# Lab Services Web

## Prérequis

Avant de lancer le serveur du lab, il est nécessaire d'avoir plusieures applications fonctionnelles.

### Installation de node et npm

Le projet étant développé pour un serveur nodejs avec l'assistance du gestionnaire npm, veuillez les installer en vous référant aux sites https://nodejs.org/en/ et https://www.npmjs.com/. Remarque : tous les serveurs nodes requièrent l'utilisation de "npm install" pour être opérationnels.

### Installation de docker et docker-compose

L'utilisation de conteneurs docker pour l'exécution du projet suppose l'utilisation de docker et docker-compose. Pour les installer, veuillez vous référer aux scripts d'installation suivants :
Docker : https://docs.docker.com/engine/installation/
Docker-compose : https://docs.docker.com/compose/install/

### Broker mqtt et simulateur

Le broker mqtt et le simulateur sont embarqués dans un docker. Après avoir cloné le dépôt git, merci de vous référer au guide d'installation à cette adresse : https://github.com/bachrc/random-sensors

### Serveur Mongodb

Le serveur mongodb est lui aussi disponible sous la forme d'un docker que vous pouvez obtenir directement sur DockerHub : https://hub.docker.com/r/_/mongo/
Exécutez la commande : $ docker run --name <nom_docker> -d mongo
nom_docker sera le nom de votre base mongo pour docker

### Lien broker-mongo

Le serveur node chargé de faire le lien entre le broker mqtt et la base mongodb est un projet git disponible à cette addresse : https://github.com/pigne/sensors-to-db

Pour lancer le serveur, utilisez la commande suivante depuis la racine du projet sensors-to-db : node sensors-to-db.js --broker=mqtt://<broker ip:port>
Puisque nous lançons toutes les application sur la même machine, l'ip sera donc 127.0.0.1, pour définir le port du broker mqtt, lancez la commande "docker ps" et repérez le port du conteneur matteolina/mosca, ci-dessous : 32768

```
alekshar@Alekshar-UM:~$ docker ps
CONTAINER ID        IMAGE                          COMMAND                  CREA
TED             STATUS              PORTS
               NAMES
ce66203aa91b        mongo                          "/entrypoint.sh mo..."   15 m
inutes ago      Up 15 minutes       27017/tcp
               mongodb
af2f4550a86d        randomsensors_random-sensors   "node test-random-..."   13 d
ays ago         Up About an hour
               randomsensors_random-sensors_1
09ab6c2900b2        matteocollina/mosca            "/usr/src/app/bin/..."   13 d
ays ago         Up About an hour    8080/tcp, 0.0.0.0:8080->80/tcp, 0.0.0.0:3276
8->1883/tcp    randomsensors_mosca_1
alekshar@Alekshar-UM:~$
```

Lancement du serveur web

Clonez le projet github : https://github.com/Alekshar/IdodLabs/tree/master/LabServicesWeb
Depuis la racine du projet, lancez le serveur node après avoir exécuté “npm install” avec la commande “sudo node server.js <url broker> <url mongo>”.
L’url du broker à indiquer est celle utilisée pour le lien broker-mongo. Si les paramètres ne sont pas spécifiés : mqtt://localhost:32768 et mongodb://localhost:/InternetOfThings sont utilisés par défaut.

Le serveur lancé, vous pouvez accéder à l’application web via http://localhost

**Documentation REST**

GET /sensor/information/:id
:id <= Identifiant du sensor

Réponse
Sensor {
    _id:string
    name:string
    location:string
    type:string
}

POST /sensor/information
Contenu de la requête
{
    id:string  <= Identifiant du sensor
    name:string
    location:string
}

Réponses
200 Success
500 Failed to update

GET /sensor/measures/:id/:from/:to
:id <= identifiant du sensor
:from <= date depuis laquelle récupérer l'historique
:to <= date de fin de la période à récupérer

Réponse
Measures[
    Measure {
        _id:string
        date:string
        sensor:string
        value:string
    }
]