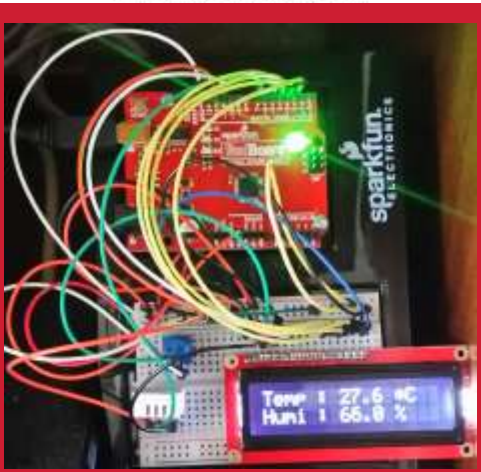
Arduino-IOT

[wk12]

cds_dht22 + node

MongoDB-II

on Time: 2015-09-02 12:48:14.192



Visualization of Signals using Arduino,

Node.js & storing signals in MongoDB

& mining data using Python

Drone-IoT-Comsi, INJE University

2nd semester, 2023

Email : chaos21c@gmail.com

My ID

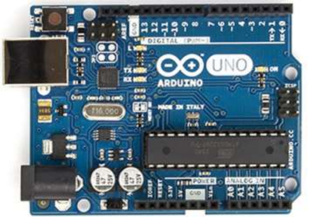
ID를 확인하고 github에 repo 만들기

ID	성명
AA01	강동하
AA02	고서진
AA03	김민재
AA04	김예원
AA05	김주호
AA06	김창욱
AA07	김현서
AA08	박종혁
AA09	서명진
AA10	유동기
AA11	
AA12	이근보
AA13	정호기

위의 id를 이용해서 github에 repo를 만드시오.

Option: 아두이노응용 실습 과제 – AAnn

Public, README.md check



[Practice]

- ◆ [wk11: mongoDB test]

- ➢ Insert documents to test mongodb & cds-dht22-node-mongodb
- ➢ Upload folder: aann-rpt11
- ➢ Use repo “aann” in github

wk11 : Practice : aann-rpt11

[Target of this week]

- Complete your works : **mongoDB test.**
- **Save your outcomes and upload outputs in github**

제출폴더명 : **aann-rpt11** 에 아래 파일을 추가

- 제출할 파일들

① **AAnn_mongo_schemas.png**

② **AAnn_mongo_update.png**

③ **dbtest.js (cds_dht22 folder)**

④ **dbtest2.js (cds_dht22 folder)**

⑤ **AAnn_iot_mongodb.png**

Email : chaos21c@gmail.com

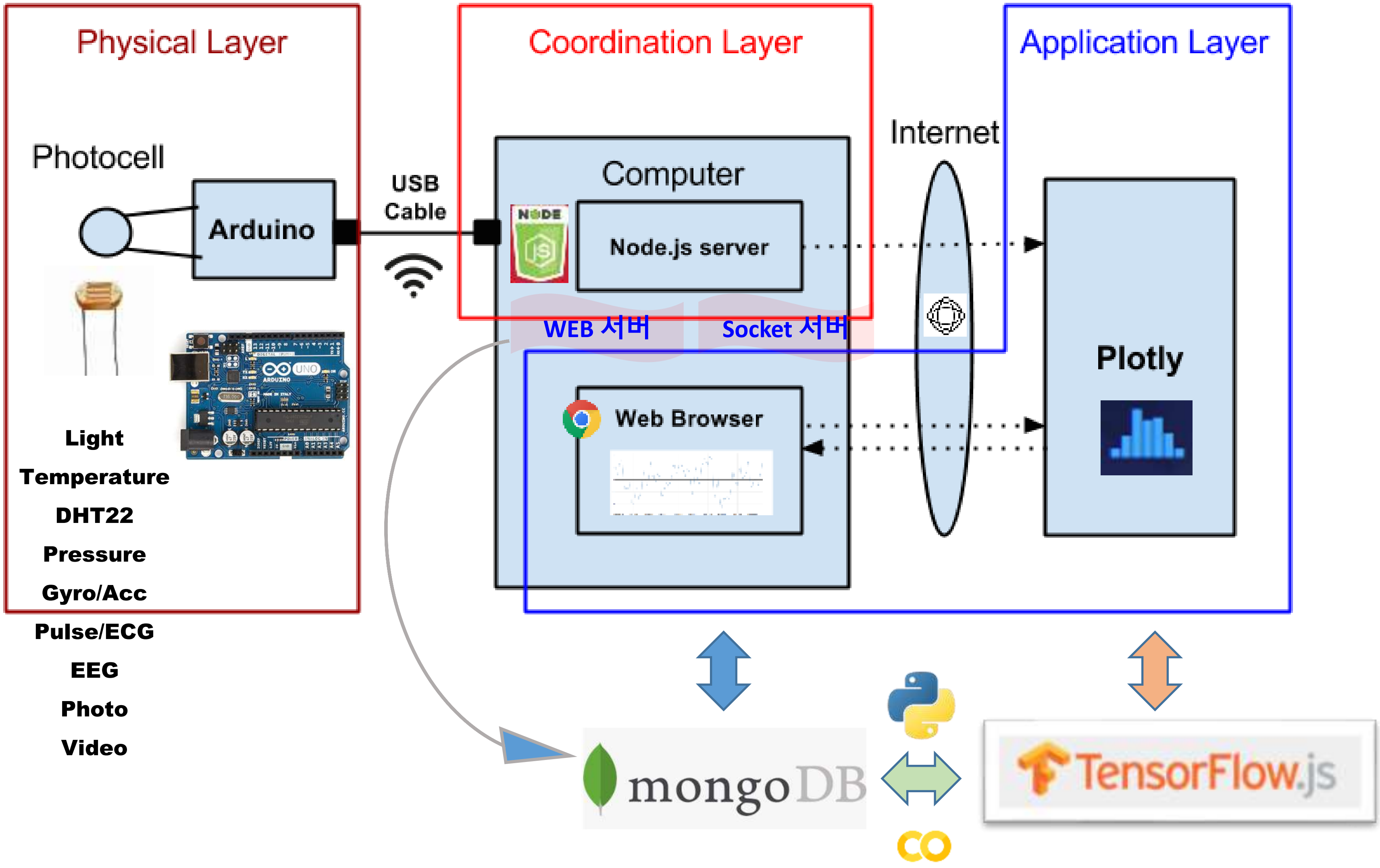
Purpose of AA

주요 수업 목표는 다음과 같다.

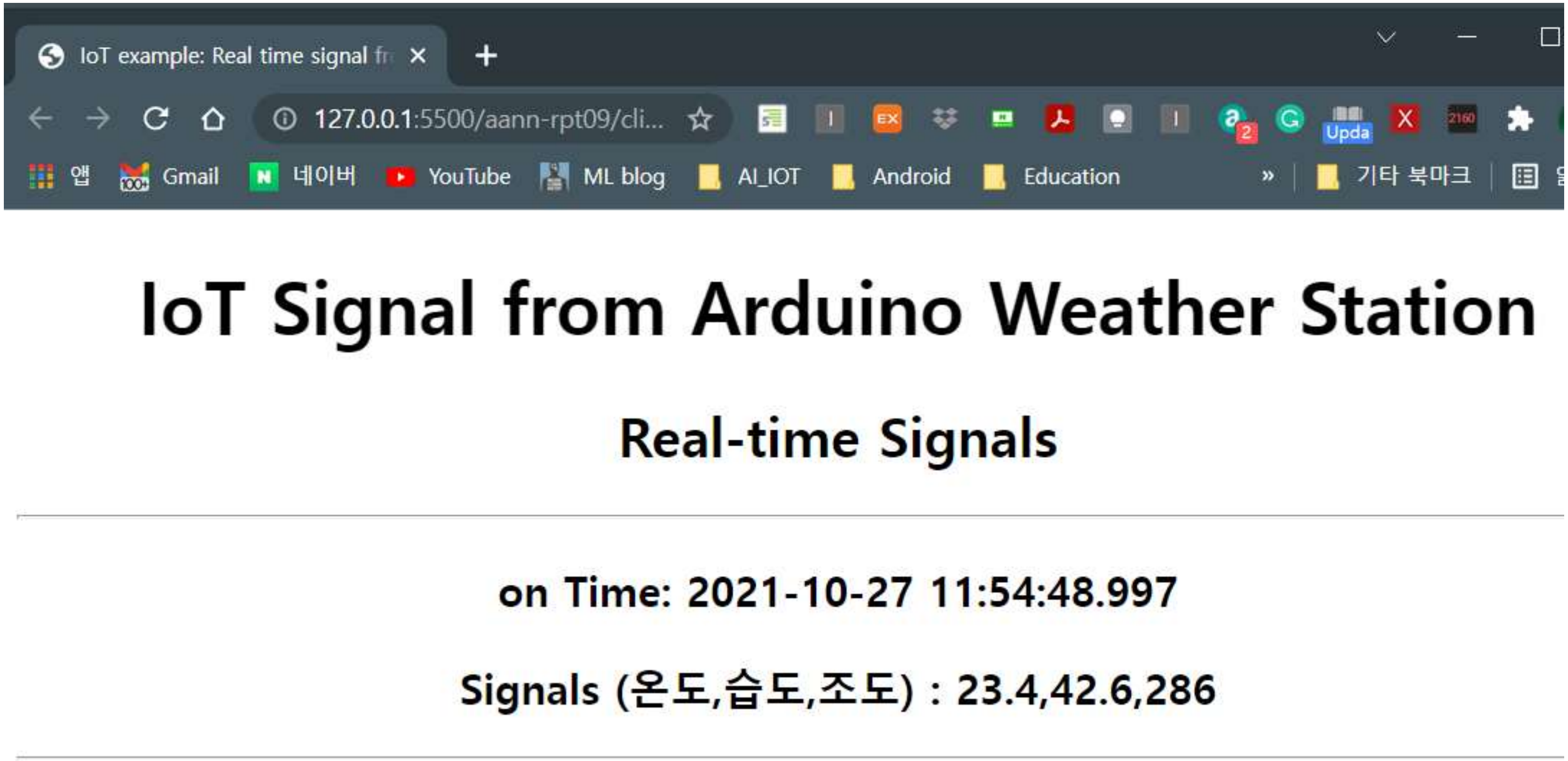
1. Node.js를 이용한 아두이노 센서 신호 처리
2. Plotly.js를 이용한 아두이노 센서 신호 시각화
3. MongoDB에 아두이노 센서 데이터 저장 및 처리

4. 저장된 IoT 데이터의 마이닝 (파이썬 코딩)

Layout [H S C]


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

on WEB monitoring Arduino data

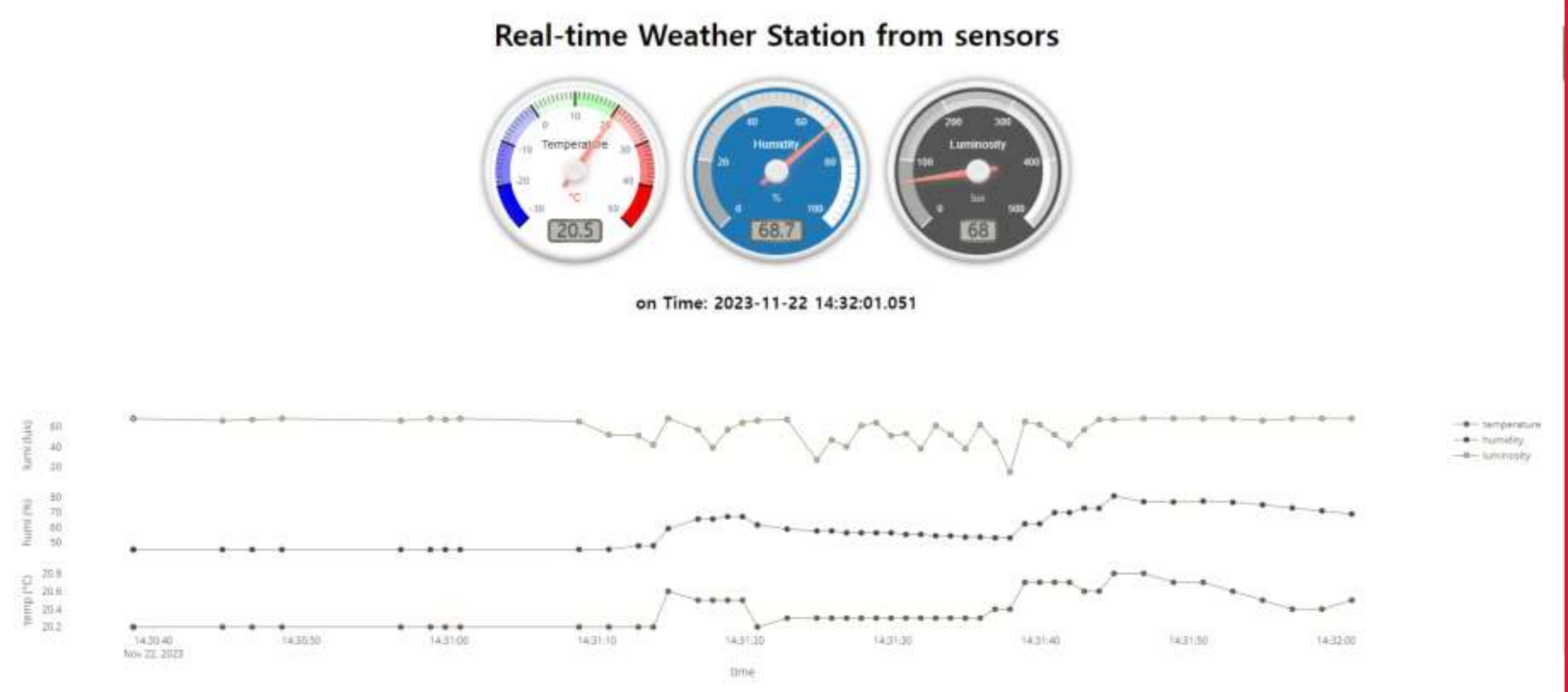

IoT example: Real time signal fr
127.0.0.1:5500/aann-rpt09/cli...
앱
Gmail
네이버
YouTube
ML blog
AI_IOT
Android
Education
기타 북마크
Upda

IoT Signal from Arduino Weather Station

Real-time Signals

on Time: 2021-10-27 11:54:48.997

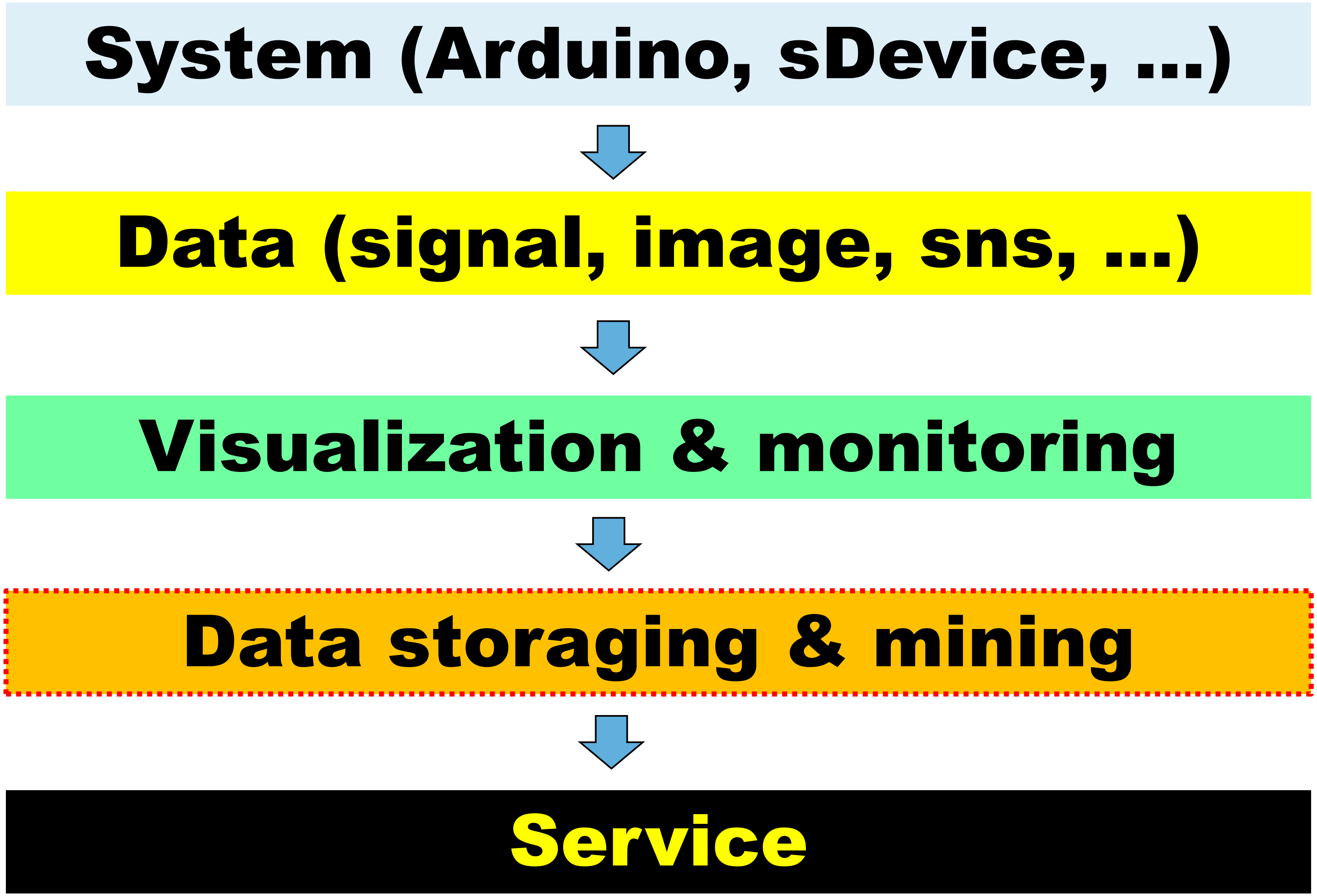
Signals (온도,습도,조도) : 23.4,42.6,286



Real-time Weather Station from sensors

on Time: 2023-11-22 14:32:01.051

A5. Introduction to IoT service



System (Arduino, sDevice, ...)

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

Visualization & monitoring

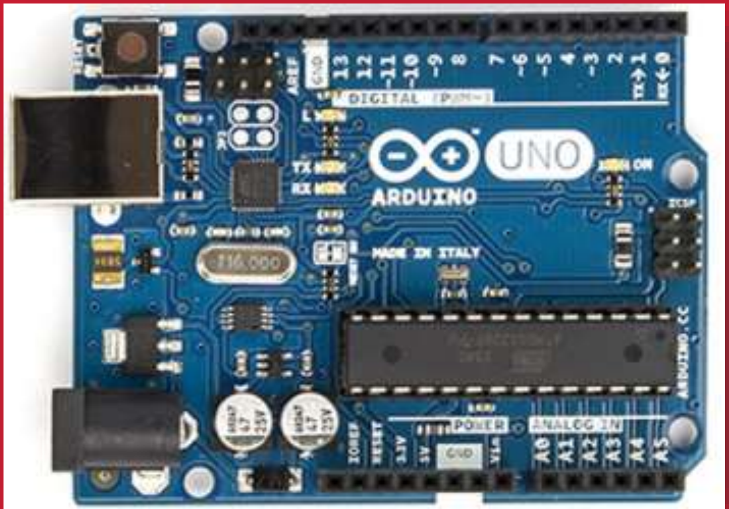
Data storaging & mining

Service

Arduino

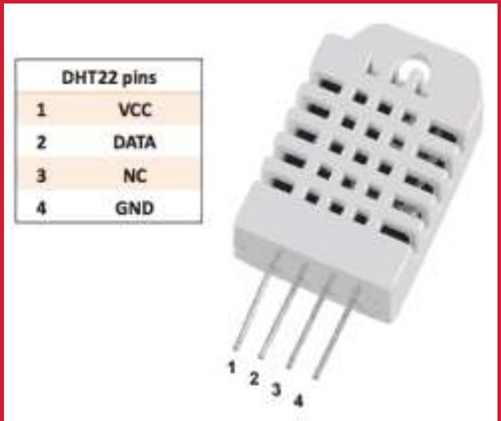
& Node.js

& MongoDB



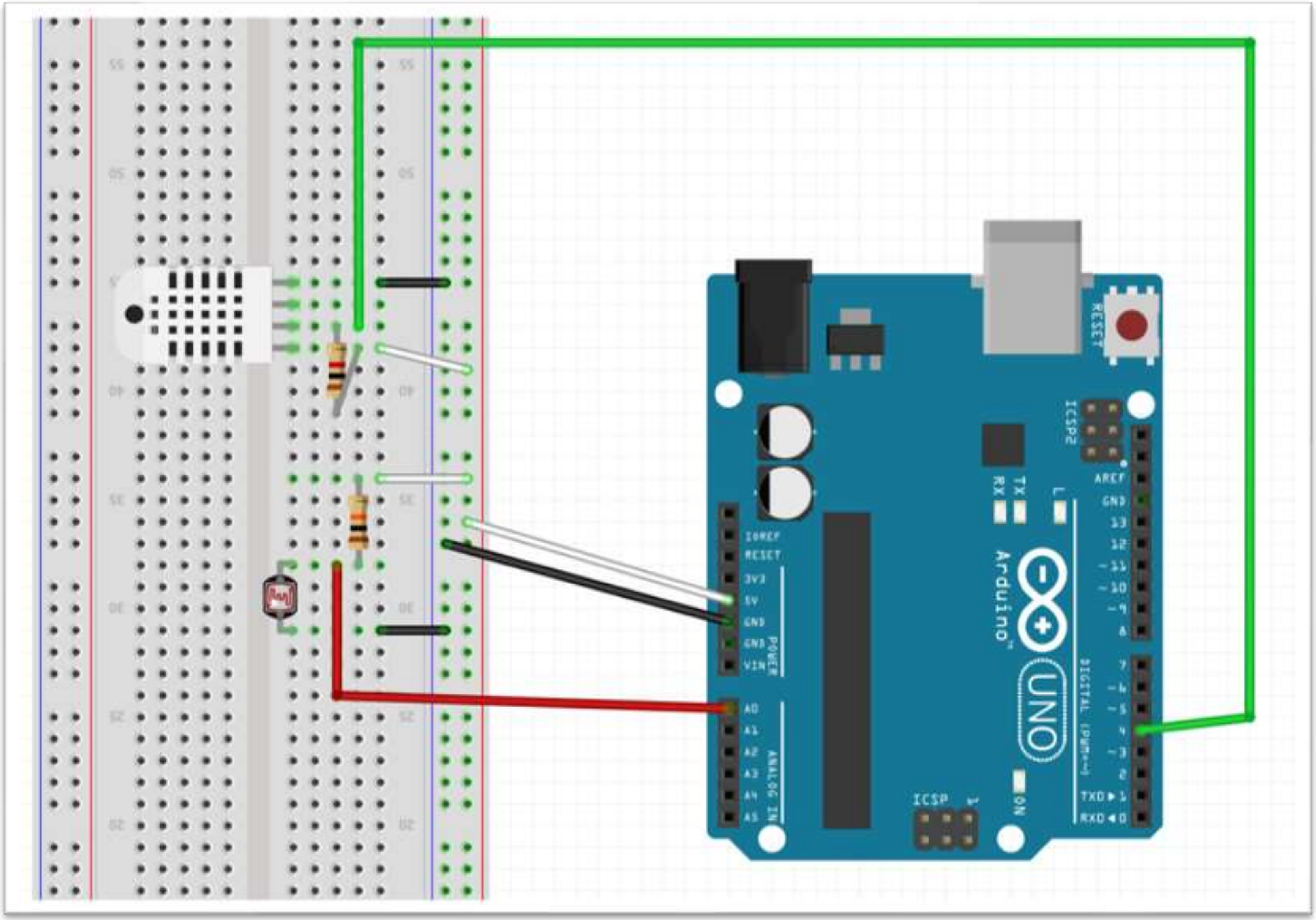
Multi-sensors

DHT22 + CdS



DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND

DHT22 + CdS circuit

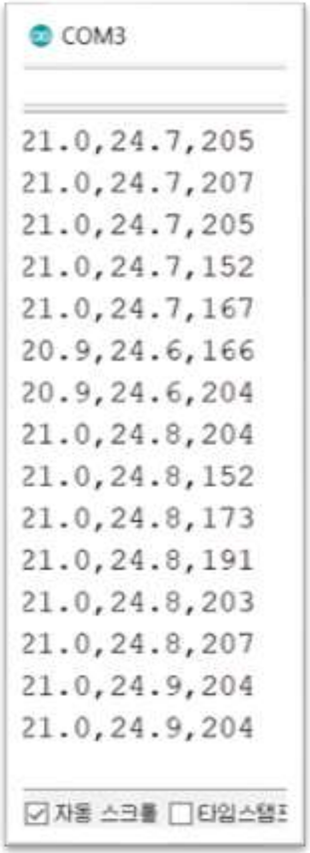


DHT22[D4] + 1 kΩ,   CdS[A0] + 10 k Ω

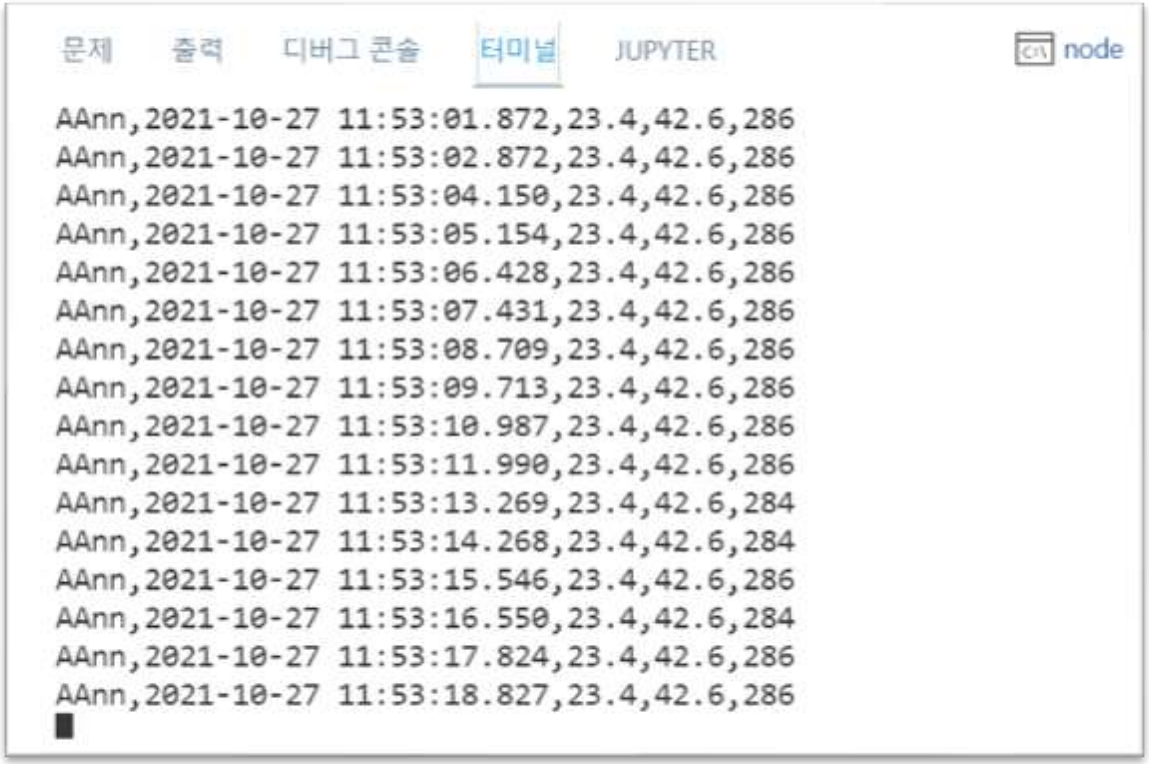
DHT22 + CdS + Node.js

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

node cds_dht22_node.js

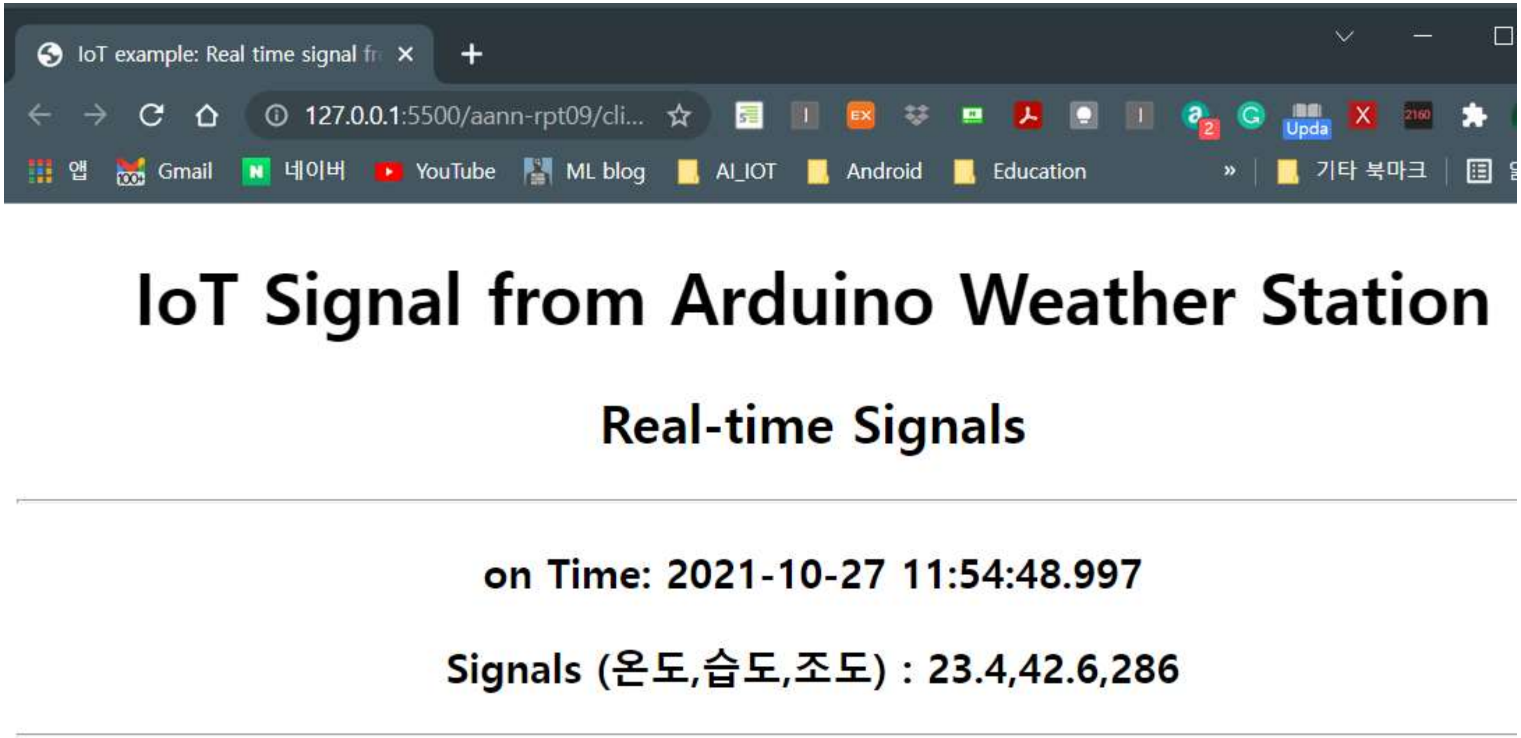


```
21.0,24.7,205
21.0,24.7,207
21.0,24.7,205
21.0,24.7,152
21.0,24.7,167
20.9,24.6,166
20.9,24.6,204
21.0,24.8,204
21.0,24.8,152
21.0,24.8,173
21.0,24.8,191
21.0,24.8,203
21.0,24.8,207
21.0,24.9,204
21.0,24.9,204
```



```
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
█
```

Arduino data on network socket



IoT Signal from Arduino Weather Station

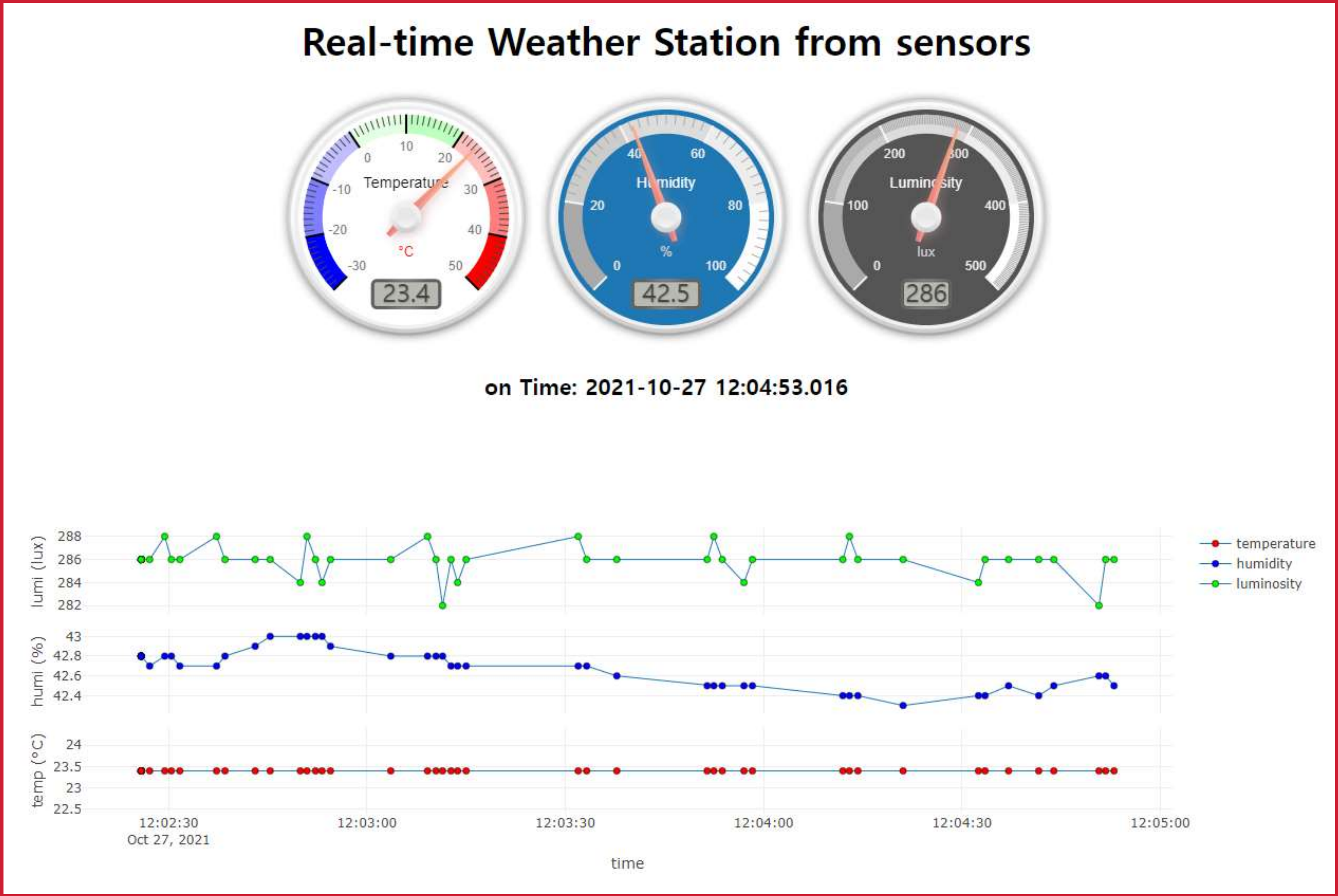
Real-time Signals

on Time: 2021-10-27 11:54:48.997

Signals (온도,습도,조도) : 23.4,42.6,286

Real-time monitoring of signals from Arduino CdS + DHT22 circuit

WEB client： client_cds_dht22.html

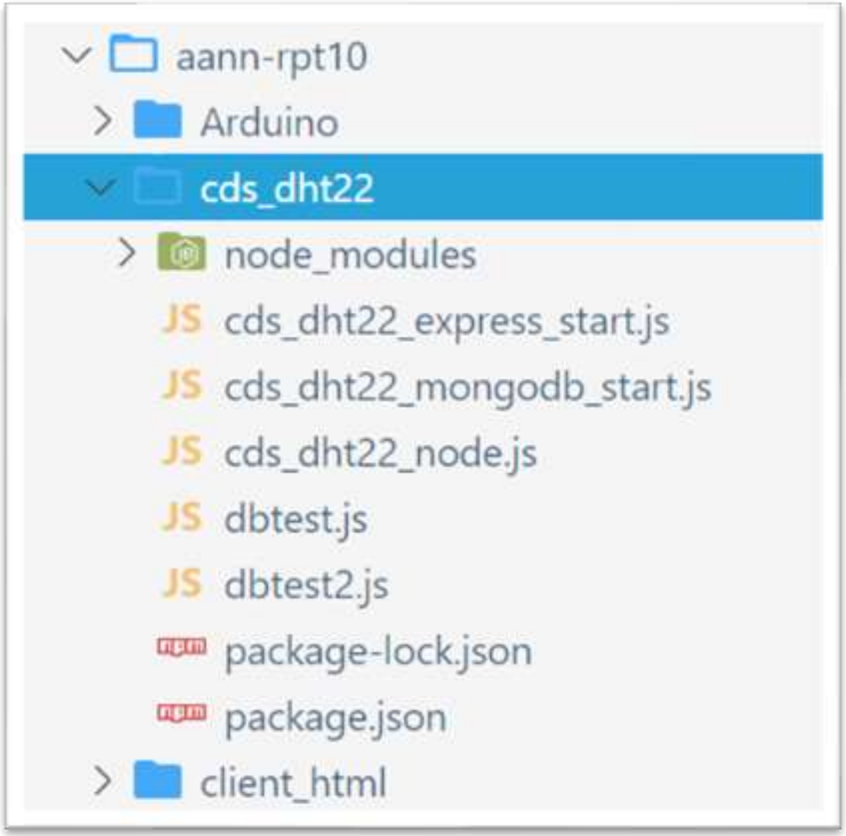


Real-time Weather Station from sensors

on Time: 2021-10-27 12:04:53.016

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

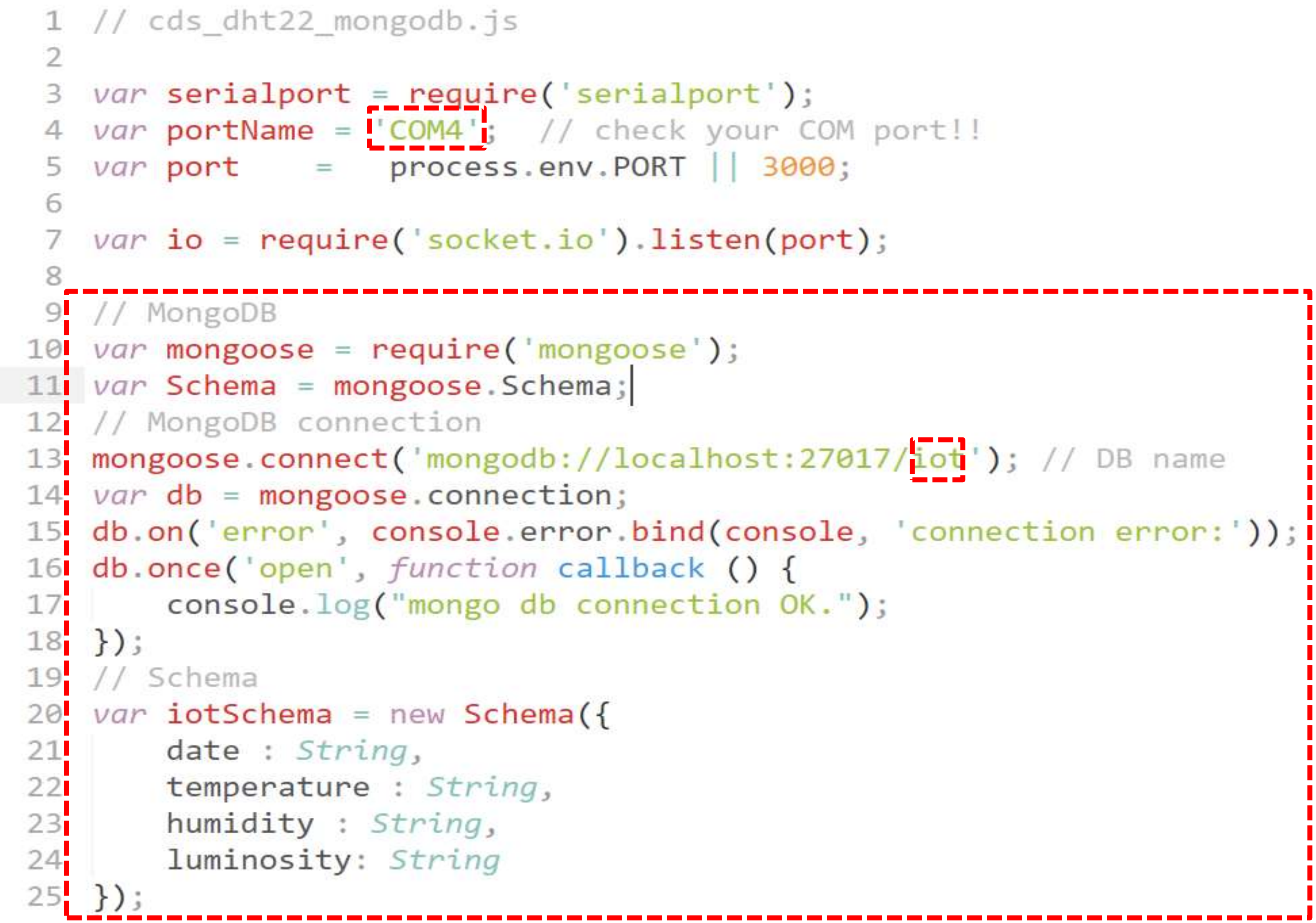
1. 작업 폴더 구조



```
aann-rpt10
  Arduino
  cds_dht22
    node_modules
    cds_dht22_express_start.js
    cds_dht22_mongodb_start.js
    cds_dht22_node.js
    dbtest.js
    dbtest2.js
    package-lock.json
    package.json
  client_html
```

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

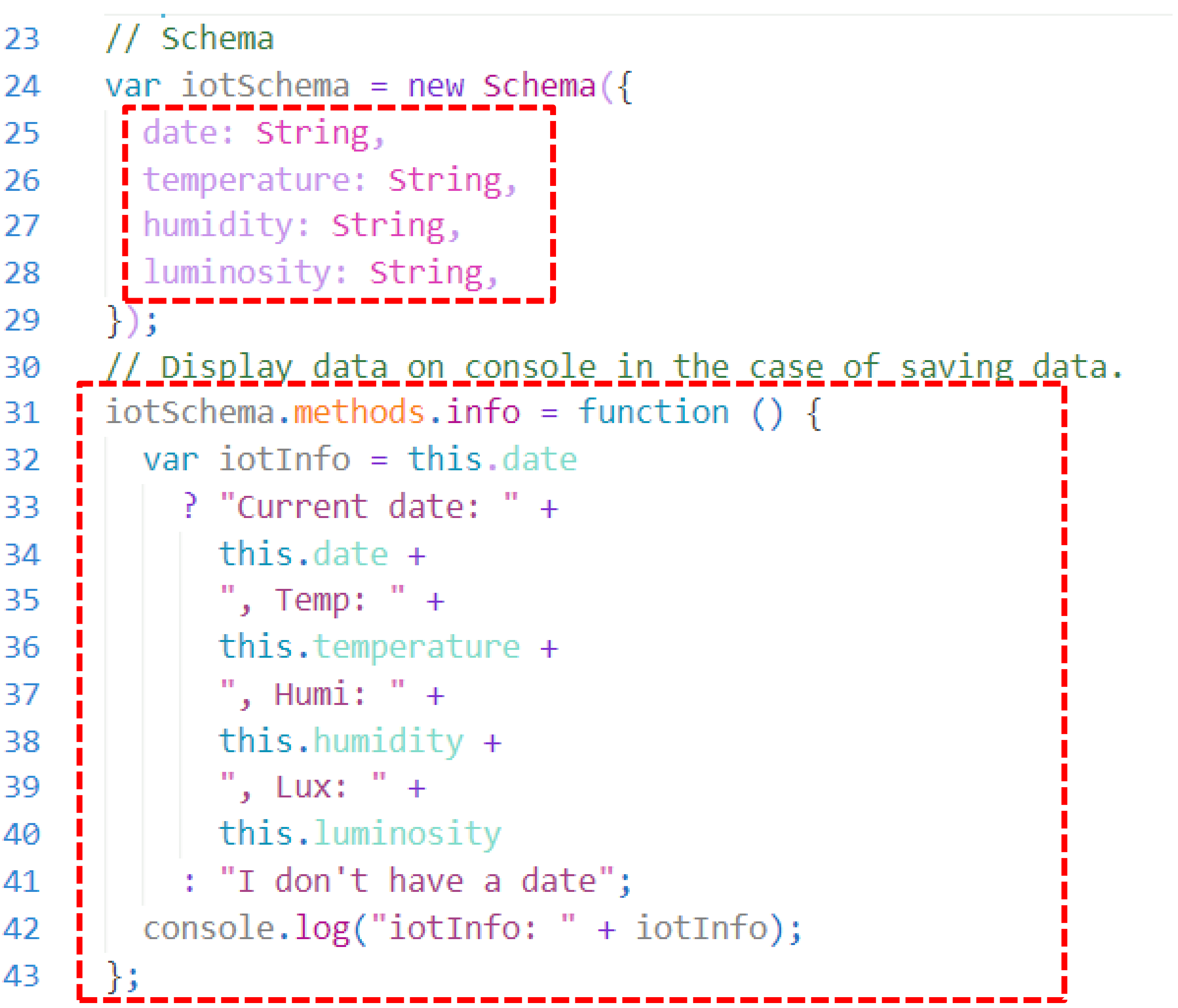
2.1 cds_dht22_mongodb.js



```
// cds_dht22_mongodb.js

var serialport = require('serialport');
var portName = 'COM4';   // 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'); // DB name
var db = mongoose.connection;
db.on('error', console.error.bind(console, 'connection error:'));
db.once('open', function callback () {
    console.log("mongo db connection OK.");
});
// Schema
var iotSchema = new Schema({
    date : String,
    temperature : String,
    humidity : String,
    luminosity: String
});
```

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

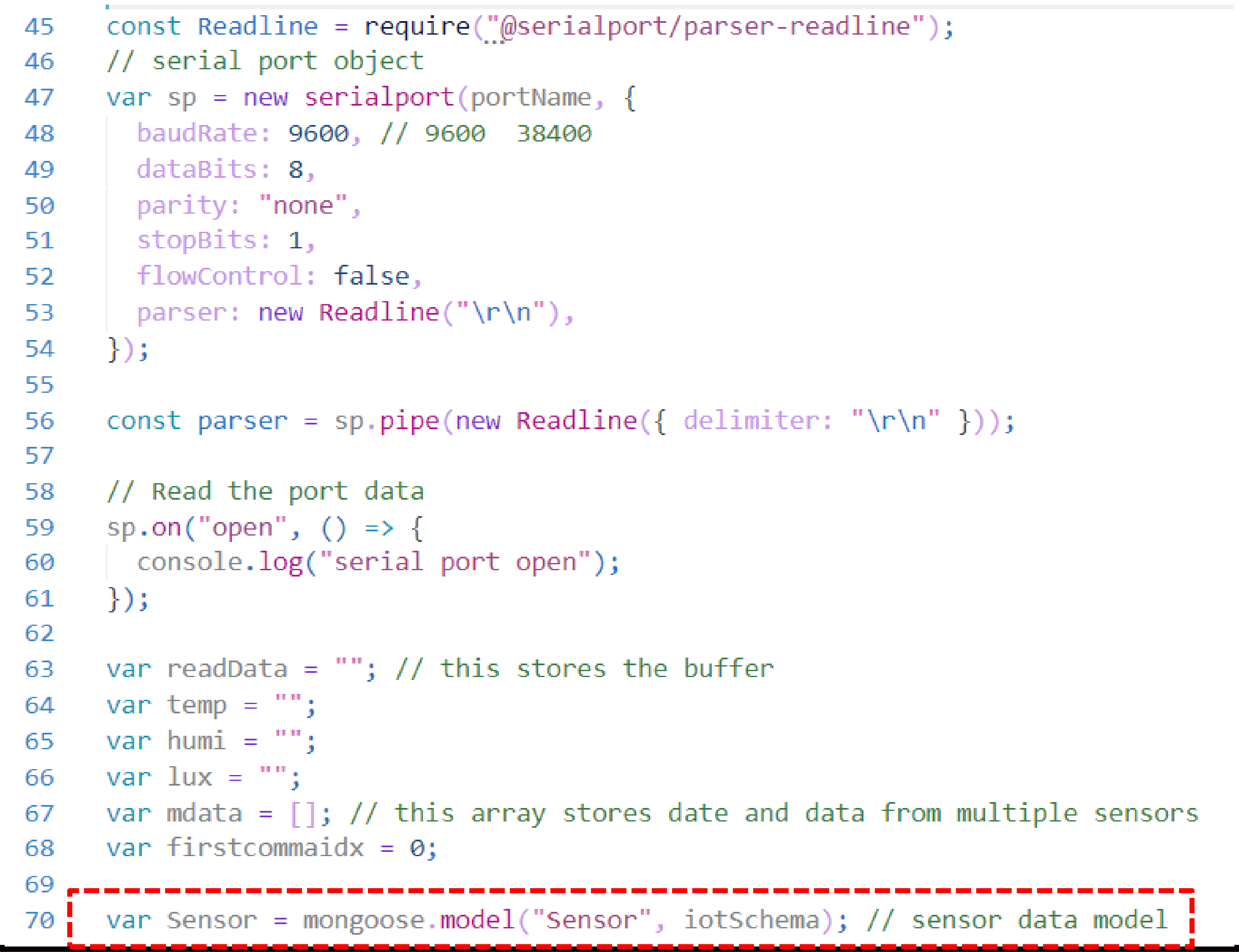
2.2 cds_dht22_mongodb.js



```
// Schema
var iotSchema = new Schema({
  date: String,
  temperature: String,
  humidity: String,
  luminosity: String,
});
// Display data on console in the case of saving data.
iotSchema.methods.info = function () {
  var iotInfo = this.date
    ? "Current date: " +
      this.date +
      ", Temp: " +
      this.temperature +
      ", Humi: " +
      this.humidity +
      ", Lux: " +
      this.luminosity
    : "I don't have a date";
  console.log("iotInfo: " + iotInfo);
};
```

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

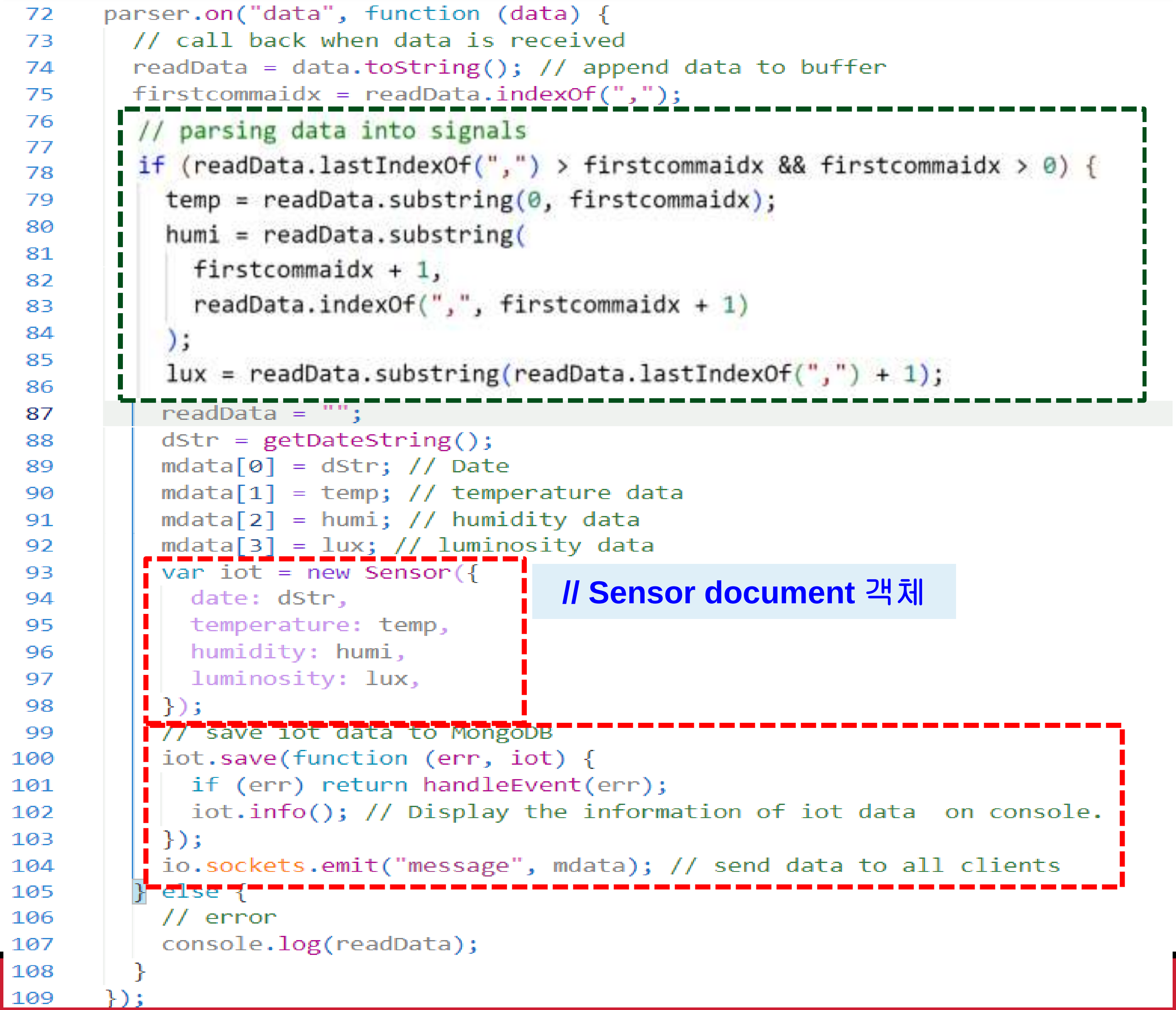
2.3 cds_dht22_mongodb.js



```
const Readline = require("@serialport/parser-readline");
// serial port object
var sp = new serialport(portName, {
  baudRate: 9600, // 9600  38400
  dataBits: 8,
  parity: "none",
  stopBits: 1,
  flowControl: false,
  parser: new Readline("\r\n"),
});

const parser = sp.pipe(new Readline({ delimiter: "\r\n" }));

// Read the port data
sp.on("open", () => {
  console.log("serial port open");
});

var readData = ""; // this stores the buffer
var temp = "";
var humi = "";
var lux = "";
var mdata = []; // this array stores date and data from multiple sensors
var firstcommaidx = 0;

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

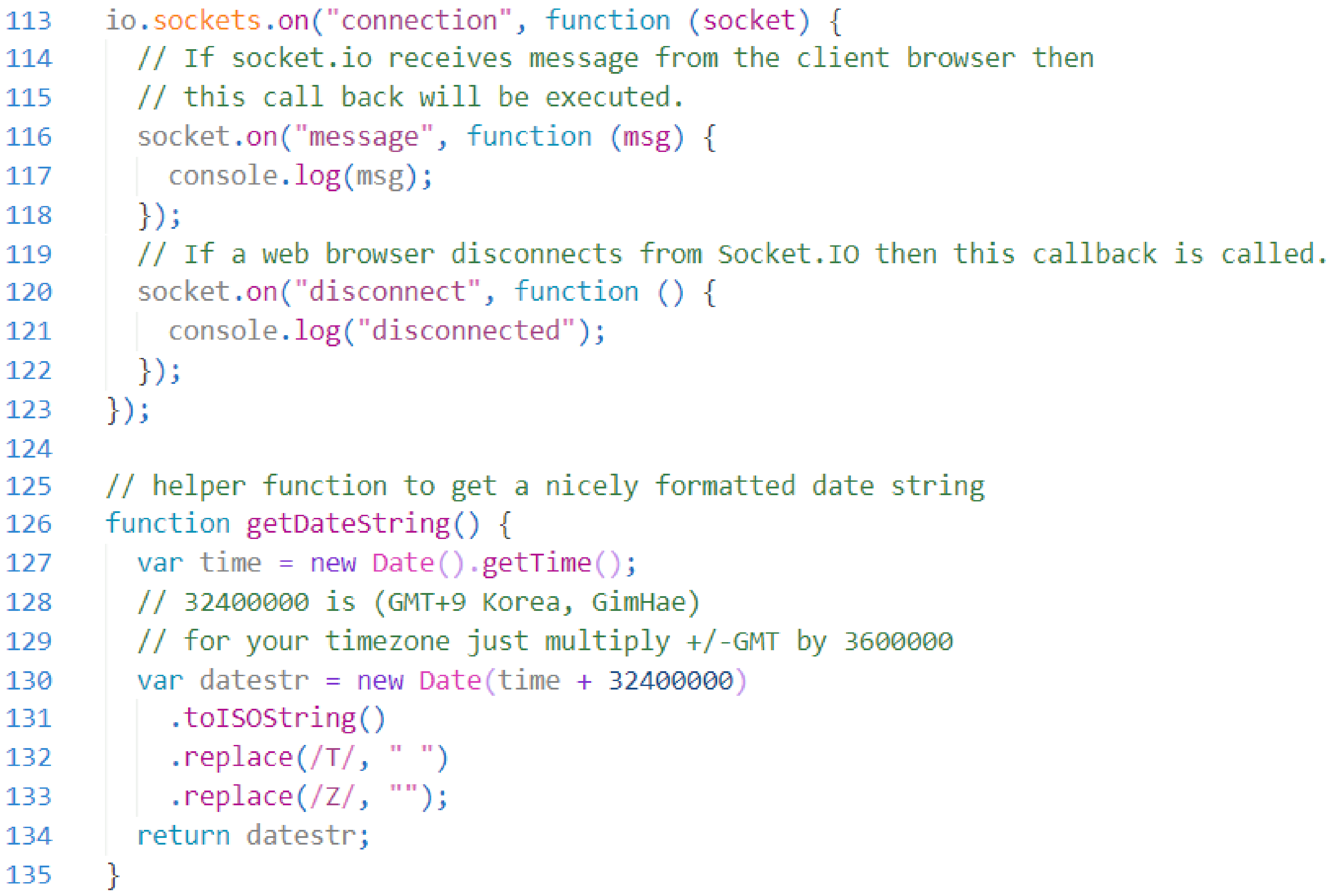


```
parser.on("data", function (data) {
  // call back when data is received
  readData = data.toString(); // append data to buffer
  firstcommaidx = readData.indexOf(",");
  // parsing data into signals
  if (readData.lastIndexOf(",") > firstcommaidx && firstcommaidx > 0) {
    temp = readData.substring(0, firstcommaidx);
    humi = readData.substring(
      firstcommaidx + 1,
      readData.indexOf(",", firstcommaidx + 1)
    );
    lux = readData.substring(readData.lastIndexOf(",") + 1);
    readData = "";
    dStr = getDateString();
    mdata[0] = dStr; // Date
    mdata[1] = temp; // temperature data
    mdata[2] = humi; // humidity data
    mdata[3] = lux; // luminosity data
    var iot = new Sensor({
      date: dStr,
      temperature: temp,
      humidity: humi,
      luminosity: lux,
    });
    // save iot data to MongoDB
    iot.save(function (err, iot) {
      if (err) return handleEvent(err);
      iot.info(); // Display the information of iot data  on console.
    });
    io.sockets.emit("message", mdata); // send data to all clients
  } else {
    // error
    console.log(readData);
  }
});
```

// Sensor document 객체

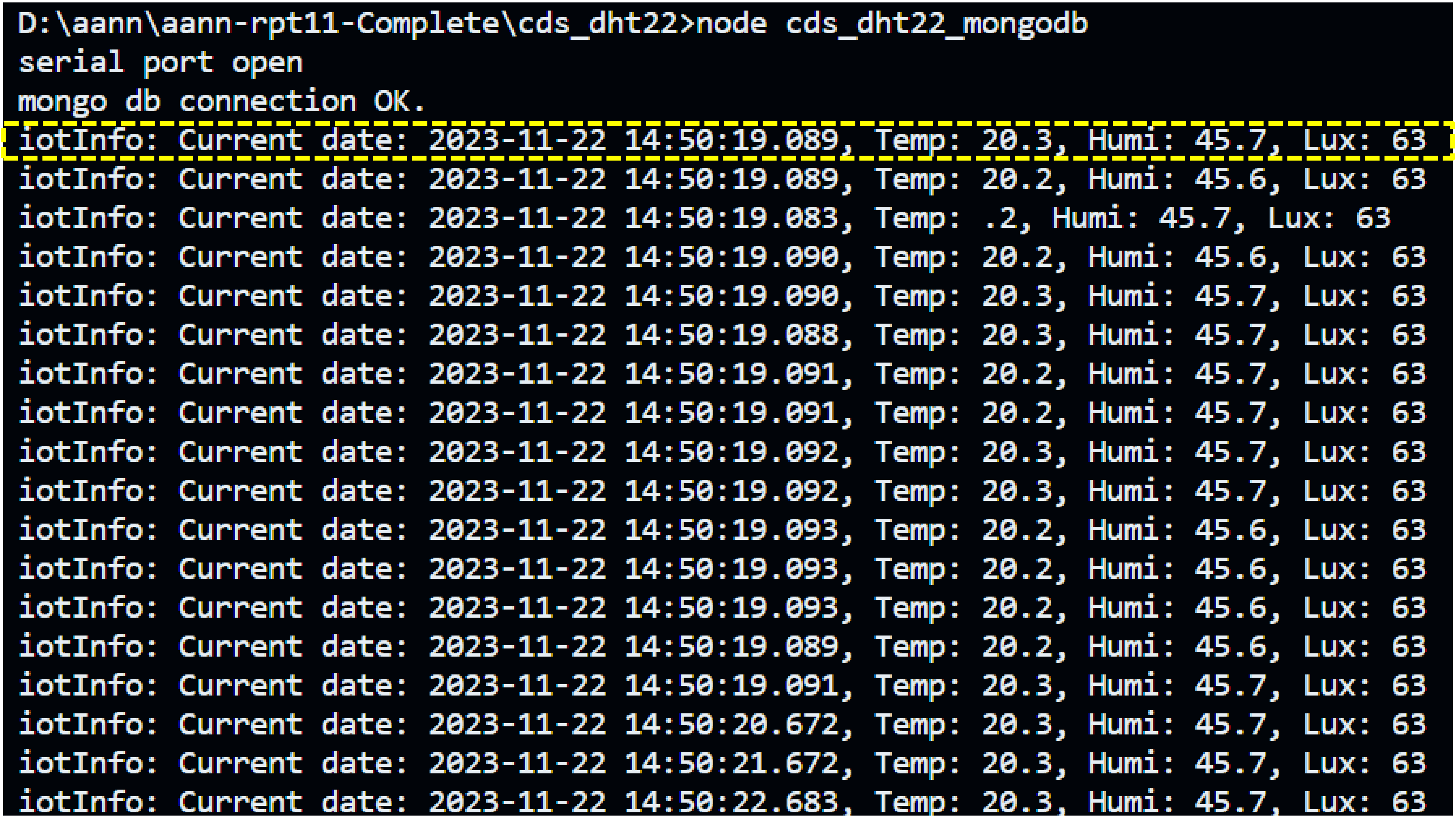
A5.9.5 DHT22 + CdS + Node.js + MongoDB

2.5 cds_dht22_mongodb.js



```
io.sockets.on("connection", function (socket) {
  // If socket.io receives message from the client browser then
  // this call back will be executed.
  socket.on("message", function (msg) {
    console.log(msg);
  });
  // If a web browser disconnects from Socket.IO then this callback is called.
  socket.on("disconnect", function () {
    console.log("disconnected");
  });
});

// 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;
}
```

2.6 [Run] node cds_dht22_mongodb.js



```
D:\aann\aann-rpt11-Complete\cds_dht22>node cds_dht22_mongodb
serial port open
mongo db connection OK.
iotInfo: Current date: 2023-11-22 14:50:19.089, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.089, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.083, Temp: .2, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.090, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.090, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.088, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.091, Temp: 20.2, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.091, Temp: 20.2, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.092, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.092, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.093, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.093, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.093, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.089, Temp: 20.2, Humi: 45.6, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:19.091, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:20.672, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:21.672, Temp: 20.3, Humi: 45.7, Lux: 63
iotInfo: Current date: 2023-11-22 14:50:22.683, Temp: 20.3, Humi: 45.7, Lux: 63
```

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

3. cds_dht22_mongodb.js → Check documents in Mongo shell

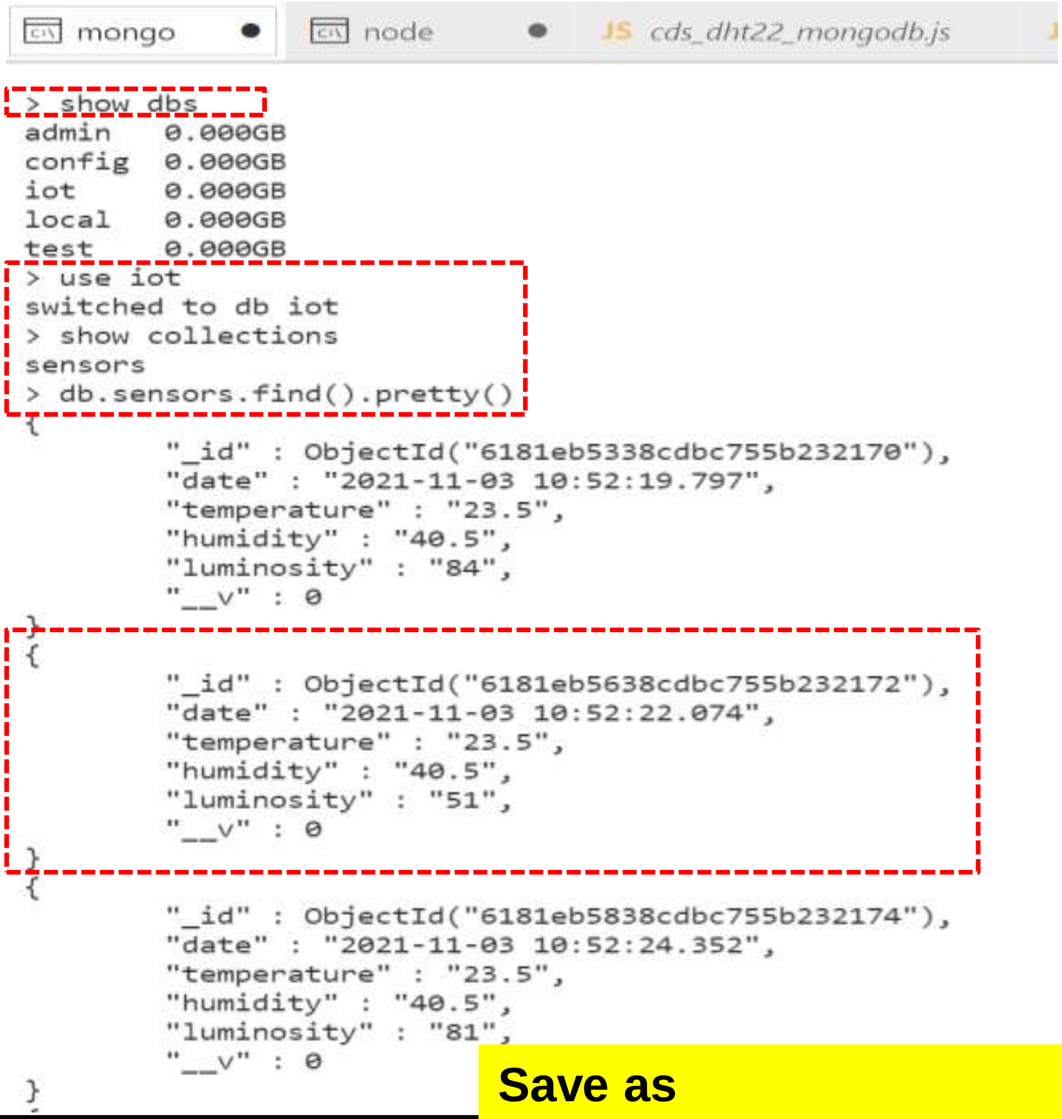
Mongo shell

> show dbs

> use iot

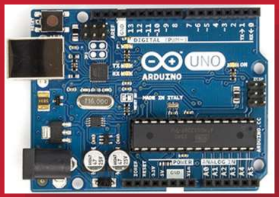
> show collections

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



```
> 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("6181eb5338cdbc755b232170"),
        "date" : "2021-11-03 10:52:19.797",
        "temperature" : "23.5",
        "humidity" : "40.5",
        "luminosity" : "84",
        "__v" : 0
}
{
        "_id" : ObjectId("6181eb5638cdbc755b232172"),
        "date" : "2021-11-03 10:52:22.074",
        "temperature" : "23.5",
        "humidity" : "40.5",
        "luminosity" : "51",
        "__v" : 0
}
{
        "_id" : ObjectId("6181eb5838cdbc755b232174"),
        "date" : "2021-11-03 10:52:24.352",
        "temperature" : "23.5",
        "humidity" : "40.5",
        "luminosity" : "81",
        "__v" : 0
}
```

Save as
AAnn_iot_mongdb.png



Arduino

& Node.js

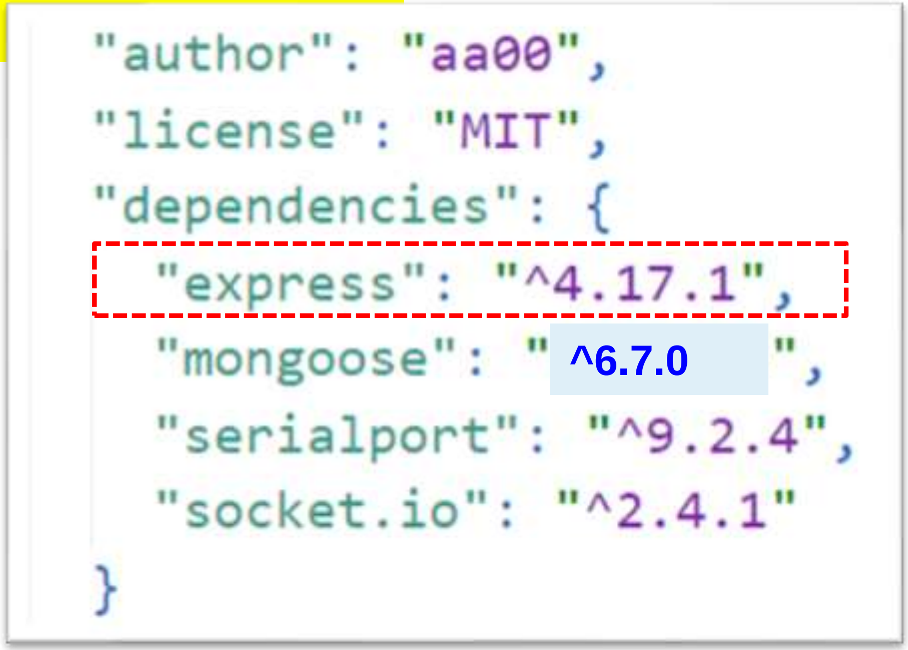
& MongoDB

& Express server

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

1.1 Install express server – package.json

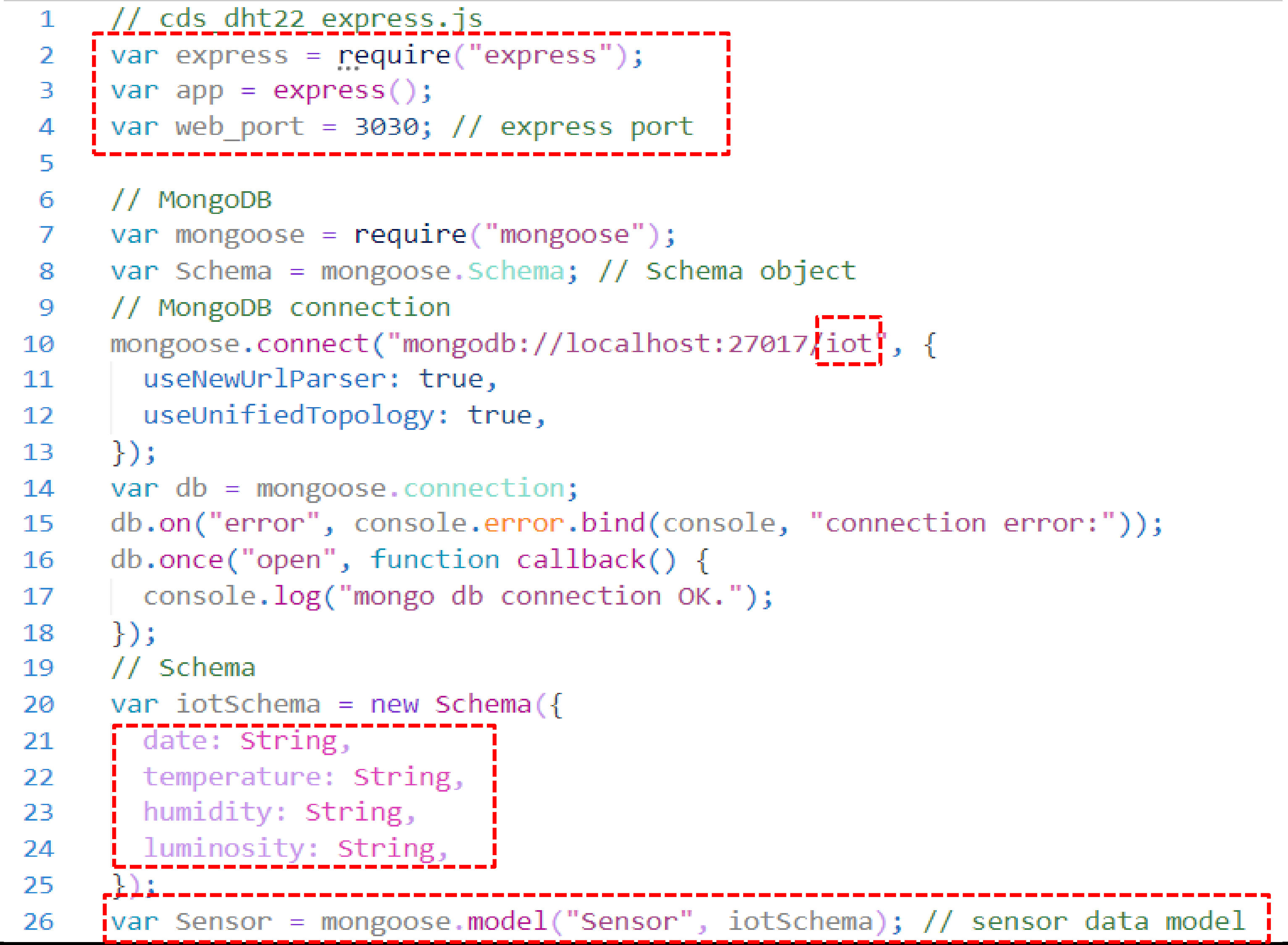
- Go to cds_dht22 project
- npm install --save express
- package.json



```
"author": "aa00",
"license": "MIT",
"dependencies": {
  "express": "^4.17.1",
  "mongoose": "^6.7.0",
  "serialport": "^9.2.4",
  "socket.io": "^2.4.1"
}
```

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

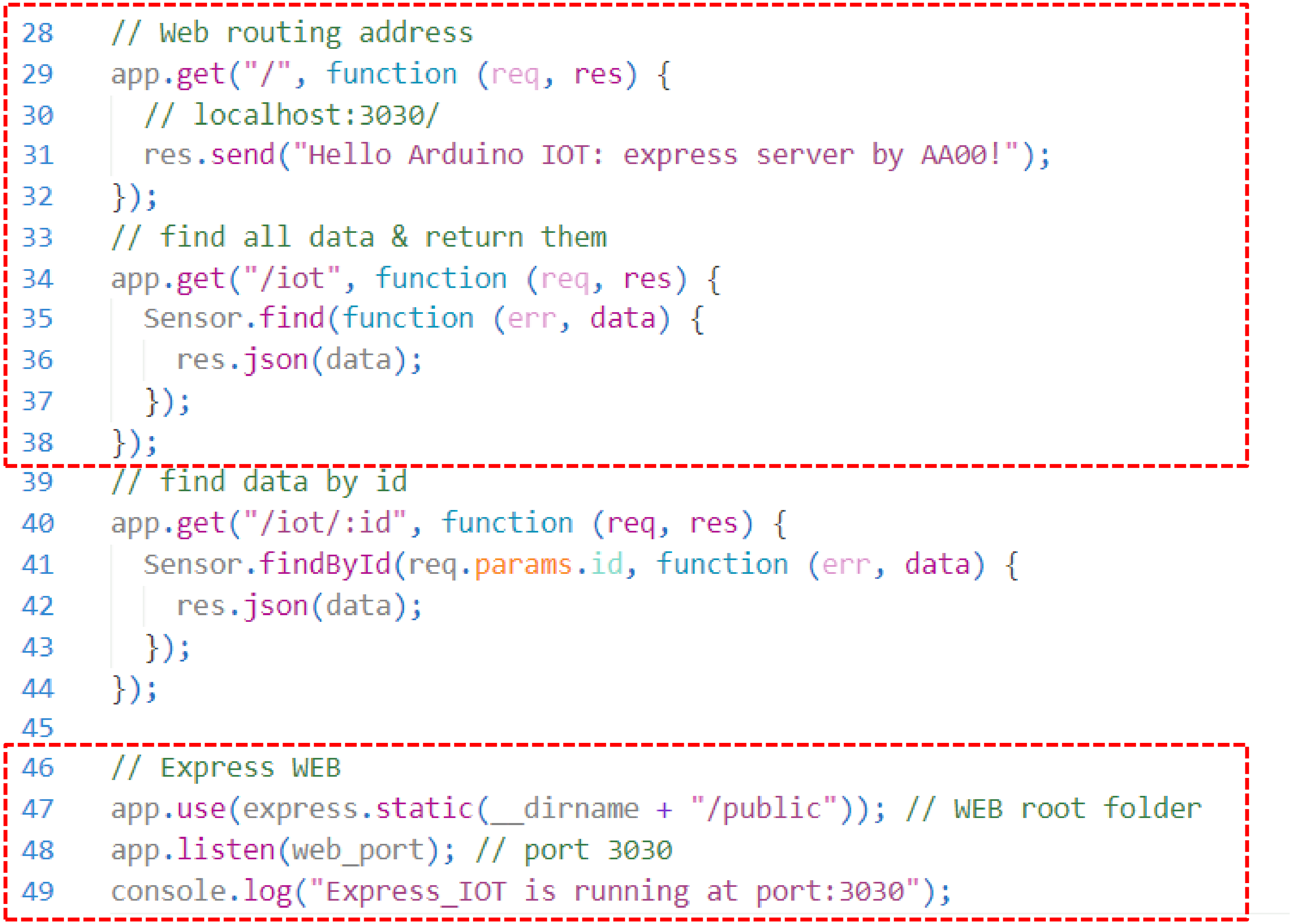
2.1 cds_dht22_express.js



```javascript
// cds_dht22_express.js
var express = require("express");
var app = express();
var web_port = 3030; // express port

// MongoDB
var mongoose = require("mongoose");
var Schema = mongoose.Schema; // Schema object
// MongoDB connection
mongoose.connect("mongodb://localhost:27017/iot", {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});
var db = mongoose.connection;
db.on("error", console.error.bind(console, "connection error:"));
db.once("open", function callback() {
  console.log("mongo db connection OK.");
});
// Schema
var iotSchema = new Schema({
  date: String,
  temperature: String,
  humidity: String,
  luminosity: String,
});
var Sensor = mongoose.model("Sensor", iotSchema); // sensor data model
```

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

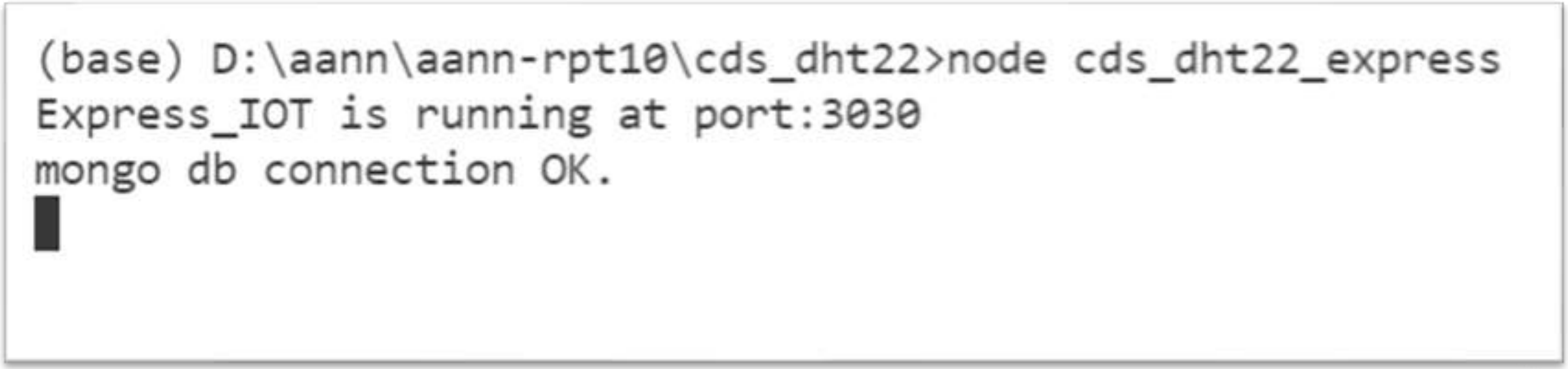
2.2 cds_dht22_express.js



```
// Web routing address
app.get("/", function (req, res) {
  // localhost:3030/
  res.send("Hello Arduino IOT: express server by AA00!");
});
// find all data & return them
app.get("/iot", function (req, res) {
  Sensor.find(function (err, data) {
    res.json(data);
  });
});
// find data by id
app.get("/iot/:id", function (req, res) {
  Sensor.findById(req.params.id, function (err, data) {
    res.json(data);
  });
});

// Express WEB
app.use(express.static(__dirname + "/public")); // WEB root folder
app.listen(web_port); // port 3030
console.log("Express_IOT is running at port:3030");
```

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

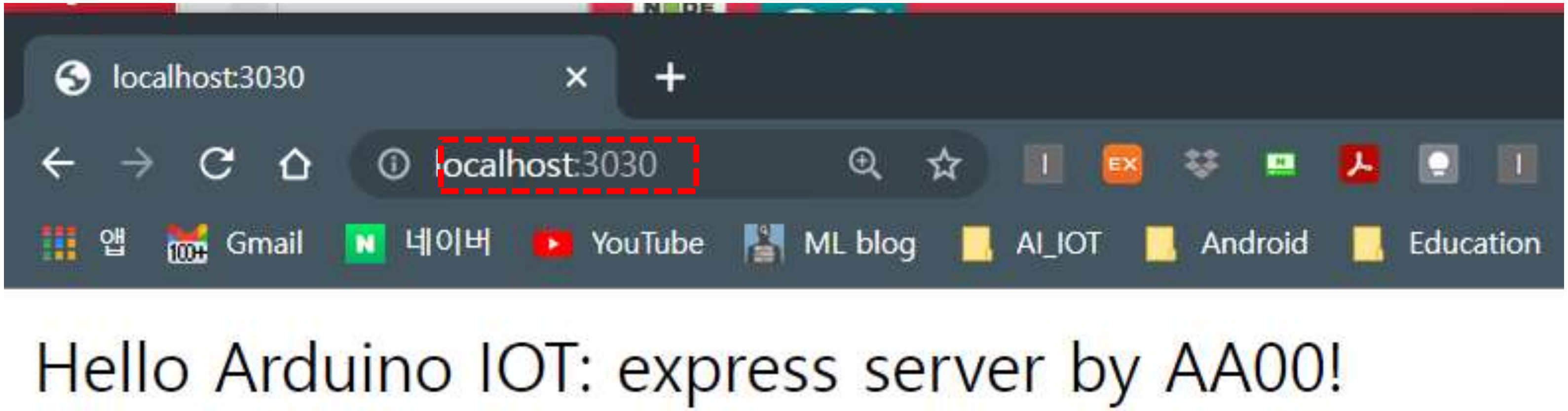
2.3 [Run] node cds_dht22_express.js



```
(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

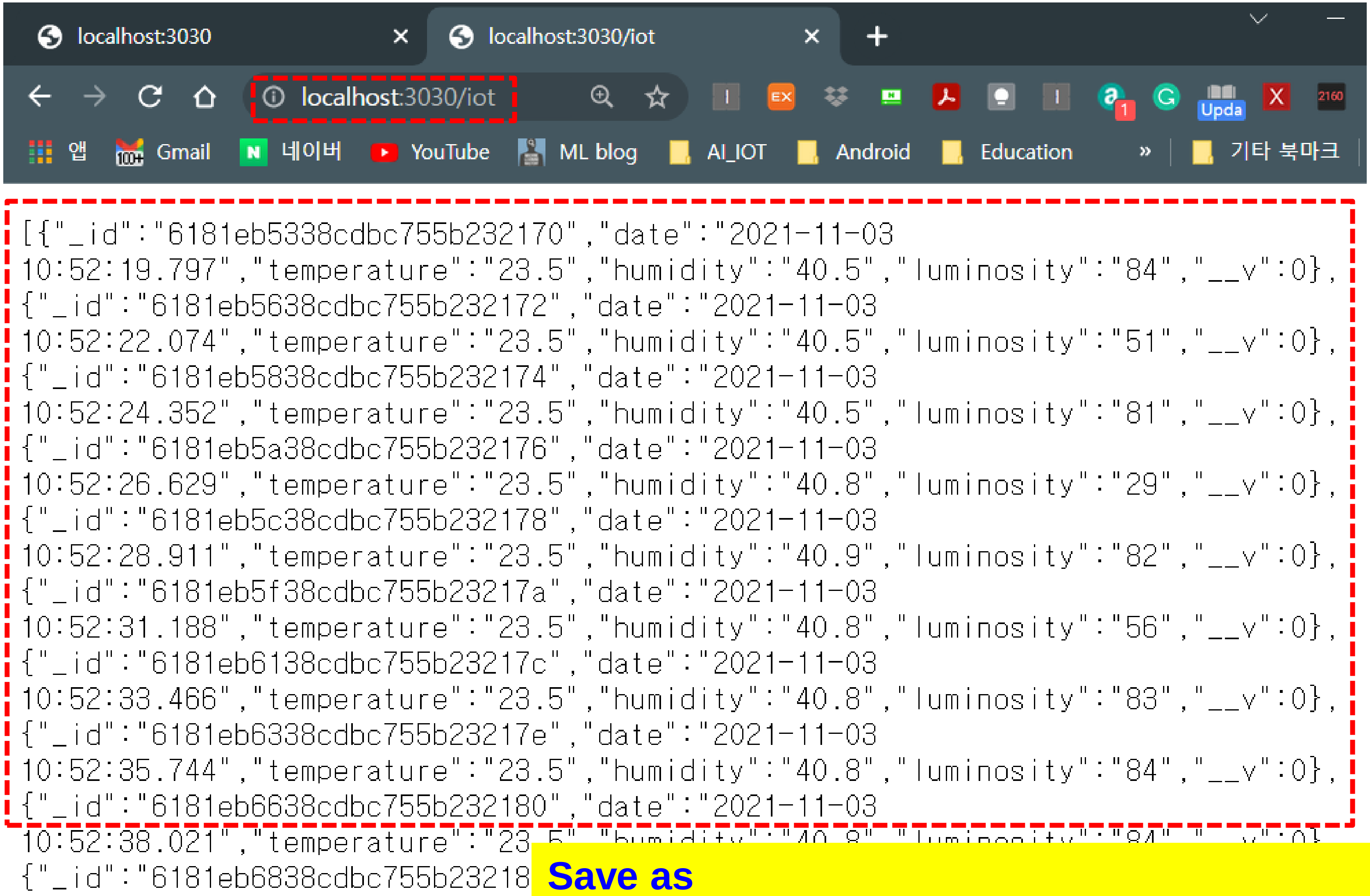
2.4 cds_dht22_express.js → routing1, http://localhost:3030/



Hello Arduino IOT: express server by AA00!

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

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



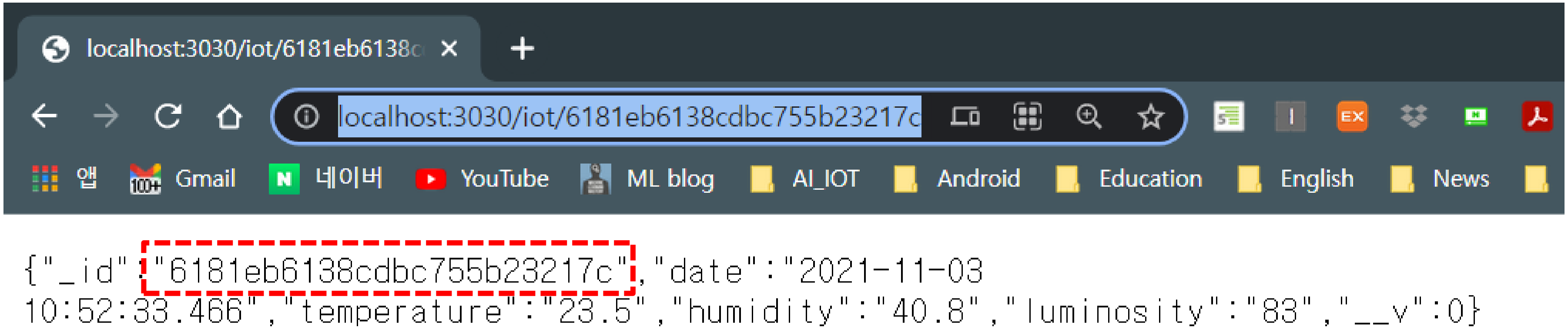
```
[{"_id":"6181eb5338cdbc755b232170","date":"2021-11-03
10:52:19.797","temperature":"23.5","humidity":"40.5","luminosity":"84","__v":0},
{"_id":"6181eb5638cdbc755b232172","date":"2021-11-03
10:52:22.074","temperature":"23.5","humidity":"40.5","luminosity":"51","__v":0},
{"_id":"6181eb5838cdbc755b232174","date":"2021-11-03
10:52:24.352","temperature":"23.5","humidity":"40.5","luminosity":"81","__v":0},
{"_id":"6181eb5a38cdbc755b232176","date":"2021-11-03
10:52:26.629","temperature":"23.5","humidity":"40.8","luminosity":"29","__v":0},
{"_id":"6181eb5c38cdbc755b232178","date":"2021-11-03
10:52:28.911","temperature":"23.5","humidity":"40.9","luminosity":"82","__v":0},
{"_id":"6181eb5f38cdbc755b23217a","date":"2021-11-03
10:52:31.188","temperature":"23.5","humidity":"40.8","luminosity":"56","__v":0},
{"_id":"6181eb6138cdbc755b23217c","date":"2021-11-03
10:52:33.466","temperature":"23.5","humidity":"40.8","luminosity":"83","__v":0},
{"_id":"6181eb6338cdbc755b23217e","date":"2021-11-03
10:52:35.744","temperature":"23.5","humidity":"40.8","luminosity":"84","__v":0},
{"_id":"6181eb6638cdbc755b232180","date":"2021-11-03
10:52:38.021","temperature":"23.5","humidity":"40.8","luminosity":"84","__v":0},
{"_id":"6181eb6838cdbc755b23218
```

Save as

AAnn_iot_mongodb_web.png

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

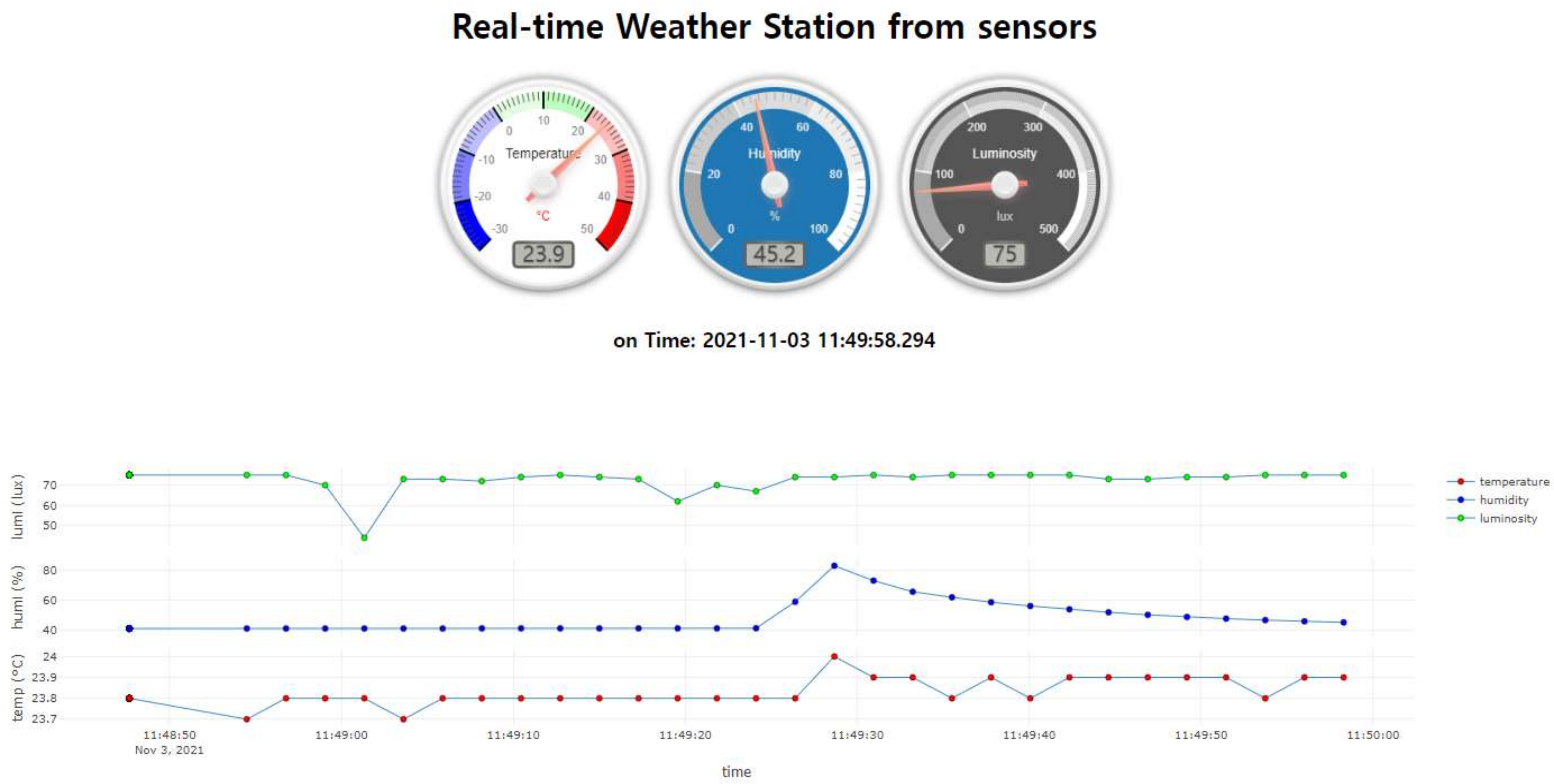
2.6 cds_dht22_express.js → routing2 http://localhost:3030/iot:id


localhost:3030/iot/6181eb6138c
localhost:3030/iot/6181eb6138cdbc755b23217c
앱
Gmail
네이버
YouTube
ML blog
AI_IOT
Android
Education
English
News

```
{"_id":"6181eb6138cdbc755b23217c","date":"2021-11-03
10:52:33.466","temperature":"23.5","humidity":"40.8","luminosity":"83","__v":0}
```

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

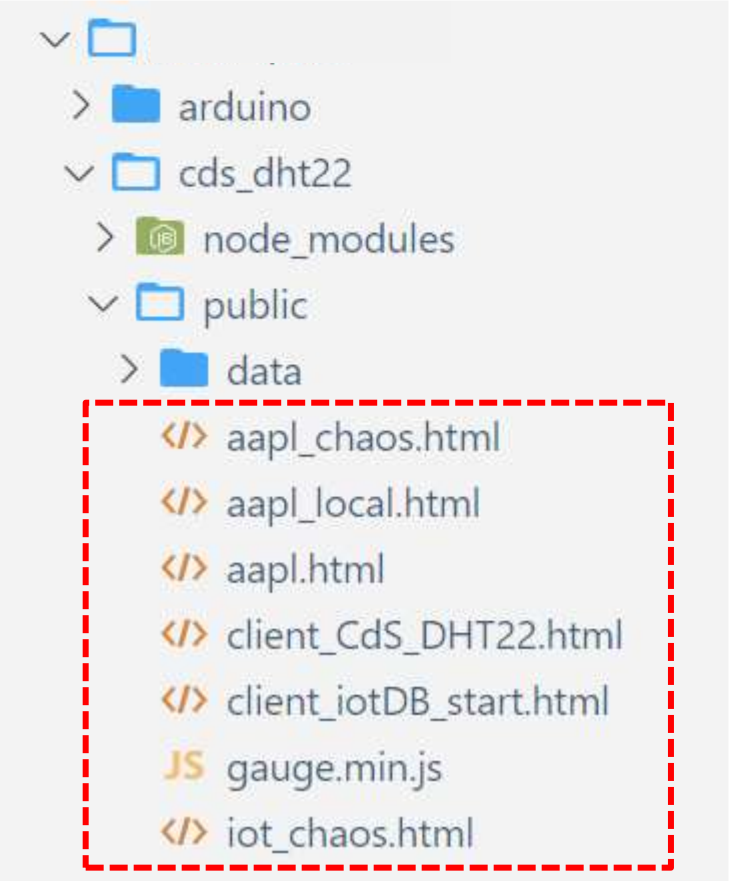
2.7 copy cds_dht22_client.html & gauge.min.js → ./public/ subfolder
http://localhost:3030/client_cds_dht22.html **(web root folder)**



Real-time Weather Station from sensors

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

2.8 CORS bug (Cross Origin Resource Sharing)



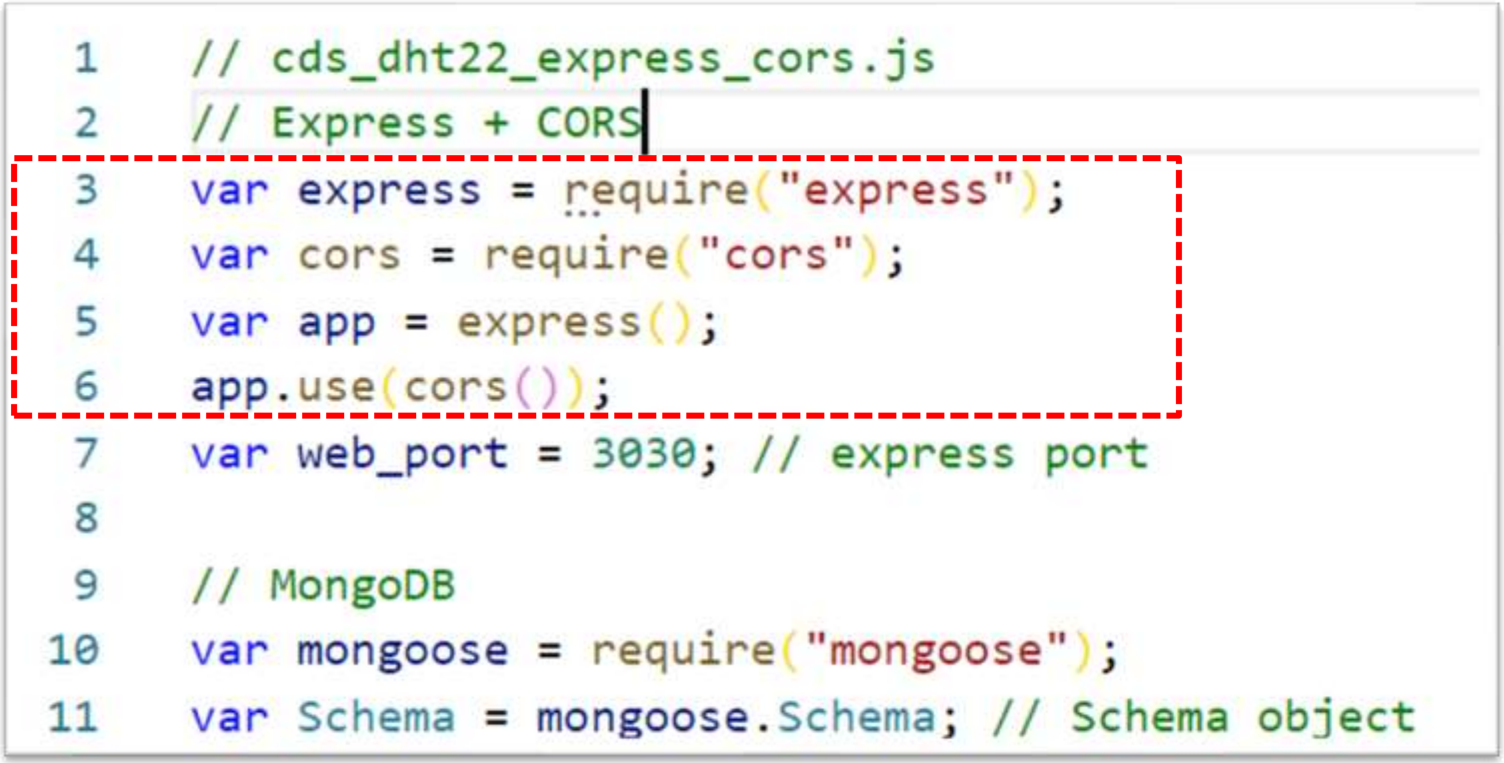
```
arduino
cds_dht22
  node_modules
  public
    data
    aapl_chaos.html
    aapl_local.html
    aapl.html
    client_CdS_DHT22.html
    client_iotDB_start.html
    gauge.min.js
    iot_chaos.html
```

*** CORS problem**

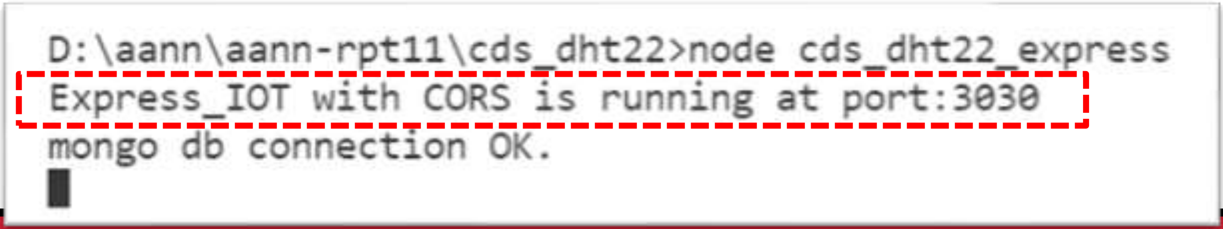
→ 원격 서버 내의 파일에 접근을 허용

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



```
// 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
```

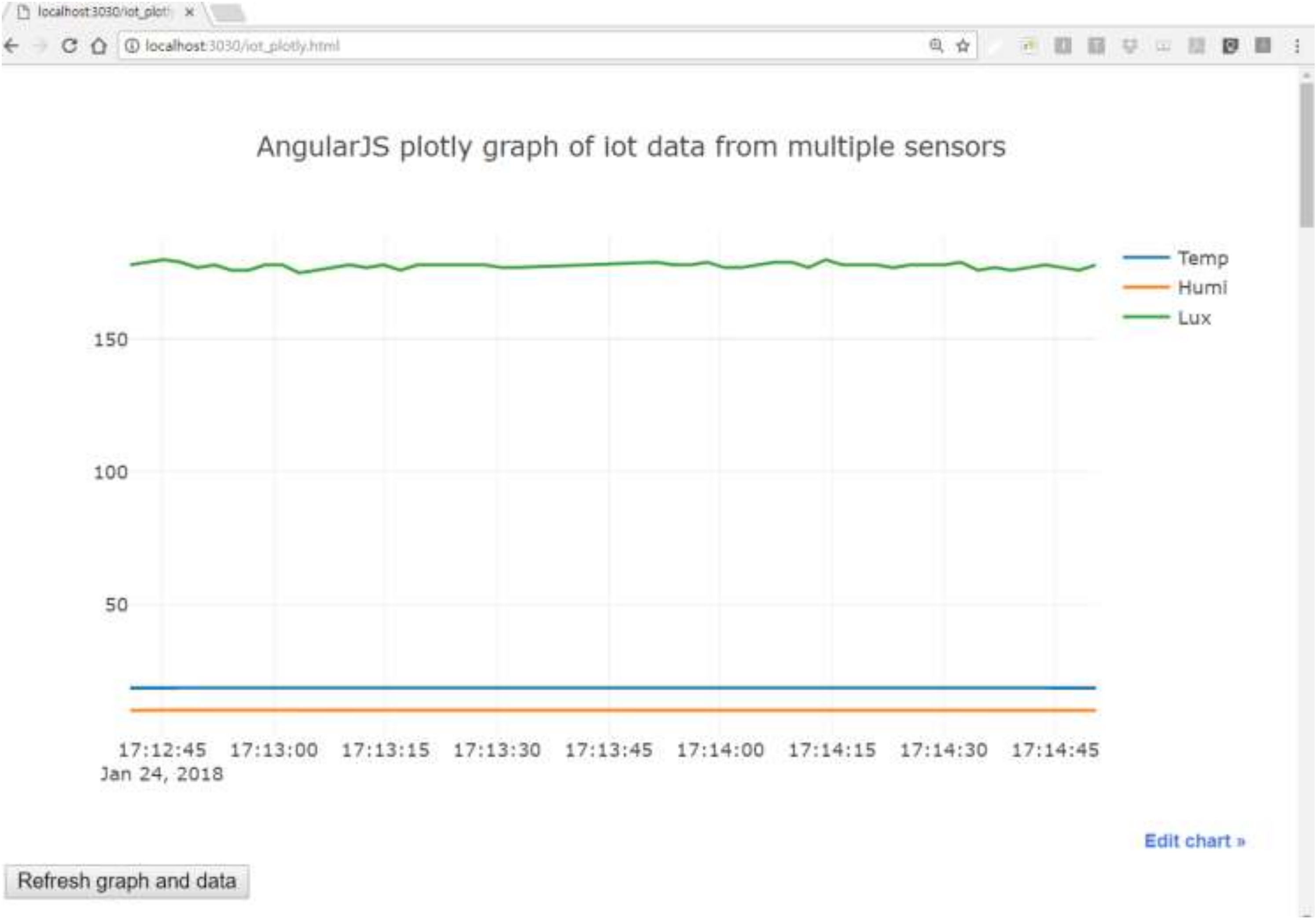


```
D:\aann\aann-rpt11\cds_dht22>node cds_dht22_express
Express_IOT with CORS is running at port:3030
mongo db connection OK.
█
```

Web monitoring

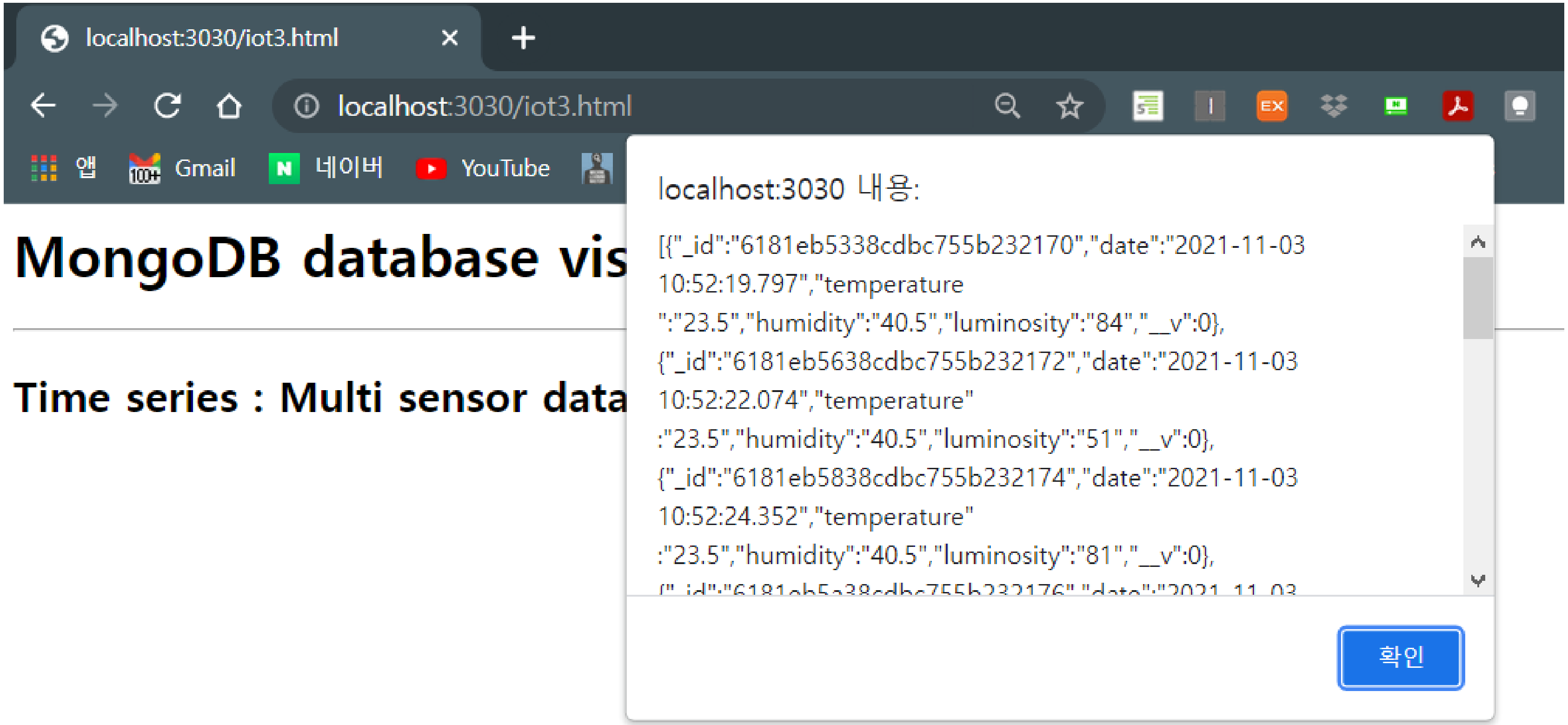
DHT22 + CdS + Node.js + MongoDB

Web monitoring – Google AngularJS



DHT22 + CdS + Node.js + MongoDB

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



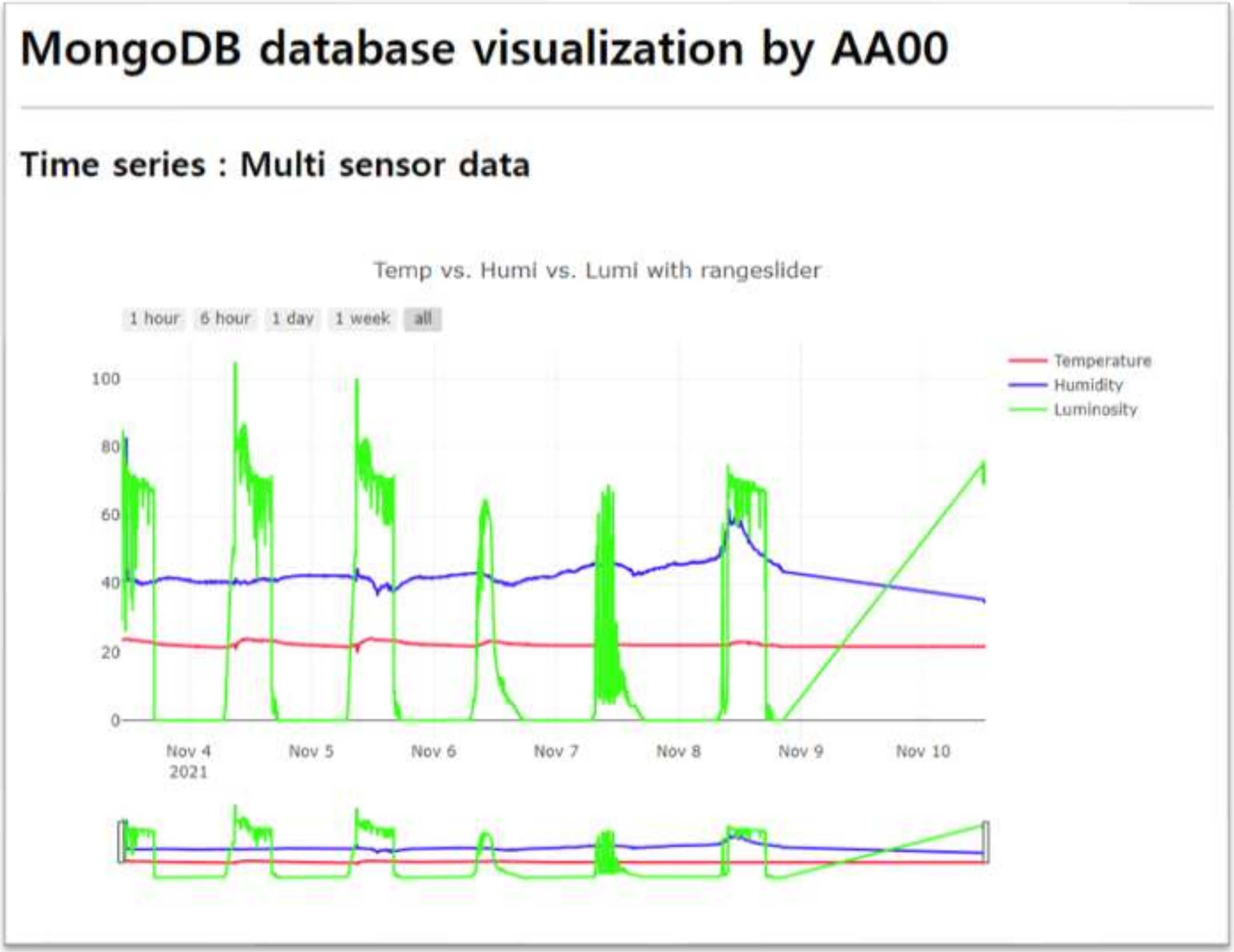
MongoDB database vis

Time series : Multi sensor data

```
[{"_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
```

DHT22 + CdS + Node.js + MongoDB

Web monitoring

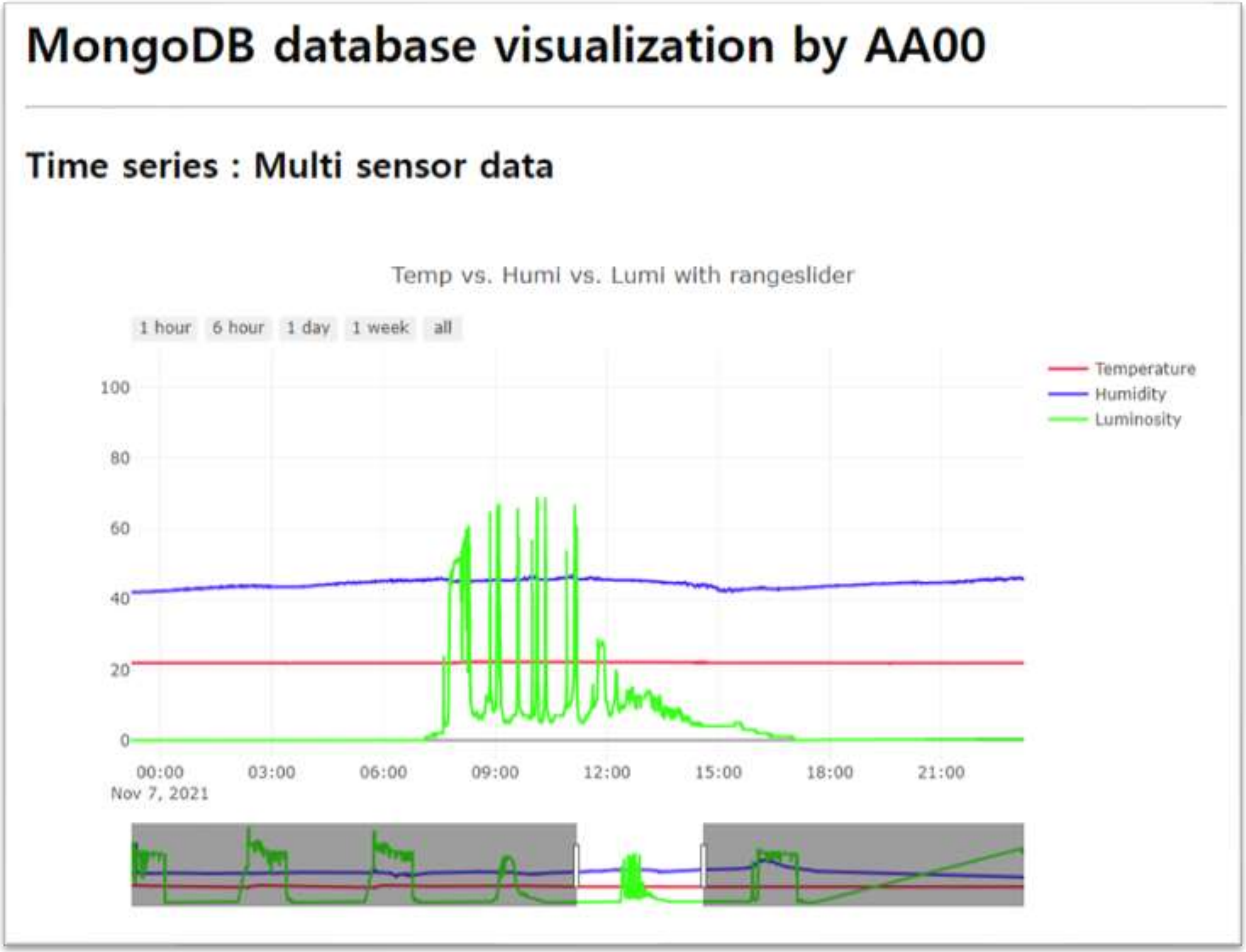


MongoDB database visualization by AA00

Time series : Multi sensor data

DHT22 + CdS + Node.js + MongoDB

Web monitoring

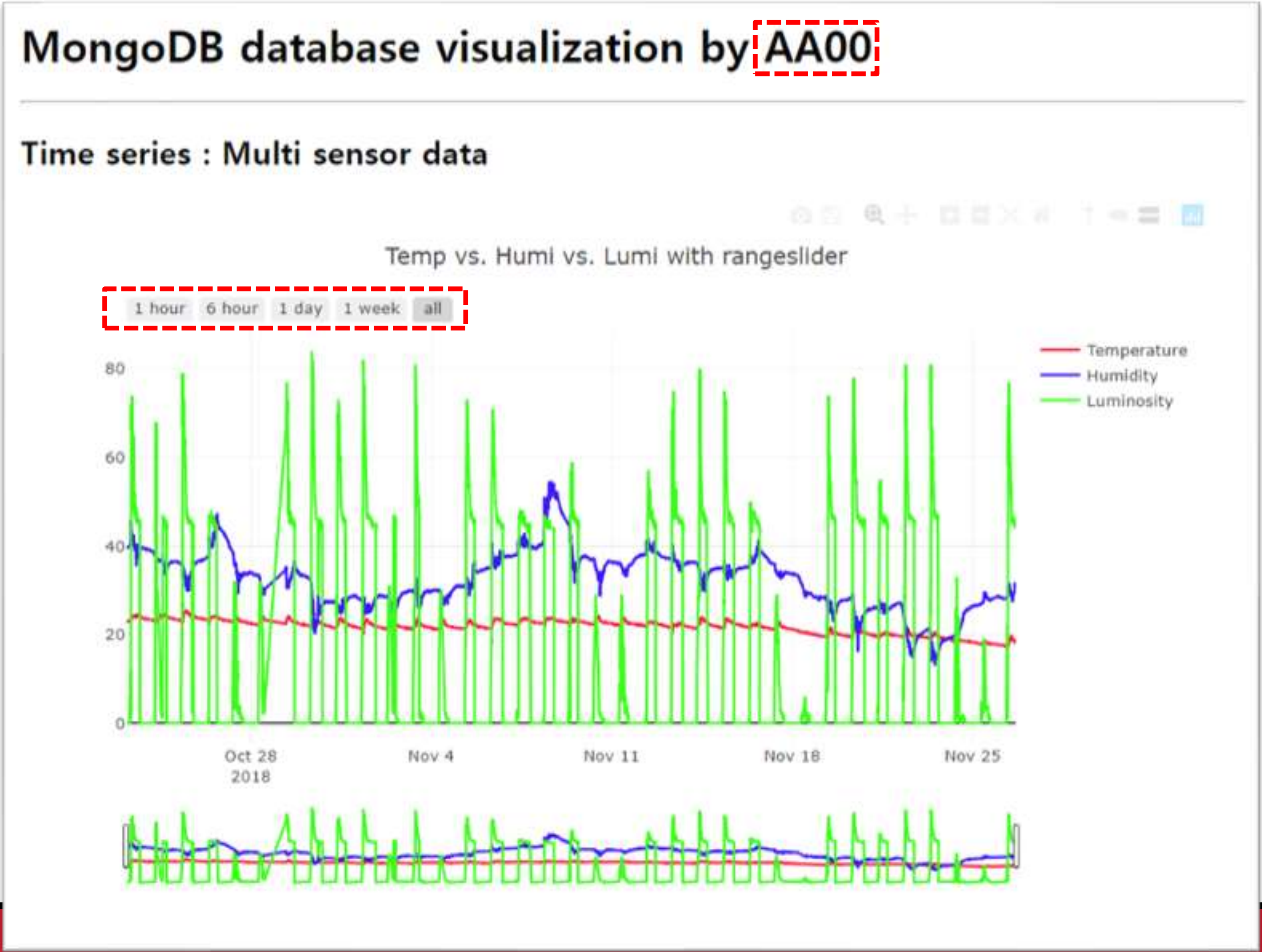


MongoDB database visualization by AA00

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)

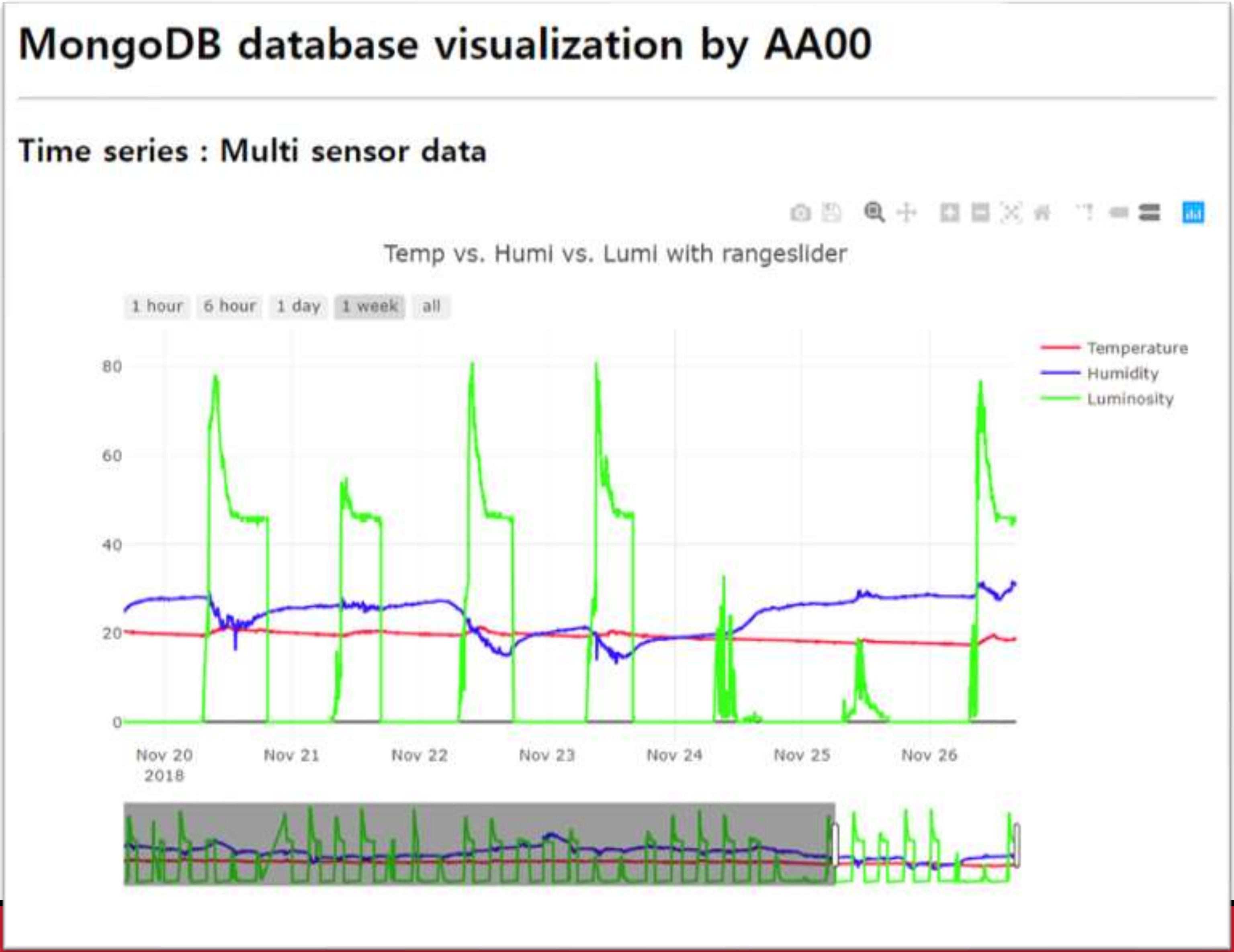


MongoDB database visualization by AA00

Time series : Multi sensor data

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

Web monitoring-2: week

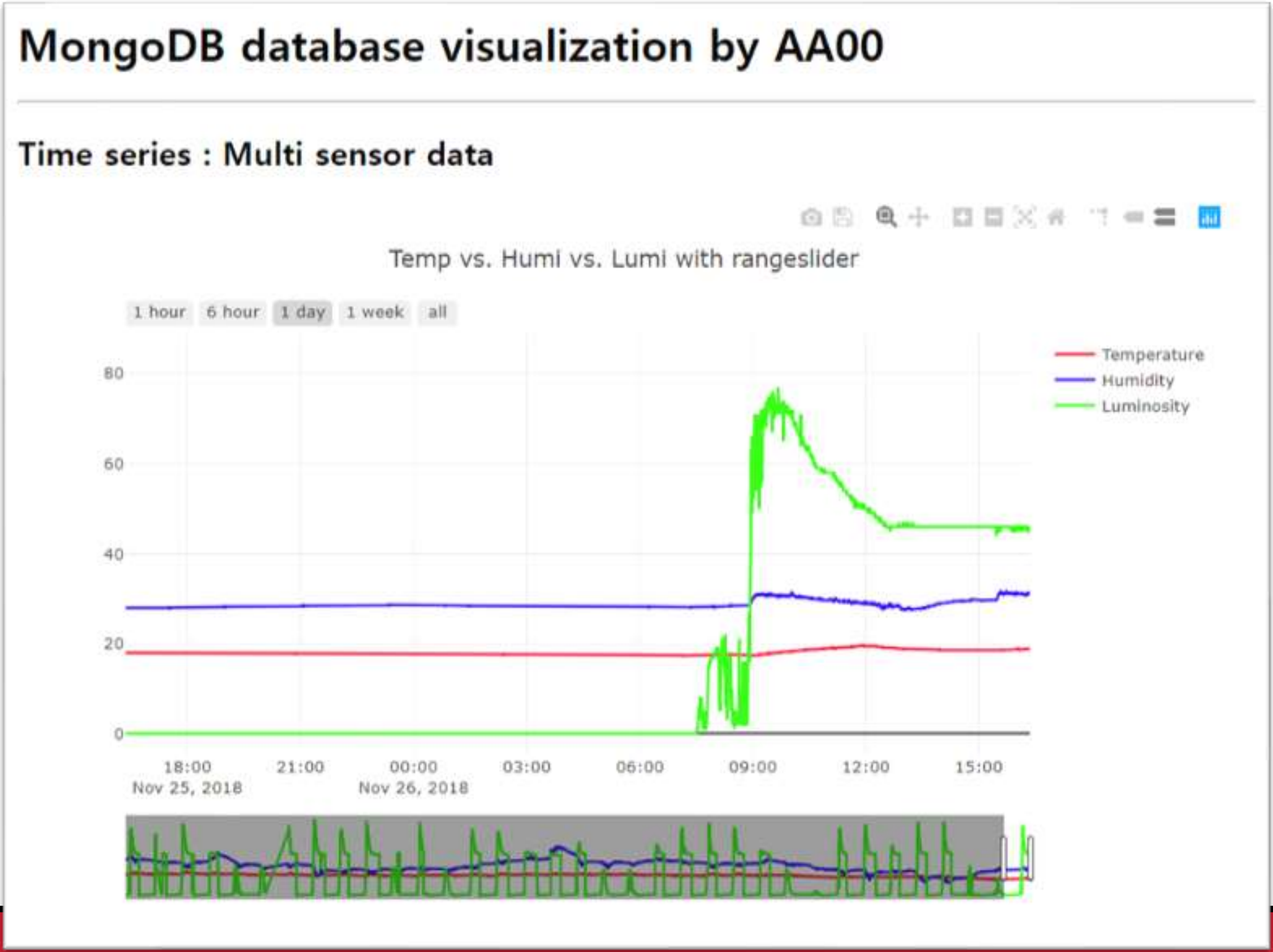


MongoDB database visualization by AA00

Time series : Multi sensor data

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

Web monitoring-3: day

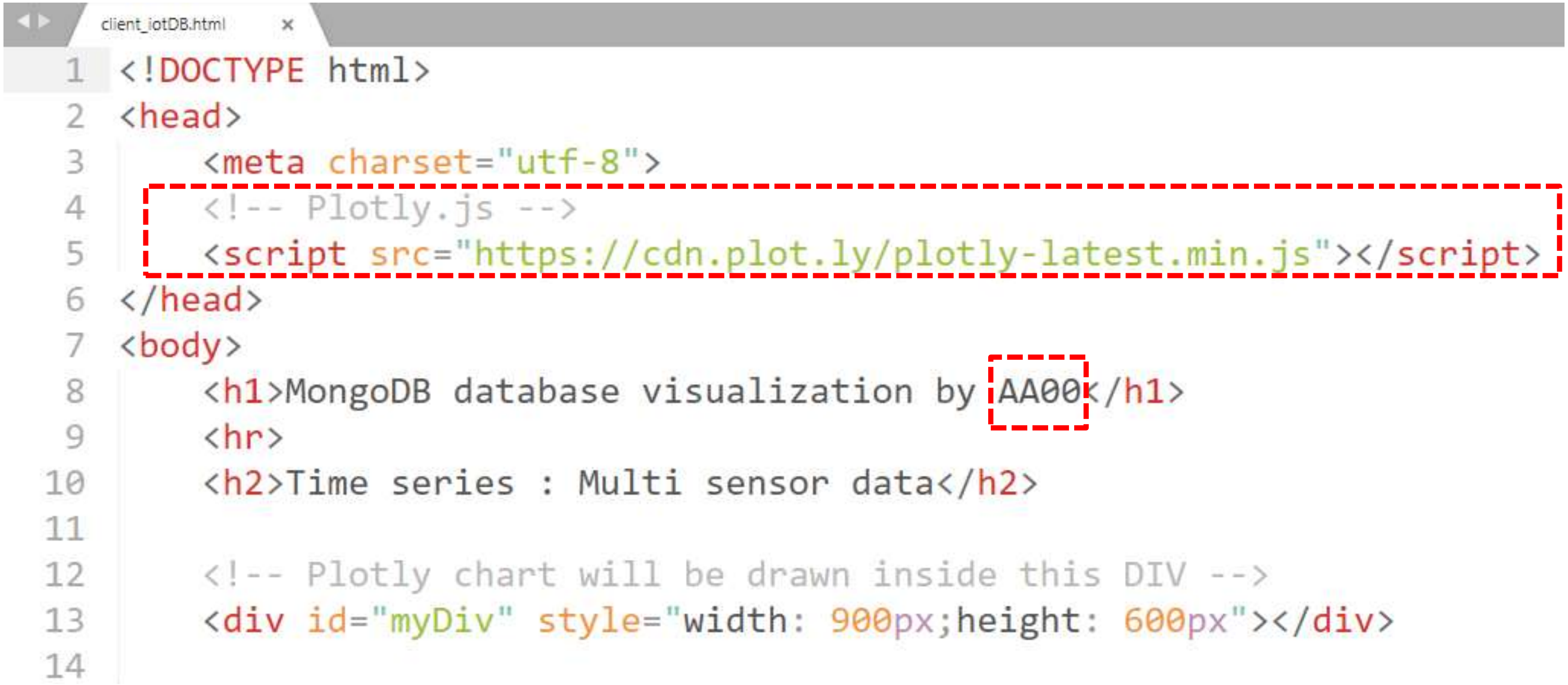


MongoDB database visualization by AA00

Time series : Multi sensor data

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

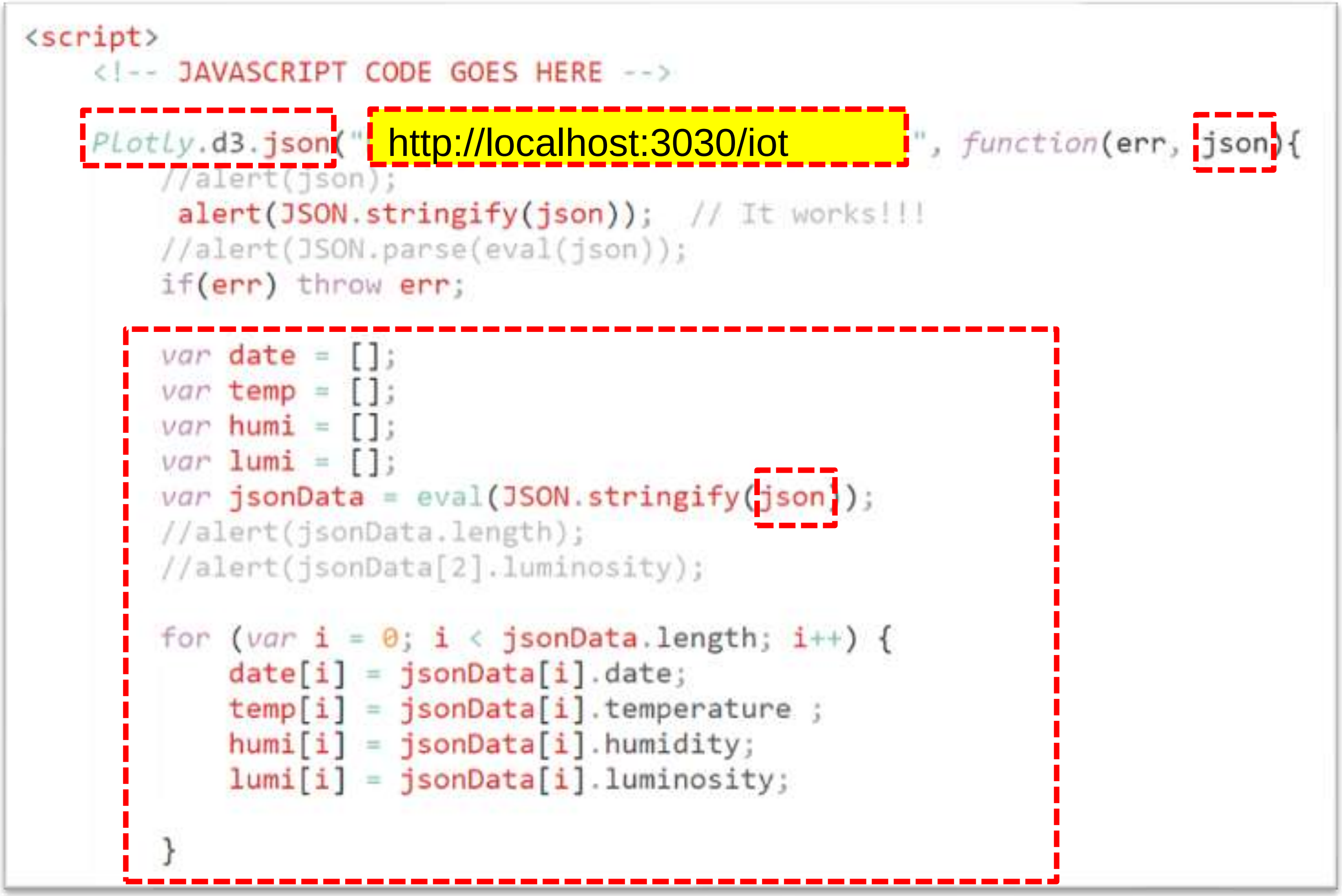
3.1 Web client: client_iotDB.html



```
<!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>

```

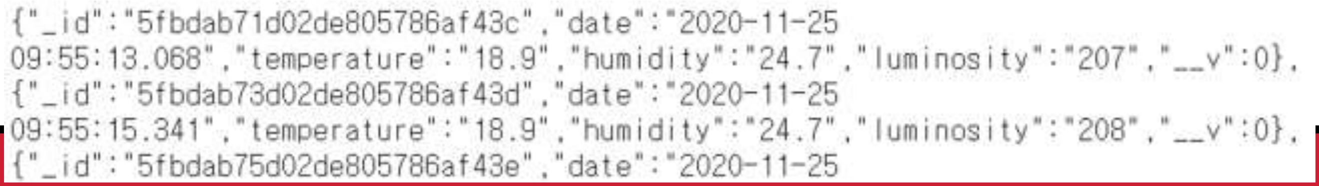
A5.9.7 DHT22 + CdS + Node.js + MongoDB

3.2 Web client: client_iotDB.html



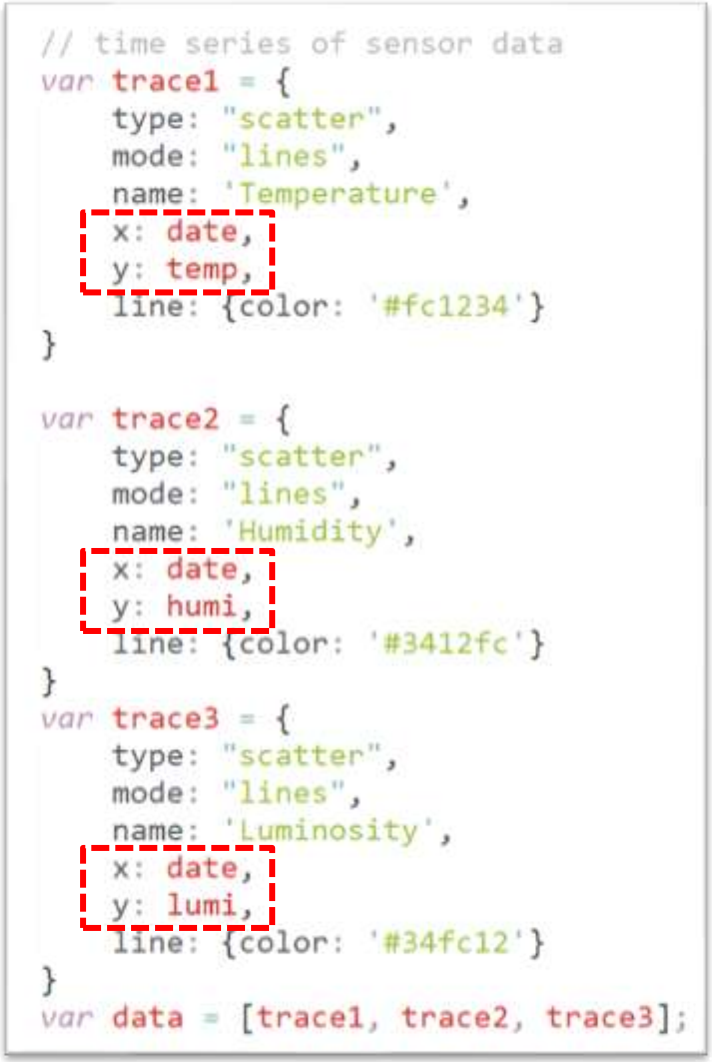
```
<script>
    <!-- JAVASCRIPT CODE GOES HERE -->

    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

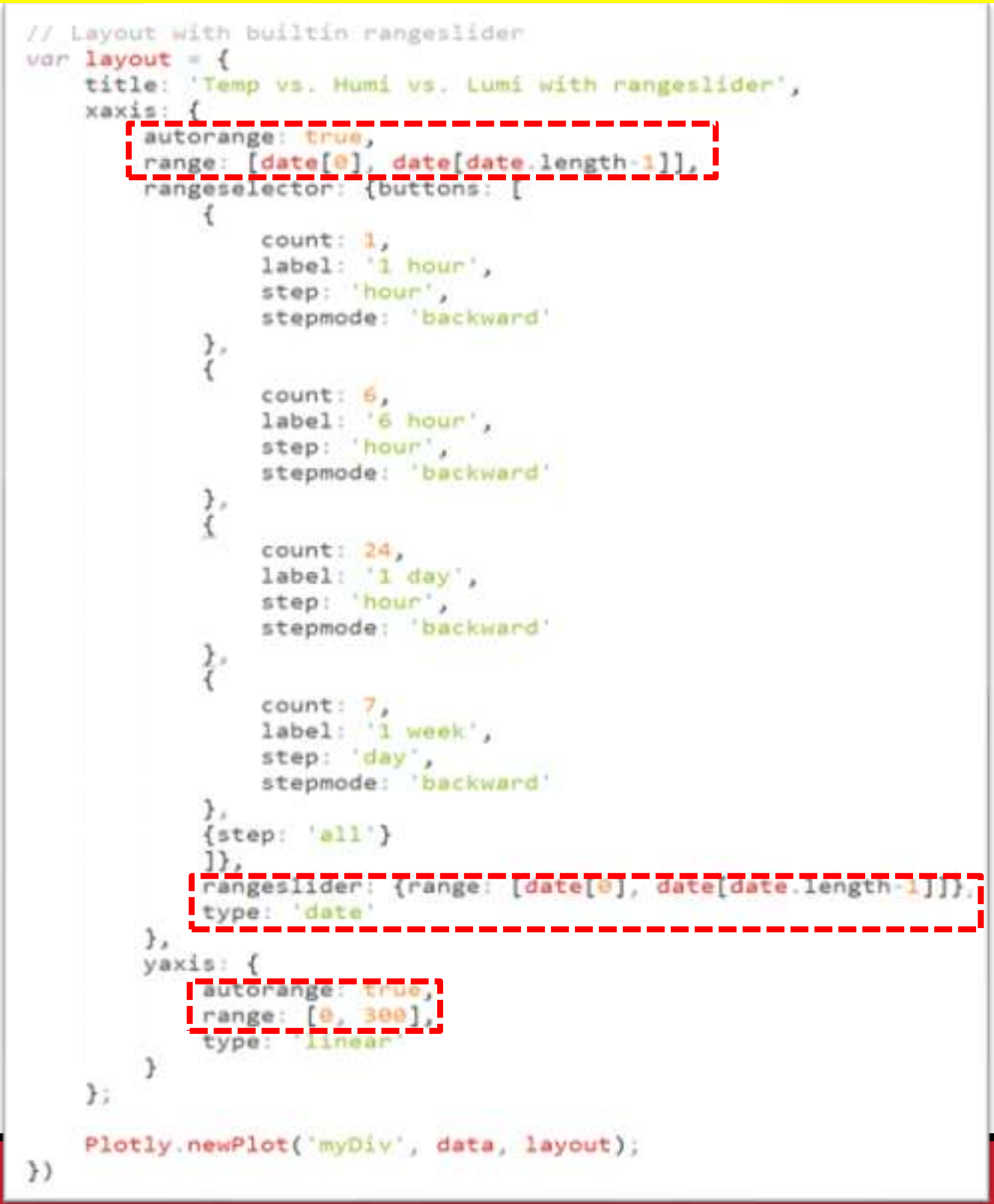


```
{"_id":"5fbdab71d02de805786af43c","date":"2020-11-25
09:55:13.068","temperature":"18.9","humidity":"24.7","luminosity":"207","__v":0},
{"_id":"5fbdab73d02de805786af43d","date":"2020-11-25
09:55:15.341","temperature":"18.9","humidity":"24.7","luminosity":"208","__v":0},
{"_id":"5fbdab75d02de805786af43e","date":"2020-11-25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

3.3 Web client: client_iotDB.html – data & layout

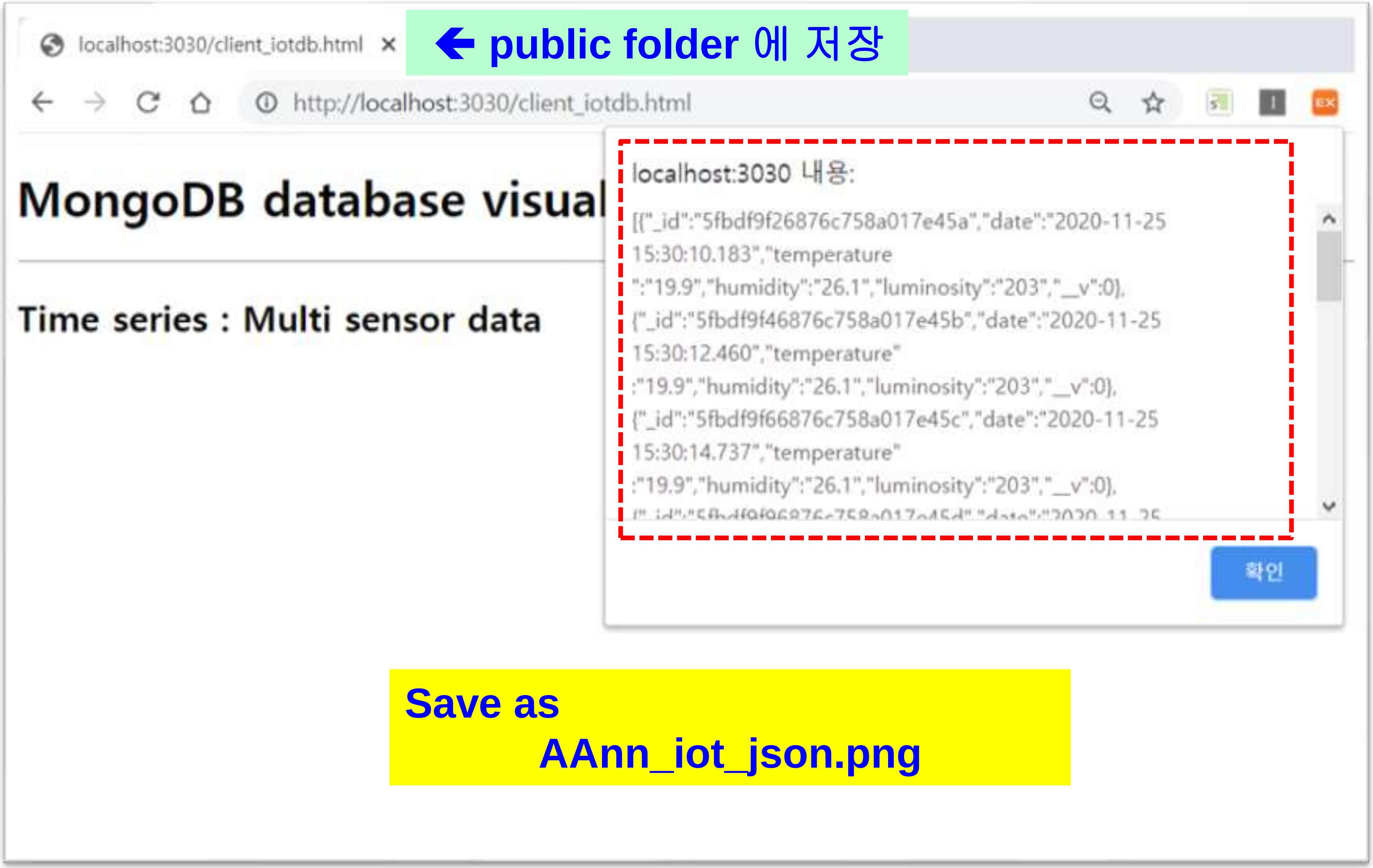


```
// time series of sensor data
var trace1 = {
    type: "scatter",
    mode: "lines",
    name: 'Temperature',
    x: date,
    y: temp,
    line: {color: '#fc1234'}
}

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);
})
```

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



localhost:3030/client_iotdb.html ×
http://localhost:3030/client_iotdb.html

← public folder 에 저장

MongoDB database visual

Time series : Multi sensor data

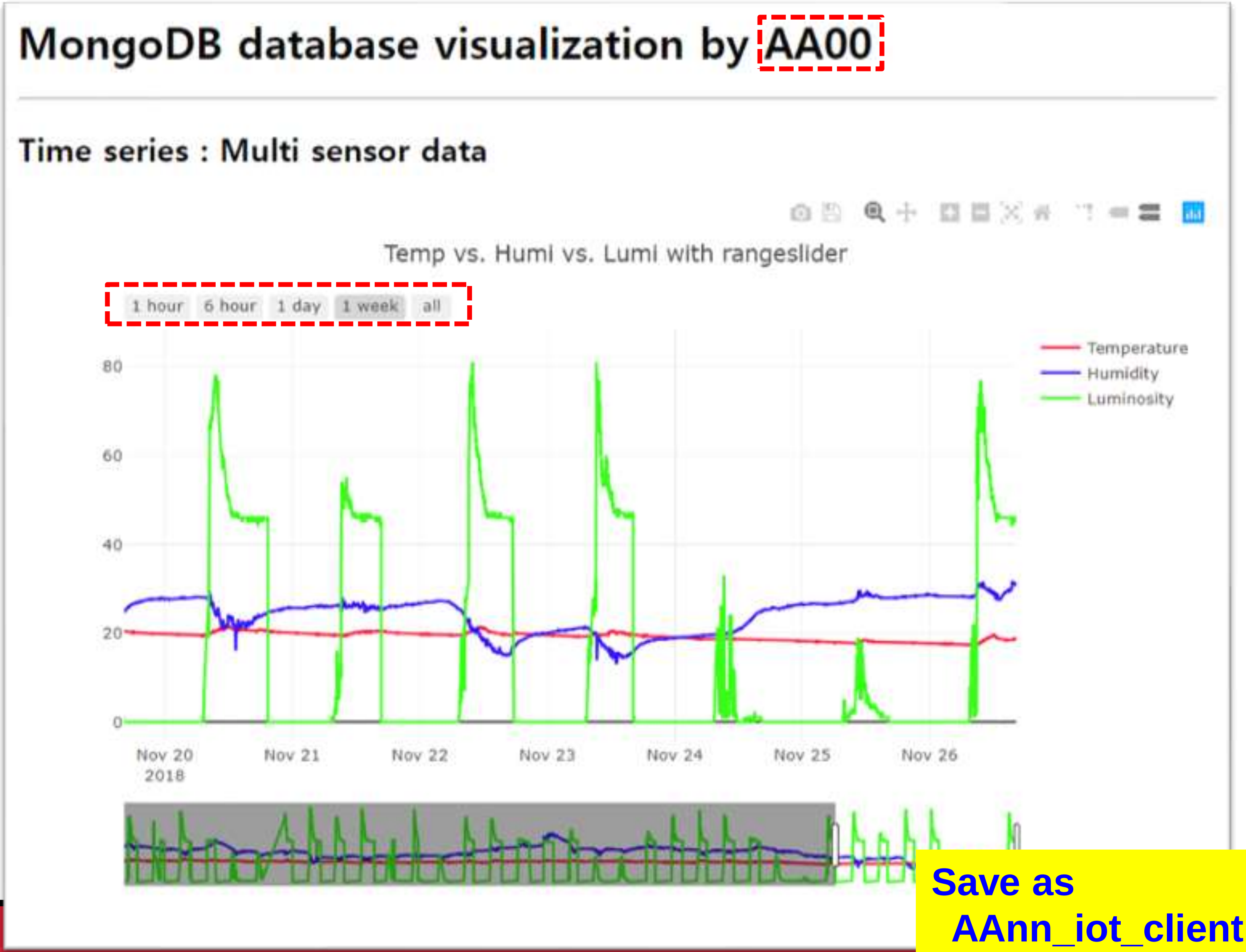
localhost:3030 내용:

```
[{"_id":"5fbdf9f26876c758a017e45a","date":"2020-11-25
15:30:10.183","temperature
":"19.9","humidity":"26.1","luminosity":"203","__v":0},
{"_id":"5fbdf9f46876c758a017e45b","date":"2020-11-25
15:30:12.460","temperature"
:"19.9","humidity":"26.1","luminosity":"203","__v":0},
{"_id":"5fbdf9f66876c758a017e45c","date":"2020-11-25
15:30:14.737","temperature"
:"19.9","humidity":"26.1","luminosity":"203","__v":0},
{"_id":"5fbdf9f96876c758a017e45d","date":"2020-11-25
```

Save as
AAnn_iot_json.png

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

3.5 Web client: client_iotDB.html – iot DB monitoring

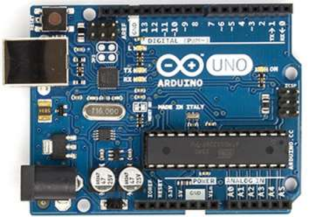


MongoDB database visualization by AA00

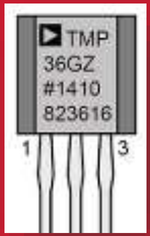
Time series : Multi sensor data

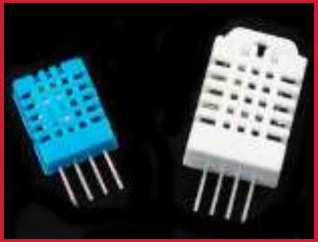
Save as

AAnn_iot_client.png



[Practice]





- [wk12]
  - ➢ RT Data storaging with MongoDB
  - ➢ Multi-sensor circuits (cds-dht22)
  - ➢ Complete your project
  - ➢ Upload folder: aann-rpt11

Wk11/12 : Practice : aann-rpt11

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

제출폴더명 : aann-rpt11

- 압축할 파일들

① AAnn_iot_mongodb_web.png

② AAnn_iot_json.png

③ AAnn_iot_client.png

④ All *.ino

⑤ All *.js

⑥ All *.html in public folder

⑦ Delete ‘node_modules’ folder

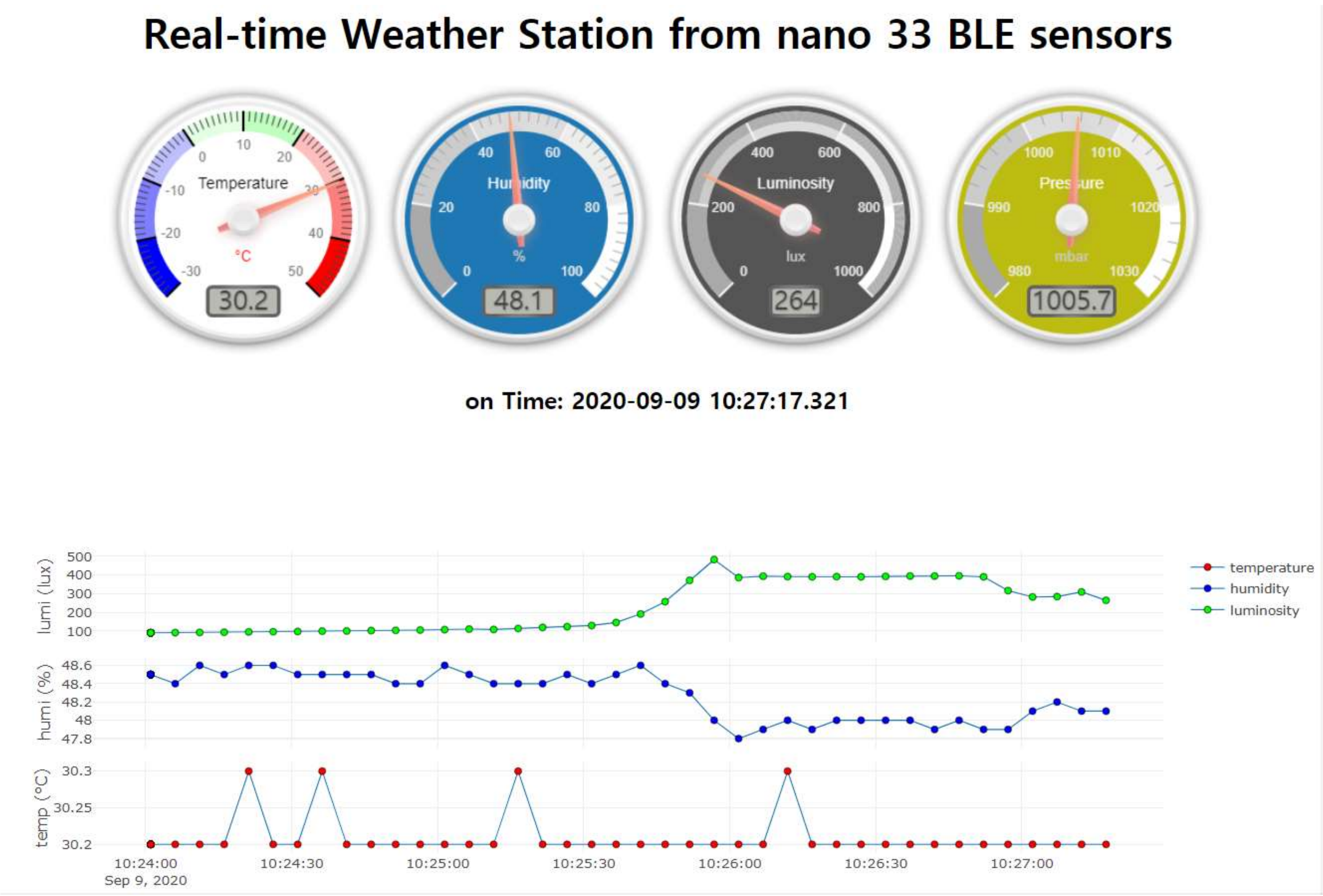
Email : chaos21c@gmail.com

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

Target of this class



Real-time Weather Station from nano 33 BLE sensors

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