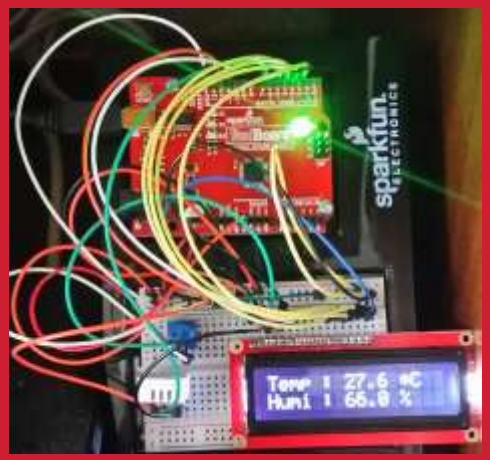
Arduino-IOT

[wk14]

Arduino + Node Data visualization



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

Drone-IoT-Comsi, INJE University

2nd semester, 2020

Email : chaos21c@gmail.com

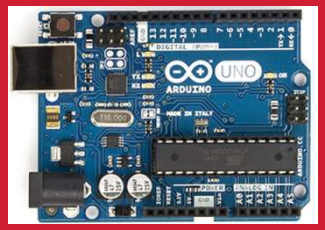
My ID

1분반-목요일 (2학년)

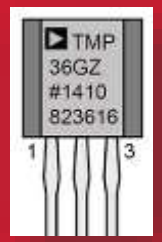
- AA1-01: 강서현
- AA1-02: 강태민
- AA1-03: 김세은
- AA1-04: 여수민
- AA1-05: 정영훈
- AA1-06: 차혁준
- AA1-07: 하태헌
- AA1-08: 김경욱
- AA1-09: 김민욱
- AA1-10: 김민성
- AA1-11: 김민준
- AA1-12: 김인수
- AA1-13: 김현식
- AA1-14: 장성운
- AA1-15: 전승진
- AA1-16: 정희철
- AA1-17: 조동현
- AA1-18: 전동빈
- AA1-19: 신종원

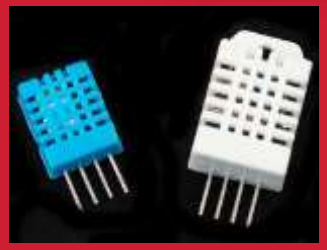
2분반-수요일 (3학년)

- AA2-01: 강민수
- AA2-02: 구병준
- AA2-03: 김종민
- AA2-04: 박성철
- AA2-05: 이승현
- AA2-06: 이창호
- AA2-07: 손성빈
- AA2-08: 안예찬
- AA2-09: 유종인
- AA2-10: 이석민
- AA2-11: 이정문
- AA2-12: 이주원
- AA2-13: 정재영
- AA2-14: 하태성
- AA2-15: 김경미
- AA2-16: 김규년
- AA2-17: 김유빈
- AA2-18: 송다은
- AA2-19: 정주은
- AA2-20: 권준표



[Review]





- [wk13]
  - RT Data storaging with MongoDB
  - Multi-sensor circuits (cds-dht22)
  - Complete your project
  - Upload folder: aax-nn-rpt10
  - Use repo “aax-nn” in github

wk13 : Practice : aax-nn-rpt10

- [Target of this week]
  - Complete your works
  - Save your outcomes and upload outputs in github

제출폴더명 : **aax-nn-rpt10**

- 압축할 파일들

① **AAnn_mongo_schemas.png**

② **AAnn_mongo_update.png**

③ AAnn_iot_mongodb.png

④ AAnn_iot_mongodb_web.png

⑤ **All *.ino**

⑥ **All *.js**

⑦ **All *.html**

Email : chaos21c@gmail.com

IOT: HSC

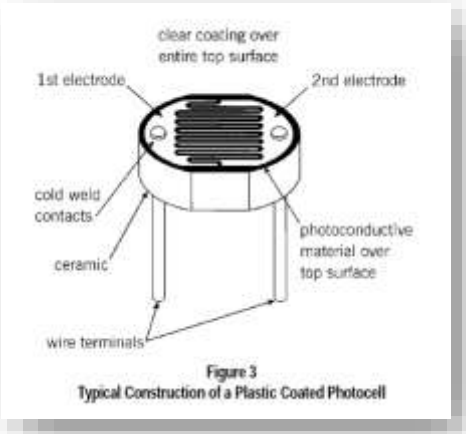
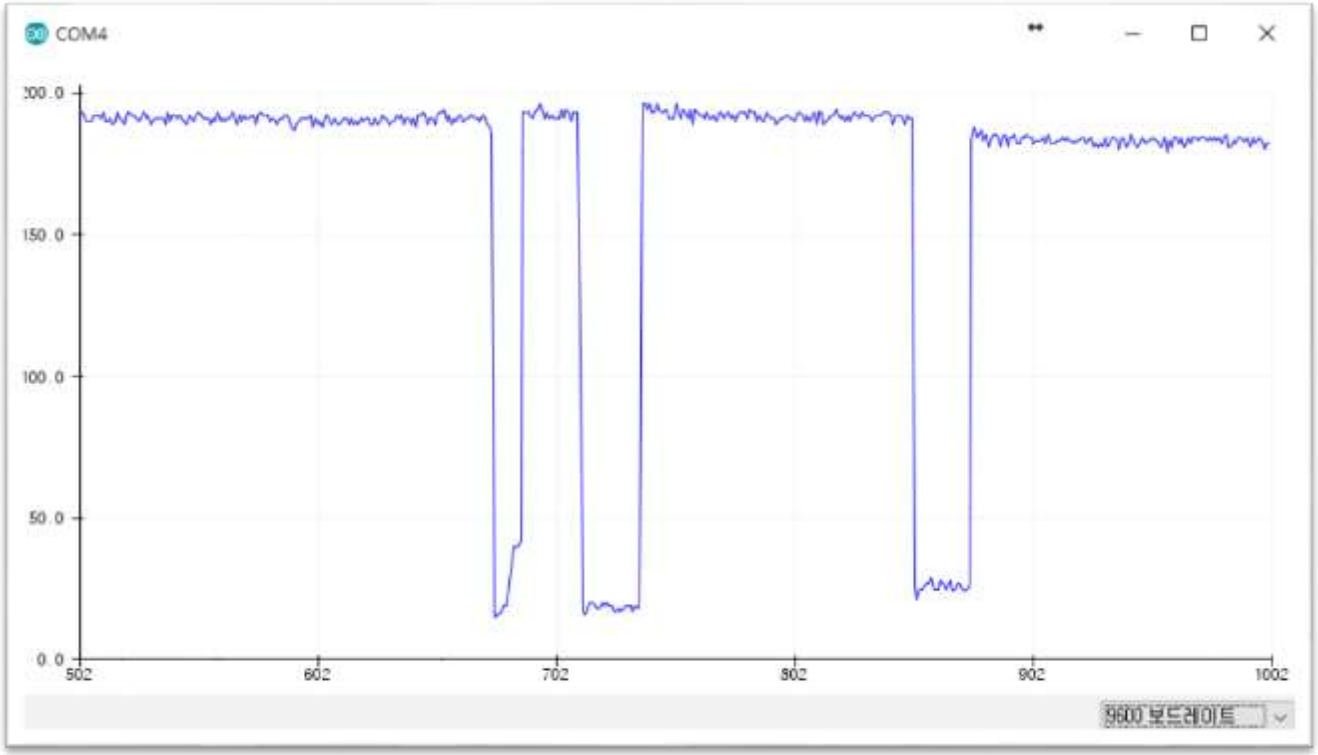
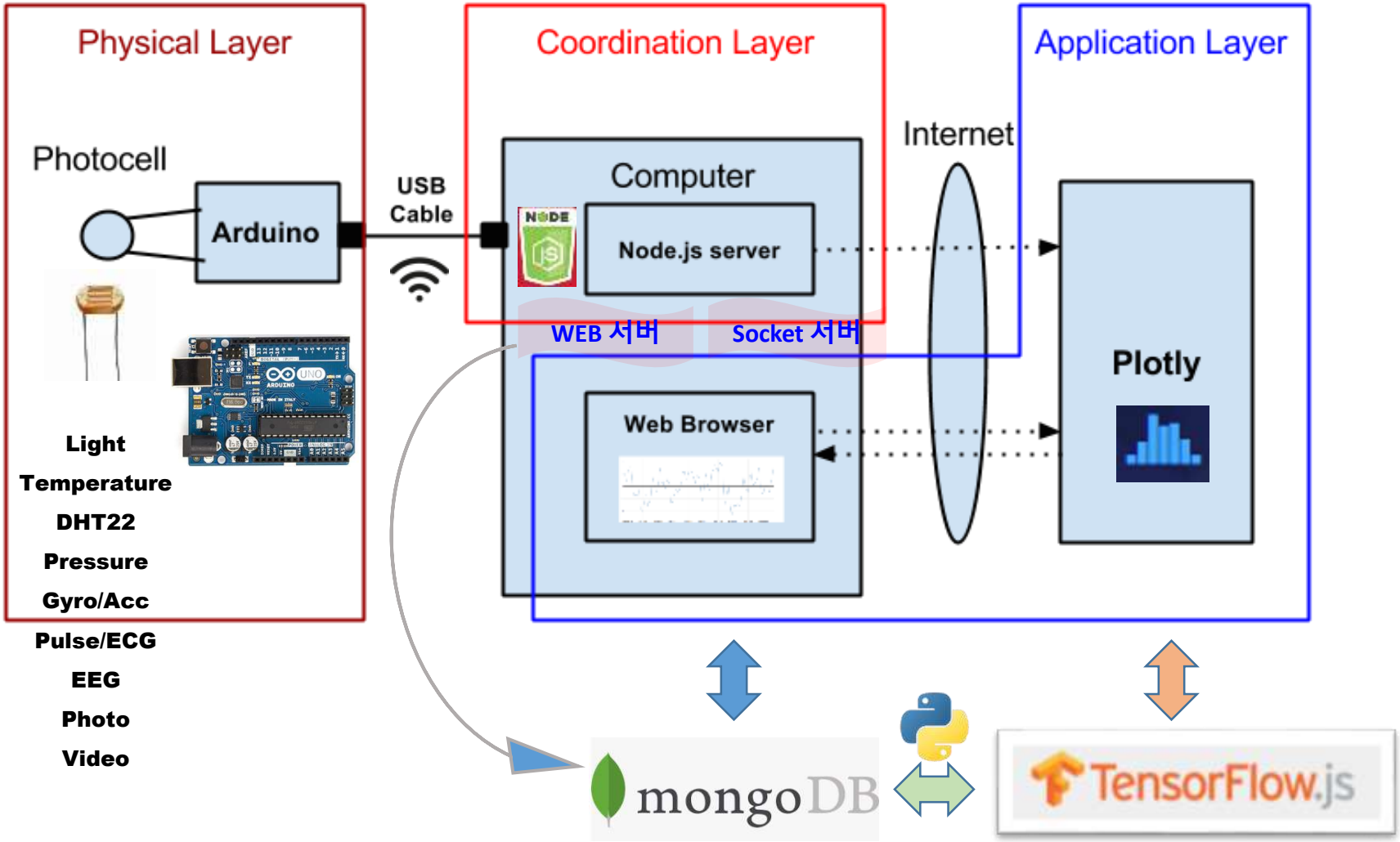


Figure 3
Typical Construction of a Plastic Coated Photocell

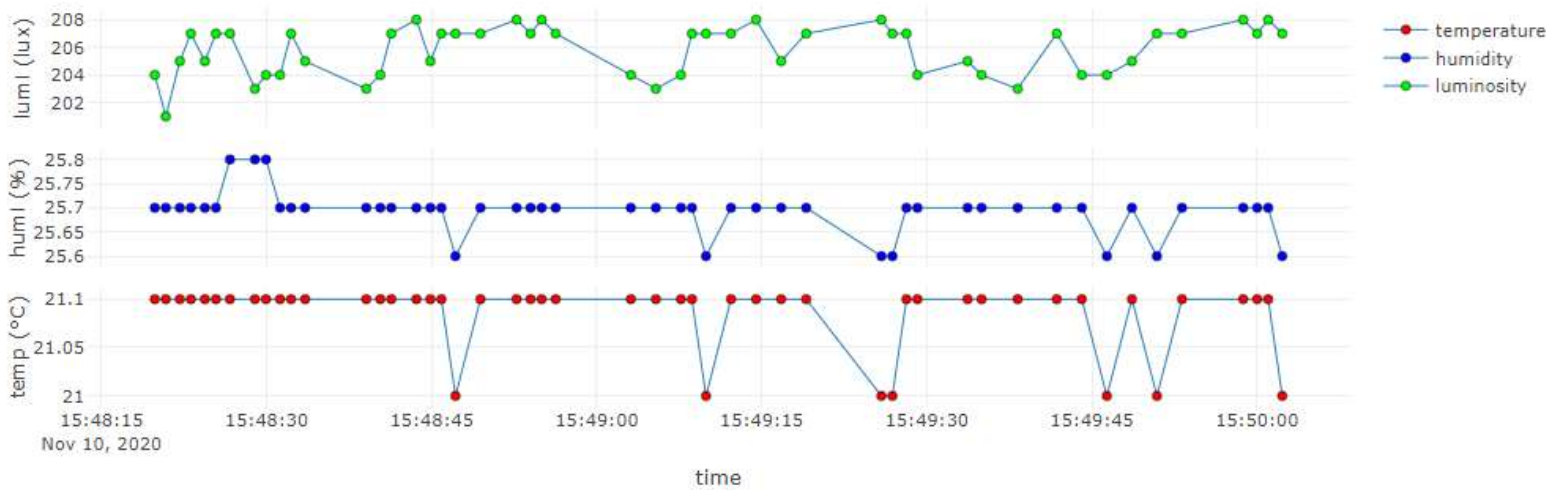


Layout [H S C]


Physical Layer
Photocell
Arduino
Light
Temperature
DHT22
Pressure
Gyro/Acc
Pulse/ECG
EEG
Photo
Video
USB
Cable
Coordination Layer
Computer
Node.js server
WEB 서버
Socket 서버
Web Browser
Internet
Application Layer
Plotly
mongoDB
TensorFlow.js

Real-time Weather Station from sensors

on Time: 2020-11-10 15:50:02.300



A5. Introduction to IoT service

System (Arduino, sDevice, ...)

↓

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

↓

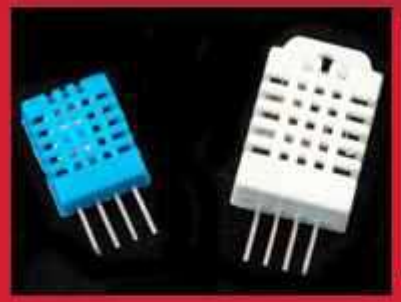
Visualization & monitoring

↓

Data storaging & mining

↓

Service



[Goal]

Arduino + Node.js

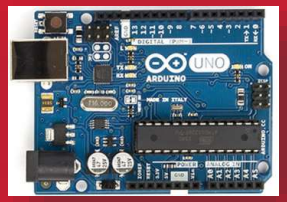
+ plotly.js

+ MongoDB

→ Data storaging

& visualization

& mining



Node.js

+

MongoDB

A5.9.4 MongoDB + Node.js：mongoose

mongoose

elegant mongodb object modeling for node.js

read the docs　discover plugins

Star　Version 5.10.15　Fork

Let's face it, **writing MongoDB validation, casting and business logic boilerplate is a drag**. That's why we wrote Mongoose.

```
const mongoose = require('mongoose');
mongoose.connect('mongodb://localhost:27017/test', {useNe

const Cat = mongoose.model('Cat', { name: String });
```

http://mongoosejs.com/

A5.9.4 MongoDB + Node.js：mongoose

Features Business Explore Marketplace Pricing This repository

Automattic / mongoose

Code Issues 250 Pull requests 2 Projects 0 Wiki Insights

MongoDB object modeling designed to work in an asynchronous environment. http://mon

8,362 commits 14 branches 420 releas

Branch: master New pull request

vkarpov15 Merge branch '4.x'

.github	fix typo
benchmarks	style: remove unused `BlogPost` variable
docs	Merge branch '4.x'

https://github.com/Automattic/mongoose

A5.9.4 MongoDB + Node.js : mongooseJS

1. Install mongoose in node.js project http://mongoosejs.com/

- Go to cds_dht22 project
- npm install --save mongoose (버전 : 5.10.15)

```
D:\Portable\vscode-portable\data\aa2-00\aa2-99-rpt09\wk11_src\Node>npm install --save mongoose
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN cds_dht22@1.0.0 No repository field.

+ mongoose@5.10.15
added 21 packages from 16 contributors and audited 149 packages in 3.331s

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

found 0 vulnerabilities


D:\Portable\vscode-portable\data\aa2-00\aa2-99-rpt09\wk11_src\Node>
```

A5.9.4 MongoDB + Node.js : mongoose

2. node.js project using mongoose (use VSCode)

- cds_dht22 project in vscode
- New file: dbtest.js
- node dbtest.js

```
// dbtest.js
var mongoose = require("mongoose");
mongoose.connect("mongodb://localhost/test", {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

var SensorSchema = new mongoose.Schema({
  data: String,
  created: Date,
});

// data model
var Sensor = mongoose.model("Sensor", SensorSchema);

var sensor1 = new Sensor({ data: "124", created: new Date() });
sensor1.save();

var sensor2 = new Sensor({ data: "573", created: new Date() });
sensor2.save();

console.log("Sensor data were saved in MongoDB");
```

```
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds_dht22>node dbtest
Sensor data were saved in MongoDB
^C
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds_dht22>
```

3. node.js project using mongoose (mongo shell)

Mongo shell

> show dbs

> use test

> show collections

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

```
> show dbs
aa99    0.000GB
admin   0.000GB
config  0.000GB
local   0.000GB
test    0.000GB
> use test
switched to db test
> show collections
sensors
> db.sensors.find()
{ "_id" : ObjectId("5fbcb10c7f7b5c2a7084834f"), "data" : "124", "created" : I
SODate("2020-11-24T07:06:52.096Z"), "__v" : 0 }
{ "_id" : ObjectId("5fbcb10c7f7b5c2a70848350"), "data" : "573", "created" : I
SODate("2020-11-24T07:06:52.100Z"), "__v" : 0 }
> db.sensors.find().pretty()
{
        "_id" : ObjectId("5fbcb10c7f7b5c2a7084834f"),
        "data" : "124",
        "created" : ISODate("2020-11-24T07:06:52.096Z"),
        "__v" : 0
}
{
        "_id" : ObjectId("5fbcb10c7f7b5c2a70848350"),
        "data" : "573",
        "created" : ISODate("2020-11-24T07:06:52.100Z"),
        "__v" : 0
}
```

4. dbtest2.js

```
// dbtest2.js
var mongoose = require("mongoose");
mongoose.connect("mongodb://localhost/test2", {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});
var SensorSchema = new mongoose.Schema({
  data: String,
  created: String,
});
// data model
var Sensor = mongoose.model("Sensor", SensorSchema);

var sensor1 = new Sensor({ data: "124", created: getDateString() });
sensor1.save();

var sensor2 = new Sensor({ data: "573", created: getDateString() });
sensor2.save();

console.log("[dbtest2.js]: Sensor data were saved in MongoDB");

// helper function to get a nicely formatted date string
function getDateString() {
  var time = new Date().getTime();
  // 32400000 is (GMT+9 Korea, GimHae)
  // for your timezone just multiply +/-GMT by 3600000
  var datestr = new Date(time + 32400000)
    .toISOString()
    .replace(/T/, " ")
    .replace(/Z/, "");
  return datestr;
}
```

```
var SensorSchema = new mongoose.Schema({
        data: String,
        created: String
});
```

A5.9.4 MongoDB + Node.js : mongoose

5. dbtest2.js (change Schema & check using mongo shell)

Mongo shell

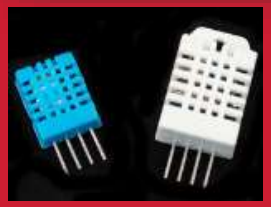
> show dbs

> use test2

> show collections

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

```
> show dbs
aa99    0.000GB
admin   0.000GB
config  0.000GB
local   0.000GB
test    0.000GB
test2   0.000GB
> use test2
switched to db test2
> db.sensors.find().pretty()
{
        "_id" : ObjectId("5fbcb31261c2ce07c4bb3401"),
        "data" : "124",
        "created" : "2020-11-24 16:15:30.214",
        "__v" : 0
}
{
        "_id" : ObjectId("5fbcb31261c2ce07c4bb3402"),
        "data" : "573",
        "created" : "2020-11-24 16:15:30.217",
        "__v" : 0
}
>
```



MongoDB from Arduino with node.js & mongoose

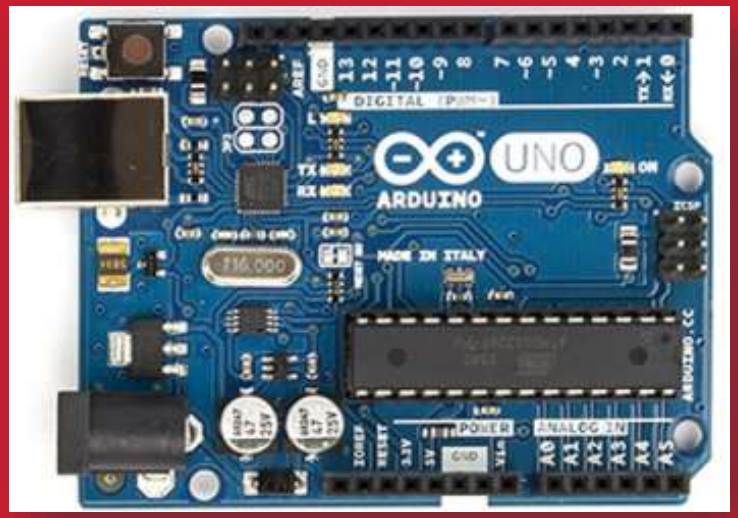
```
> show dbs
aa00     0.000GB
admin    0.000GB
config   0.000GB
iot      0.000GB
iot2     0.000GB
iot3     0.001GB
local    0.000GB
test     0.000GB
test2    0.000GB
>
```

```
mongo db connection OK.
info() - Current date is 2015-11-26 12:04:21.411, Lumi: 67
info() - Current date is 2015-11-26 12:04:26.415, Lumi: 67
info() - Current date is 2015-11-26 12:04:31.416, Lumi: 67
info() - Current date is 2015-11-26 12:04:36.422, Lumi: 104
info() - Current date is 2015-11-26 12:04:41.427, Lumi: 92
info() - Current date is 2015-11-26 12:04:46.432, Lumi: 410
info() - Current date is 2015-11-26 12:04:51.432, Lumi: 67
info() - Current date is 2015-11-26 12:04:56.438, Lumi: 66
```

Arduino

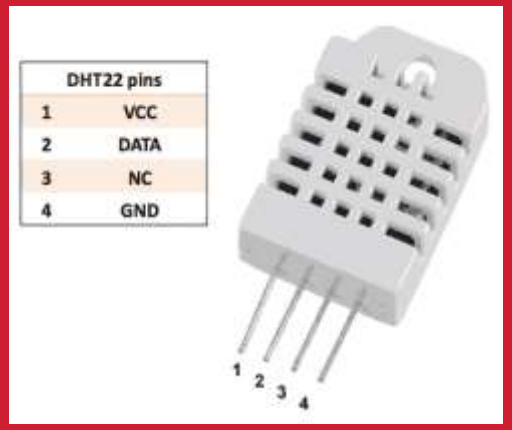
& Node.js

& MongoDB

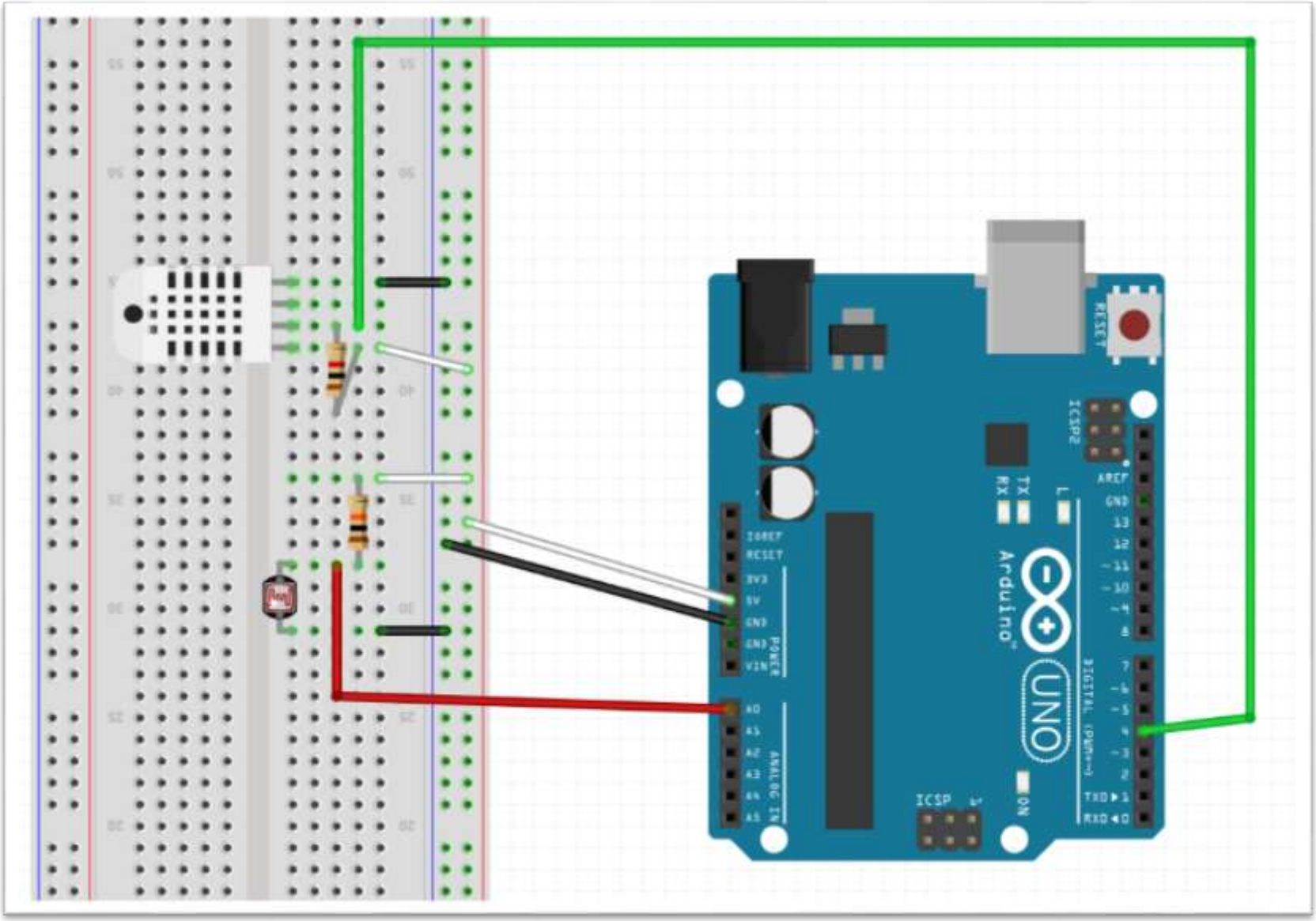


Multi-sensors

DHT22 + CdS

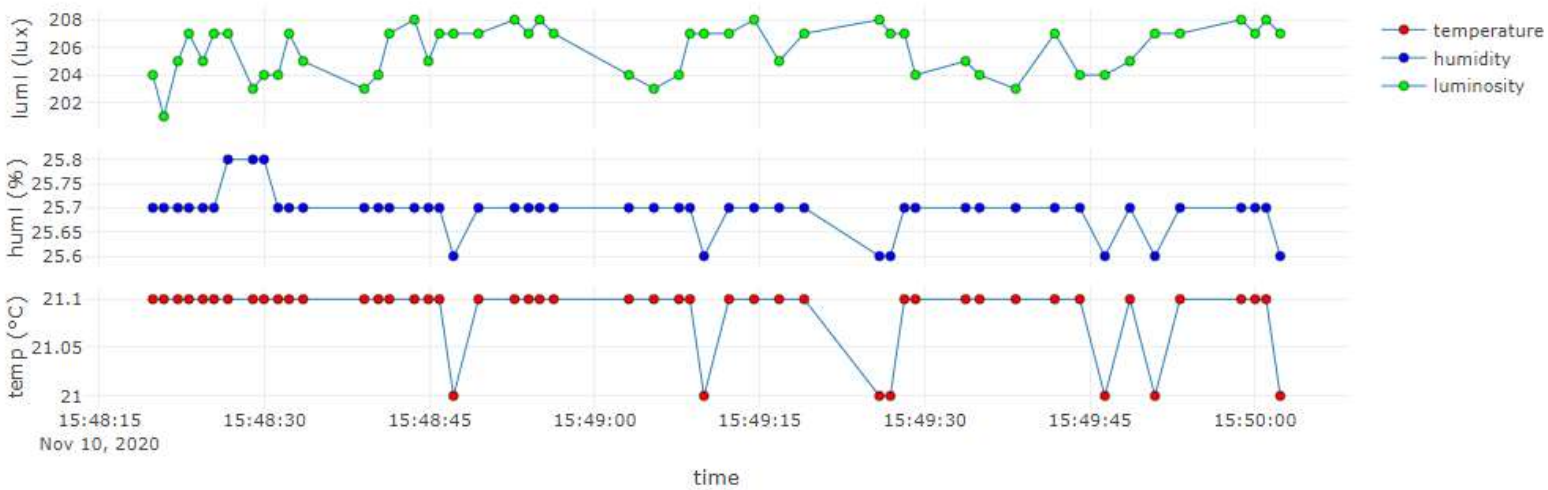


DHT22 + CdS : circuit



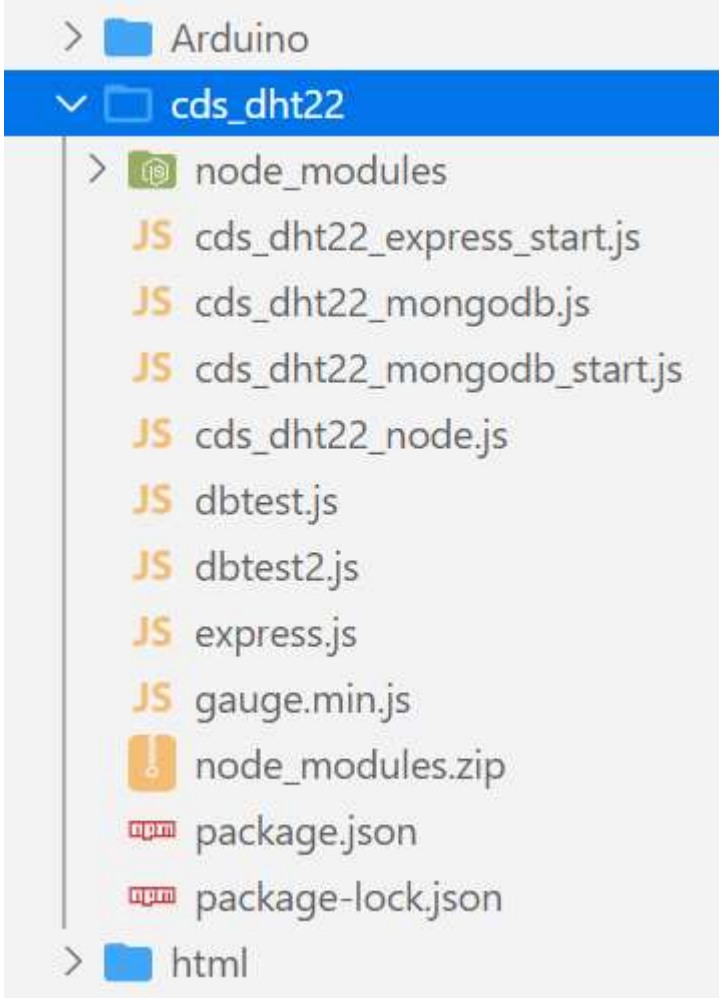
Real-time Weather Station from sensors

on Time: 2020-11-10 15:50:02.300



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

1. 작업 폴더 구조 [2020]



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

2.1 cds_dht22_mongodb.js

```
// cds_dht22_mongodb.js

var serialport = require("serialport");
var portName = "COM3"; // check your COM port!!
var port = process.env.PORT || 3000;

var io = require("socket.io").listen(port);

// MongoDB
var mongoose = require("mongoose");
var Schema = mongoose.Schema;
// MongoDB connection
mongoose.connect("mongodb://localhost:27017/iot", {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

```

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

2.1 cds_dht22_mongodb.js

```
18  var db = mongoose.connection;
19  db.on("error", console.error.bind(console, "connection error:"));
20  db.once("open", function callback() {
21    console.log("mongo db connection OK.");
22  });
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

2.2 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;
```

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

2.2 cds_dht22_mongodb.js

```
70  var Sensor = mongoose.model("Sensor", iotSchema); // sensor data model
71
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
77    // parsing data into signals
78    if (readData.lastIndexOf(",") > firstcommaidx && firstcommaidx > 0) {
79      temp = readData.substring(
80        firstcommaidx + 1,
81        readData.indexOf(",", firstcommaidx + 1)
82      );
83      humi = readData.substring(
84        readData.indexOf(",", firstcommaidx + 1) + 1,
85        readData.lastIndexOf(",")
86      );
87      lux = readData.substring(readData.lastIndexOf(",") + 1);
88      readData = "";
```

2.2 cds_dht22_mongodb.js

```
90      dStr = getDateString();
91      mdata[0] = dStr; // Date
92      mdata[1] = temp; // temperature data
93      mdata[2] = humi; // humidity data
94      mdata[3] = lux; // luminosity data
95      var iot = new Sensor({
96        date: dStr,
97        temperature: temp,
98        humidity: humi,
99        luminosity: lux,
100     });
101     // save iot data to MongoDB
102     iot.save(function (err, iot) {
103       if (err) return handleEvent(err);
104       iot.info(); // Display the information of iot data  on console.
105     });
106     io.sockets.emit("message", mdata); // send data to all clients
107   } else {
108     // error
109     console.log(readData);
110   }
111 });
```

2.4 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
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

2.5 node cds_dht22_mongodb.js [vscode 터미널에서 실행]

```
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds_dht22>node cds_dht22_mongodb
serial port open
mongo db connection OK.
iotInfo: Current date: 2020-11-25 09:27:08.915, Temp: 15.9, Humi: 23.7, Lux: 312
iotInfo: Current date: 2020-11-25 09:27:09.915, Temp: 15.9, Humi: 23.7, Lux: 315
iotInfo: Current date: 2020-11-25 09:27:11.193, Temp: 15.9, Humi: 23.7, Lux: 312
iotInfo: Current date: 2020-11-25 09:27:12.192, Temp: 15.9, Humi: 23.7, Lux: 312
iotInfo: Current date: 2020-11-25 09:27:13.470, Temp: 15.9, Humi: 23.7, Lux: 310
iotInfo: Current date: 2020-11-25 09:27:14.470, Temp: 15.9, Humi: 23.7, Lux: 310
iotInfo: Current date: 2020-11-25 09:27:15.747, Temp: 15.9, Humi: 23.7, Lux: 310
iotInfo: Current date: 2020-11-25 09:27:16.747, Temp: 15.9, Humi: 23.7, Lux: 312
iotInfo: Current date: 2020-11-25 09:27:18.024, Temp: 15.9, Humi: 23.7, Lux: 312
iotInfo: Current date: 2020-11-25 09:27:19.024, Temp: 15.9, Humi: 23.7, Lux: 315
iotInfo: Current date: 2020-11-25 09:27:20.302, Temp: 15.9, Humi: 23.7, Lux: 312
```

3. cds_dht22_mongodb.js → Check documents in Mongo shell

Mongo shell

> show dbs

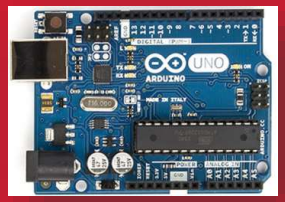
> use iot

> show collections

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

```
> show dbs
aa99    0.000GB
admin   0.000GB
config  0.000GB
iot     0.000GB
local   0.000GB
test    0.000GB
test2   0.000GB
> use iot
switched to db iot
> show collections
sensors
> db.sensors.find().pretty()
{
        "_id" : ObjectId("5fbda4354fe24d3218cf1400"),
        "date" : "2020-11-25 09:24:21.094",
        "temperature" : "16.1",
        "humidity" : "23.6",
        "luminosity" : "315",
        "__v" : 0
}
{
        "_id" : ObjectId("5fbda4354fe24d3218cf1401"),
        "date" : "2020-11-25 09:24:21.098",
        "temperature" : "16.2",
        "humidity" : "23.6",
        "luminosity" : "312",
        "__v" : 0
}
```

Save as AAnn_iot_mongdb.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:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds_dht22>npm install --save express

npm WARN cds_dht22@1.0.0 No repository field.

+ express@4.17.1
added 51 packages from 33 contributors and audited 200 packages in 3.078s

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

found 0 vulnerabilities
```

1.2 Install express server

- Go to cds_dht22 project
- npm install --save express
- package.json

```
"name": "cds_dht22",
"version": "1.0.0",
"description": "cds-dht22-node project",
"main": "cds_dht22_node.js",
▷ Debug
"scripts": {
  "test": "echo \"Error: no test specifi
},
"author": "aa00",
"license": "MIT",
"dependencies": {
  "express": "^4.17.1",
  "mongoose": "^5.10.15",
  "serialport": "^9.0.1",
  "socket.io": "^2.3.0"
}
```

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 m
```

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

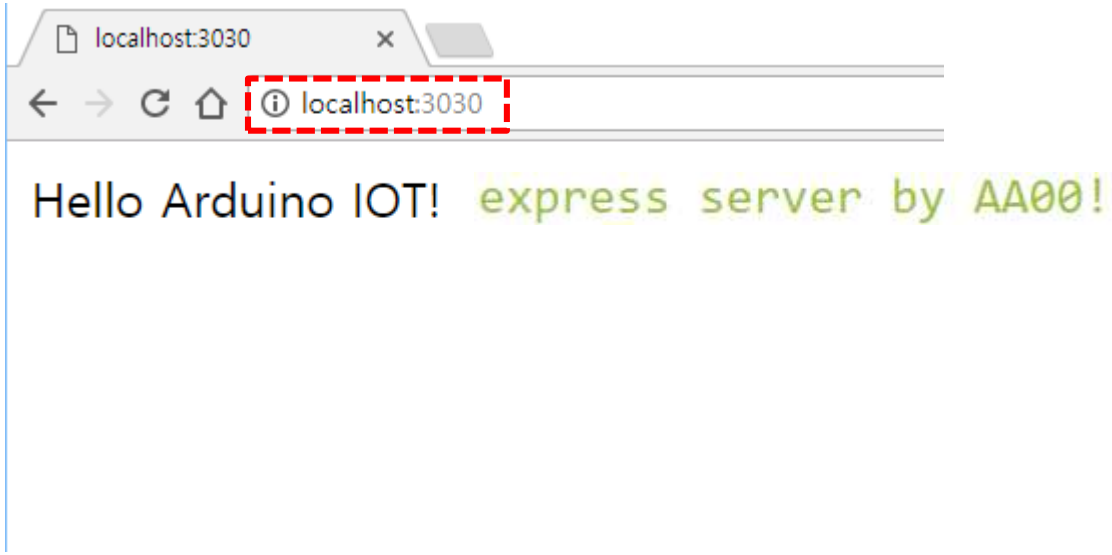
2.3 cds_dht22_express.js → Run
(cds_dht22_mongodb.js 는 현재 running!)

```
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_sta
rt\cds_dht22>node cds_dht22_mongodb
serial port open
mongo db connection OK.
iotInfo: Current date: 2020-11-25 10:16:06.207, Temp: 19
.4, Humi: 24.3, Lux: 212
iotInfo: Current date: 2020-11-25 10:16:08.484, Temp: 19
.4, Humi: 24.3, Lux: 213
iotInfo: Current date: 2020-11-25 10:16:10.757, Temp: 19
.4, Humi: 24.3, Lux: 212
iotInfo: Current date: 2020-11-25 10:16:13.034, Temp: 19
.4, Humi: 24.3, Lux: 213
iotInfo: Current date: 2020-11-25 10:16:15.312, Temp: 19
.4, Humi: 24.3, Lux: 212
iotInfo: Current date: 2020-11-25 10:16:17.585, Temp: 19
.4, Humi: 24.3, Lux: 212
```

```
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds
_dht22>node cds_dht22_express
Express_IOT is running at port:3030
mongo db connection OK.
```

Now, two servers are running

2.4 cds_dht22_express.js → routing1, http://localhost:3030/



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

2.5 cds_dht22_express.js → routing2 http://localhost:3030/iot

localhost:3030/iot

http://localhost:3030/iot


```
[{"_id":"5fbdab6ad02de805786af439","date":"2020-11-25
09:55:06.241","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab6cd02de805786af43a","date":"2020-11-25
09:55:08.514","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab6ed02de805786af43b","date":"2020-11-25
09:55:10.791","temperature":"18.9","humidity":"24.6","luminosity":"207","__v":0},
{"_id":"5fbdab71d02de805786af43c","date":"2020-11-25
09:55:13.068","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab73d02de805786af43d","date":"2020-11-25
09:55:15.341","temperature":"18.9","humidity":"24.7","luminosity":"208","__v":0},
{"_id":"5fbdab75d02de805786af43e","date":"2020-11-25
09:55:17.619","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab77d02de805786af43f","date":"2020-11-25
09:55:19.896","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab7ad02de805786af440","date":"2020-11-25
09:55:22.169","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab7cd02de805786af441","date":"2020-11-25
09:55:24.446","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab7ed02de805786af442","date":"2020-11-25
09:55:26.723","temperature":"18.9","h
```


Save as
AAnn_iot_mongodb_web.png

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

2.6 cds_dht22_express.js → routing2 http://localhost:3030/iot:id

localhost:3030/iot/5fbdab71d02

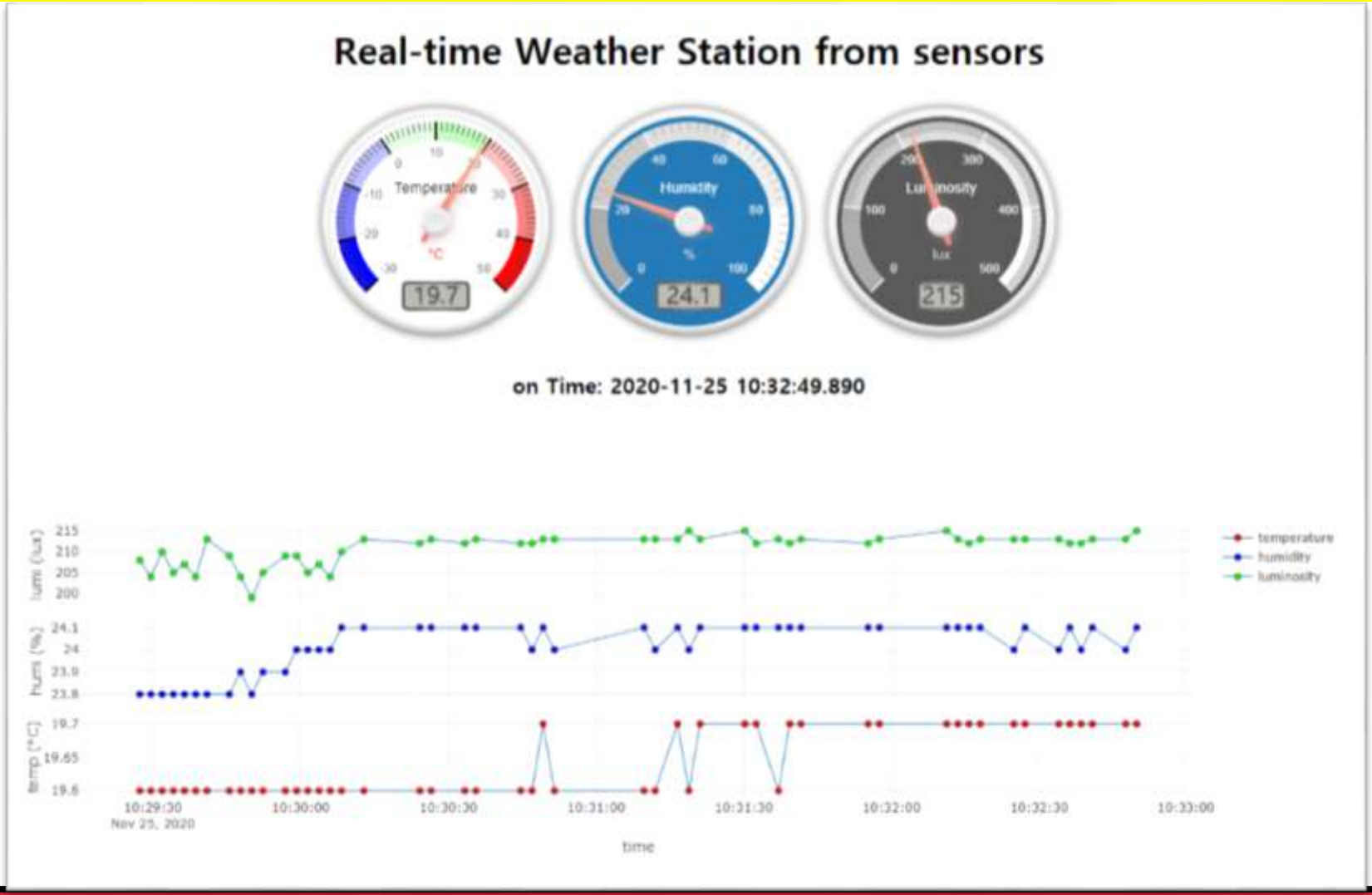
http://localhost:3030/iot/5fbdab71d02de805786af43c


```
{"_id":"5fbdab71d02de805786af43c","date":"2020-11-25
09:55:13.068","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0}
```

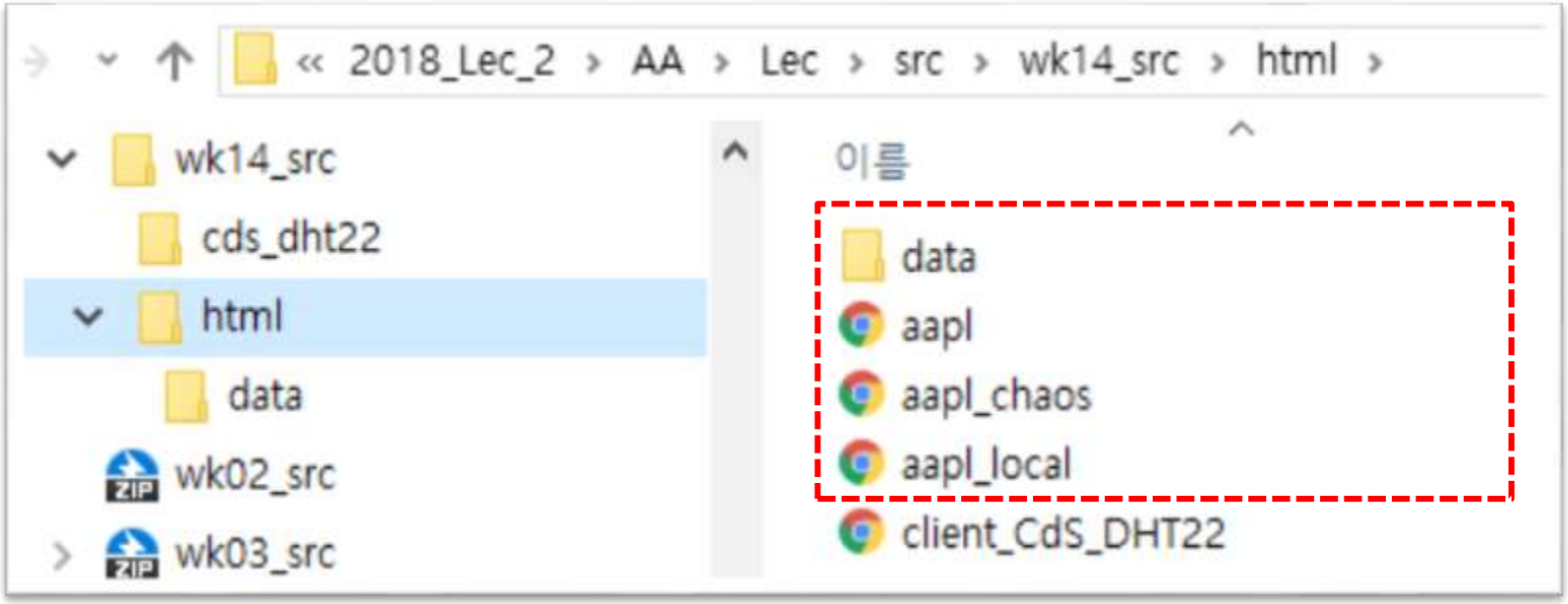
A5.9.6 DHT22 + CdS + Node.js + MongoDB

2.7 copy cds_dht22_client.html & gauge.min.js → ./public/ subfolder

http://localhost:3030/client_cds_dht22.html (web root folder)



2.8 CORS bug (Cross Origin Resource Sharing)



Apple 사의 주가그래프를 그리는 html client 3개를 실행하고 결과를 비교.
→ Local file에 접근 불허 ?
→ CORS problem

→ public 폴더로 html,data를 복사한 후에 비교.

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

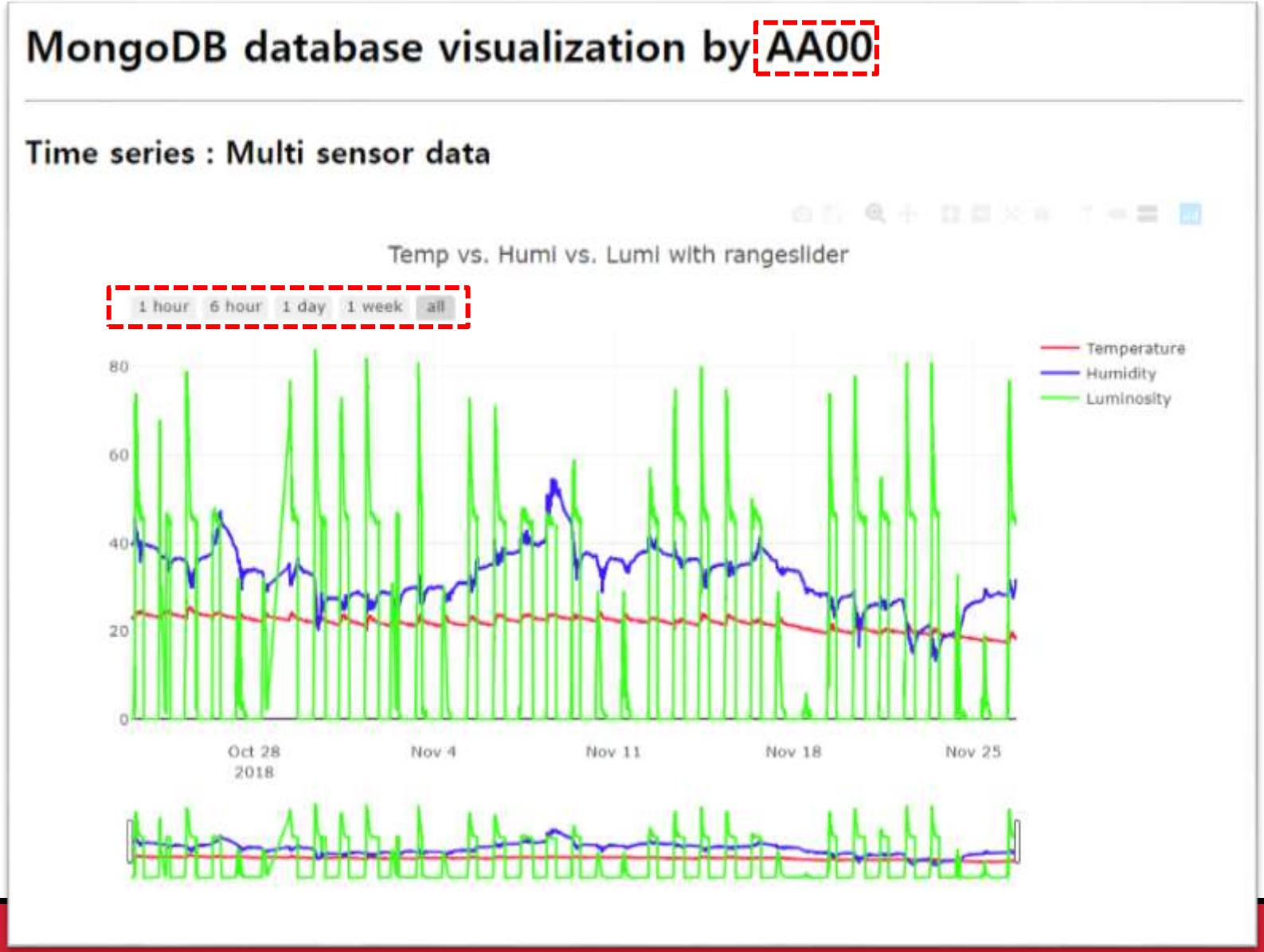
2.9 CORS patch on the express server → cds_dht22_express.js
Node cmd에서 'cors' module 설치 (version 2.8.4 이상)
npm install –save cors

```
// cds_dht22_express_cors.js
// Express + CORS
var express = require("express");
var cors = require("cors");
var app = express();
app.use(cors());
var web_port = 3030; // express port

// MongoDB
var mongoose = require("mongoose");
var Schema = mongoose.Schema; // Schema object
```

Web monitoring

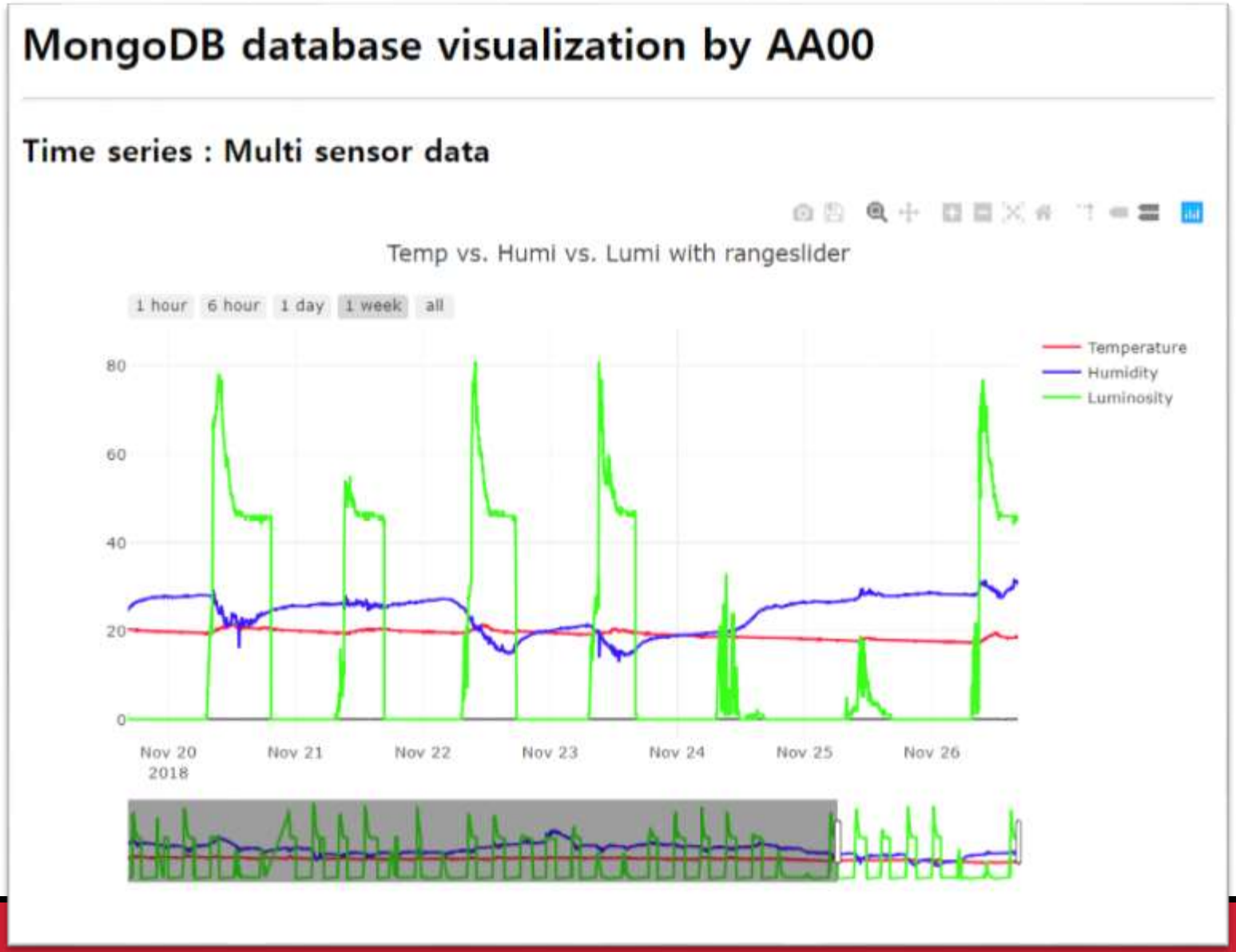
A5.9.7 DHT22 + CdS + Node.js + MongoDB

3.5 Web client: client_iotDB.html – iot DB monitoring (public 폴더에서 제공)



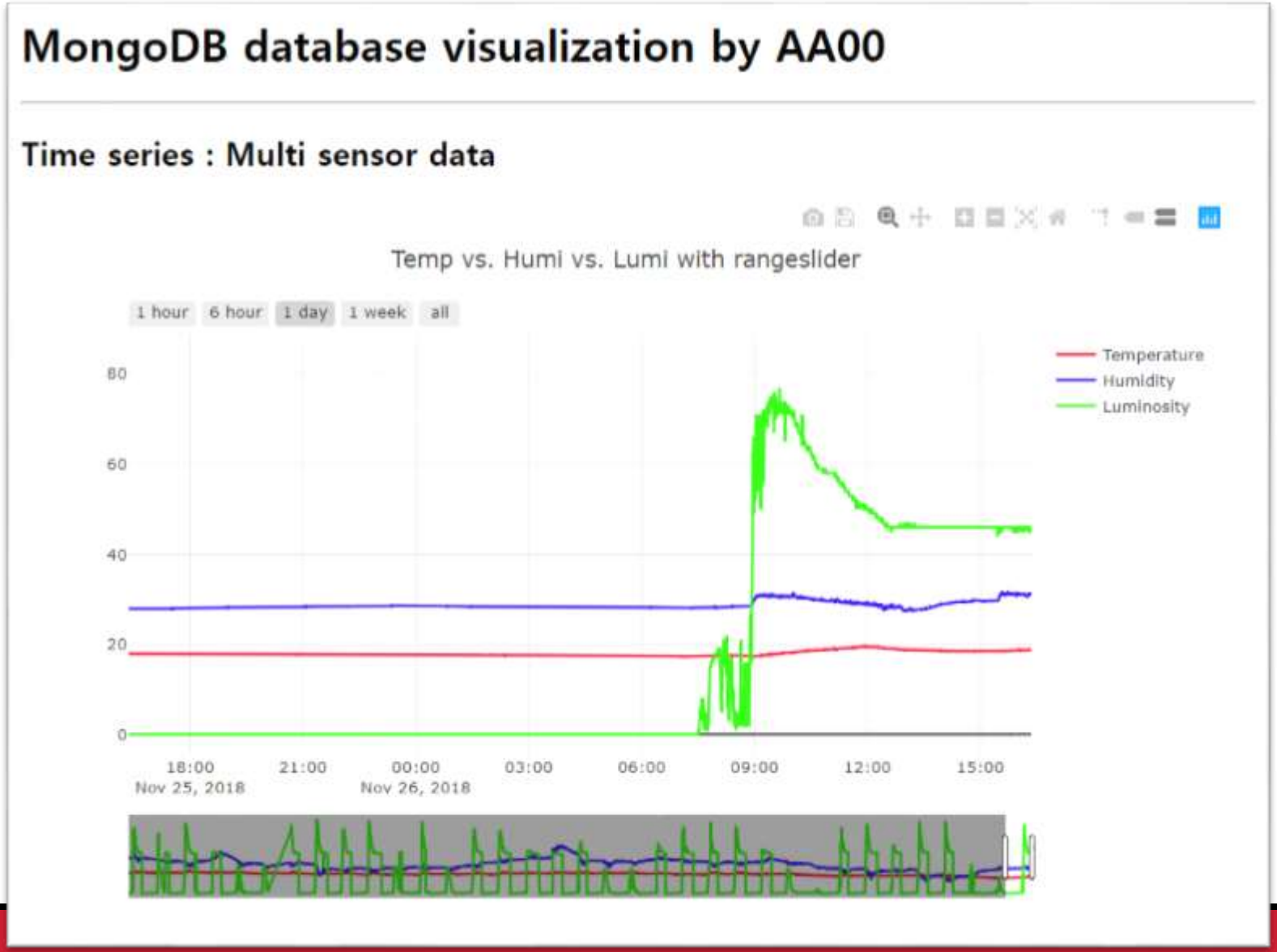
A5.9.7 DHT22 + CdS + Node.js + MongoDB

Web monitoring-2: week



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

Web monitoring-3: day



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

3.1 Web client: client_iotDB.html

client_iotDB.html

```html
<!DOCTYPE html>
<head>
    <meta charset="utf-8">
    <!-- Plotly.js -->
    <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
</head>
<body>
    <h1>MongoDB database visualization by AA00</h1>
    <hr>
    <h2>Time series : Multi sensor data</h2>

    <!-- Plotly chart will be drawn inside this DIV -->
    <div id="myDiv" style="width: 900px;height: 600px"></div>

```

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":"5fbdab71d02de805786af43c","date":"2020-11-25
09:55:13.068","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab73d02de805786af43d","date":"2020-11-25
09:55:15.341","temperature":"18.9","humidity":"24.7","luminosity":"208","__v":0},
{"_id":"5fbdab75d02de805786af43e","date":"2020-11-25
```

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

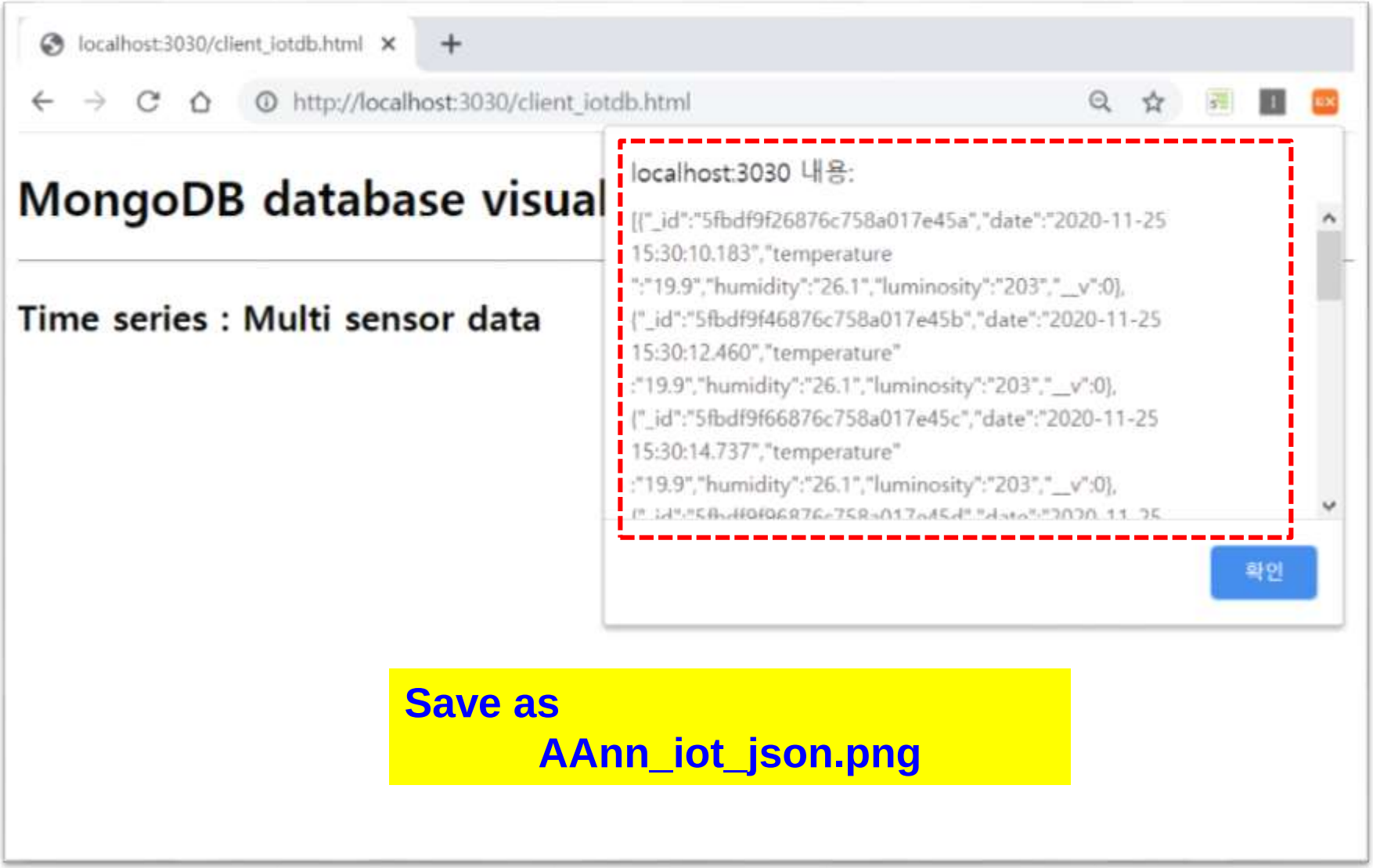
3.3 Web client: client_iotDB.html – data & layout

```
// 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'
    },
    yaxis: {
        autorange: true,
        range: [0, 300],
        type: 'linear'
    }
};

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

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

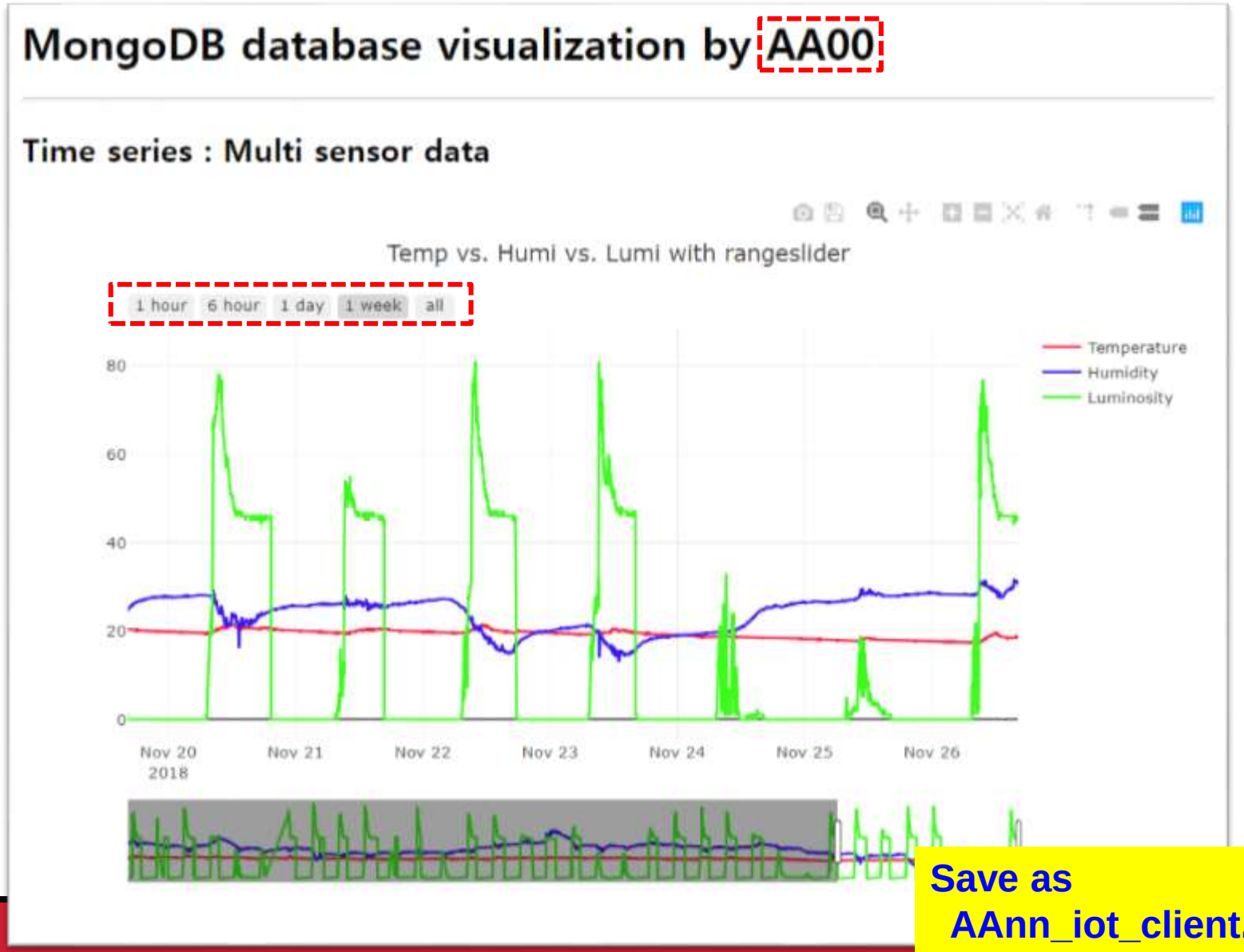
3.4 Web client: client_iotDB.html – load iot data in json file



Save as
AAnn_iot_json.png

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

3.5 Web client: client_iotDB.html – iot DB monitoring



Save as AAnn_iot_client.png

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: “2020-12-02 16:26:05”}}) → 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 (windows cmd 창에서 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

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

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

A5.9.8 MongoDB management

[Tip] iot db의 최근 데이터 500개를 csv 파일 (iot_s500.csv)로 저장할 때,

- **mongoexport -d=iot -c=sensors --sort="{_id: -1}" --limit=500 --fields =date,temperature,humidity,luminosity --type=csv --out=iot_s500.csv**

```
D:\Portable\vscode-portable\data\aa2-00\src\wk13_src_start\cds_dht22>mongoexport -d=iot
 -c=sensors --sort="{_id: -1}" --limit=500 --fields=date,temperature,humidity,luminosit
y --type=csv --out=iot_s500.csv
2020-12-01T16:40:37.472+0900    connected to: mongodb://localhost/
2020-12-01T16:40:37.547+0900    exported 500 records
```

iot_s500.csv

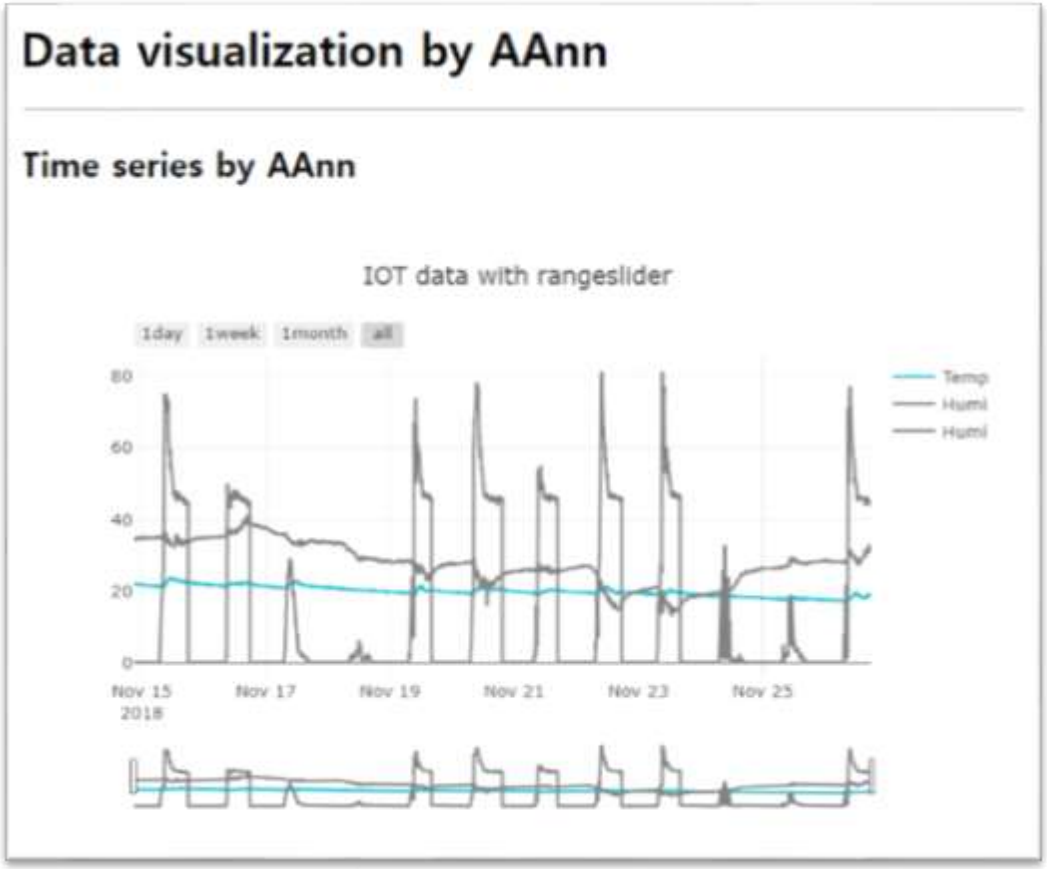
aa2-00 > src > wk13_src_start > cds_dht22 > iot_s500.csv

```
date,temperature,humidity,luminosity
2020-12-01 16:40:36.756,22.3,21.1,180
2020-12-01 16:40:34.482,22.3,21.1,179
2020-12-01 16:40:32.205,22.3,21.1,179
2020-12-01 16:40:29.928,22.4,21.2,179
2020-12-01 16:40:27.654,22.3,21.1,179
```

A5.9.8 MongoDB management

[DIY]

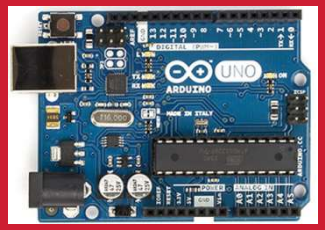
1. iot db의 최근 데이터 1000개를 csv 파일 (AAnn_s1000.csv)로 저장하시오.
2. 저장된 AAnn_s1000.csv 파일을 public/data 폴더에 복사.
3. csv 파일을 이용하는 Rangeslider가 포함된 웹 클라이언트 client_iot.html 파일을 만드시오.
4. Localhost:3030/client_iot.html 로 실행하고 확인.



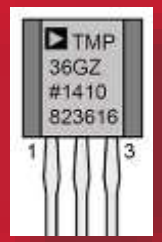
iot_chaos.html을

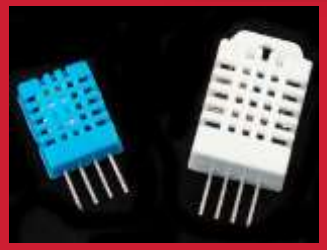
client_iot.html로

이름 변경해서 코드를 완성하시오.



[Practice]





- [wk14]
  - RT Data visualization with MongoDB
  - Multi-sensor circuits (cds-dht22)
  - Complete your project
  - Upload folder: aax-nn-rpt10
  - Use repo “aax-nn” in github

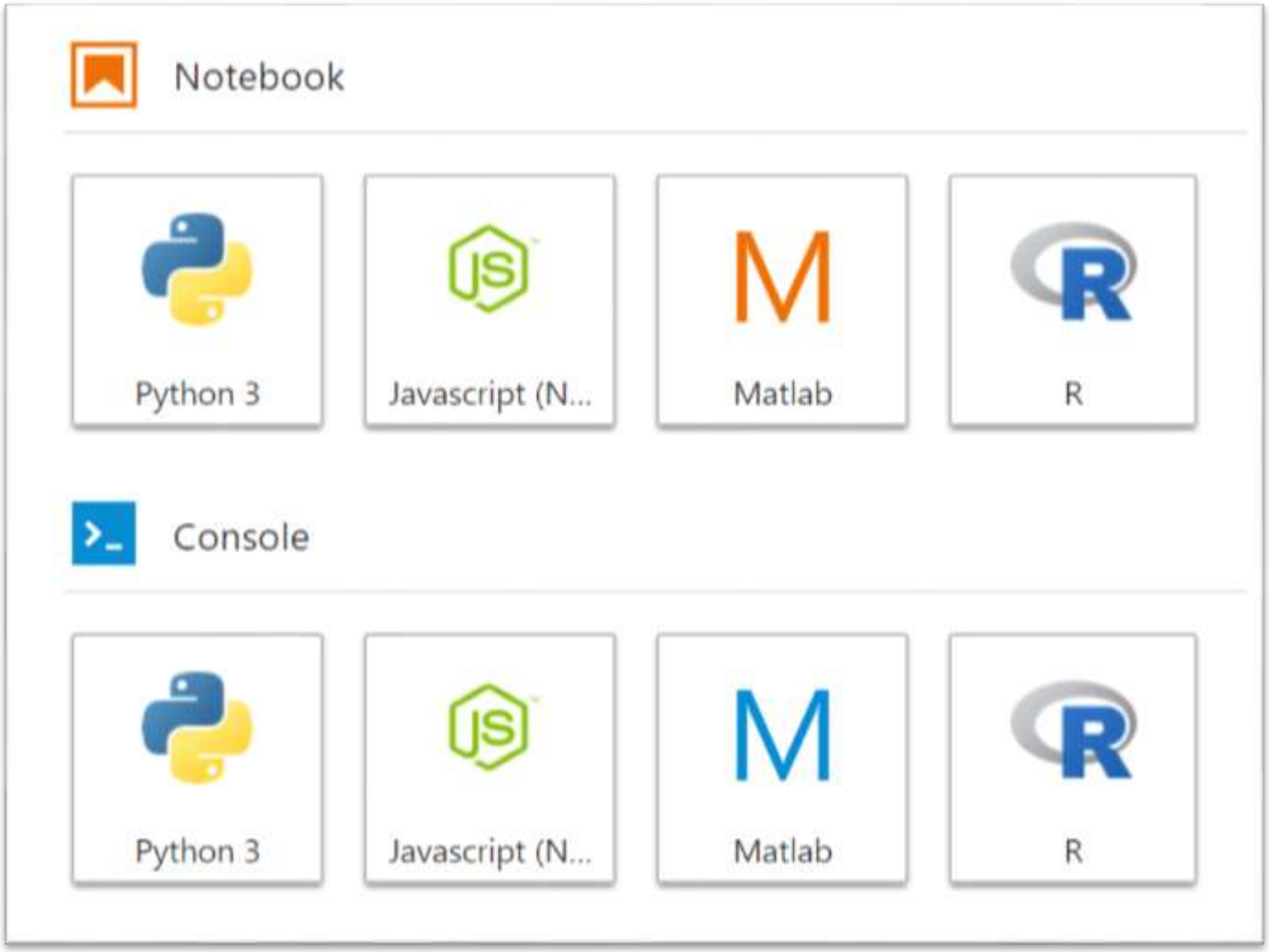
wk14 : Practice : aax-nn-rpt10

- [Target of this week]
  - Complete your works
  - Save your outcomes and upload outputs in github

제출폴더명 : **aax-nn-rpt10**
- 제출할 파일들 ① AAnn_iot_mongodb.png ② AAnn_iot_mongodb_web.png ③ AAnn_iot_json.png ④ AAnn_iot_client.png ⑤ All *.js ⑥ public/All *.html ⑦ public/data/All data (*.csv)

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]
```
import pandas as pd
```

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

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

[3]
```
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

3.3 Make dataframe from json data

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

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

Json 객체에서 필요한 항목을 선택해서 pandas의 dataframe을 구성한다.

```
[ ]  iot_data.shape
```

```
(340230, 4)
```

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

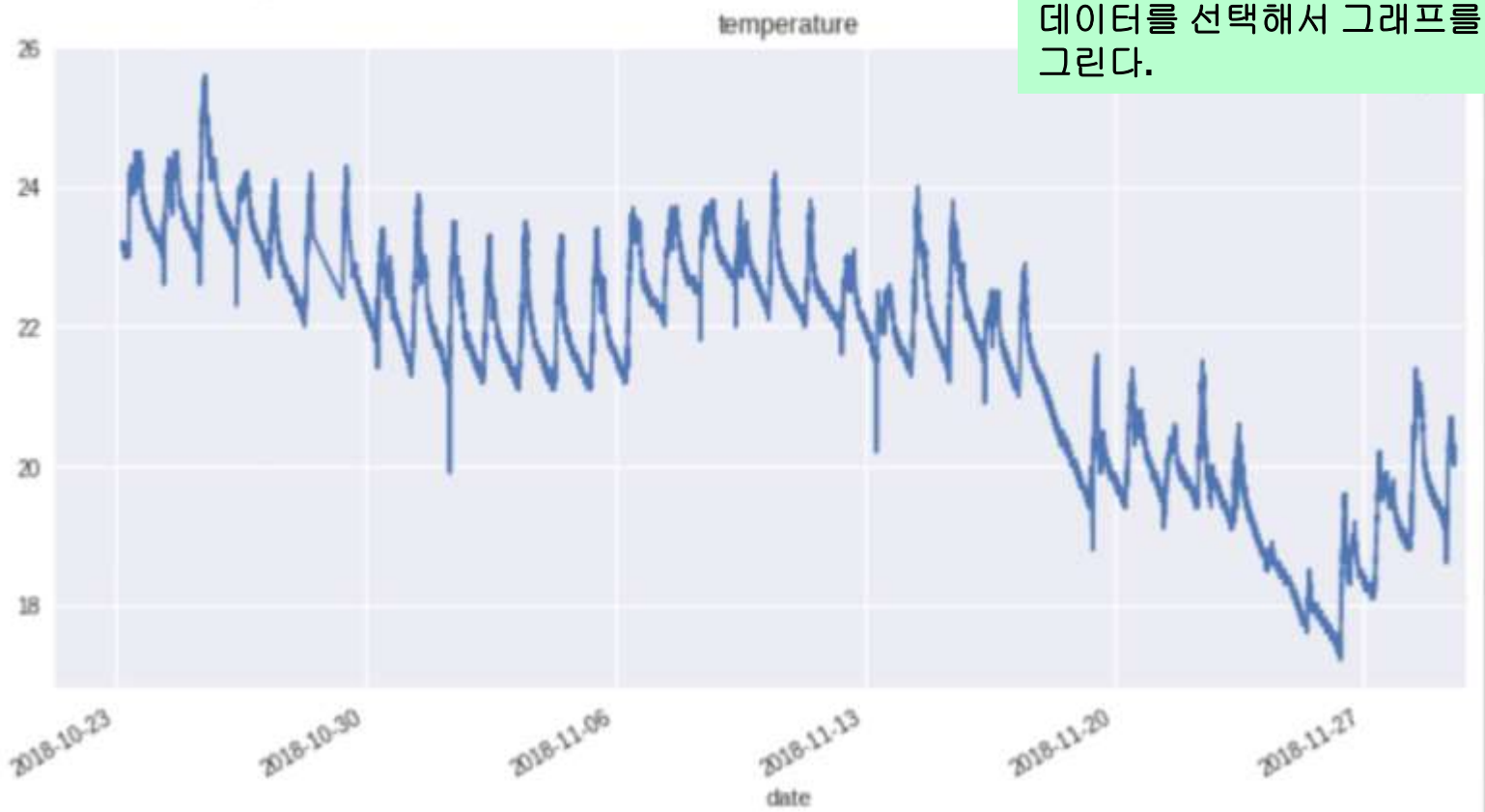
	date	temperature	humidity	luminosity
0	2018-10-23 04:25:21.349	23.2	39.7	0
1	2018-10-23 04:25:31.594	23.2	39.7	0
2	2018-10-23 04:25:41.855	23.2	39.7	0
3	2018-10-23 04:25:52.100	23.2	39.7	0
4	2018-10-23 04:26:02.360	23.2	39.7	0

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>
```



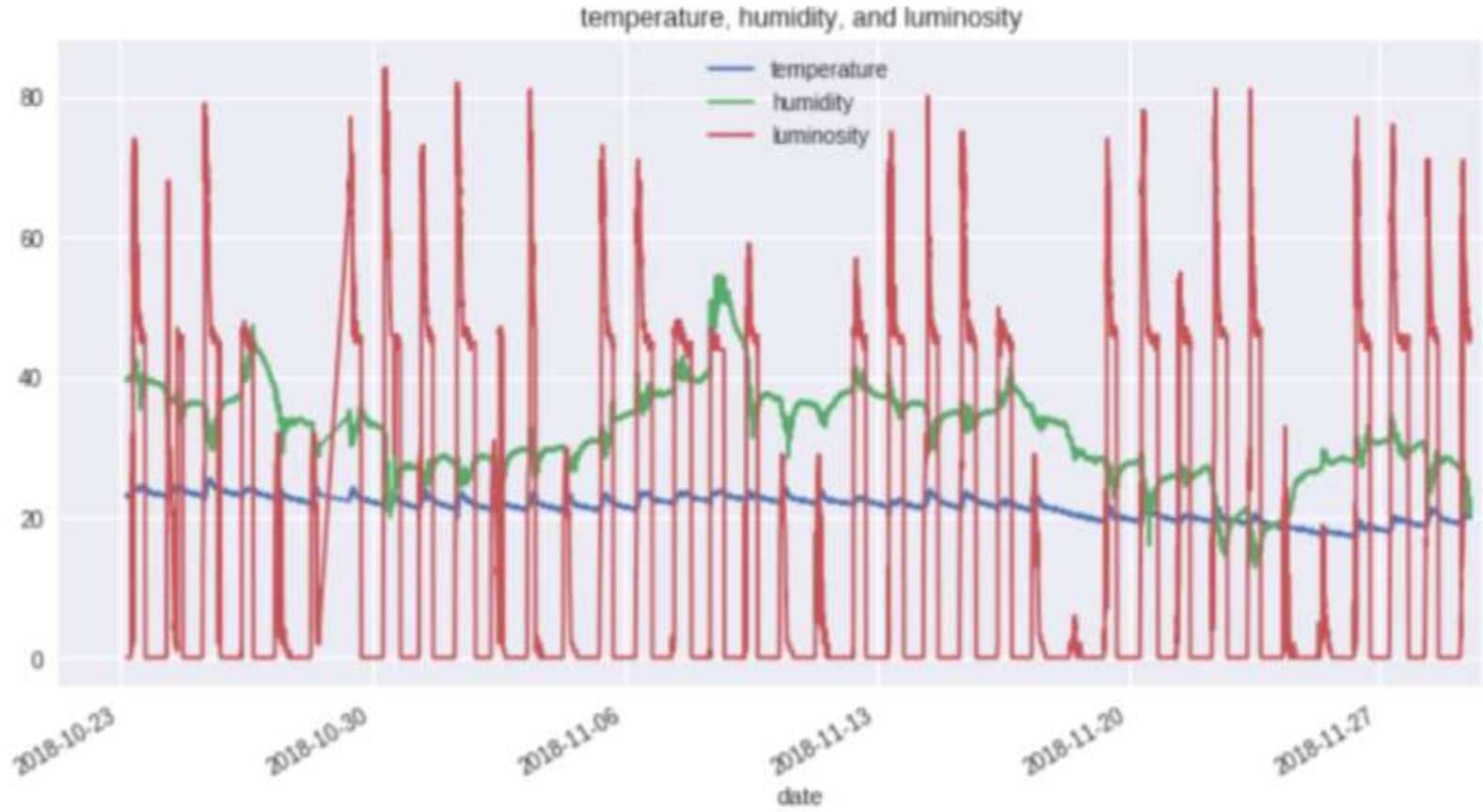
Dataframe에서 시간과 온도 데이터를 선택해서 그래프를 그린다.

3.4.2 Plot iot data (time series)

```
# Plot of ['temperature','humidity','luminosity']
iot_data.plot(x='date', y=['temperature','humidity','luminosity'], figsize=(12,6),
              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>
```

Dataframe에서 시간과 세 개의 센서 데이터를 전부 선택해서 그래프를 그린다.

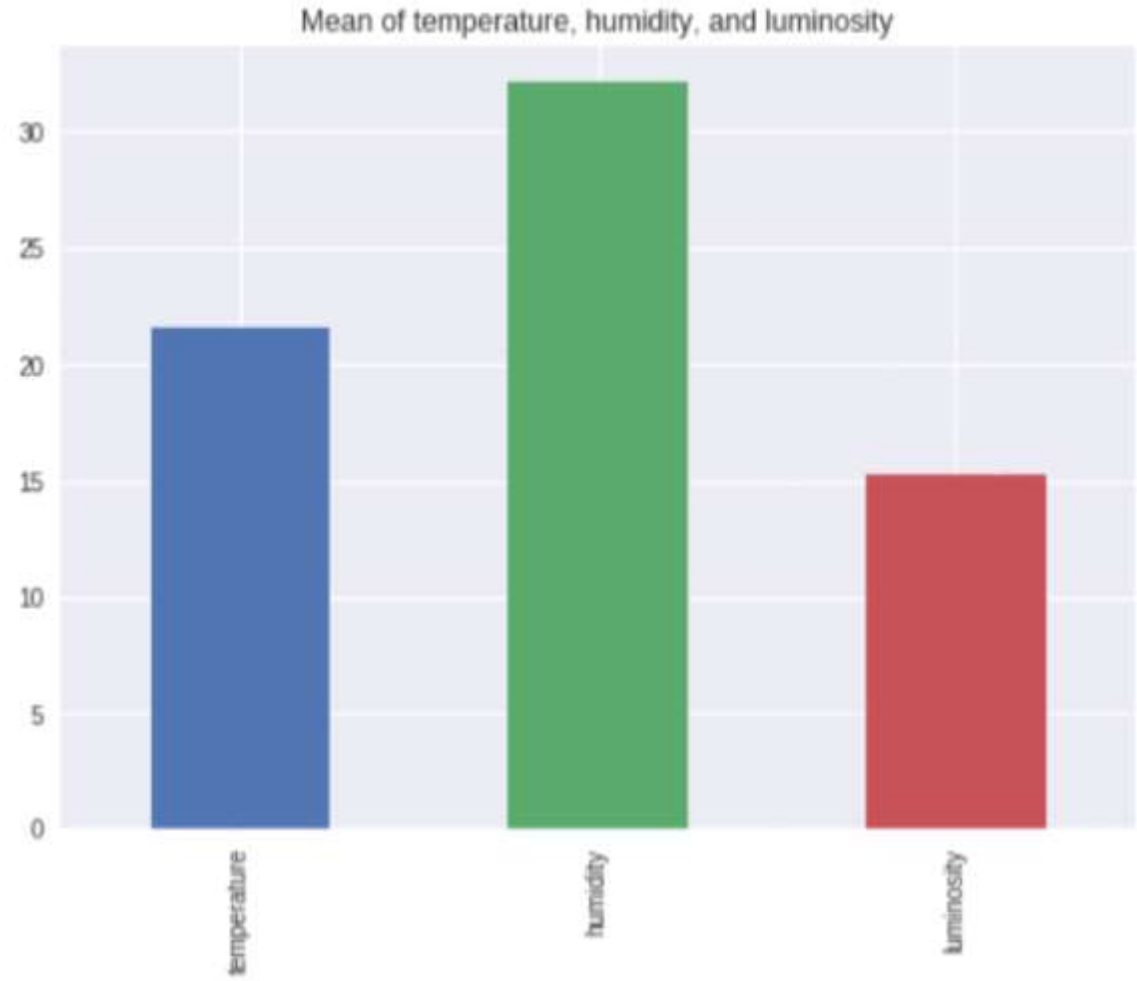


3.5 Plot mean of sensor data

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

Dataframe에서 세 개의 센서 데이터의 평균을 구해서 그래프를 그린다.

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



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

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

```
[ ]  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
```

```
iot_data.head()
```

date	temperature	humidity	luminosity
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

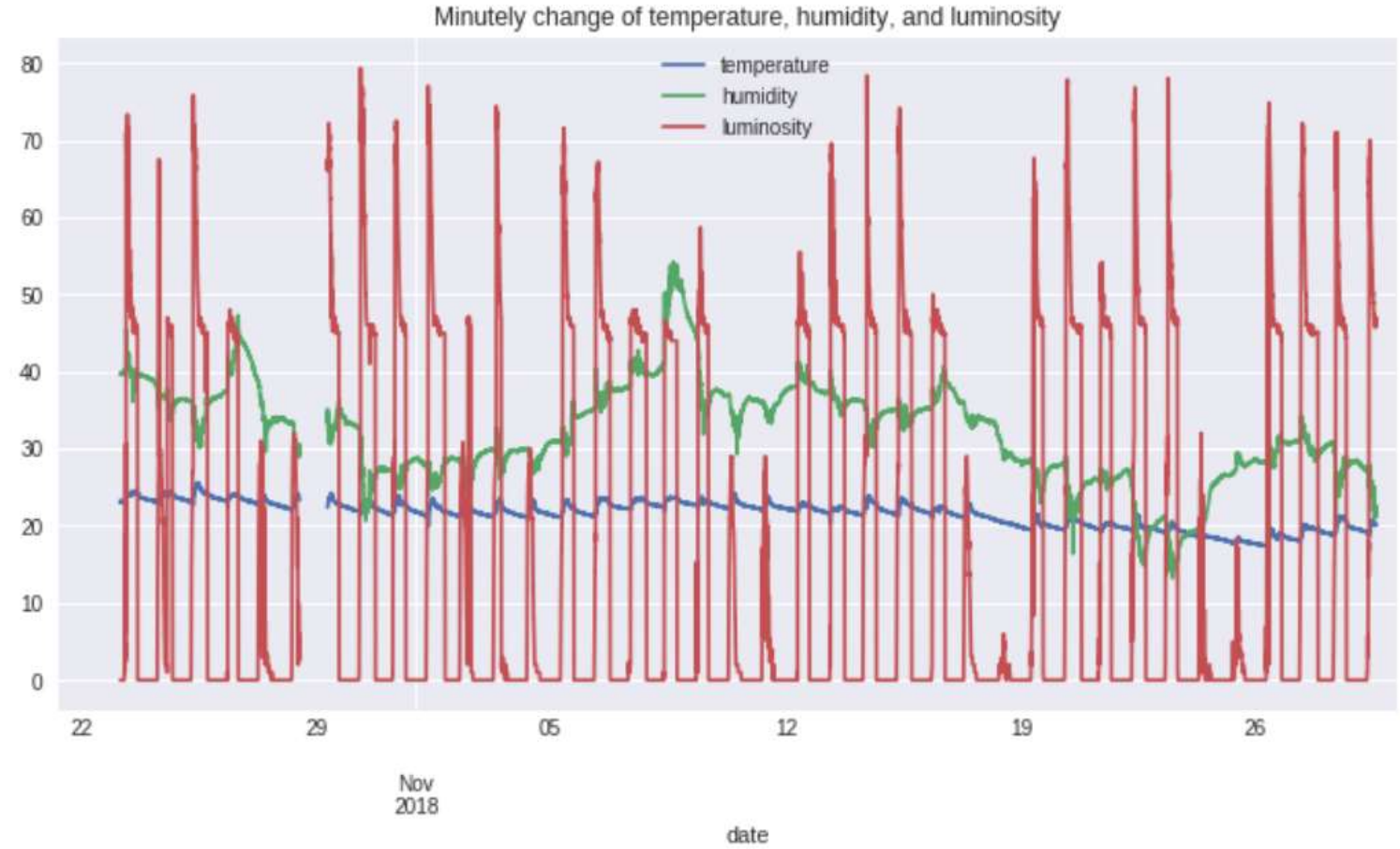
시간(date)을 **timestamp** 형태의 **Index**로 변경해서 데이터를 재구성한다.

A5.9.8 IOT data mining

3.6.2 Plot the change of sensor data over various time spans.
1 분당 평균 그래프

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

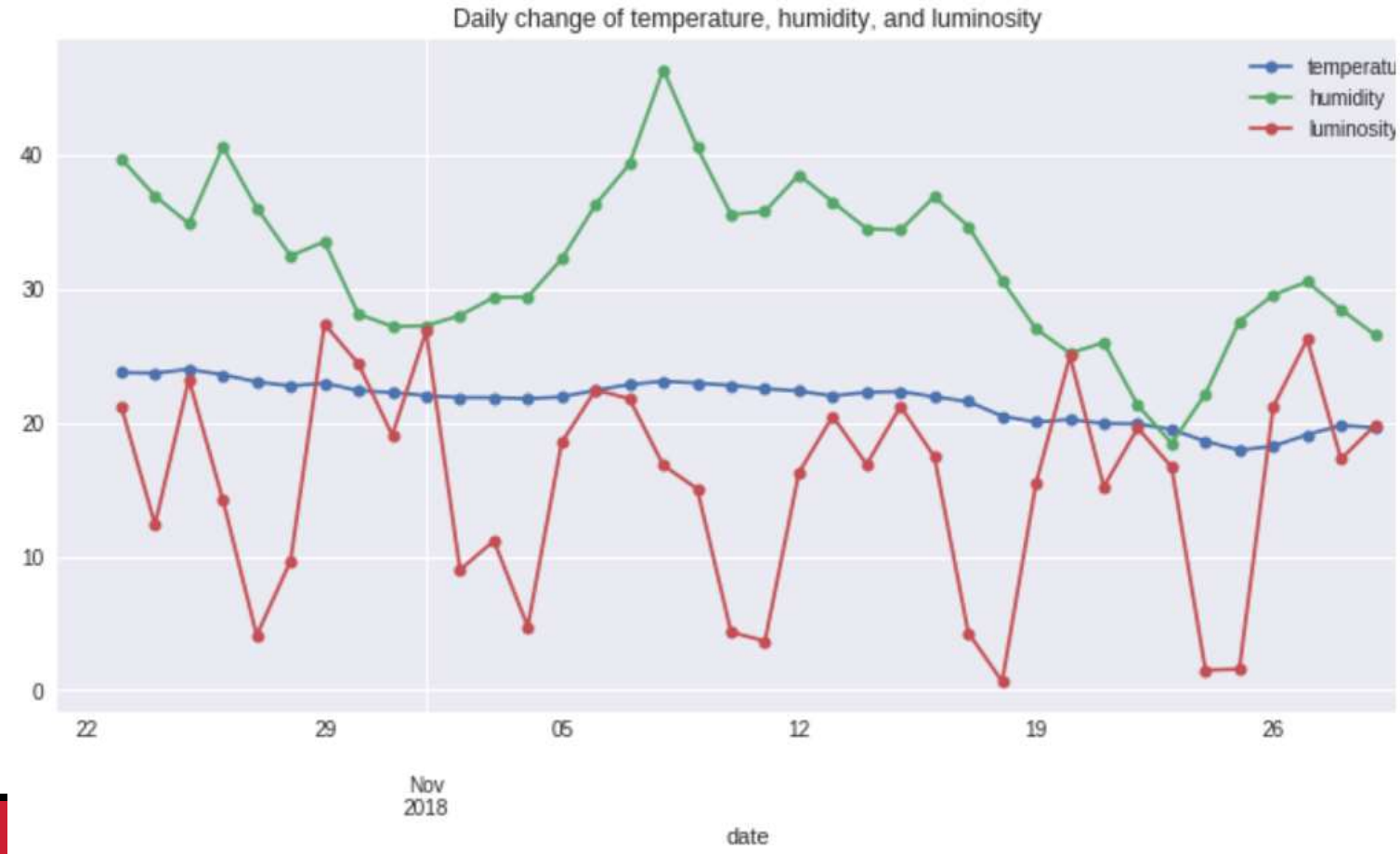
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5b2b57c630>
```



3.6.3 Plot the change of sensor data over various time spans.
1 일당 평균 그래프

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

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

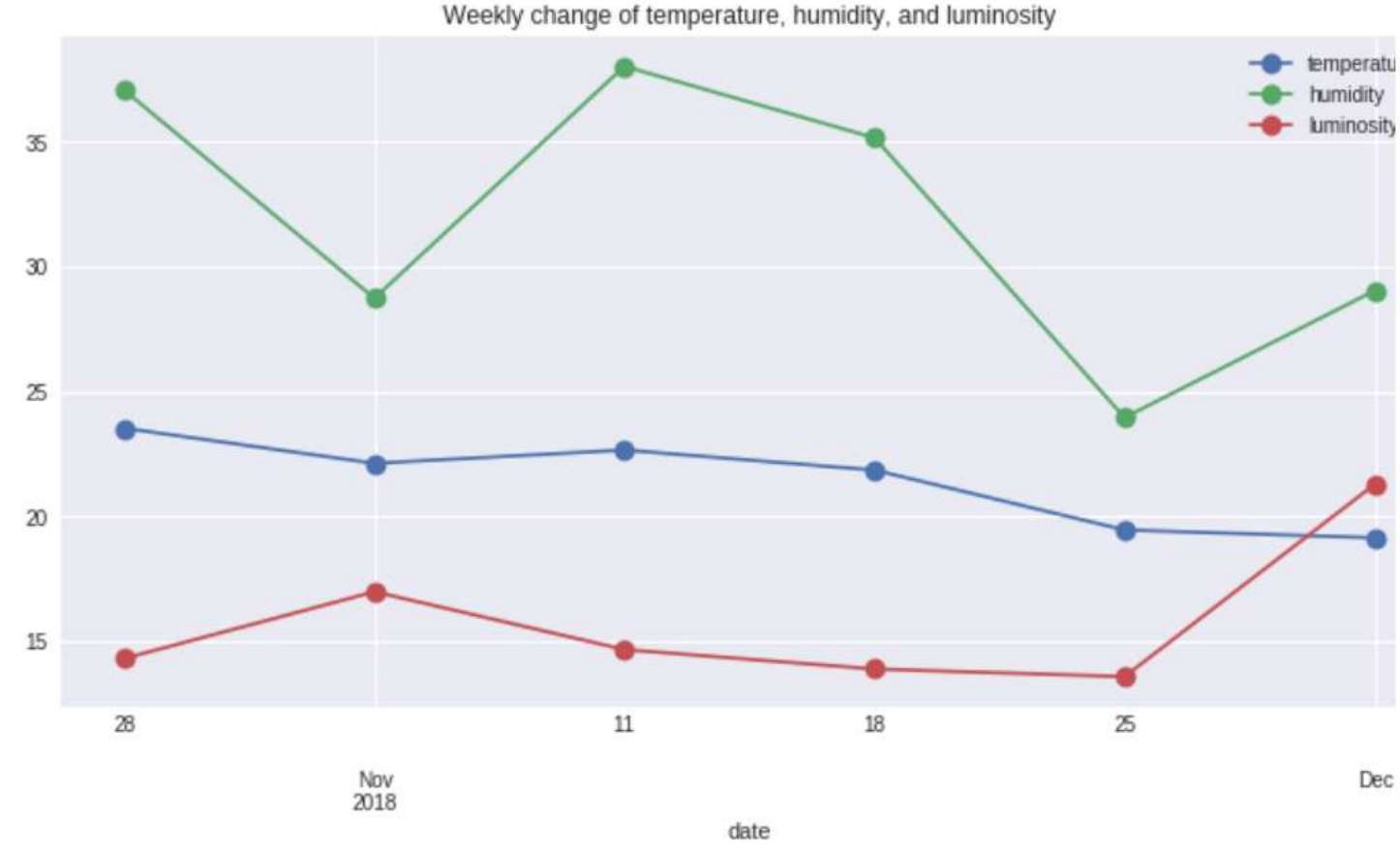


A5.9.8 IOT data mining

3.6.3 Plot the change of sensor data over various time spans.
1 주당 평균 그래프

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

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

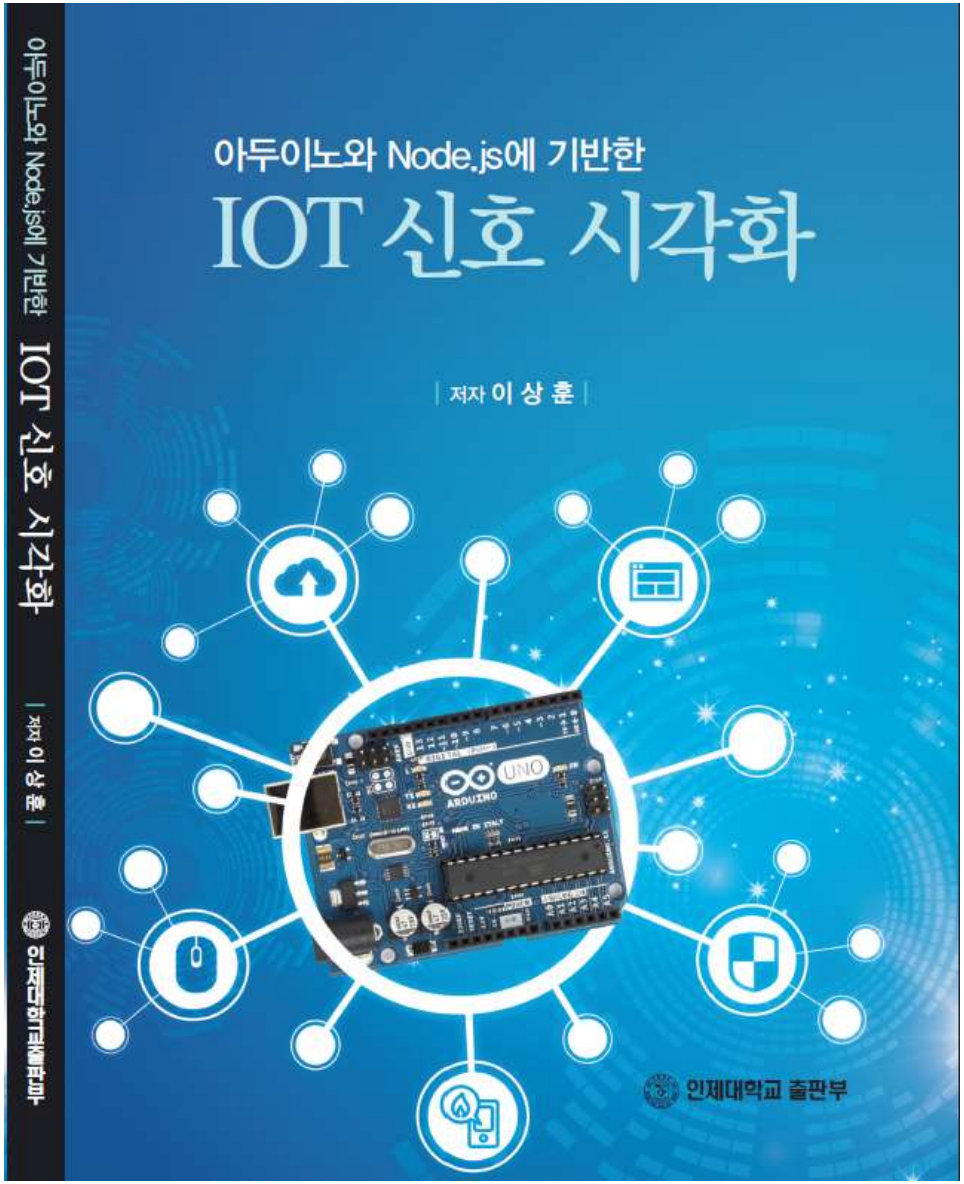


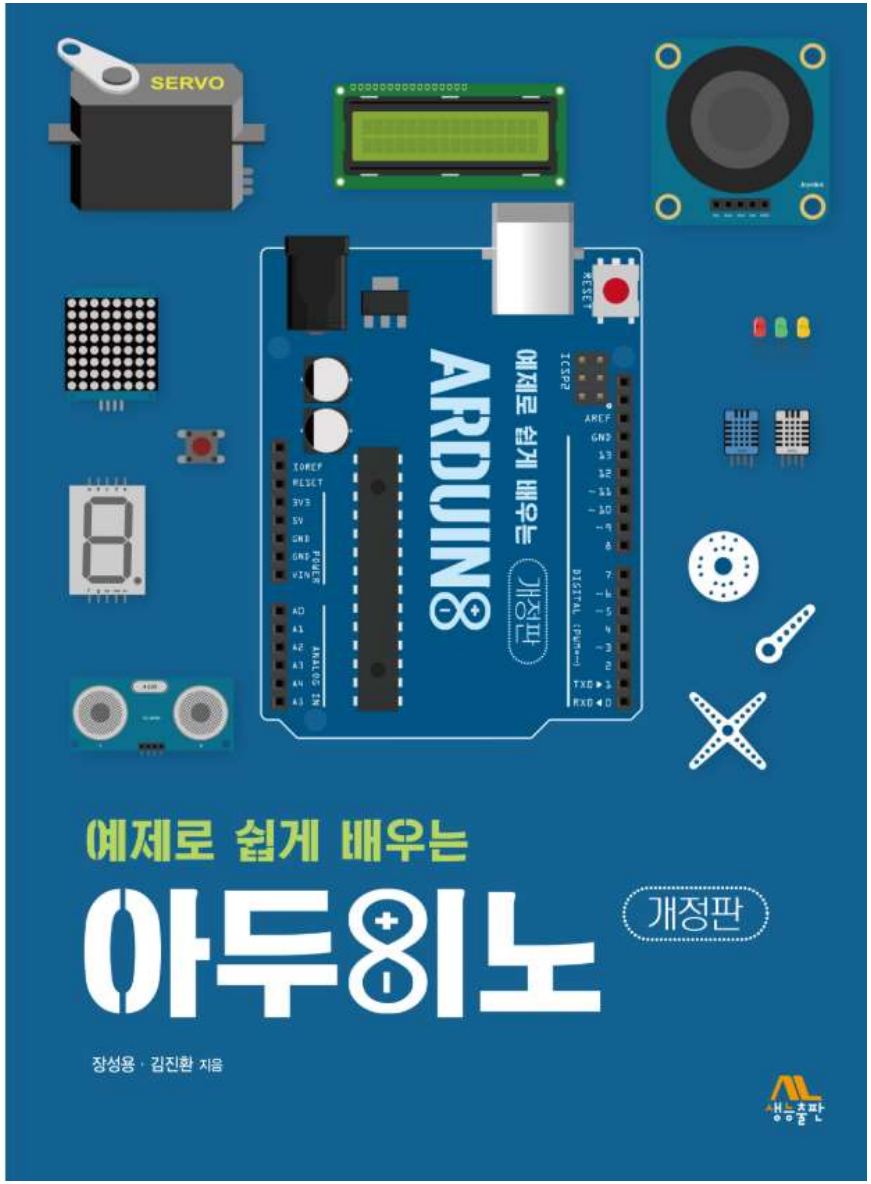
Lecture materials

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

주교재 및 참고도서


아두이노와 Node.js에 기반한
IOT 신호 시각화
| 저자 이 상 훈 |
인제대학교 출판부
아두이노와 Node.js에 기반한 IOT 신호 시각화
| 저자 이 상 훈 |
인제대학교출판부


SERVO
ARDUINO
예제로 쉽게 배우는
개정판
예제로 쉽게 배우는
아두이노
개정판
장성용 · 김진환 지음

Target of this class

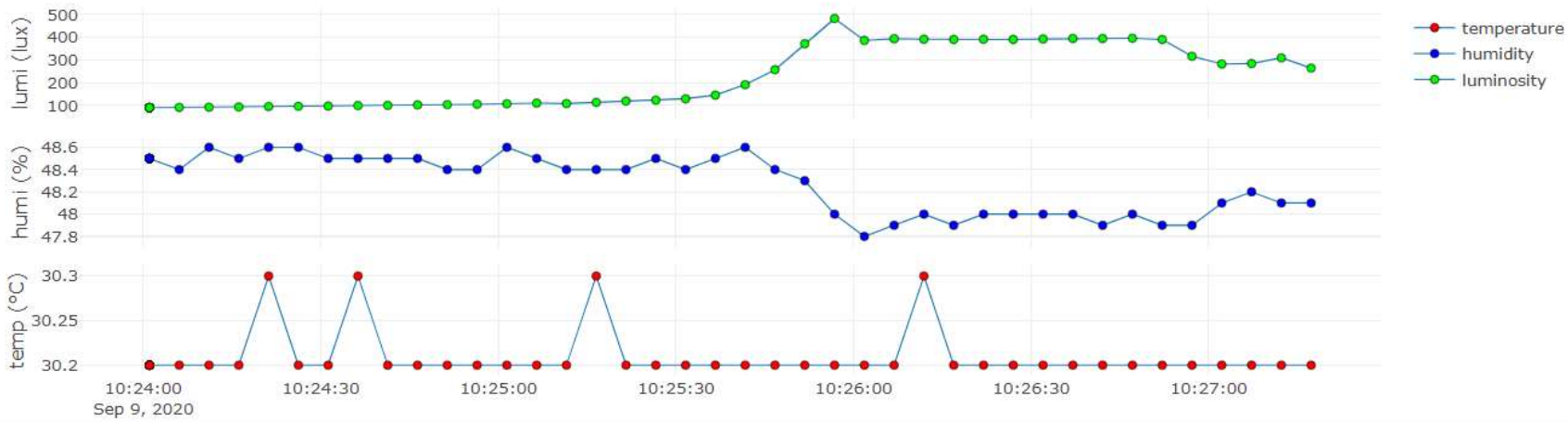
Real-time Weather Station from sensors

on Time: 2018-01-22 17:58:31.012

Target of this class

Real-time Weather Station from nano 33 BLE sensors

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



Another target of this class