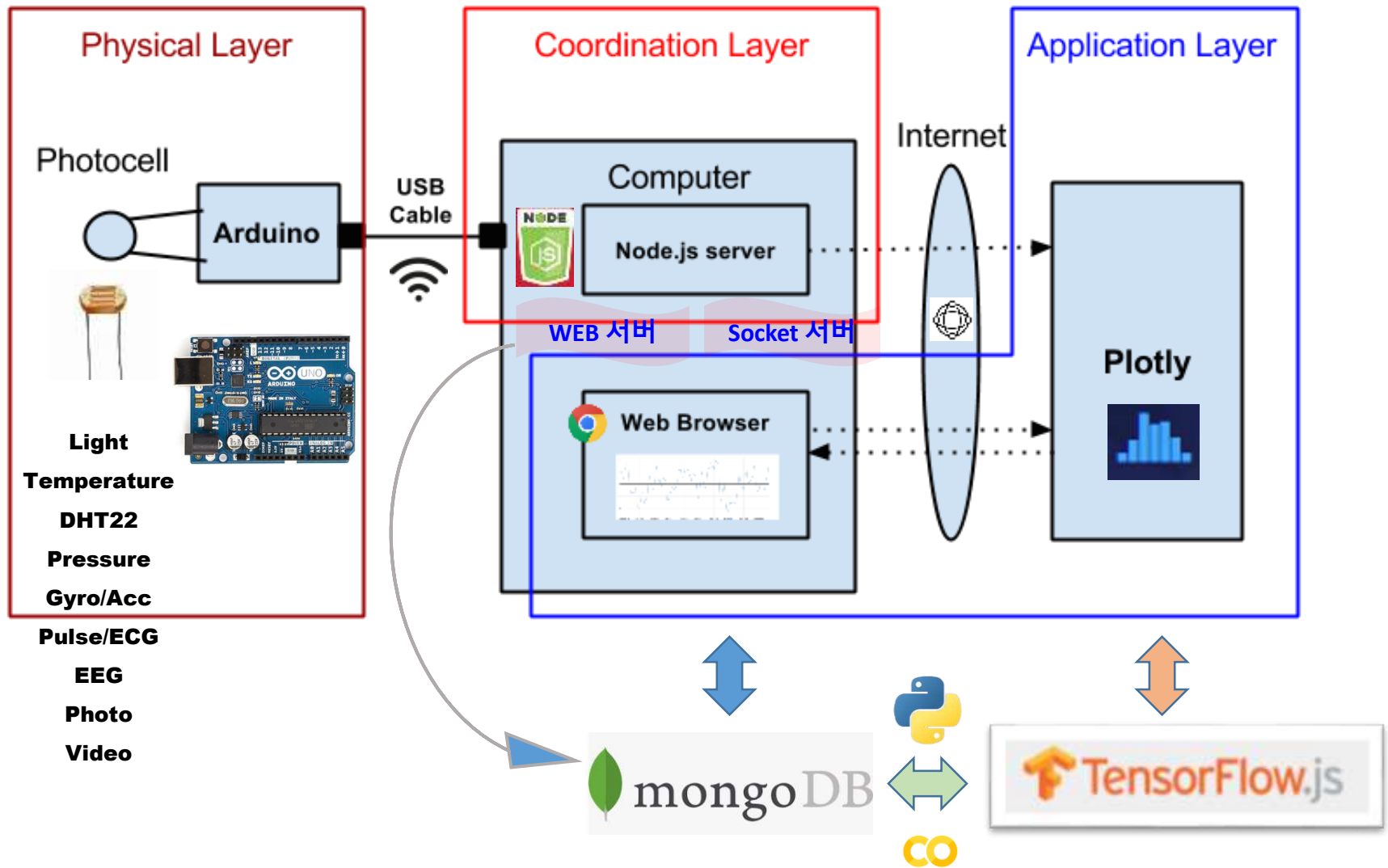
1. Node.js를 이용한 아두이노 센서 신호 처리
2. Plotly.js를 이용한 아두이노 센서 신호 시각화
3. MongoDB에 아두이노 센서 데이터 저장 및 처리



## 4. 저장된 IoT 데이터의 마이닝 (파이썬 코딩)



# Layout [H S C]





# DHT22 + CdS + Node.js + MongoDB

**Network socket server : port=3000**

```
D:\aann\aann-rpt12\cds_dht22>node cds_dht22_mongodb
serial port open
mongo db connection OK.
iotInfo: Current date: 2021-11-17 10:15:23.935, Temp: 21.1, Humi: 33.2, Lux: 87
iotInfo: Current date: 2021-11-17 10:15:26.213, Temp: 21.1, Humi: 33.2, Lux: 87
iotInfo: Current date: 2021-11-17 10:15:28.490, Temp: 21.1, Humi: 33.2, Lux: 87
iotInfo: Current date: 2021-11-17 10:15:30.767, Temp: 21.1, Humi: 33.2, Lux: 87
iotInfo: Current date: 2021-11-17 10:15:33.045, Temp: 21.1, Humi: 33.2, Lux: 87
iotInfo: Current date: 2021-11-17 10:15:35.322, Temp: 21.1, Humi: 33.2, Lux: 87
iotInfo: Current date: 2021-11-17 10:15:37.605, Temp: 21.1, Humi: 33.2, Lux: 87
```

█

**Express server : port=3030**

```
D:\aann\aann-rpt12\cds_dht22>node cds_dht22_express
Express_IOT with CORS is running at port:3030
mongo db connection OK.
```

█

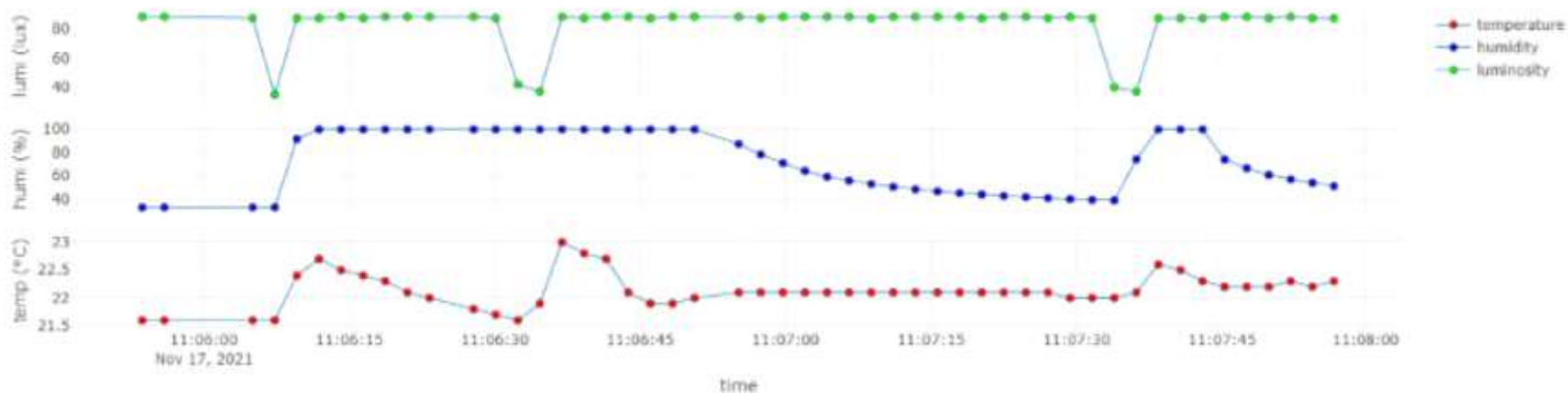


Web monitoring: [http://localhost:3030/client\\_cds\\_dht22.html](http://localhost:3030/client_cds_dht22.html)

## Real-time Weather Station from sensors



on Time: 2021-11-17 11:07:56.680





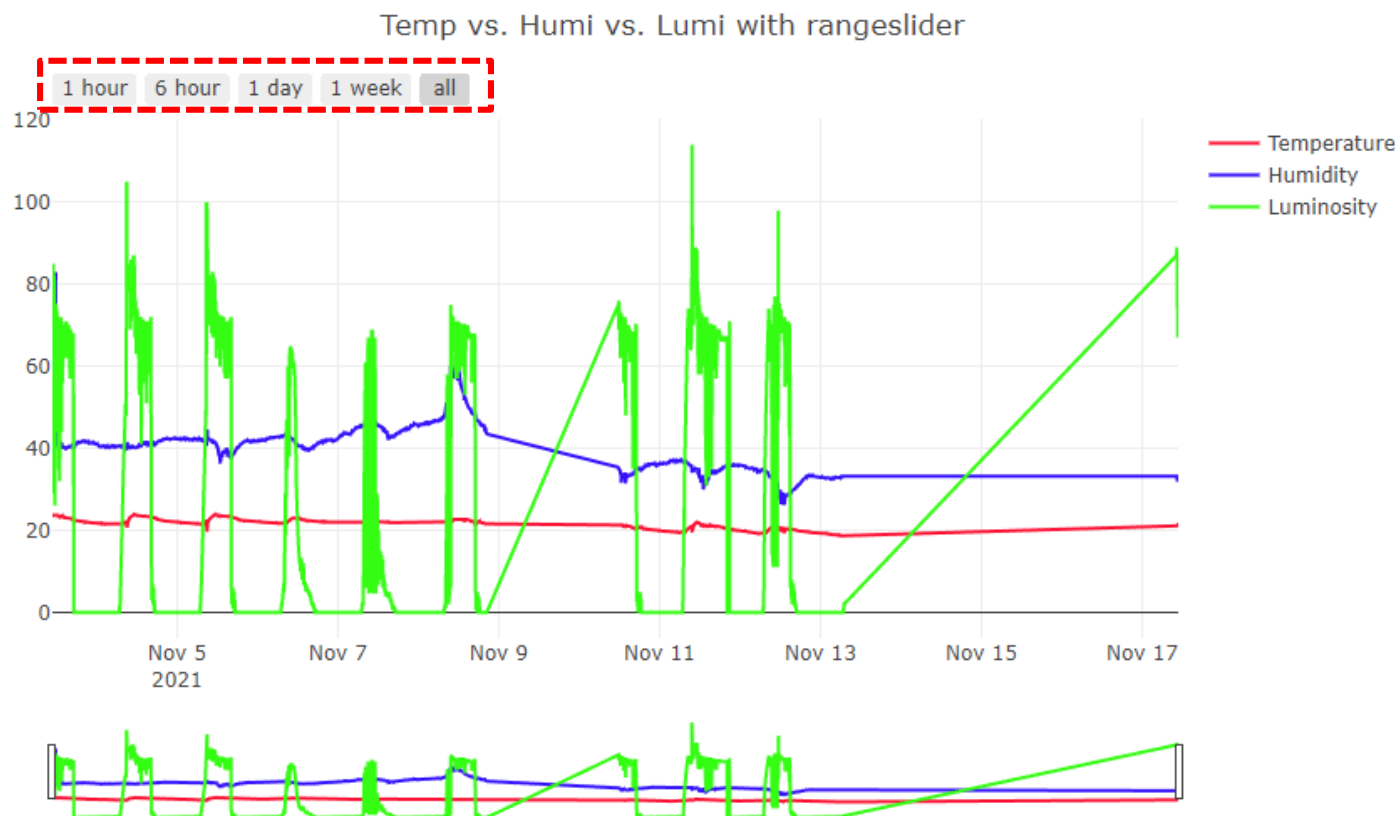
# MongoDB shell: mongo

```
> show dbs
admin 0.000GB
config 0.000GB
iot 0.013GB
iot10 0.000GB
local 0.000GB
test 0.000GB
> use iot
switched to db iot
> show collections
sensors
> db.sensors.find().sort({'_id':-1}).limit(10)
{ "_id" : ObjectId("619466cc0374269b9cc2cd3b"), "date" : "2021-11-17 11:19:56.573", "temperature" : "21.6", "humidity" : "32.4", "luminosity" : "85",
  "_v" : 0 }
{ "_id" : ObjectId("619466ca0374269b9cc2cd39"), "date" : "2021-11-17 11:19:54.296", "temperature" : "21.7", "humidity" : "32.4", "luminosity" : "86",
  "_v" : 0 }
{ "_id" : ObjectId("619466c80374269b9cc2cd37"), "date" : "2021-11-17 11:19:52.018", "temperature" : "21.7", "humidity" : "32.4", "luminosity" : "85",
  "_v" : 0 }
{ "_id" : ObjectId("619466c50374269b9cc2cd35"), "date" : "2021-11-17 11:19:49.741", "temperature" : "21.7", "humidity" : "32.4", "luminosity" : "85",
  "_v" : 0 }
{ "_id" : ObjectId("619466c30374269b9cc2cd33"), "date" : "2021-11-17 11:19:47.463", "temperature" : "21.6", "humidity" : "32.3", "luminosity" : "85",
  "_v" : 0 }
{ "_id" : ObjectId("619466c10374269b9cc2cd31"), "date" : "2021-11-17 11:19:45.186", "temperature" : "21.6", "humidity" : "32.3", "luminosity" : "86",
  "_v" : 0 }
{ "_id" : ObjectId("619466be0374269b9cc2cd2f"), "date" : "2021-11-17 11:19:42.908", "temperature" : "21.7", "humidity" : "32.4", "luminosity" : "85",
  "_v" : 0 }
{ "_id" : ObjectId("619466bc0374269b9cc2cd2d"), "date" : "2021-11-17 11:19:40.630", "temperature" : "21.7", "humidity" : "32.4", "luminosity" : "86",
  "_v" : 0 }
{ "_id" : ObjectId("619466ba0374269b9cc2cd2b"), "date" : "2021-11-17 11:19:38.348", "temperature" : "21.7", "humidity" : "32.4", "luminosity" : "85",
  "_v" : 0 }
{ "_id" : ObjectId("619466b80374269b9cc2cd29"), "date" : "2021-11-17 11:19:36.071", "temperature" : "21.7", "humidity" : "32.4", "luminosity" : "85",
  "_v" : 0 }
```

Web monitoring: [http://localhost:3030/client\\_iotdb.html](http://localhost:3030/client_iotdb.html)

## MongoDB database visualization by AA00

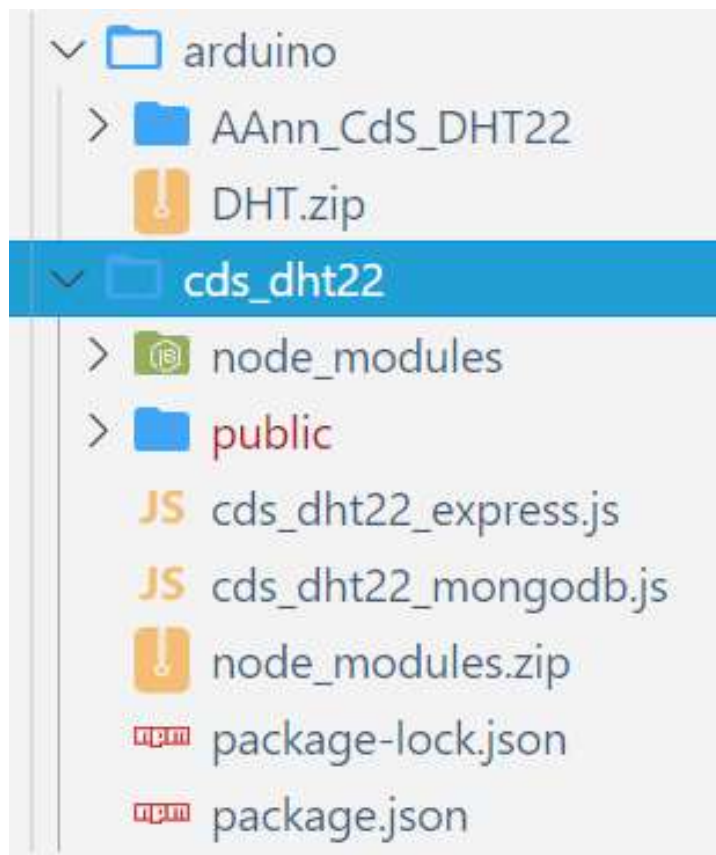
### Time series : Multi sensor data

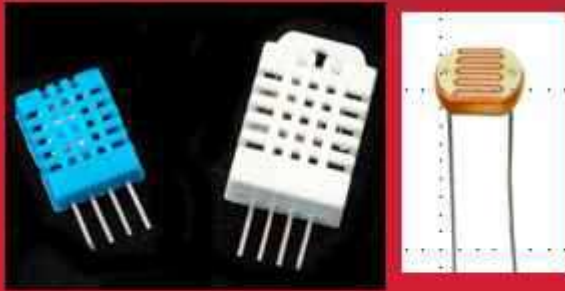




## A5.9.5 DHT22 + CdS + Node.js + MongoDB

### 1. 작업 폴더 구조 [2021]





[Goal]

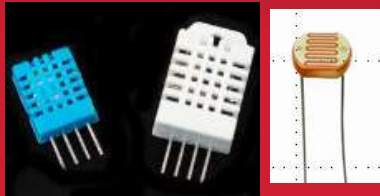
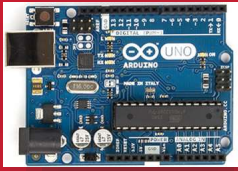
Arduino + Node.js

+ plotly.js

+ MongoDB

→ Data storaging

& visualization



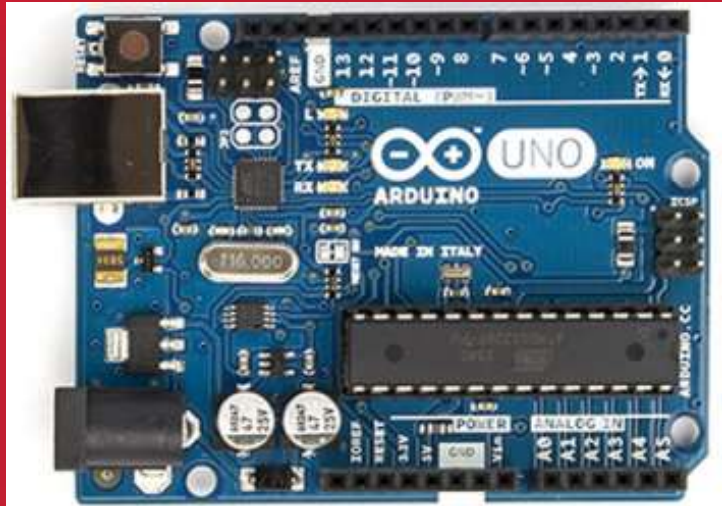
# MongoDB from Arduino with node.js & mongoose

```
> show dbs
aa00      0.000GB
admin     0.000GB
config    0.000GB
iot        0.000GB
iot2       0.000GB
iot3       0.001GB
local     0.000GB
test      0.000GB
test2     0.000GB
>
```

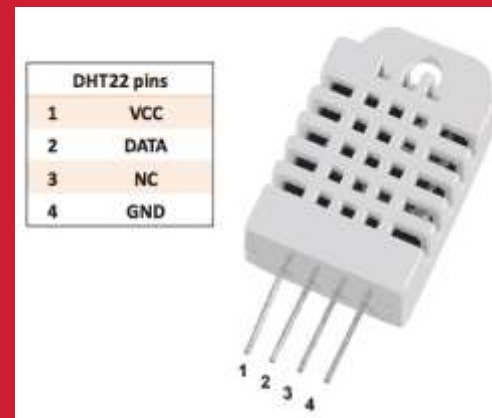
```
mongo db connection OK.
iotInfo: Current date: 2021-11-03 10:52:19.797, Temp: 23.5, Humi: 40.5, Lux: 84
iotInfo: Current date: 2021-11-03 10:52:22.074, Temp: 23.5, Humi: 40.5, Lux: 51
iotInfo: Current date: 2021-11-03 10:52:24.352, Temp: 23.5, Humi: 40.5, Lux: 81
iotInfo: Current date: 2021-11-03 10:52:26.629, Temp: 23.5, Humi: 40.8, Lux: 29
iotInfo: Current date: 2021-11-03 10:52:28.911, Temp: 23.5, Humi: 40.9, Lux: 82
iotInfo: Current date: 2021-11-03 10:52:31.188, Temp: 23.5, Humi: 40.8, Lux: 56
iotInfo: Current date: 2021-11-03 10:52:33.466, Temp: 23.5, Humi: 40.8, Lux: 83
iotInfo: Current date: 2021-11-03 10:52:35.744, Temp: 23.5, Humi: 40.8, Lux: 84
iotInfo: Current date: 2021-11-03 10:52:38.021, Temp: 23.5, Humi: 40.8, Lux: 84
iotInfo: Current date: 2021-11-03 10:52:40.299, Temp: 23.5, Humi: 40.8, Lux: 84
iotInfo: Current date: 2021-11-03 10:52:42.576, Temp: 23.5, Humi: 40.8, Lux: 84
iotInfo: Current date: 2021-11-03 10:52:44.854, Temp: 23.5, Humi: 40.8, Lux: 84
```



# Arduino & Node.js & MongoDB



**Multi-sensors**  
**DHT22 + CdS**









# DHT22 + CdS + Node.js

## [3] Result: Parsed streaming data from dht22 & CdS (Run in Terminal)

COM3

```
21.0,24.7,205  
21.0,24.7,207  
21.0,24.7,205  
21.0,24.7,152  
21.0,24.7,167  
20.9,24.6,166  
20.9,24.6,204  
21.0,24.8,204  
21.0,24.8,152  
21.0,24.8,173  
21.0,24.8,191  
21.0,24.8,203  
21.0,24.8,207  
21.0,24.9,204  
21.0,24.9,204
```

☒ 자동 스크롤 ☐ 타임스탬프



문제 출력 디버그 콘솔 터미널 JUPYTER node

```
AAnn,2021-10-27 11:53:01.872,23.4,42.6,286  
AAnn,2021-10-27 11:53:02.872,23.4,42.6,286  
AAnn,2021-10-27 11:53:04.150,23.4,42.6,286  
AAnn,2021-10-27 11:53:05.154,23.4,42.6,286  
AAnn,2021-10-27 11:53:06.428,23.4,42.6,286  
AAnn,2021-10-27 11:53:07.431,23.4,42.6,286  
AAnn,2021-10-27 11:53:08.709,23.4,42.6,286  
AAnn,2021-10-27 11:53:09.713,23.4,42.6,286  
AAnn,2021-10-27 11:53:10.987,23.4,42.6,286  
AAnn,2021-10-27 11:53:11.990,23.4,42.6,286  
AAnn,2021-10-27 11:53:13.269,23.4,42.6,284  
AAnn,2021-10-27 11:53:14.268,23.4,42.6,284  
AAnn,2021-10-27 11:53:15.546,23.4,42.6,286  
AAnn,2021-10-27 11:53:16.550,23.4,42.6,284  
AAnn,2021-10-27 11:53:17.824,23.4,42.6,286  
AAnn,2021-10-27 11:53:18.827,23.4,42.6,286  
█
```





# A5.9.5 DHT22 + CdS + Node.js + MongoDB

## 2.1 cds\_dht22\_mongodb.js

```
1 // cds_dht22_mongodb.js
2
3 var serialport = require('serialport');
4 var portName = 'COM4'; // check your COM port!!
5 var port = process.env.PORT || 3000;
6
7 var io = require('socket.io').listen(port);
8
9 // MongoDB
10 var mongoose = require('mongoose');
11 var Schema = mongoose.Schema;
12 // MongoDB connection
13 mongoose.connect('mongodb://localhost:27017/iot'); // DB name
14 var db = mongoose.connection;
15 db.on('error', console.error.bind(console, 'connection error:'));
16 db.once('open', function callback () {
17   console.log("mongo db connection OK.");
18 });
19 // Schema
20 var iotSchema = new Schema({
21   date : String,
22   temperature : String,
23   humidity : String,
24   luminosity: String
25 });
```



# A5.9.5 DHT22 + CdS + Node.js + MongoDB

## 2.2 cds\_dht22\_mongodb.js

```
23 // Schema
24 var iotSchema = new Schema({
25   date: String,
26   temperature: String,
27   humidity: String,
28   luminosity: String,
29 });
30 // Display data on console in the case of saving data.
31 iotSchema.methods.info = function () {
32   var iotInfo = this.date
33     ? "Current date: " +
34       this.date +
35       ", Temp: " +
36       this.temperature +
37       ", Humi: " +
38       this.humidity +
39       ", Lux: " +
40       this.luminosity
41     : "I don't have a date";
42   console.log("iotInfo: " + iotInfo);
43 };
```



# A5.9.5 DHT22 + CdS + Node.js + MongoDB

## 2.3 cds\_dht22\_mongodb.js

```
45 const Readline = require("@serialport/parser-readline");
46 // serial port object
47 var sp = new serialport(portName, {
48   baudRate: 9600, // 9600 38400
49   dataBits: 8,
50   parity: "none",
51   stopBits: 1,
52   flowControl: false,
53   parser: new Readline("\r\n"),
54 });
55
56 const parser = sp.pipe(new Readline({ delimiter: "\r\n" }));
57
58 // Read the port data
59 sp.on("open", () => {
60   console.log("serial port open");
61 });
62
63 var readData = ""; // this stores the buffer
64 var temp = "";
65 var humi = "";
66 var lux = "";
67 var mdata = []; // this array stores date and data from multiple sensors
68 var firstcommaidx = 0;
69
70 var Sensor = mongoose.model("Sensor", iotSchema); // sensor data model
```



# A5.9.5 DHT22 + CdS + Node.js + MongoDB

## 2.4 cds\_dht22\_mongodb.js – parsing data & save data in MongoDB

```
72 parser.on("data", function (data) {
73   // call back when data is received
74   readData = data.toString(); // append data to buffer
75   firstcommaidx = readData.indexOf(",");
76   // parsing data into signals
77   if (readData.lastIndexOf(",") > firstcommaidx && firstcommaidx > 0) {
78     temp = readData.substring(
79       firstcommaidx + 1,
80       readData.indexOf(",", firstcommaidx + 1)
81     );
82     humi = readData.substring(
83       readData.indexOf(",", firstcommaidx + 1) + 1,
84       readData.lastIndexOf(",")
85     );
86     lux = readData.substring(readData.lastIndexOf(",") + 1);
87     readData = "";
88     dStr = getDateString();
89     mdata[0] = dStr; // Date
90     mdata[1] = temp; // temperature data
91     mdata[2] = humi; // humidity data
92     mdata[3] = lux; // luminosity data
93     var iot = new Sensor({
94       date: dStr,
95       temperature: temp,
96       humidity: humi,
97       luminosity: lux,
98     });
99     // save iot data to MongoDB
100    iot.save(function (err, iot) {
101      if (err) return handleError(err);
102      iot.info(); // Display the information of iot data on console.
103    });
104    io.sockets.emit("message", mdata); // send data to all clients
105  } else {
106    // error
107    console.log(readData);
108  }
109 });
```



## A5.9.5 DHT22 + CdS + Node.js + MongoDB

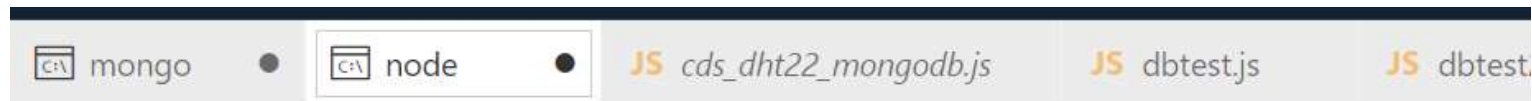
### 2.5 cds\_dht22\_mongodb.js

```
113 io.sockets.on("connection", function (socket) {
114   // If socket.io receives message from the client browser then
115   // this call back will be executed.
116   socket.on("message", function (msg) {
117     console.log(msg);
118   });
119   // If a web browser disconnects from Socket.IO then this callback is called.
120   socket.on("disconnect", function () {
121     console.log("disconnected");
122   });
123 });
124
125 // helper function to get a nicely formatted date string
126 function getDateString() {
127   var time = new Date().getTime();
128   // 32400000 is (GMT+9 Korea, GimHae)
129   // for your timezone just multiply +/-GMT by 3600000
130   var datestr = new Date(time + 32400000)
131     .toISOString()
132     .replace(/T/, " ")
133     .replace(/Z/, "");
134   return datestr;
135 }
```



# A5.9.5 DHT22 + CdS + Node.js + MongoDB

## 2.6 [Run] node cds\_dht22\_mongodb.js



```
D:\aann\aann-rpt10\cds_dht22>node cds_dht22_mongodb
```

```
serial port open
```

```
mongo db connection OK.
```

```
iotInfo: Current date: 2021-11-03 10:52:19.797, Temp: 23.5, Humi: 40.5, Lux: 84
```

```
iotInfo: Current date: 2021-11-03 10:52:22.074, Temp: 23.5, Humi: 40.5, Lux: 51
```

```
iotInfo: Current date: 2021-11-03 10:52:24.352, Temp: 23.5, Humi: 40.5, Lux: 81
```

```
iotInfo: Current date: 2021-11-03 10:52:26.629, Temp: 23.5, Humi: 40.8, Lux: 29
```

```
iotInfo: Current date: 2021-11-03 10:52:28.911, Temp: 23.5, Humi: 40.9, Lux: 82
```

```
iotInfo: Current date: 2021-11-03 10:52:31.188, Temp: 23.5, Humi: 40.8, Lux: 56
```

```
iotInfo: Current date: 2021-11-03 10:52:33.466, Temp: 23.5, Humi: 40.8, Lux: 83
```

```
iotInfo: Current date: 2021-11-03 10:52:35.744, Temp: 23.5, Humi: 40.8, Lux: 84
```

```
iotInfo: Current date: 2021-11-03 10:52:38.021, Temp: 23.5, Humi: 40.8, Lux: 84
```

```
iotInfo: Current date: 2021-11-03 10:52:40.299, Temp: 23.5, Humi: 40.8, Lux: 84
```

```
iotInfo: Current date: 2021-11-03 10:52:42.576, Temp: 23.5, Humi: 40.8, Lux: 84
```

```
iotInfo: Current date: 2021-11-03 10:52:44.854, Temp: 23.5, Humi: 40.8, Lux: 84
```

```
iotInfo: Current date: 2021-11-03 10:52:47.136, Temp: 23.5, Humi: 40.8, Lux: 84
```

```
iotInfo: Current date: 2021-11-03 10:52:49.413, Temp: 23.5, Humi: 40.7, Lux: 84
```

```
iotInfo: Current date: 2021-11-03 10:52:51.691, Temp: 23.5, Humi: 40.7, Lux: 84
```

```
iotInfo: Current date: 2021-11-03 10:52:53.969, Temp: 23.5, Humi: 40.7, Lux: 84
```







# A5.9.5 DHT22 + CdS + Node.js + MongoDB

## 3. cds\_dht22\_mongodb.js → Check documents in Mongo shell

### Mongo shell

> show dbs

> use iot

> show collections

> db.sensors.find()  
.pretty()

```
mongo node JS cds_dht22_mongodb.js
> show dbs
admin    0.000GB
config  0.000GB
iot      0.000GB
local    0.000GB
test     0.000GB
> use iot
switched to db iot
> show collections
sensors
> db.sensors.find().pretty()
{
  "_id" : ObjectId("6181eb5338cdb755b232170"),
  "date" : "2021-11-03 10:52:19.797",
  "temperature" : "23.5",
  "humidity" : "40.5",
  "luminosity" : "84",
  "__v" : 0
}
{
  "_id" : ObjectId("6181eb5638cdb755b232172"),
  "date" : "2021-11-03 10:52:22.074",
  "temperature" : "23.5",
  "humidity" : "40.5",
  "luminosity" : "51",
  "__v" : 0
}
{
  "_id" : ObjectId("6181eb5838cdb755b232174"),
  "date" : "2021-11-03 10:52:24.352",
  "temperature" : "23.5",
  "humidity" : "40.5",
  "luminosity" : "81",
  "__v" : 0
}
```

Save as

AAnn\_iot\_mongodb.png



# Arduino & Node.js & MongoDB & Express server







# A5.9.6 DHT22 + CdS + Node.js + MongoDB

## 1.1 Install express server

➤ Go to cds\_dht22 project

➤ `npm install --save express`

➤ `package.json`

```
D:\aann\aann-rpt10\cds_dht22>npm install --save express
npm WARN cds_dht22@1.0.0 No repository field.

+ express@4.17.1
added 50 packages from 33 contributors, removed 67 packages,
66s

4 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

D:\aann\aann-rpt10\cds_dht22>
```



## 1.2 Install express server – package.json

- Go to cds\_dht22 project
- `npm install --save express`
- `package.json`

```
"author": "aa00",  
"license": "MIT",  
"dependencies": {  
  "express": "^4.17.1",  
  "mongoose": "^6.0.12",  
  "serialport": "^9.2.4",  
  "socket.io": "^2.4.1"  
}
```



# A5.9.6 DHT22 + CdS + Node.js + MongoDB

## 2.1 cds\_dht22\_express.js

```
1 // cds_dht22_express.js
2 var express = require("express");
3 var app = express();
4 var web_port = 3030; // express port
5
6 // MongoDB
7 var mongoose = require("mongoose");
8 var Schema = mongoose.Schema; // Schema object
9 // MongoDB connection
10 mongoose.connect("mongodb://localhost:27017/iot", {
11   useNewUrlParser: true,
12   useUnifiedTopology: true,
13 });
14 var db = mongoose.connection;
15 db.on("error", console.error.bind(console, "connection error:"));
16 db.once("open", function callback() {
17   console.log("mongo db connection OK.");
18 });
19 // Schema
20 var iotSchema = new Schema({
21   date: String,
22   temperature: String,
23   humidity: String,
24   luminosity: String,
25 });
26 var Sensor = mongoose.model("Sensor", iotSchema); // sensor data model
```



## A5.9.6 DHT22 + CdS + Node.js + MongoDB

### 2.2 cds\_dht22\_express.js

```
28 // Web routing address
29 app.get("/", function (req, res) {
30   // localhost:3030/
31   res.send("Hello Arduino IOT: express server by AA00!");
32 });
33 // find all data & return them
34 app.get("/iot", function (req, res) {
35   Sensor.find(function (err, data) {
36     res.json(data);
37   });
38 });
39 // find data by id
40 app.get("/iot/:id", function (req, res) {
41   Sensor.findById(req.params.id, function (err, data) {
42     res.json(data);
43   });
44 });
45
46 // Express WEB
47 app.use(express.static(__dirname + "/public")); // WEB root folder
48 app.listen(web_port); // port 3030
49 console.log("Express_IOT is running at port:3030");
```



## A5.9.6 DHT22 + CdS + Node.js + MongoDB

### 2.3 [Run ] node cds\_dht22\_express.js

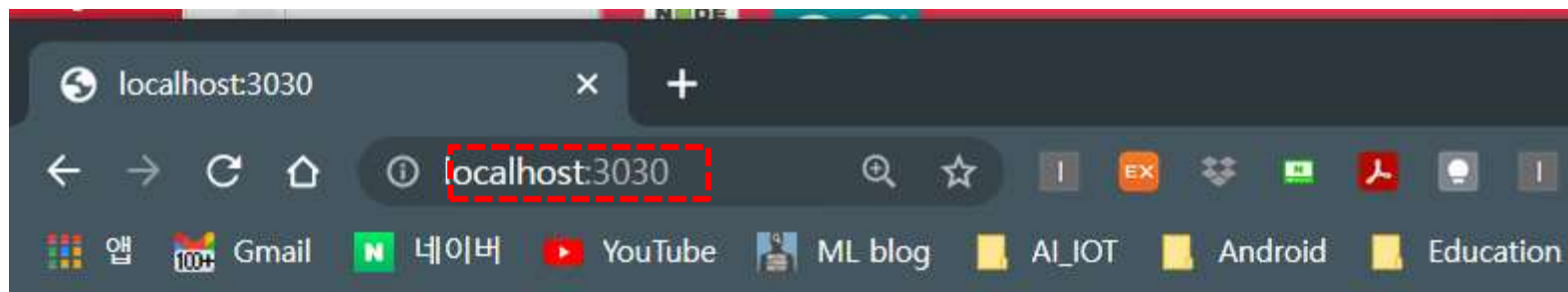
```
(base) D:\aann\aann-rpt10\cds_dht22>node cds_dht22_express
Express_IOT is running at port:3030
mongo db connection OK.
```





## A5.9.6 DHT22 + CdS + Node.js + MongoDB

2.4 cds\_dht22\_express.js → routing1, <http://localhost:3030/>



Hello Arduino IOT: express server by AA00!

**AAnn\_iot\_mongodb\_web.png**



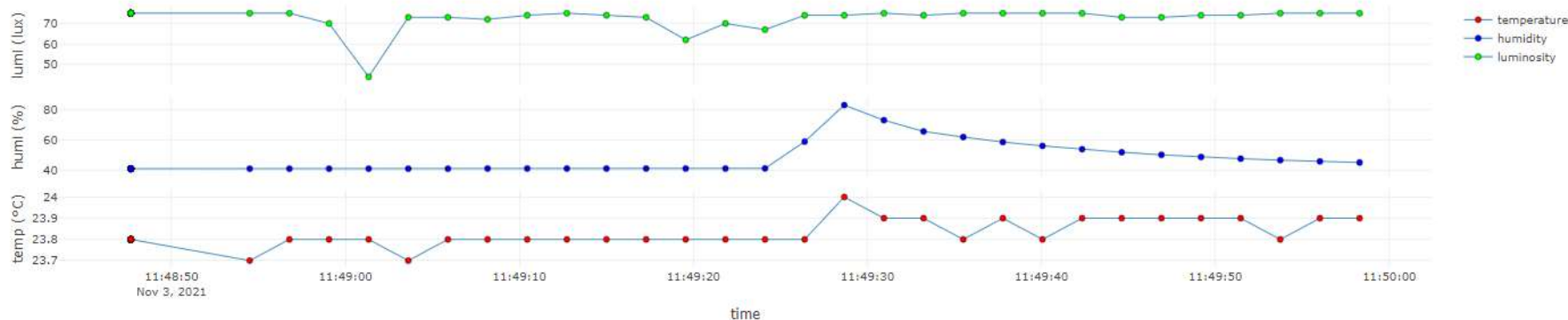
# A5.9.6 DHT22 + CdS + Node.js + MongoDB

2.7 copy `cds_dht22_client.html` & `gauge.min.js` → `./public/` subfolder  
[http://localhost:3030/client\\_cds\\_dht22.html](http://localhost:3030/client_cds_dht22.html) (web root folder)

## Real-time Weather Station from sensors



on Time: 2021-11-03 11:49:58.294







## 2.8 CORS bug (Cross Origin Resource Sharing)

```
✓ aann-rpt11
  > arduino
  ✓ cds_dht22
    > node_modules
    ✓ public
      > data
        </> aapl_chaos.html
        </> aapl_local.html
        </> aapl.html
        </> client_CdS_DHT22.html
        </> client_iotDB_start.html
        JS gauge.min.js
        </> iot_chaos.html
```

### \* CORS problem

→ 원격 서버 내의 파일에 접근을 허용



## A5.9.6 DHT22 + CdS + Node.js + MongoDB

2.9 **CORS patch** on the express server → [cds\\_dht22\\_express.js](#)

Node cmd에서 'cors' module 설치 (version 2.8.4 이상)

**npm install --save cors**

```
1  // cds_dht22_express_cors.js
2  // Express + CORS
3  var express = require("express");
4  var cors = require("cors");
5  var app = express();
6  app.use(cors());
7  var web_port = 3030; // express port
8
9  // MongoDB
10 var mongoose = require("mongoose");
11 var Schema = mongoose.Schema; // Schema object
```

```
D:\aann\aann-rpt11\cds_dht22>node cds_dht22_express
Express_IOT with CORS is running at port:3030
mongo db connection OK.
```



DHT22 + CdS + Node.js + MongoDB

# Web monitoring



# A5.9.8 DHT22 + CdS + Node.js + MongoDB

## 3.1 Web client: [client\\_iotDB.html](#)

```
client_iotDB.html x
1 <!DOCTYPE html>
2 <head>
3   <meta charset="utf-8">
4   <!-- Plotly.js -->
5   <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
6 </head>
7 <body>
8   <h1>MongoDB database visualization by AA00</h1>
9   <hr>
10  <h2>Time series : Multi sensor data</h2>
11
12  <!-- Plotly chart will be drawn inside this DIV -->
13  <div id="myDiv" style="width: 900px;height: 600px"></div>
14
```



# A5.9.7 DHT22 + CdS + Node.js + MongoDB

## 3.2 Web client: [client\\_iotDB.html](#)

```
<script>
  <!-- JAVASCRIPT CODE GOES HERE -->

  Plotly.d3.json("http://localhost:3030/iot", function(err, json){
    //alert(json);
    alert(JSON.stringify(json)); // It works!!!
    //alert(JSON.parse(eval(json)));
    if(err) throw err;

    var date = [];
    var temp = [];
    var humi = [];
    var lumi = [];
    var jsonData = eval(JSON.stringify(json));
    //alert(jsonData.length);
    //alert(jsonData[2].luminosity);

    for (var i = 0; i < jsonData.length; i++) {
      date[i] = jsonData[i].date;
      temp[i] = jsonData[i].temperature ;
      humi[i] = jsonData[i].humidity;
      lumi[i] = jsonData[i].luminosity;
    }
  })
}
```

**JSON  
file**

```
{ "_id": "5fbdb71d02de805786af43c", "date": "2020-11-25
09:55:13.068", "temperature": "18.9", "humidity": "24.7", "luminosity": "207", "__v": 0 },
{ "_id": "5fbdb73d02de805786af43d", "date": "2020-11-25
09:55:15.341", "temperature": "18.9", "humidity": "24.7", "luminosity": "208", "__v": 0 },
{ "_id": "5fbdb75d02de805786af43e", "date": "2020-11-25
```



# A5.9.7 DHT22 + CdS + Node.js + MongoDB

## 3.3 Web client: [client\\_iotDB.html](#) – data & layout

```
// time series of sensor data
var trace1 = {
  type: "scatter",
  mode: "lines",
  name: 'Temperature',
  x: date,
  y: temp,
  line: {color: '#fc1234'}
}

var trace2 = {
  type: "scatter",
  mode: "lines",
  name: 'Humidity',
  x: date,
  y: humi,
  line: {color: '#3412fc'}
}

var trace3 = {
  type: "scatter",
  mode: "lines",
  name: 'Luminosity',
  x: date,
  y: lumi,
  line: {color: '#34fc12'}
}

var data = [trace1, trace2, trace3];
```

```
// Layout with builtin rangeslider
var layout = {
  title: 'Temp vs. Humi vs. Lumi with rangeslider',
  xaxis: {
    autorange: true,
    range: [date[0], date[date.length-1]],
    rangeselector: {buttons: [
      {
        count: 1,
        label: '1 hour',
        step: 'hour',
        stepmode: 'backward'
      },
      {
        count: 6,
        label: '6 hour',
        step: 'hour',
        stepmode: 'backward'
      },
      {
        count: 24,
        label: '1 day',
        step: 'hour',
        stepmode: 'backward'
      },
      {
        count: 7,
        label: '1 week',
        step: 'day',
        stepmode: 'backward'
      },
      {step: 'all'}
    ]},
    rangeslider: {range: [date[0], date[date.length-1]],
      type: 'date'
    },
    type: 'date'
  },
  yaxis: {
    autorange: true,
    range: [0, 300],
    type: 'linear'
  }
}

Plotly.newPlot('myDiv', data, layout);
})
```



# DHT22 + CdS + Node.js + MongoDB

Web monitoring: <http://localhost:3030/iot>

localhost:3030/iot3.html

localhost:3030/iot3.html

앱 Gmail 네이버 YouTube

## MongoDB database viewer

Time series : Multi sensor data

localhost:3030 내용:

```
[{"_id":"6181eb5338cdbc755b232170","date":"2021-11-03 10:52:19.797","temperature":23.5,"humidity":40.5,"luminosity":84,"__v":0}, {"_id":"6181eb5638cdbc755b232172","date":"2021-11-03 10:52:22.074","temperature":23.5,"humidity":40.5,"luminosity":51,"__v":0}, {"_id":"6181eb5838cdbc755b232174","date":"2021-11-03 10:52:24.352","temperature":23.5,"humidity":40.5,"luminosity":81,"__v":0}, {"_id":"6181eb5a38cdbc755b232176","date":"2021-11-03 10:52:26.630","temperature":23.5,"humidity":40.5,"luminosity":81,"__v":0}]
```

확인



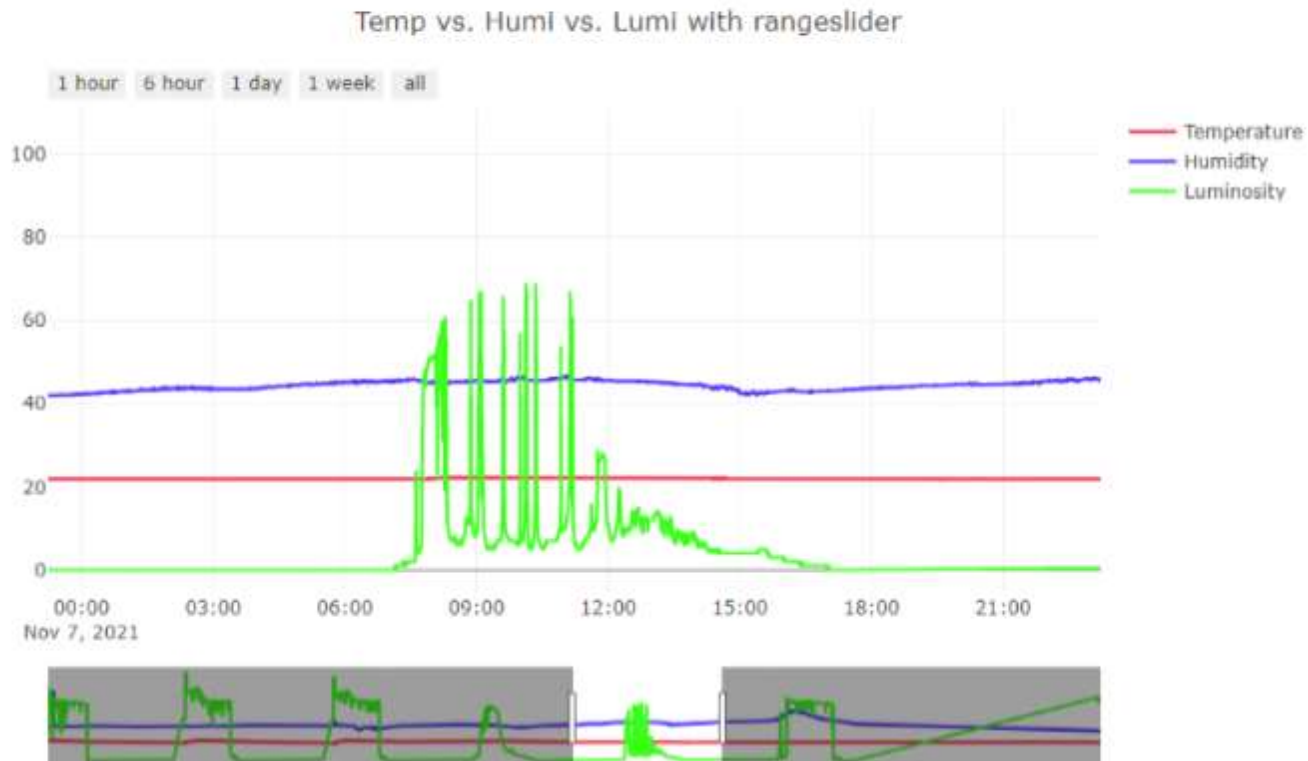


# DHT22 + CdS + Node.js + MongoDB

## Web monitoring

### MongoDB database visualization by AA00

#### Time series : Multi sensor data





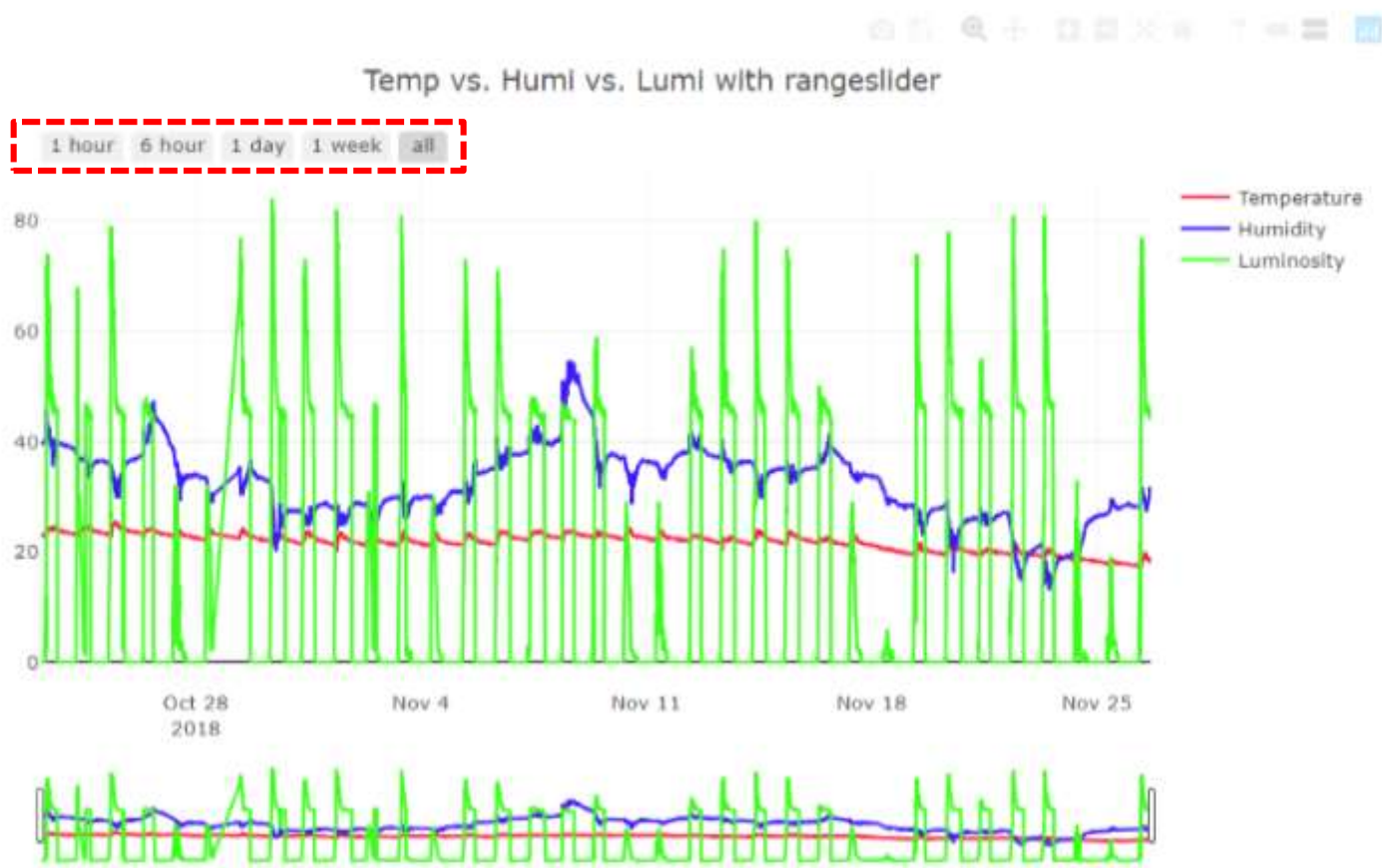


# A5.9.7 DHT22 + CdS + Node.js + MongoDB

## 3.5 Web client: [client\\_iotDB.html](#) – iot DB monitoring (public 폴더에서 제공)

### MongoDB database visualization by AA00

Time series : Multi sensor data





## A5. Introduction to IoT service

**System (Arduino, sDevice, ...)**



**Data (signal, image, sns, ...)**



**Visualization & monitoring**



**Data storing & mining**



**Service**



# MongoDB data management

- Query in mongo shell
- Export & import MongoDB
- Using and understanding iot data with Python (or R)



## A5.9.8 MongoDB management

### 1. Query in Mongo shell (문서 검색)

`db.sensors.count()` → sensors collection에 있는 도큐먼트 (문서)의 수

`db.sensors.find().sort({_id: 1}).limit(10)` → 오래된 document 10개 추출

`db.sensors.find().sort({_id: -1}).limit(10)` → 최근 document 10개 추출

`db.sensors.find({date: {$gt: "2021-11-17 15:16:05"}})` → 특정 시간 이후 document 추출

`db.sensors.find( {temperature: {$gt: "25"}} )` → 온도가 25도를 넘는 document 추출

<https://docs.mongodb.com/manual/tutorial/query-documents/>



## A5.9.8 MongoDB management

### 2. Import or export MongoDB (VSCode 터미널에서 실행)

- **mongoimport** --d=dbName --c=collectionName --type=csv --headerline --file= fileName.csv
- **mongoexport** --d=dbName --c=collectionName --fields=<field1,field2,...> --limit=nn --type=csv --out=fileName.csv

**json 또는 csv 파일로 import/export**

**[Help] mongoimport --help**

<https://docs.mongodb.com/manual/reference/program/mongoimport/>

<https://docs.mongodb.com/manual/reference/program/mongoexport/>



## A5.9.8 MongoDB management

➤ **mongoimport** /db:dbName /collection:collectionName /type:csv /headerline /file: fileName.csv

```
D:\aann\aann-rpt11\cds_dht22\public\data>mongoimport /db:iot10 /collection:sensors
/type:csv /headerline /file:sensor10.csv
2021-11-10T12:52:08.616+0900    connected to: mongodb://localhost/
2021-11-10T12:52:08.803+0900    10 document(s) imported successfully. 0 document(s)
failed to import.
```

**Import sensor10.csv  
to MongoDB: iot10**

```
> show dbs
admin    0.000GB
config  0.000GB
iot      0.009GB
iot10    0.000GB
local    0.000GB
test     0.000GB
> use iot10
switched to db iot10
```



## A5.9.8 MongoDB management

[Tip] **iot db**의 최근 데이터 **500**개를 **csv** 파일 (**iot\_s500.csv**)로 저장할 때,

➤ **mongoexport /db:iot /collection:sensors /sort:"{\_id: -1}" /limit:500 /fields:date,temperature,humidity,luminosity /type:csv /out:iot\_s500.csv**

```
D:\aann\aann-rpt11\cds_dht22\public\data>mongoexport /db:iot /collection:sensors
/sort:"{_id: -1}" /limit:500 /fields:date,temperature,humidity,luminosity /type
:csv /out:iot_s500.csv
2021-11-10T13:08:45.875+0900    connected to: mongodb://localhost/
2021-11-10T13:08:45.941+0900    exported 500 records
```

```
D:\aann\aann-rpt11\cds_dht22\public\data>dir
```

D 드라이브의 볼륨: DATA

볼륨 일련 번호: 82D1-4852

D:\aann\aann-rpt11\cds\_dht22\public\data 디렉터리

2021-11-10	오후 01:08	<DIR>	.
2021-11-10	오후 01:08	<DIR>	..
2017-11-16	오전 09:58		60,220 aapl.csv
2018-11-26	오후 05:50		3,628,861 iot_chaos.csv
2021-11-10	오후 01:08		18,537 iot_s500.csv
2017-11-16	오후 01:18		135,008 ppg5k.csv
2018-05-26	오후 12:52		397 sensor10.csv
5개 파일			3,843,023 바이트
2개 디렉터리			2,410,432,798,720 바이트 남음

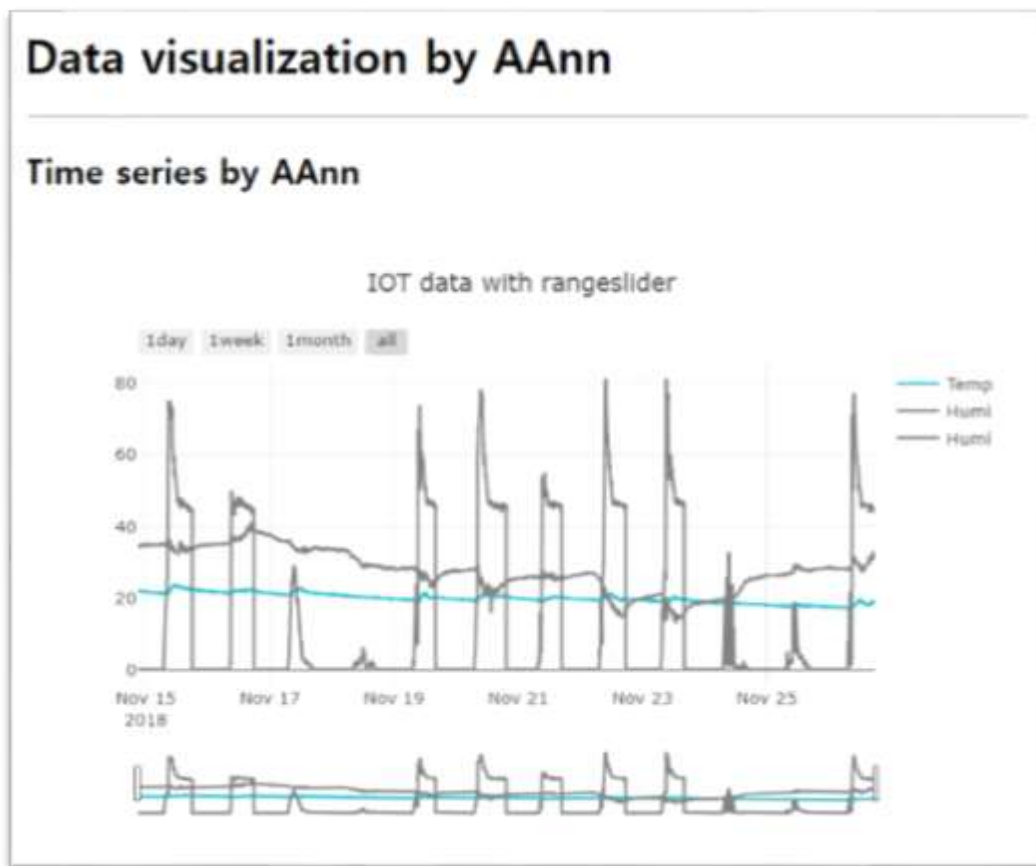




## A5.9.8 MongoDB management

### [DIY]

1. `iot db`의 최근 데이터 1000개를 `csv` 파일 ([AAnn\\_s1000.csv](#))로 저장하시오.
2. 저장된 `AAnn_s1000.csv` 파일을 `public/data` 폴더에 복사.
3. `csv` 파일을 이용하는 `Rangeslider`가 포함된 웹 클라이언트 [client\\_iot.html](#) 파일을 만드시오.
4. `localhost:3030/client_iot.html` 로 실행하고 확인.



**iot\_chaos.html**로

**client\_iot.html**로

이름 변경해서 코드를 완성하시오.



3. How to use and understand iot data? → Python(or R) in Colab/Jupyter lab



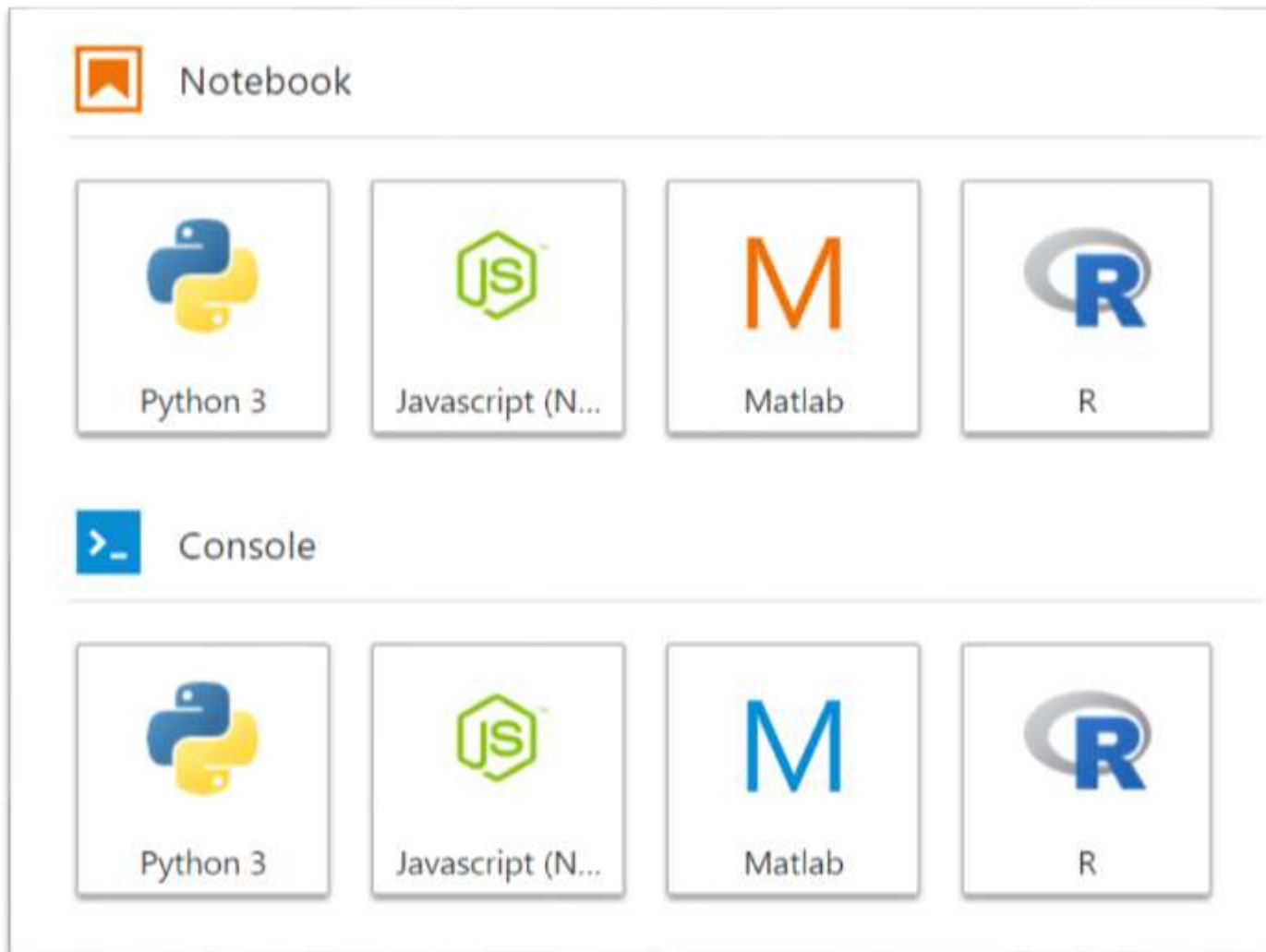
# IoT Data

# Mining



# IoT data mining

3. How to use and understand iot data? → Python(or R) in Colab/Jupyter lab





# IoT data mining

How to use and understand iot data? → [Google Colab](#)

 Open in Colab

## Pandas: access to the remote json from MongoDB

- The json file is generated on the fly from the express server of Node.js.
- The data stored in MongoDB are saved in the json file.
- The data are composed of three time series; temperature, humidity, and luminosity.

```
In [0]: import pandas as pd
```

```
In [0]: # loading json file from MongoDB via web (CORS, port=3030)
url="http://chaos.inje.ac.kr:3030/iot"
df=pd.read_json(url)
print('Large data was retrieved successfully from MongoDB!')
```

```
In [0]: df.head()
```



## A5.9.8 IOT data mining

### 3.1 How to use and understand iot data? → [iot\\_csv.ipynb](#), [iot\\_json.ipynb](#)

 [Redwoods](#) / [Arduino](#)

 Code

 Issues 0

 Pull requests 0

 Projects 0

 Wiki

Branch: master ▼

[Arduino](#) / [ar-iot](#) / [py-pandas](#) /



Redwoods Add files via upload

..

 data

Add files via upload

 [iot\\_csv.ipynb](#)

Colaboratory를 통해 생성됨

 [iot\\_json.ipynb](#)

Colaboratory를 통해 생성됨



## A5.9.8 MongoDB management

### 3.2 Loading data ... → `iot_json.ipynb`

```
[1] 1 | import pandas as pd
```

```
[2] 1 | # loading json file from MongoDB via web (CORS, port=3030)  
2 | url="http://chaos.inje.ac.kr:3030/iot"  
3 | j1=pd.read_json(url)
```

1. Express 서버에서 MongoDB에 접속한다.  
2. 아두이노에서 만들어져 전송되어 MongoDB에 저장되고 있는 센서 데이터를 json 파일로 가져온다.

```
[3] 1 | j1.head()
```



	__v	_id	date	humidity	luminosity	temperature
0	0	5bce24218d1ec32774d781a9	2018-10-23 04:25:21.349	39.7	0	23.2
1	0	5bce242b8d1ec32774d781aa	2018-10-23 04:25:31.594	39.7	0	23.2
2	0	5bce24358d1ec32774d781ab	2018-10-23 04:25:41.855	39.7	0	23.2
3	0	5bce24408d1ec32774d781ac	2018-10-23 04:25:52.100	39.7	0	23.2
4	0	5bce244a8d1ec32774d781ad	2018-10-23 04:26:02.360	39.7	0	23.2



## A5.9.8 IOT data mining

### 3.3 Make dataframe from json data

#### ▼ Dataframe with date and three sensor values(temperature, humidity, luminosity)

```
[ ] 1 | iot_data = j1[['date', 'temperature', 'humidity', 'luminosity']]
```

```
[ ] 1 | iot_data.shape
```

Json 객체에서 필요한 항목을  
선택해서 **pandas의 dataframe**을  
구성한다.

(340230, 4)

```
[ ] 1 | iot_data.head()
```



	date	temperature	humidity	luminosity
0	2018-10-23 04:25:21.349	23.2	39.7	0
1	2018-10-23 04:25:31.594	23.2	39.7	0
2	2018-10-23 04:25:41.855	23.2	39.7	0
3	2018-10-23 04:25:52.100	23.2	39.7	0
4	2018-10-23 04:26:02.360	23.2	39.7	0



## 3.4.1 Plot iot data (time series)

Plot time series of sensor data

```
[ ] | iot_data.plot(x='date', y='temperature', figsize=(12,6), title='temperature')
```



<matplotlib.axes.\_subplots.AxesSubplot at 0x7f5b2596e438>

temperature



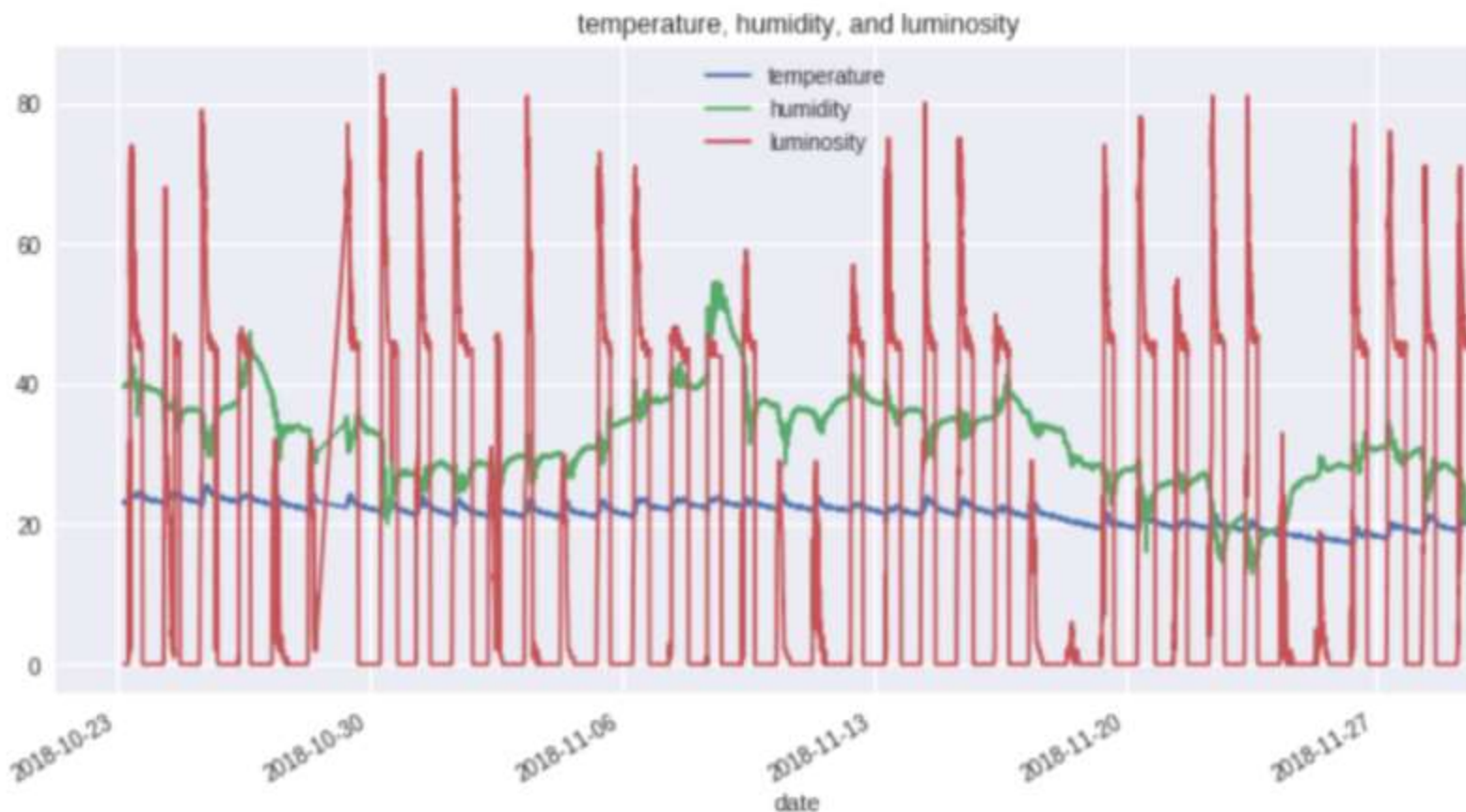
**Dataframe**에서 시간과 온도 데이터를 선택해서 그래프를 그린다.

## 3.4.2 Plot iot data (time series)

```
1 # Plot of ['temperature', 'humidity', 'luminosity']
2 iot_data.plot(x='date', y=['temperature', 'humidity', 'luminosity'], figsize=(12,6),
3               title='temperature, humidity, and luminosity')
```

```
/usr/local/lib/python3.6/dist-packages/pandas/plotting/_core.py:1716:
series.name = label
<matplotlib.axes._subplots.AxesSubplot at 0x7f5b28813128>
```

**Dataframe**에서 시간과 세 개의 센서 데이터를 전부 선택해서 그래프를 그린다.





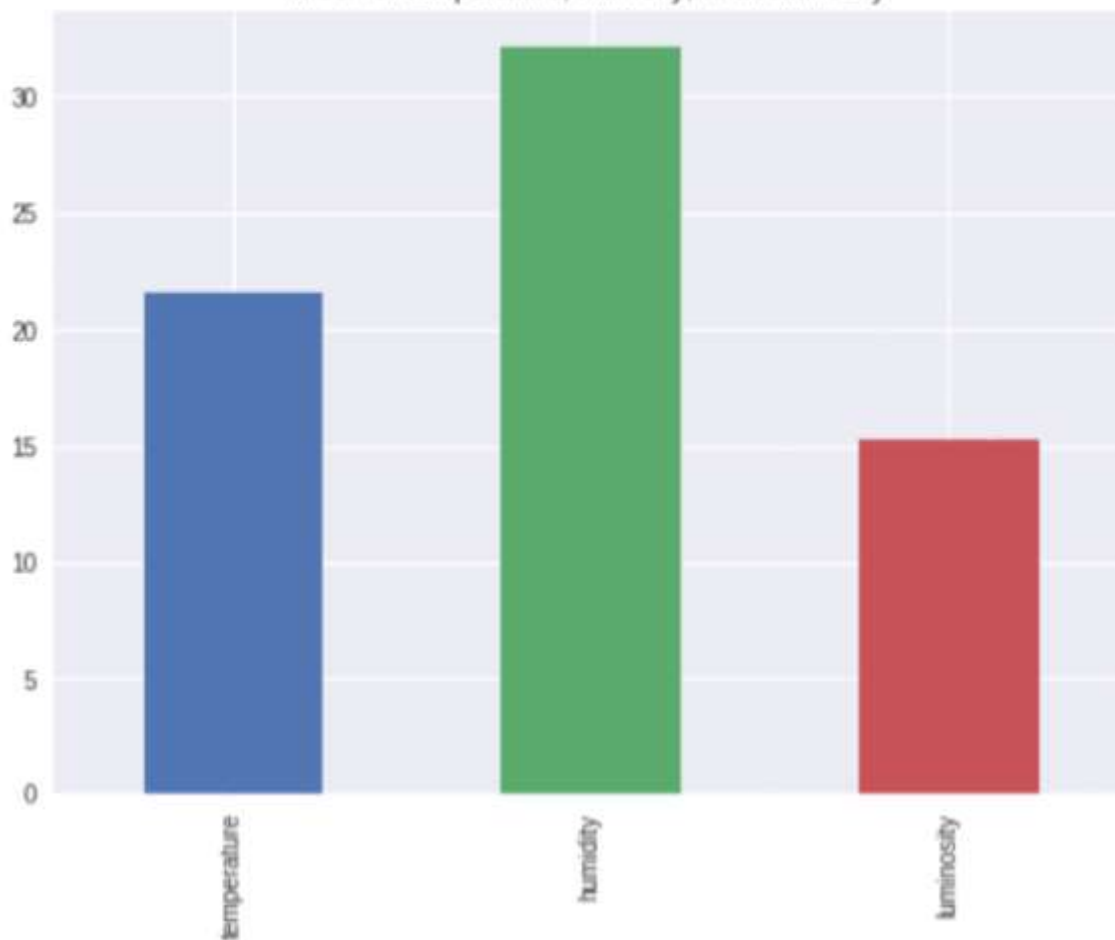
## A5.9.8 IOT data mining

### 3.5 Plot mean of sensor data

```
1 iot_data[['temperature', 'humidity', 'luminosity']].mean().plot.bar(figsize=(8,6),  
2 title="Mean of temperature, humidity, and luminosity")
```

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f5b297d9470>

Mean of temperature, humidity, and luminosity



**Dataframe**에서 세 개의 센서 데이터의 평균을 구해서 그래프를 그린다.



## A5.9.8 IOT data mining

### 3.6.1 Plot the change of sensor data over various time spans.

Set date as index of timestamp

```
[ ] 1 | iot_data.set_index('date', inplace=True)
```

```
[ ] 1 | iot_data.info() # timestamp index
```

```
<class 'pandas.core.frame.DataFrame'>  
DatetimeIndex: 307849 entries, 2018-10-23  
Data columns (total 3 columns):  
temperature    307849 non-null float64  
humidity       307849 non-null float64  
luminosity     307849 non-null int64  
dtypes: float64(2), int64(1)  
memory usage: 9.4 MB
```

```
1 | iot_data.head()
```

	temperature	humidity	luminosity
date			
2018-10-23 04:25:21.349	23.2	39.7	0
2018-10-23 04:25:31.594	23.2	39.7	0
2018-10-23 04:25:41.855	23.2	39.7	0
2018-10-23 04:25:52.100	23.2	39.7	0
2018-10-23 04:26:02.360	23.2	39.7	0

시간(date)을 **timestamp** 형태의 **Index**로 변경해서 데이터를 재구성한다.



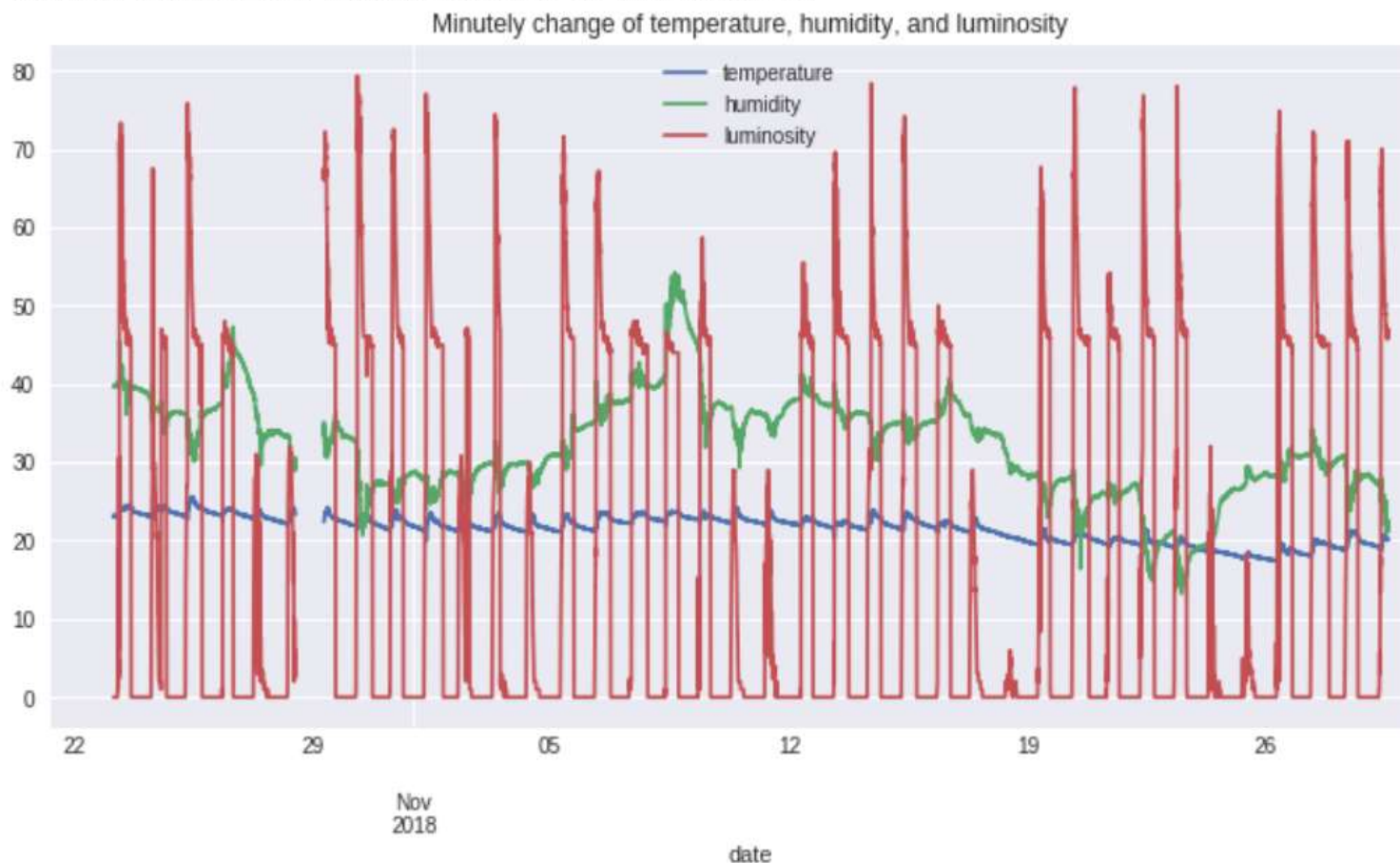
## A5.9.8 IOT data mining

### 3.6.2 Plot the change of sensor data over various time spans.

#### 1 분당 평균 그래프

```
1 # Plot mean of the iot data per every minute  
2 iot_data.resample('60S').mean().plot(figsize=(12,6),  
3 title='Minutely change of temperature, humidity, and lumi
```

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f5b2b57c630>





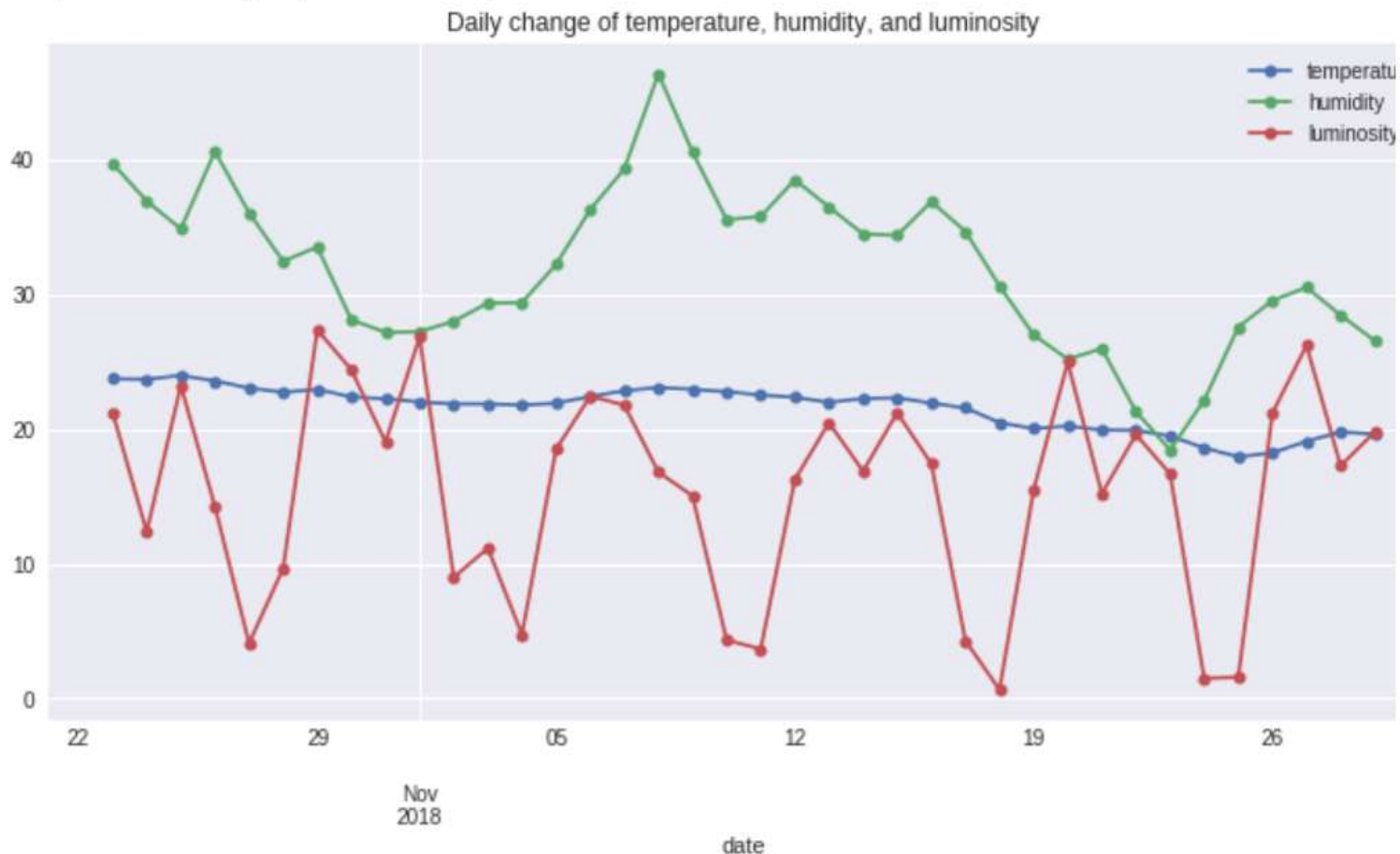
## A5.9.8 IOT data mining

### 3.6.3 Plot the change of sensor data over various time spans.

#### 1 일당 평균 그래프

```
1 # Plot mean of the iot data per every day
2 iot_data.resample('D').mean().plot(kind='line', marker='o', ms=6, figsize=(12,6),
3                                     title='Daily change of temperature, humidity, and luminosit
```

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f5b2c7fb7f0>







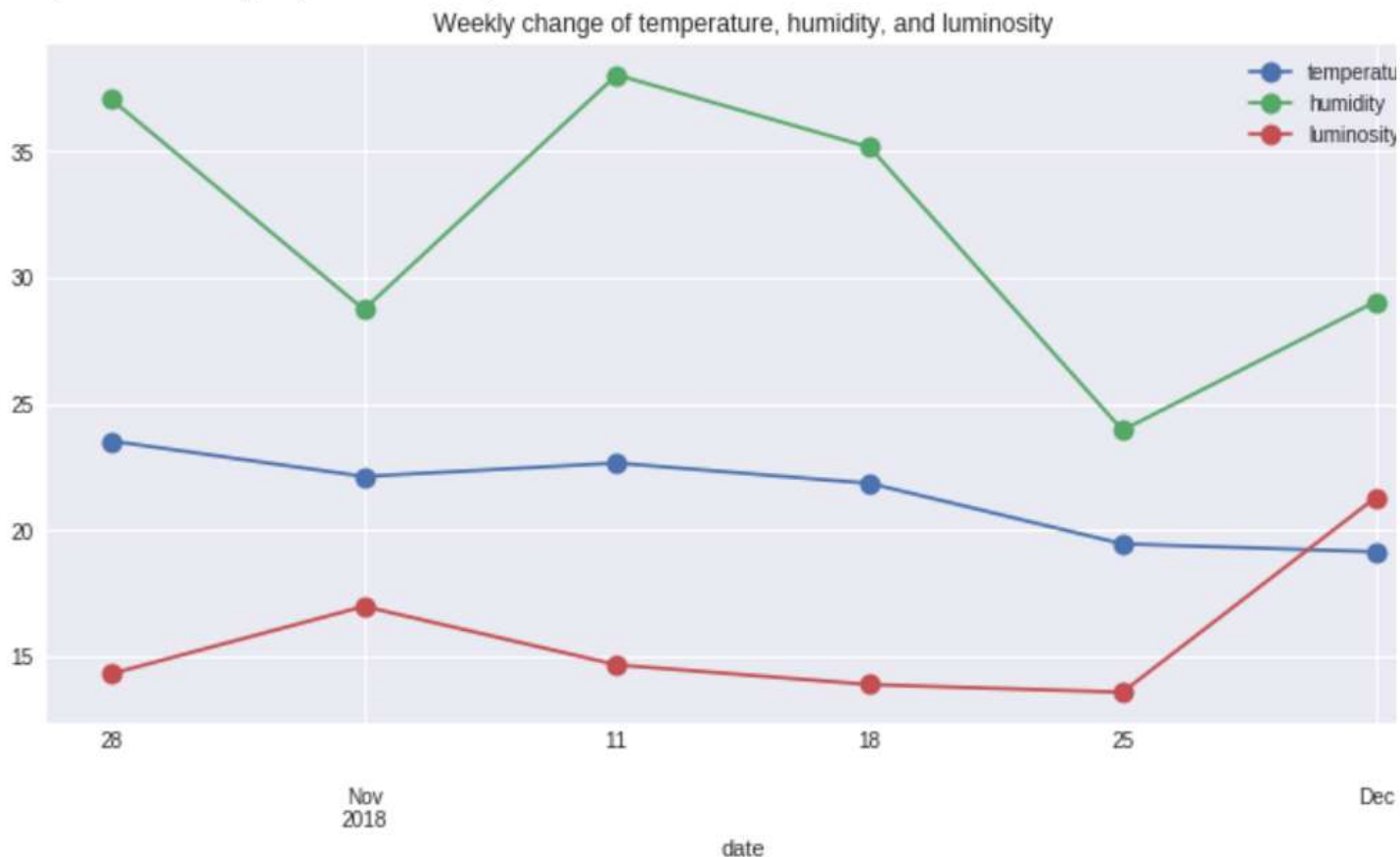
## A5.9.8 IOT data mining

### 3.6.3 Plot the change of sensor data over various time spans.

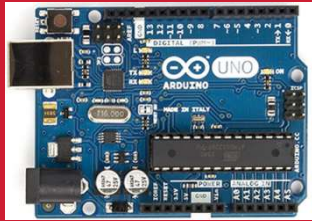
#### 1 주당 평균 그래프

```
1 # Plot mean of the iot data per every week
2 iot_data.resample('W').mean().plot(kind='line', marker='o', ms=10,
3                                     figsize=(12,6),
4                                     title='Weekly change of temperature, humidity, and luminosi
```

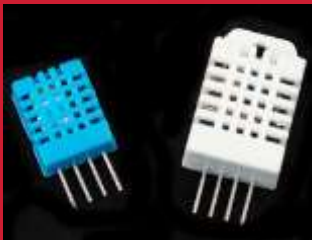
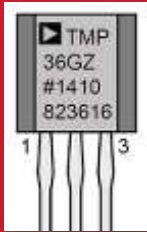
<matplotlib.axes.\_subplots.AxesSubplot at 0x7f5b2c8f8748>







# [Practice]



## ◆ [wk12]

- Data Mining of IoT Data
- Multi-sensor circuits (cds-dht22)
- Complete your project
- Upload folder: aann-rpt12
- Use repo “aann” in github

# wk12 : Practice : aann-rpt12

## ◆ [Target of this week]

- Complete your works
- Save your outcomes and upload outputs in github

제출폴더명 : **aann-rpt12**

- 제출할 파일들

- ① **iot\_csv.ipynb**
- ② **iot\_ison.ipynb**
- ③ **All \*.js**
- ④ **public/All \*.html**
- ⑤ **client\_iot.html**
- ⑥ **public/data/All data (\*.csv)**
- ⑦ **AAnn\_s1000.csv**

## ● References & good sites

- ✓ <http://www.arduino.cc> Arduino Homepage
- ✓ <http://www.nodejs.org/ko> Node.js
- ✓ <https://plot.ly/> plotly
- ✓ <https://www.mongodb.com/> MongoDB
- ✓ <http://www.w3schools.com> By w3schools
- ✓ <http://www.github.com> GitHub

# Target of this class

## Real-time Weather Station from nano 33 BLE sensors



on Time: 2020-09-09 10:27:17.321

