



Arduino-IoT

[wk10]

Arduino + Node

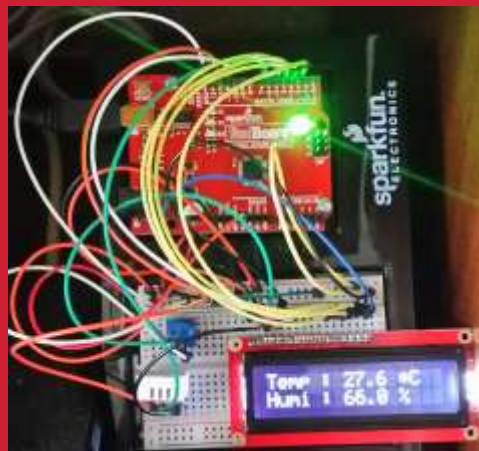
Data storing I

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

Drone-IoT-Comsi, INJE University

2nd 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]

◆ [wk09: mid-exam.]

- RT Data Visualization with node.js
- Multiple data and Usage of gauge.js
- Complete your real-time WEB charts
- Upload folder: aann-rpt09
- Use repo “aann” in github

wk09 : Practice : aann-rpt09

◆ [Target of this week]

- Complete your works : **mid-exam.**
- Save your outcomes and upload outputs in github

제출폴더명 : **aann-rpt09**

- 제출할 파일들

- ① **AAnn_cds_dht22_data.png**
- ② **AAnn_signals_cds_dht22.html**
- ③ **AAnn_cds_dht22.html**
- ④ **AAnn_cds_dht22.png**
- ⑤ **All *.ino**
- ⑥ **All *.js**
- ⑦ **All *.html**

Purpose of AA

주요 수업 목표는 다음과 같다.

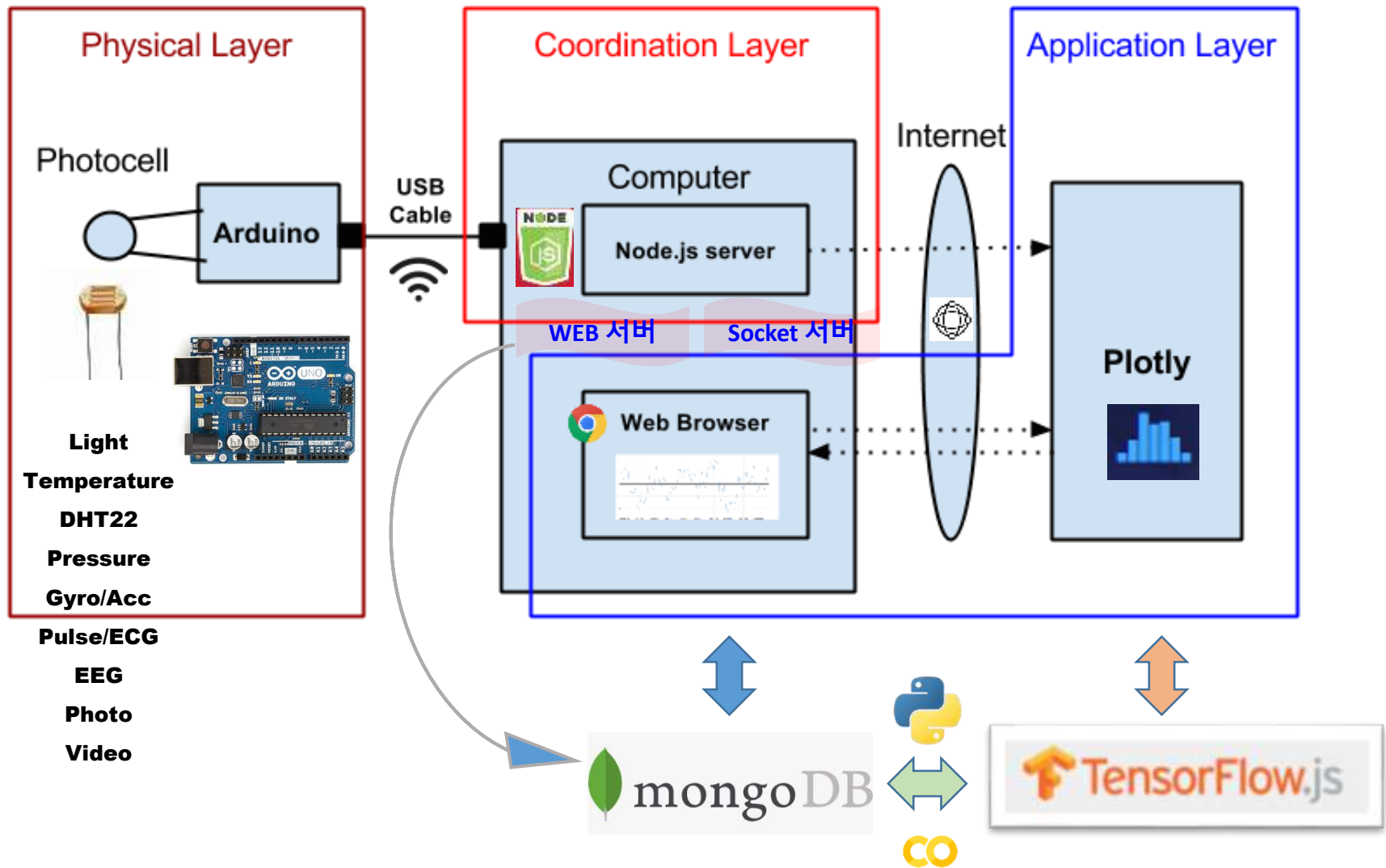
1. Node.js를 이용한 아두이노 센서 신호 처리
2. Plotly.js를 이용한 아두이노 센서 신호 시각화
3. MongoDB에 아두이노 센서 데이터 저장 및 처리



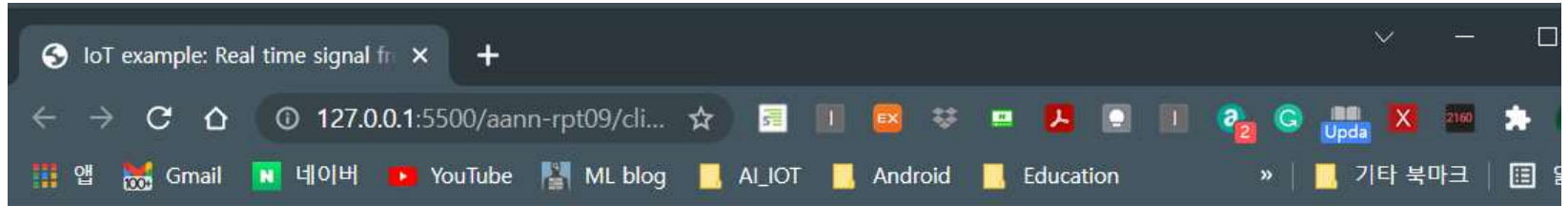
4. 저장된 IoT 데이터의 마이닝 (파이썬 코딩)



Layout [H S C]



on WEB monitoring Arduino data



IoT Signal from Arduino Weather Station

Real-time Signals

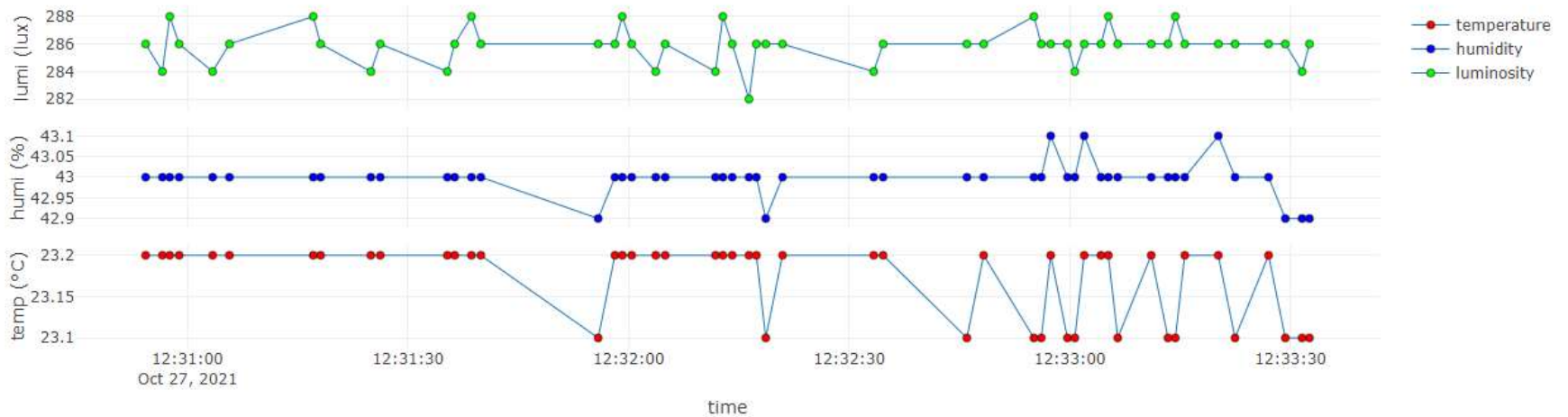
on Time: 2021-10-27 11:54:48.997

Signals (온도, 습도, 조도) : 23.4, 42.6, 286

Real-time Weather Station from sensors



on Time: 2021-10-27 12:33:32.600





A5. Introduction to IoT service

System (Arduino, sDevice, ...)



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



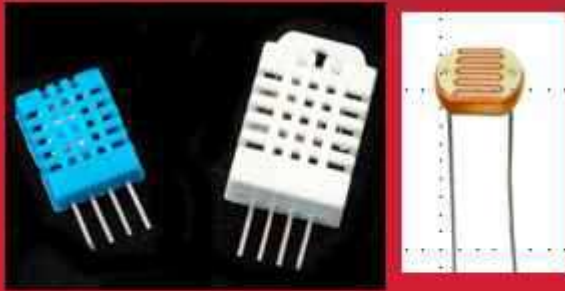
Visualization & monitoring



Data storing & mining



Service



[Goal]

Arduino + Node.js

+ plotly.js

+ MongoDB

→ Data storaging

& visualization

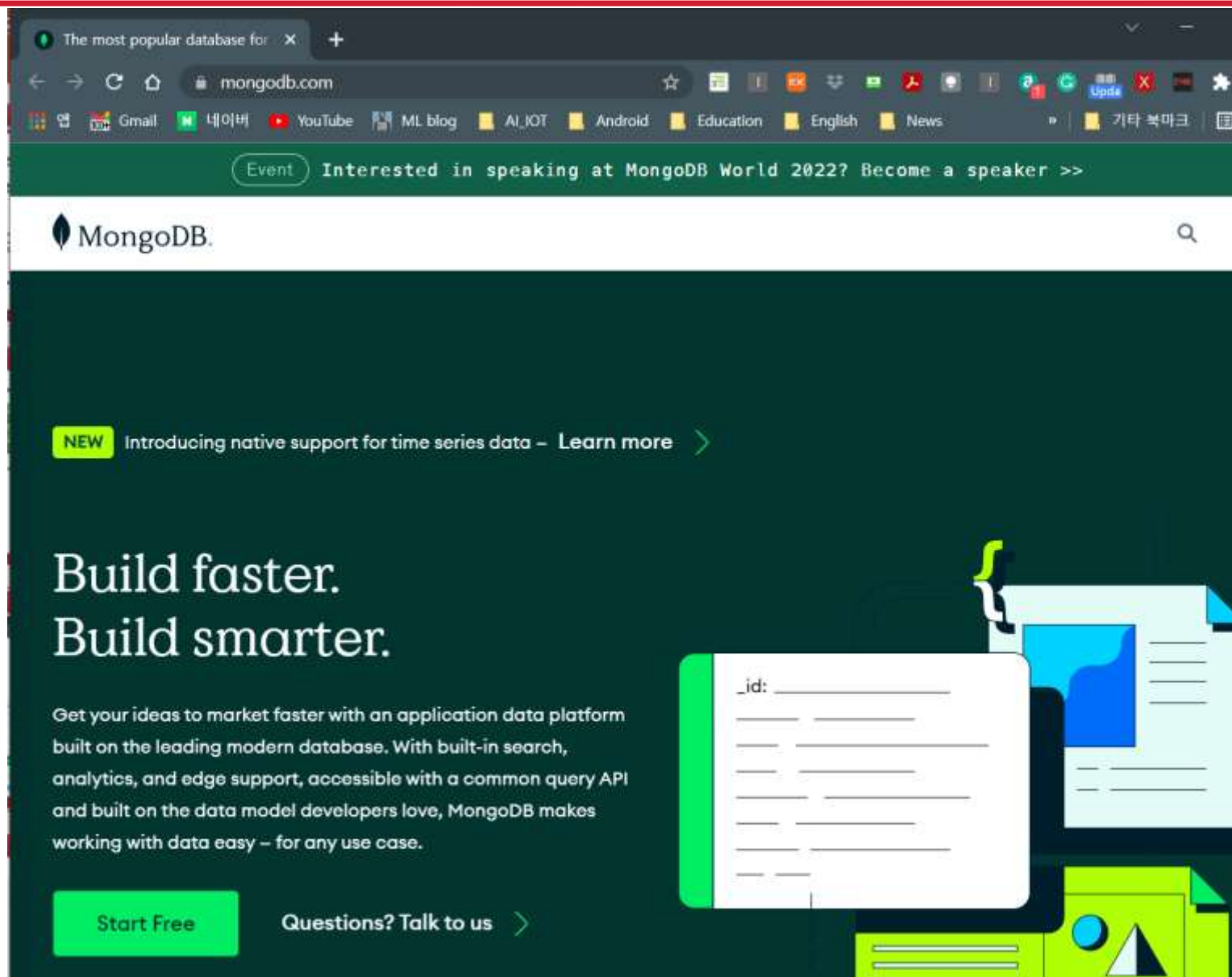


A5.9 MongoDB





A5.9 MongoDB



The screenshot shows the MongoDB website homepage in a web browser. The browser's address bar displays 'mongodb.com'. The page features a dark green header with the MongoDB logo and a search icon. Below the header, a green banner promotes an event: 'Interested in speaking at MongoDB World 2022? Become a speaker >>'. The main content area has a dark green background with the text 'Build faster. Build smarter.' and a paragraph describing MongoDB's capabilities. A yellow 'NEW' badge highlights 'Introducing native support for time series data - Learn more >'. At the bottom, there are two buttons: 'Start Free' and 'Questions? Talk to us >'. The background of the main content area includes a stylized illustration of a document with a blue graph and a yellow folder.

The most popular database for

mongodb.com

Event Interested in speaking at MongoDB World 2022? Become a speaker >>

MongoDB.

NEW Introducing native support for time series data – Learn more >

Build faster. Build smarter.

Get your ideas to market faster with an application data platform built on the leading modern database. With built-in search, analytics, and edge support, accessible with a common query API and built on the data model developers love, MongoDB makes working with data easy – for any use case.

Start Free Questions? Talk to us >



A5.9 MongoDB



MongoDB는 **C++**로 작성된 오픈소스 문서지향(**Document-Oriented**) 적 **Cross-platform** 데이터베이스이며, 뛰어난 확장성과 성능을 자랑합니다. 또한, 현존하는 **NoSQL** 데이터베이스 중 인지도 1위를 유지하고 있습니다.

NoSQL?

흔히 **NoSQL**이라고 해서 아, **SQL**이 없는 데이터베이스구나! 라고 생각 할 수도 있겠지만, 진짜 의미는 **Not Only SQL** 입니다. 기존의 **RDBMS**의 한계를 극복하기 위해 만들어진 새로운 형태의 데이터저장소 입니다. **관계형 DB**가 아니므로, **RDMS**처럼 고정된 스키마 및 **JOIN**이 존재하지 않습니다.

Document?

Document Oriented 데이터베이스라는데.. 여기서 말하는 **Document**가 뭘까요? 문서? 이게 그냥 '문서'로 번역해버리면 조금은 애매합니다. 문서라고 하면 보통 워드/엑셀에 사용되는 그런 문서가 떠오르는데요, 그것과는 다릅니다. **Document**는 **RDMS**의 **record**와 비슷한 개념인데요, 이의 데이터 구조는 한개이상의 **key-value pair**으로 이루어져 있습니다. **MongoDB** 샘플 **Document**를 확인 해 볼까요?

```
{ "_id": ObjectId("5099803df3f4948bd2f98391"),
```

```
"username": "velopert",
```

```
"name": { first: "M.J.", last: "Kim" } }
```



A5.9 MongoDB



여기서 **_id, username, name** 은 **key** 이고 그 오른쪽에 있는 값들은 **value** 입니다.

_id 는 12bytes의 hexadecimal 값으로서, 각 document의 유일함(uniqueness)을 제공합니다.

이 값의 첫 4bytes 는 현재 timestamp, 다음 3bytes 는 machine id, 다음 2bytes 는 MongoDB 서버의 프로세스id, 마지막 3bytes 는 순차번호입니다 추가될때마다 값이 높아진다는거지요.

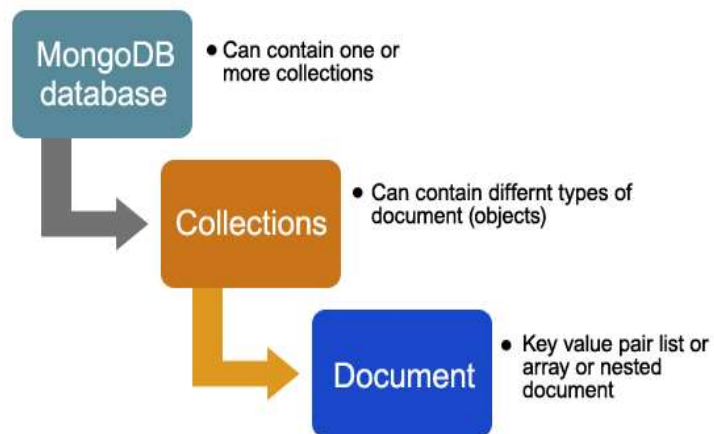
Document는 동적(dynamic)의 schema 를 갖고있습니다. 같은 Collection 안에 있는 Document 끼리 다른 schema 를 갖고 있을 수 있는데요, 쉽게 말하면 서로 다른 데이터 (즉 다른 key) 들을 가지고 있을 수 있습니다.

Collection?

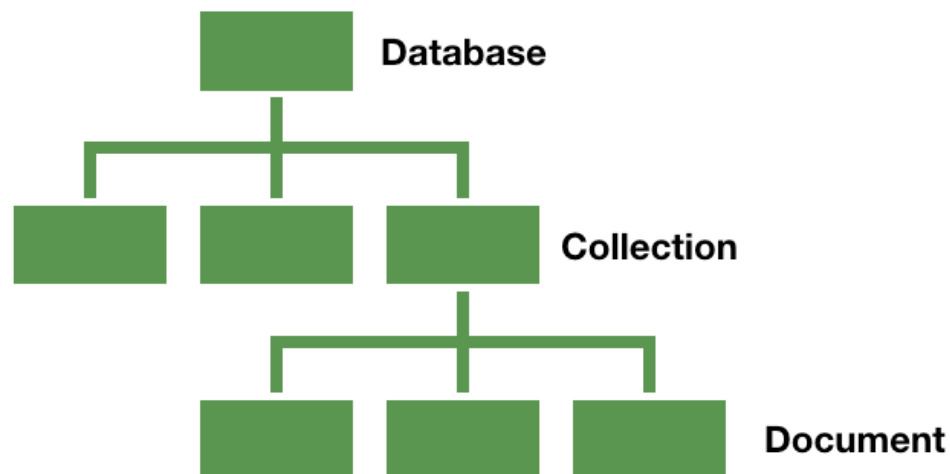
Collection은 MongoDB Document의 그룹입니다. Document들이 Collection 내부에 위치하고 있습니다. RDMS의 table 과 비슷한 개념이지만 RDMS와 달리 schema 를 따로 가지고 있지않습니다. Document 부분설명에 나와있듯이 각 Document들이 동적인 schema 를 가지고 있으니까요

Database?

Database는 Collection들의 물리적인 컨테이너입니다. 각 Database는 파일시스템에 여러파일들로 저장됩니다.



<https://cdn.educba.com/academy/wp-content/uploads/2019/04/MongoDB-chart2.jpg>



<https://i.imgur.com/Att4uVC.png>

MongoDB Community Download

mongodb.com/try/download/community

MongoDB

Atlas
MongoDB as a service

On-premises
MongoDB locally

Tools
Boost productivity

Mobile & Edge
Realm Datastore

MongoDB Enterprise Server

MongoDB Community Server

The Community version of our distributed database offers a flexible document data model along with support for ad-hoc queries, secondary indexing, and real-time aggregations to provide powerful ways to access and analyze your data.

The database is also offered as a fully-managed service with MongoDB Atlas. Get access to advanced functionality such as auto-scaling, serverless instances (in preview), full-text search, and data distribution across regions

Available Downloads

Version: 4.2.17

Platform: Windows

Package: msi

[Download](#) [Copy Link](#)

[Current releases & packages](#)

<https://www.mongodb.com/try/download/community>

MongoDB 4.2.11 2008R2Plus SSL (64 bit) Service Customi... — □ ×

Service Configuration

Specify optional settings to configure MongoDB as a service.

☒ Install MongoD as a Service

☒ Run service as Network Service user

☐ Run service as a local or domain user:

Account Domain:

Account Name:

Account Password:

Service Name:

Data Directory:

Log Directory:

< Back Next > Cancel

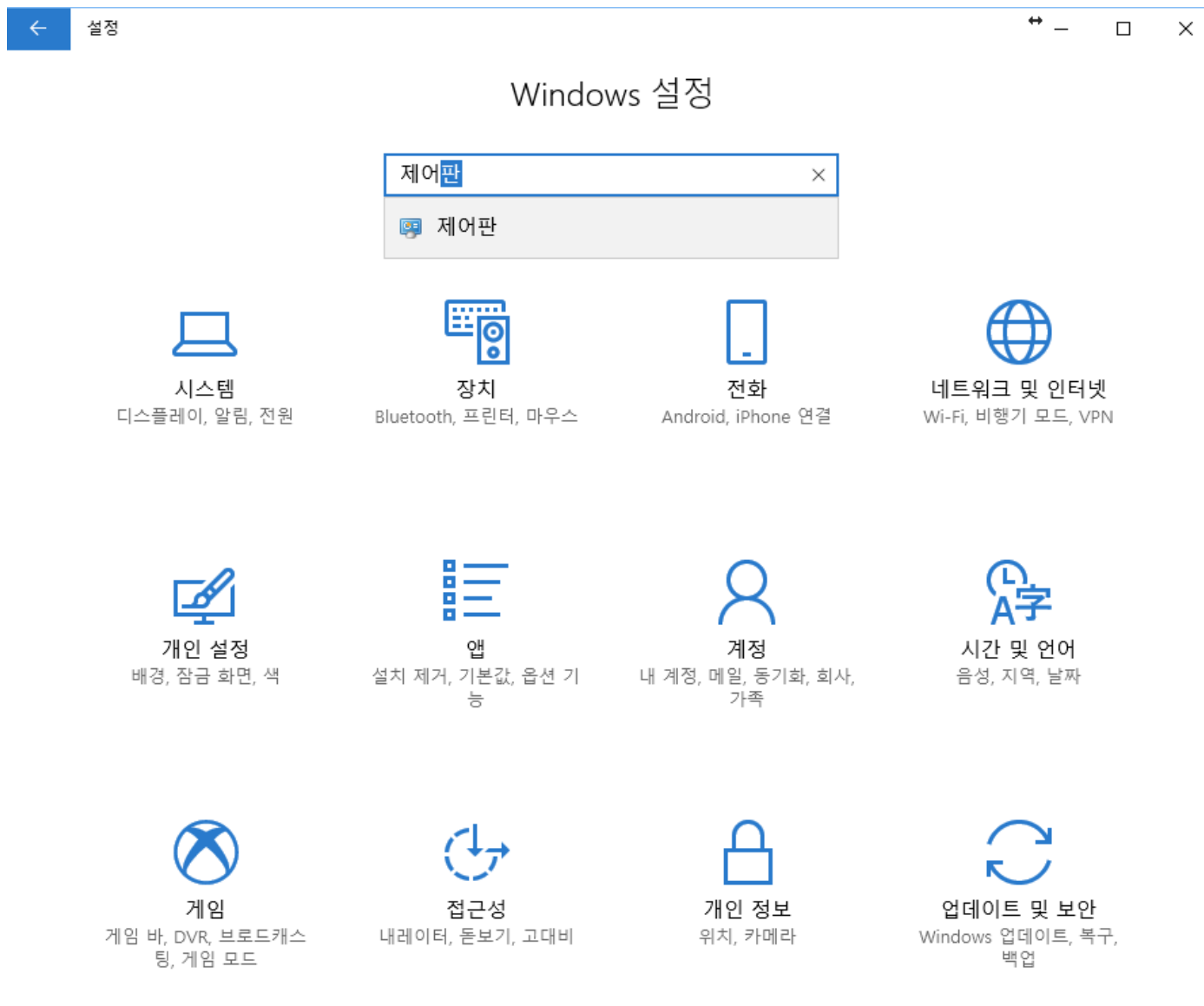
A5.9.1 MongoDB install - 1



윈도우10: 설정 > 시스템 > 정보

[중요] 시스템 환경변수 : **PATH** 에 경로 추가

C:\Program Files\MongoDB\Server\4.2\bin





A5.9.1 MongoDB install – 3

윈도우10: 설정 > ‘제어판’ 검색 > 모든 제어판 항목에서 ‘시스템’ 선택
> 고급 시스템 설정

시스템

← → ▾ ▴ [시스템] > 제어판 > 모든 제어판 항목 > 시스템 ▾ [제어판 검색] 🔍

제어판 홈

- 장치 관리자
- 원격 설정
- 시스템 보호
- 고급 시스템 설정**

컴퓨터에 대한 기본 정보 보기

Windows 버전

Windows 10 Home

© 2017 Microsoft Corporation. All rights reserved.

시스템

프로세서: Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz 3.40 GHz

설치된 메모리(RAM): 32.0GB

시스템 종류: 64비트 운영 체제, x64 기반 프로세서

펜 및 터치: 이 디스플레이에 사용할 수 있는 펜 또는 터치식 입력이 없습니다.

컴퓨터 이름, 도메인 및 작업 그룹 설정

컴퓨터 이름: yish-HCit

전체 컴퓨터 이름: yish-HCit

컴퓨터 설명:

작업 그룹: WORKGROUP

Windows 정품 인증

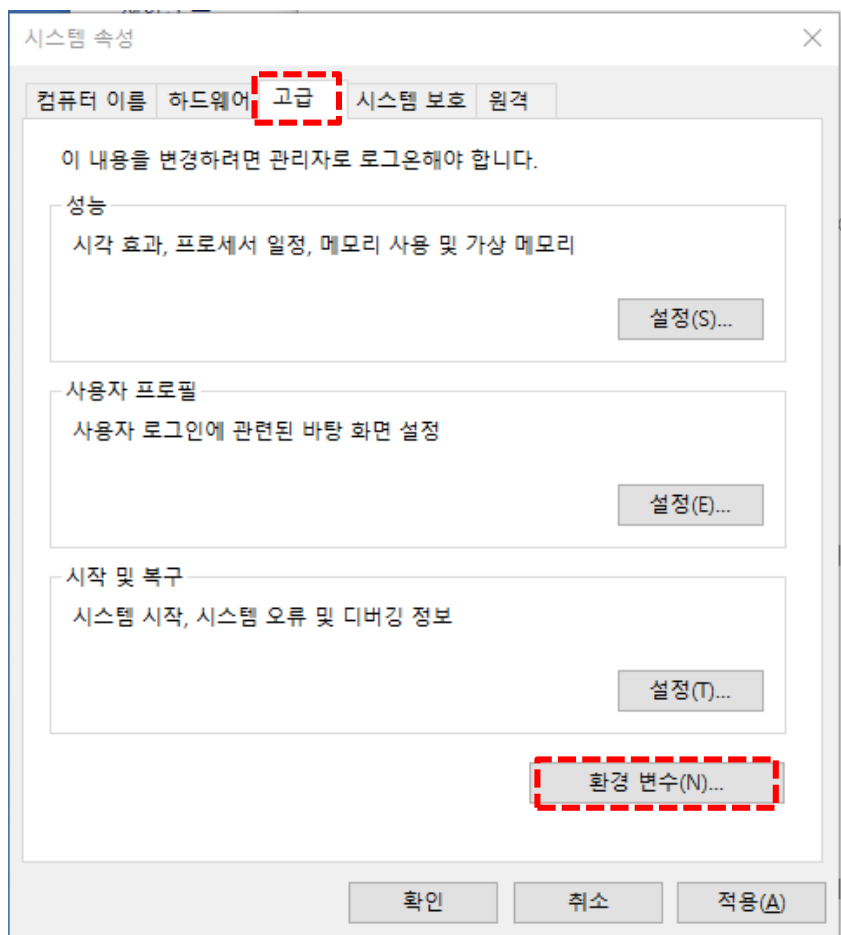
Windows 정품 인증을 받았습니다. [Microsoft 소프트웨어 사용 조건 읽기](#)

제품 ID: 00325-96080-39821-AAOEM

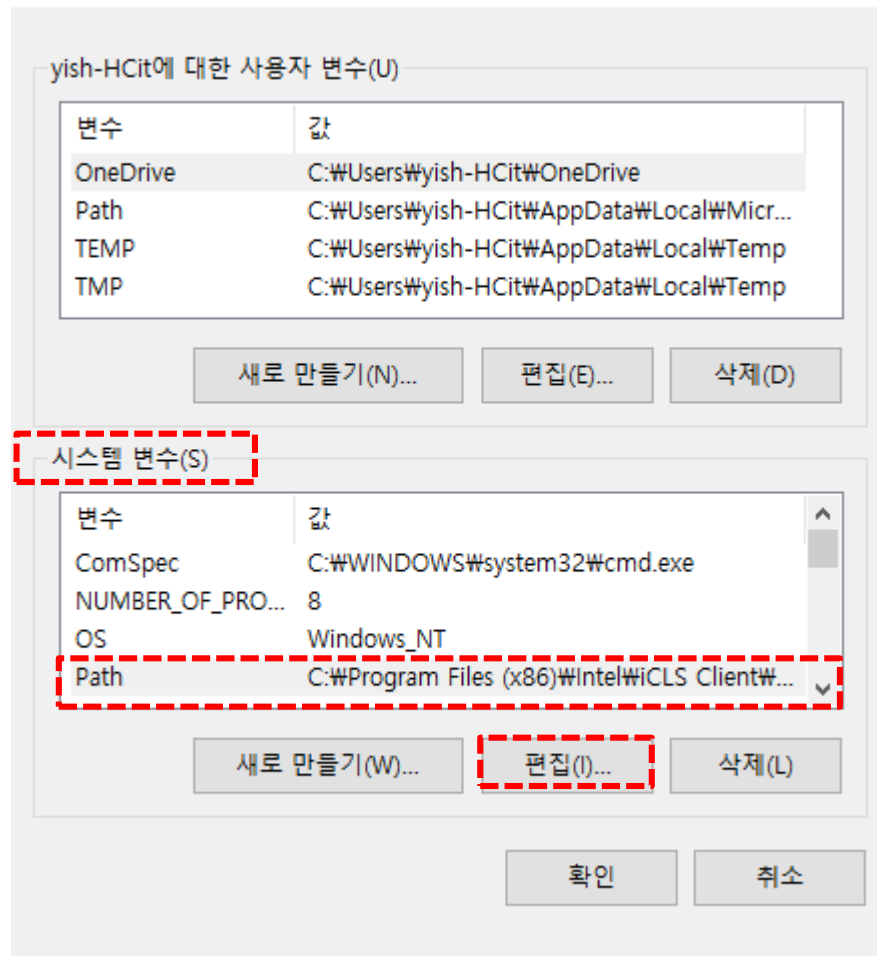
참고 항목

보안 및 유지 관리

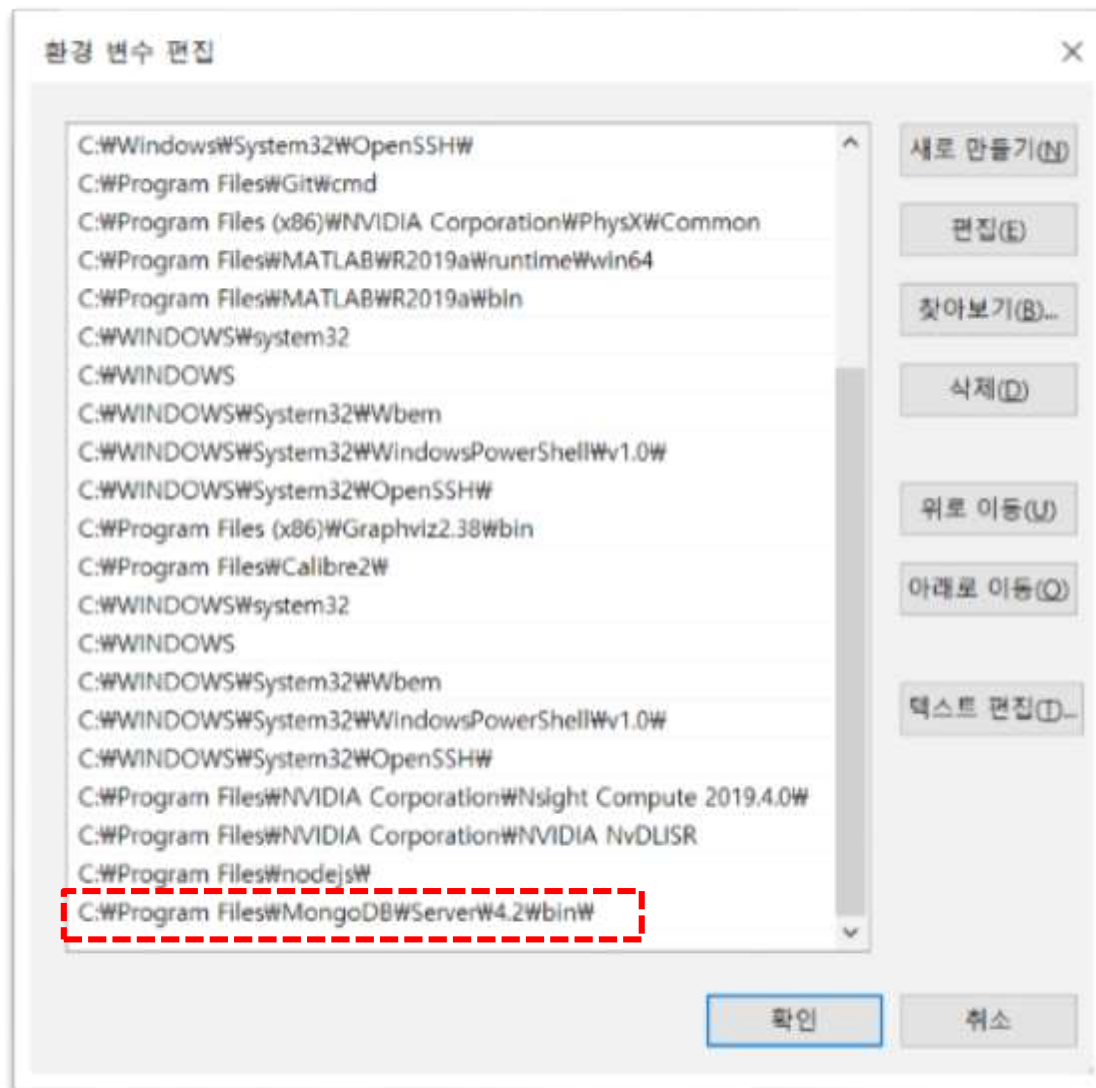
환경 변수 설정



환경 변수



환경 변수 추가





A5.9.2 MongoDB shell - 1

1. Mongo shell 실행

> mongo

C:\ 명령 프롬프트 - mongo

```
D:\>mongo
MongoDB shell version v4.2.11-rc1
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("862dc45b-18a7-4f26-8006-b28d58799e64") }
MongoDB server version: 4.2.11-rc1
Server has startup warnings:
2020-11-18T09:43:57.884+0900 I CONTROL [initandlisten]
2020-11-18T09:43:57.884+0900 I CONTROL [initandlisten] ** WARNING: Access control
is not enabled for the database.
2020-11-18T09:43:57.884+0900 I CONTROL [initandlisten] **           Read and write
access to data and configuration is unrestricted.
2020-11-18T09:43:57.885+0900 I CONTROL [initandlisten]
```

If, Connect failed... → DB 데몬 실행

2. MongoDB 저장소 만들기 → D drive

- **md mongodb**
- **cd mongodb**
- **dir**
- **md data**
- **md log**
- **dir**

```

명령 프롬프트
D:\mongodb>dir
D 드라이브의 볼륨: DATA
볼륨 일련 번호: 82D1-4852

D:\mongodb 디렉터리

2020-11-18 오전 09:39 <DIR> .
2020-11-18 오전 09:39 <DIR> ..
2020-11-18 오전 10:11 <DIR> data
2020-11-18 오전 09:39 <DIR> log
0개 파일
4개 디렉터리 2,332,408,369,152 바이트 남음

D:\mongodb>
  
```

사용 PC 환경에 맞게 실행 (특히, 경로 지정)

실습실 환경에 맞춰서 D:에 **mongodb data** 폴더 지정

3. Run MongoDB by using **mongod.exe**

➤ **mongod --dbpath d:\mongodb\data**

명령 프롬프트 - mongod -dbpath d:\mongodb\data

D:\mongodb>md data

D:\mongodb>**mongod -dbpath d:\mongodb\data**

```
2018-01-22T19:27:32.931-0700 I CONTROL [initandlisten] MongoDB starting : pid=18820 port=27017
dbpath=d:\mongodb\data 64-bit host=yish-HCit
2018-01-22T19:27:32.931-0700 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2
008 R2
2018-01-22T19:27:32.932-0700 I CONTROL [initandlisten] db version v3.6.2
2018-01-23T11:27:33.699+0900 I COMMAND [initandlisten] setting featureCompatibilityVersion to
3.6
2018-01-23T11:27:33.706+0900 I STORAGE [initandlisten] createCollection: local.startup_log wit
h generated UUID: 06b3b7cb-62fe-4be5-a929-2a7478650a9b
2018-01-23T11:27:34.211+0900 I FTDC [initandlisten] Initializing full-time diagnostic data
capture with directory 'd:/mongodb/data/diagnostic.data'
2018-01-23T11:27:34.215+0900 I NETWORK [initandlisten] waiting for connections on port 27017
```

사용 **PC** 환경에 맞게 실행 (특히, 경로 지정)

4. Run mongo shell : `mongo.exe` [use new cmd or Terminal]

➤ mongo

Run in Terminal

mongo

```

D:\aann\aaann-rpt10>mongo
MongoDB shell version v4.2.11-rc1
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongod
Implicit session: session { "id" : UUID("ef9fb07b-cc3b-49ed-8a49-2df43ec1200a") }
MongoDB server version: 4.2.11-rc1
Server has startup warnings:
2021-10-24T18:59:43.599+0900 I CONTROL [initandlisten]
2021-10-24T18:59:43.599+0900 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2021-10-24T18:59:43.599+0900 I CONTROL [initandlisten] **           Read and write access to data and configuration is unrestricted.
2021-10-24T18:59:43.599+0900 I CONTROL [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
>

```



A5.9.2 MongoDB shell - 5

5. mongo shell :

Run new cmd

mongo

show dbs

use local

show
collections

help

exit

명령 프롬프트 - mongo

```
> show dbs
admin 0.000GB
config 0.000GB
local 0.000GB
> use local
switched to db local
> show collections
startup_log
> help
```

```
db.help()
db.mycoll.help()
sh.help()
rs.help()
help admin
help connect
help keys
help misc
help mr
```

```
help on db methods
help on collection methods
sharding helpers
replica set helpers
administrative help
connecting to a db help
key shortcuts
misc things to know
mapreduce
```

```
show dbs
show collections
show users
show profile
```

```
show database names
show collections in current database
show users in current database
show most recent system.profile entries with time >=
```

lms

```
show logs
show log [name]
```

```
show the accessible logger names
prints out the last segment of log in memory, 'global'
```

is default

```
use <db_name>
db.foo.find()
db.foo.find( { a : 1 } )
it
```

```
set current database
list objects in collection foo
list objects in foo where a == 1
result of the last line evaluated; use to further ite
```

rate

```
DBQuery.shellBatchSize = x
exit
```

```
set default number of items to display on shell
quit the mongo shell
```



A5.9.3 MongoDB shell coding

1. make my own db (aann) & insert one record (document)

use aa00

show collections

insert record with new collection "user"

db.user.insert({first:"Redwoods", last:"Yi"})

show collections

→ "user"

show dbs

db.user.find()

```
명령 프롬프트 - mongo
> use aa00
switched to db aa00
> show collections
> db.user.insert({first:"Redwoods", last:"Yi"})
WriteResult({ "nInserted" : 1 })
> show collections
user
> show dbs
aa00      0.000GB
admin     0.000GB
config    0.000GB
local     0.000GB
> _
```



A5.9.3 MongoDB shell coding

2. insert more records with different schema & show records

insert record2

insert record3

show collections

db.user.find()

db.user.find().pretty()

```
명령 프롬프트 - mongo
> db.user.insert({first:"Chaos", last:"Kim"})
WriteResult({"nInserted" : 1})
> db.user.insert({first:"Gildong", last:"Hong"})
WriteResult({"nInserted" : 1})
> show collections
user
> db.user.find()
{ "_id" : ObjectId("5a66b44b9f0d55608f5f7582"), "first" : "Redwoods", "last" : "Yi" }
{ "_id" : ObjectId("5a66b5759f0d55608f5f7583"), "first" : "Chaos", "last" : "Kim" }
{ "_id" : ObjectId("5a66b5869f0d55608f5f7584"), "first" : "Gildong", "last" : "Hong" }
> db.user.find().pretty()
{
  "_id" : ObjectId("5a66b44b9f0d55608f5f7582"),
  "first" : "Redwoods",
  "last" : "Yi"
}
{
  "_id" : ObjectId("5a66b5759f0d55608f5f7583"),
  "first" : "Chaos",
  "last" : "Kim"
}
{
  "_id" : ObjectId("5a66b5869f0d55608f5f7584"),
  "first" : "Gildong",
  "last" : "Hong"
}
>
```

_id 는 12bytes의 hexadecimal 값으로서, 각 document의 유일함(uniqueness)을 제공합니다.

이 값의 첫 4bytes는 현재 timestamp, 다음 3bytes는 machine id, 다음 2bytes는 MongoDB 서버의 프로세스id, 마지막 3bytes는 순차번호입니다.



A5.9.3 MongoDB shell coding

3. insert more records with different schema & show records

insert record4
with firstName key

`db.user.find()`

`db.user.find().pretty()`

```
명령 프롬프트 - mongo
> db.user.insert({firstName:"Fractal", last:"Park"})
WriteResult({"nInserted": 1})
> db.user.find().pretty()
{
  "_id" : ObjectId("5a66b44b9f0d55608f5f7582"),
  "first" : "Redwoods",
  "last" : "Yi"
}
{
  "_id" : ObjectId("5a66b5759f0d55608f5f7583"),
  "first" : "Chaos",
  "last" : "Kim"
}
{
  "_id" : ObjectId("5a66b5869f0d55608f5f7584"),
  "first" : "Gildong",
  "last" : "Hong"
}
{
  "_id" : ObjectId("5a66b6439f0d55608f5f7585"),
  "firstName" : "Fractal",
  "last" : "Park"
}
>
```

**Dynamic
schema**

동적스키마

Note that there are two kinds of schemas in JSON.
Save as

[AAnn_mongo_schemas.png](#)

4. remove one of records (or documents)

remove record3

`db.user.find().pretty()`

명령 프롬프트 - mongo

```
> db.user.remove({last:"Kim"})
writeResult({ nRemoved : 1 })
> db.user.find().pretty()
{
  "_id" : ObjectId("5a66b44b9f0d55608f5f7582"),
  "first" : "Redwoods",
  "last" : "Yi"
}
{
  "_id" : ObjectId("5a66b5869f0d55608f5f7584"),
  "first" : "Gildong",
  "last" : "Hong"
}
{
  "_id" : ObjectId("5a66b6439f0d55608f5f7585"),
  "firstName" : "Fractal",
  "last" : "Park"
}
>
```

5. update a record

```
db.user.update({last:"Hong"},{$set:{first:"GilDong", age:21}})
```

update record2

```
db.user.find().pretty()
```

명령 프롬프트 - mongo

```
> db.user.update({last:"Hong"},{$set:{first:"GilDong", age:21}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.user.find().pretty()
{
  "_id" : ObjectId("5a66b44b9f0d55608f5f7582"),
  "first" : "Redwoods",
  "last" : "Yi"
}
{
  "_id" : ObjectId("5a66b5869f0d55608f5f7584"),
  "first" : "GilDong",
  "last" : "Hong",
  "age" : 21
}
{
  "_id" : ObjectId("5a66b6439f0d55608f5f7585"),
  "firstName" : "Fractal",
  "last" : "Park"
}
> _
```

Note that it is possible to change schema.
Save as

[AAnn_mongo_update.png](#)



Node.js



+

MongoDB





A5.9.4 MongoDB + Node.js : mongoose

Fork me on GitHub

mongoose

elegant **mongodb** object modeling for **node.js**

read the docs

discover plugins

 Star

Version 6.0.12

 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');


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

<http://mongoosejs.com/>






A5.9.4 MongoDB + Node.js : mongoose

 [Features](#) [Business](#) [Explore](#) [Marketplace](#) [Pricing](#) [This repository](#)


Automatic / 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 releases

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 : 6.0.12)

```
문제   출력   디버그 콘솔   터미널
cmd + v  [icon] [icon] [icon] [icon]

D:\aann\aann-rpt10\cds_dht22>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@6.0.12
added 24 packages from 60 contributors, removed 10 packages and audited 68
packages in 2.941s

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

found 0 vulnerabilities
```



A5.9.4 MongoDB + Node.js : mongoose

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

- cds_dht22 project
- New file: dbtest.js
- Run:
node dbtest

```
aann > aann-rpt10 > cds_dht22 > JS dbtest.js > ...  
1 // dbtest.js  
2 var mongoose = require("mongoose");  
3 mongoose.connect("mongodb://localhost:27020/test", {  
4   useNewUrlParser: true,  
5   useUnifiedTopology: true,  
6 });  
7  
8 var SensorSchema = new mongoose.Schema({  
9   data: String,  
10  created: Date,  
11 });  
12  
13 // data model  
14 var Sensor = mongoose.model("Sensor", SensorSchema);  
15  
16 var sensor1 = new Sensor({ data: "124", created: new Date() });  
17 sensor1.save();  
18  
19 var sensor2 = new Sensor({ data: "573", created: new Date() });  
20 sensor2.save();  
21  
22 console.log("Sensor data were saved in MongoDB");
```

Sensor data were saved in MongoDB



A5.9.4 MongoDB + Node.js : mongoose

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

Mongo shell

> show dbs

> use test

> show collections

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

```
mongo  cmd  JS dbtest.js
> show dbs
admin    0.000GB
config  0.000GB
iot      0.000GB
local    0.000GB
test     0.000GB
test2    0.000GB
> use test
switched to db test
> show collections
sensors
> db.sensor.find()
> db.sensors.find()
{ "_id" : ObjectId("6180e44fbc986ac8a5297195"), "data" : "124", "created" : ISODate("2021-11-02T07:10:07.689Z"), "__v" : 0 }
{ "_id" : ObjectId("6180e44fbc986ac8a5297196"), "data" : "573", "created" : ISODate("2021-11-02T07:10:07.692Z"), "__v" : 0 }
>
```




A5.9.4 MongoDB + Node.js : mongoose

4. dbtest2.js

./ aann > aann-rpt10 > cds_dht22 > JS dbtest2.js > getDateString

```
1 // dbtest2.js
2 var mongoose = require("mongoose");
3 mongoose.connect("mongodb://localhost/test2", {
4   useNewUrlParser: true,
5   useUnifiedTopology: true,
6 });
7 var SensorSchema = new mongoose.Schema({
8   data: String,
9   created: String,
10 });
11 // data model
12 var Sensor = mongoose.model("Sensor", SensorSchema);
13
14 var sensor1 = new Sensor({ data: "124", created: getDateString() });
15 sensor1.save();
16 var sensor2 = new Sensor({ data: "573", created: getDateString() });
17 sensor2.save();
18
19 console.log("[dbtest2.js]: Sensor data were saved in MongoDB");
20 // helper function to get a nicely formatted date string
21 function getDateString() {
22   var time = new Date().getTime();
23   // 32400000 is (GMT+9 Korea, GimHae)
24   // for your timezone just multiply +/-GMT by 3600000
25   var datestr = new Date(time + 32400000)
26     .toISOString()
27     .replace(/T/, " ")
28     .replace(/Z/, "");
29   return datestr;
30 }
```




A5.9.4 MongoDB + Node.js : mongoose

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

Mongo shell

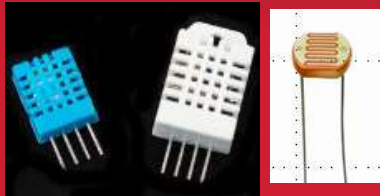
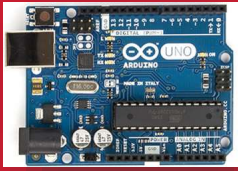
> show dbs

> use test2

> show collections

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

```
> show dbs
admin    0.000GB
config  0.000GB
iot      0.000GB
local    0.000GB
test     0.000GB
test2    0.000GB
> use test2
switched to db test2
> db.sensors.find().pretty()
{
  "_id" : ObjectId("6181d4c4501d6bc94cfa6903"),
  "data" : "124",
  "created" : "2021-11-03 09:16:04.448",
  "__v" : 0
}
{
  "_id" : ObjectId("6181d4c4501d6bc94cfa6904"),
  "data" : "573",
  "created" : "2021-11-03 09:16:04.451",
  "__v" : 0
}
>
```



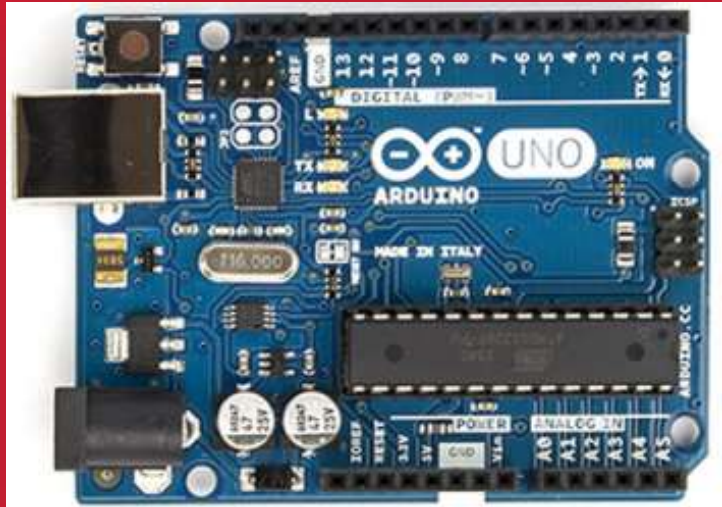
MongoDB from Arduino with node.js & mongoose

```
> 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
>
```

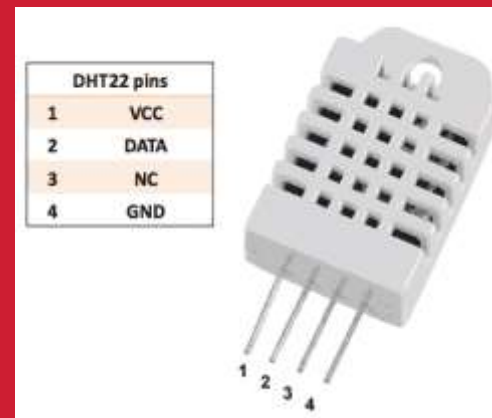
```
mongo db connection OK.
iotInfo: Current date: 2021-11-03 10:52:19.797, Temp: 23.5, Humi: 40.5, Lux: 84
iotInfo: Current date: 2021-11-03 10:52:22.074, Temp: 23.5, Humi: 40.5, Lux: 51
iotInfo: Current date: 2021-11-03 10:52:24.352, Temp: 23.5, Humi: 40.5, Lux: 81
iotInfo: Current date: 2021-11-03 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
```

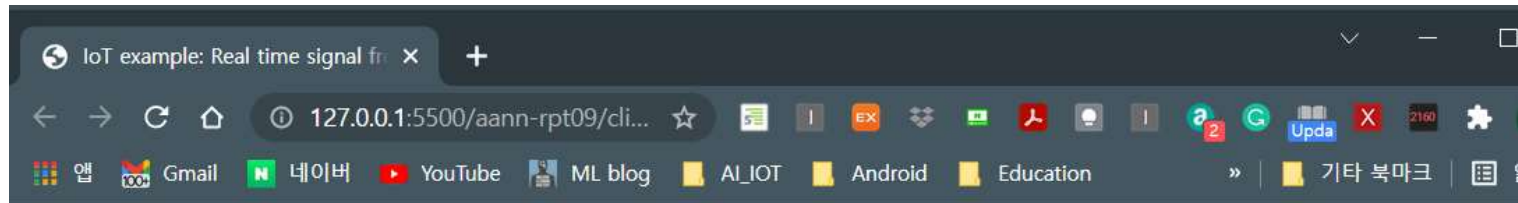
☒ 자동 스크롤 ☐ 타임스탬프



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  
█
```


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

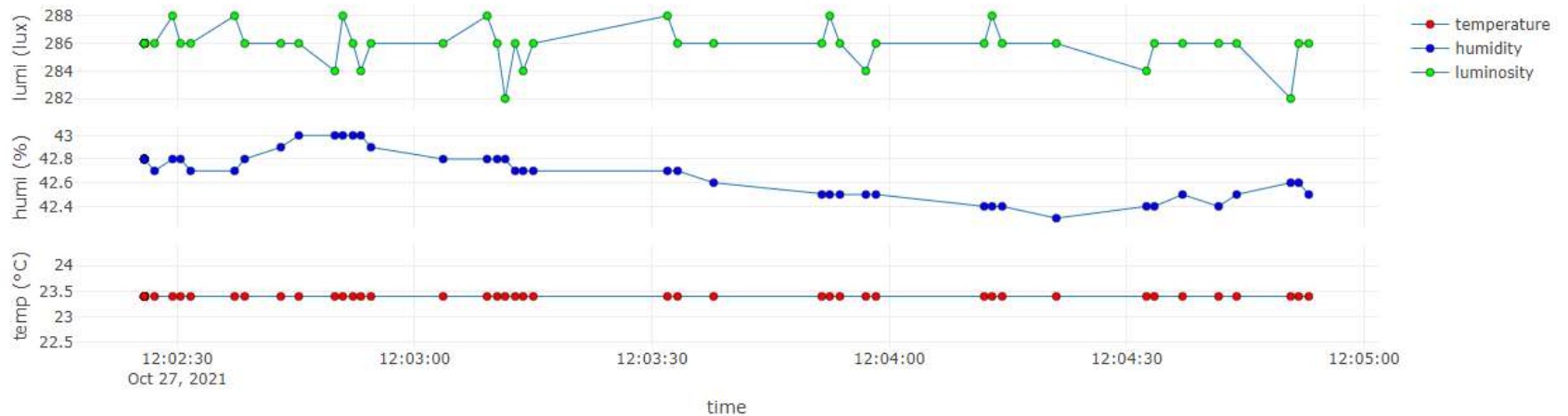
**Real-time monitoring of signals from Arduino
CdS + DHT22 circuit**

WEB client : client_cds_dht22.html

Real-time Weather Station from sensors



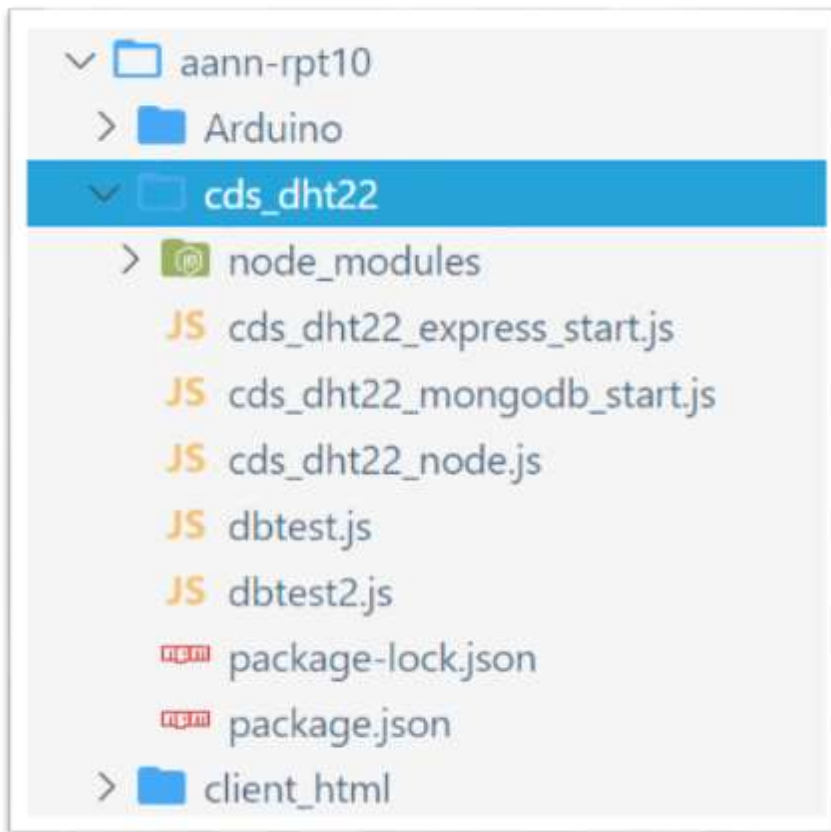
on Time: 2021-10-27 12:04:53.016





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

1. 작업 폴더 구조 [2021]





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

```
27 iotSchema.methods.info = function () {
28     var iotInfo = this.date
29     ? "Current date: " + this.date + ", Temp: " + this.temperature
30     + ", Humi: " + this.humidity + ", Lux: " + this.luminosity
31     : "I don't have a date"
32     console.log("iotInfo: " + iotInfo);
33 }
34
35 // serial port object
36 var sp = new serialport(portName,{
37     baudRate: 9600,    // 9600  38400
38     dataBits: 8,
39     parity: 'none',
40     stopBits: 1,
41     flowControl: false,
42     parser: serialport.parsers.readline('\r\n') // new serialport.parsers
43 });
44
45 var readData = ''; // this stores the buffer
46 var temp = '';
47 var humi = '';
48 var lux = '';
49 var mdata = []; // this array stores date and data from multiple sensors
50 var firstcommaidx = 0;
51
52 var Sensor = mongoose.model("Sensor", iotSchema); // sensor data model
```



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

2.3 cds_dht22_mongodb.js

```
54 sp.on('data', function (data) { // call back when data is received
55   readData = data.toString(); // append data to buffer
56   firstcommaidx = readData.indexOf(',');
57
58   // parsing data into signals
59   if (readData.lastIndexOf(',') > firstcommaidx && firstcommaidx > 0) {
60     temp = readData.substring(firstcommaidx + 1, readData.indexOf(',', firstcommaidx+1));
61     humi = readData.substring(readData.indexOf(',', firstcommaidx+1) + 1, readData.lastIndexOf(','));
62     lux = readData.substring(readData.lastIndexOf(',')+1);
63
64     readData = '';
65
66     dStr = getDateString();
67     mdata[0]=dStr; // Date
68     mdata[1]=temp; // temperature data
69     mdata[2]=humi; // humidity data
70     mdata[3]=lux; // luminosity data
71     //console.log(mdata);
72     var iot = new Sensor({date:dStr, temperature:temp, humidity:humi, luminosity:lux});
73     // save iot data to MongoDB
74     iot.save(function(err, iot) {
75       if(err) return handleEvent(err);
76       iot.info(); // Display the information of iot data on console.
77     })
78     io.sockets.emit('message', mdata); // send data to all clients
79   } else { // error
80     console.log(readData);
81   }
82 });
```



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

2.4 cds_dht22_mongodb.js

```
85 io.sockets.on('connection', function (socket) {
86     // If socket.io receives message from the client browser then
87     // this call back will be executed.
88     socket.on('message', function (msg) {
89         console.log(msg);
90     });
91     // If a web browser disconnects from Socket.IO then this callback
92     socket.on('disconnect', function () {
93         console.log('disconnected');
94     });
95 });
96
97 // helper function to get a nicely formatted date string
98 function getDateString() {
99     var time = new Date().getTime();
100     // 32400000 is (GMT+9 Korea, GimHae)
101     // for your timezone just multiply +/-GMT by 3600000
102     var datestr = new Date(time + 32400000).
103     toISOString().replace(/T/, ' ').replace(/Z/, '');
104     return datestr;
105 }
```



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

2.5 [Run] node cds_dht22_mongodb.js

mongo

node

JS cds_dht22_mongodb.js

JS dbtest.js

JS dbtest

```
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("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_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>
```




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

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



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

2.2 cds_dht22_express.js

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

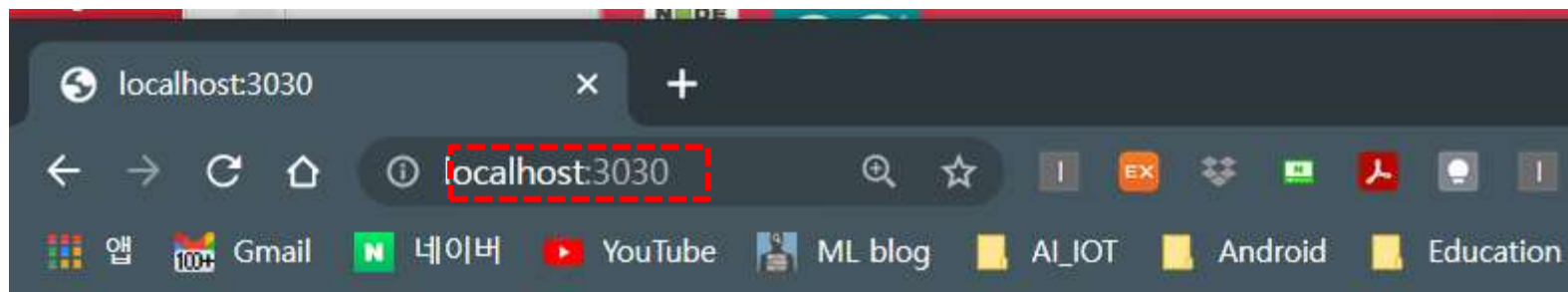
```
(base) D:\aann\aann-rpt10\cds_dht22>node cds_dht22_express
Express_IOT is running at port:3030
mongo db connection OK.
```





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

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



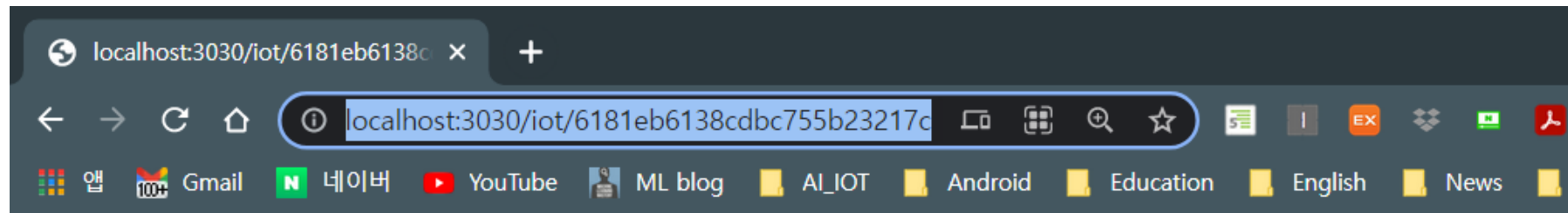
Hello Arduino IOT: express server by AA00!

AAnn_iot_mongodb_web.png



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

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





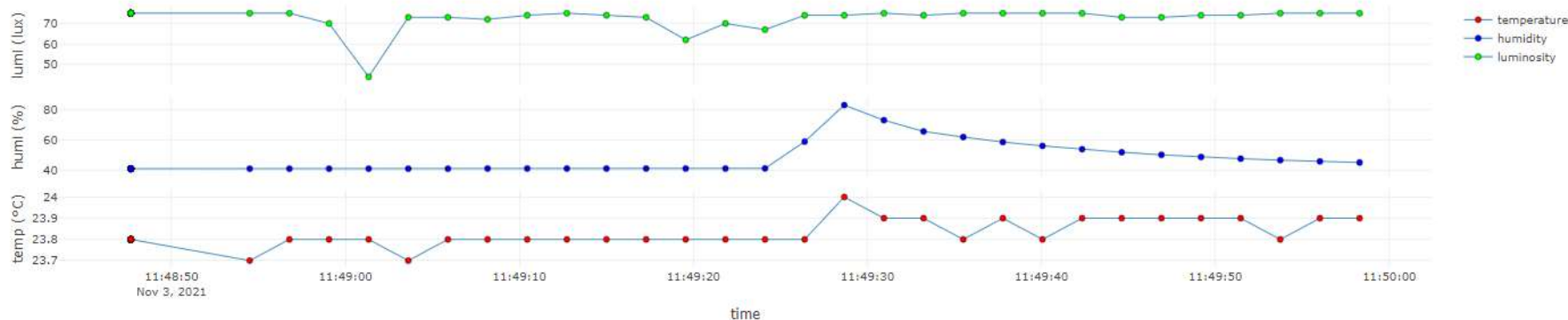
A5.9.6 DHT22 + CdS + Node.js + MongoDB

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



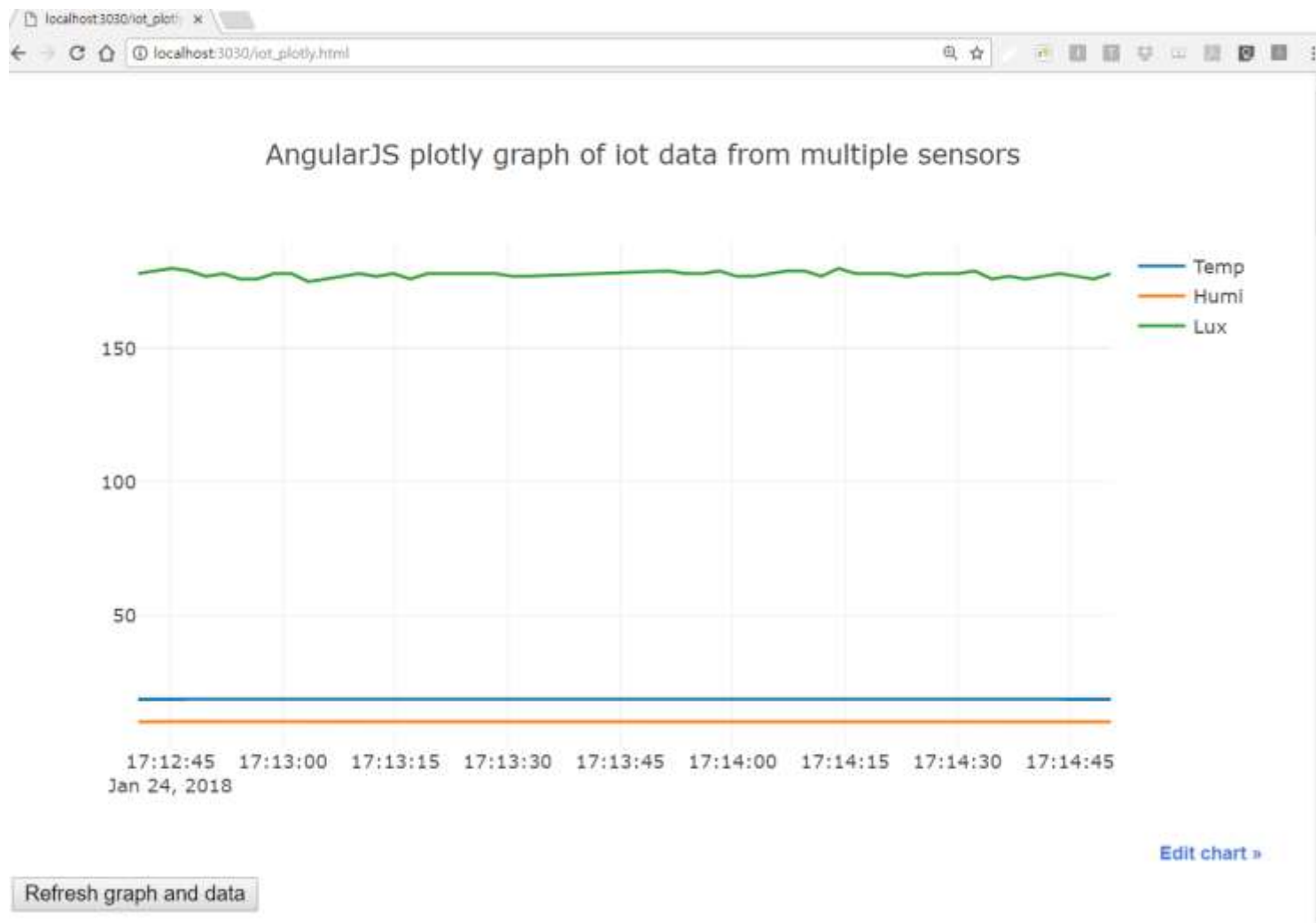
on Time: 2021-11-03 11:49:58.294





DHT22 + CdS + Node.js + MongoDB

[Next week] Web monitoring





DHT22 + CdS + Node.js + MongoDB

[Next week] Web monitoring

chaos.inje.ac.kr:3030/iot3

chaos.inje.ac.kr:3030/iot3.html

MongoDB datab

Time series : Multi ser

chaos.inje.ac.kr:3030 내용:

```
[{"_id":"5aa584d0ea0bd2064cb1f9ab","date":"2018-03-12 04:34:40.662","temperature":"16.6","humidity":"24.9","luminosity":"0", "__v":0}, {"_id":"5aa584daea0bd2064cb1f9ac","date":"2018-03-12 04:34:50.923","temperature":"16.6","humidity":"24.9","luminosity":"0", "__v":0}, {"_id":"5aa584e5ea0bd2064cb1f9ad","date":"2018-03-12 04:35:01.168","temperature":"16.6","humidity":"24.9","luminosity":"0", "__v":0}, {"_id":"5aa584efea0bd2064cb1f9ae","date":"2018-03-12 04:35:11.429","temperature":"16.6","humidity":"24.9","luminosity":"0", "__v":0}, {"_id":"5aa584f9ea0bd2064cb1f9af","date":"2018-03-12 04:35:21.674","temperature":"16.6","humidity":"24.9","luminosity":"0", "__v":0}]
```

확인

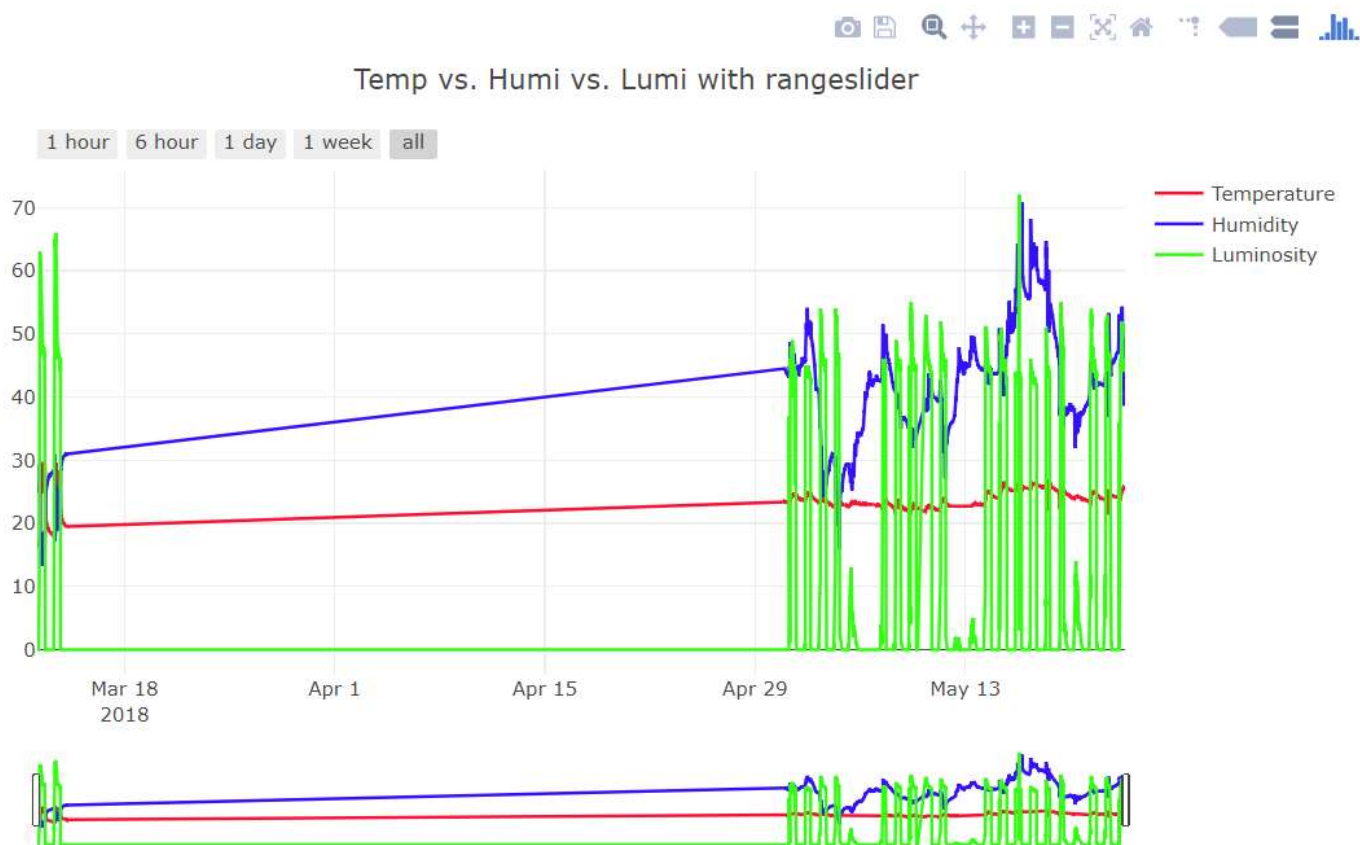


DHT22 + CdS + Node.js + MongoDB

[Next week] Web monitoring

MongoDB database visualization by AA00

Time series : Multi sensor data

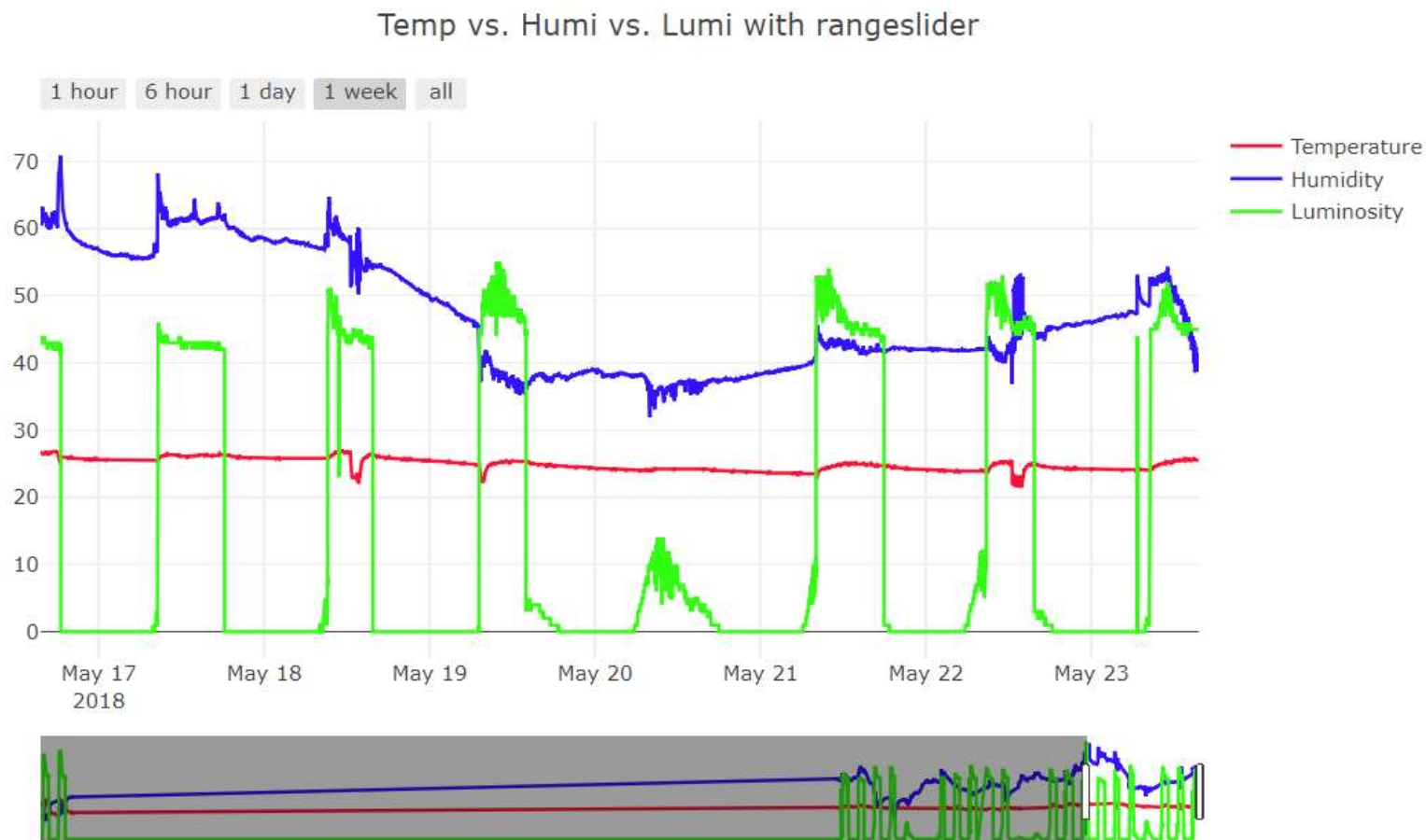




DHT22 + CdS + Node.js + MongoDB

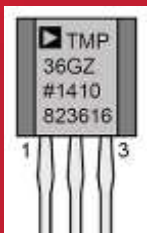
[Next week] Web monitoring

Time series : Multi sensor data



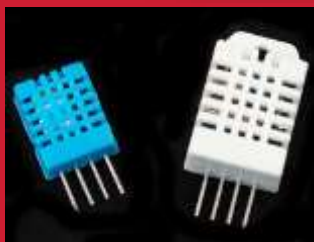


[Practice]



◆ [wk10]

- RT Data storaging with MongoDB
- Multi-sensor circuits (cds-dht22)
- Complete your project
- Upload folder: aann-rpt10



wk10 : Practice : aann-rpt10

◆ [Target of this week]

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

제출폴더명 : **aann-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**

● 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

