# AQROPOL

## Documentation Utilisateur


*Participants au projet :*

Jean-Baptiste Bordier
Manon Derocles
Mélian Ferrachat
Julien Garnier
Alan Gaubert
Adrien Leblanc
Thibault Leclair
Antoine Leval
Pierre Miola
Bashar Nemeh
Noureddine Kadri
Antoine Posnic
Fanny Prieur
Hantavolaniaina Sylvia Rabe

*Mr François Bodin*
*Enseignant référent du projet AQROPOL*
*Enseignant chercheur à l'université de Rennes 1(IRISA)*

# Table des matières

# Installation et mise en route des capteurs

Le projet Nuc/Capteurs se compose de 5 modules avec des dépendances entre eux. Tout le projet est développé majoritairement en Java et à l'aide de Maven. Le code source du projet est téléchargeable à l'adresse suivante:https://github.com/AQROPOL/aqropolNUC.

## I. Les modules

Le projet parent "aqropolNUC" se compose des quatre sous modules suivants :

"*aqropolNUC/aqropolrest*" : Responsable de la partie REST, communication avec Android (via http). Le mobile peut faire des requêtes GET sur les capteurs, les mesures et connaître l'identifiant du Nuc, il peut également faire une requête POST avec un tableau JSON clé (identifiant du nuc) : valeur (dernier hash connu) afin de supprimer les données déjà collectées et postées sur le serveur.

"*aqropolNUC/aqropolmomProducer*" : Librairie java permettant très simplement d'instancier un producteur AMQP pour poster les données des capteurs sur le broker rabbitMQ

"*aqropolNUC/aqropolmomConsumer*" : Librairie java permettant d'instancier un consommateur amqp pour recevoir les données publiées par les capteurs et les faire persister en base de données.

"*aqropolNUC/measurepmsensor*" : Module permettant de lire les données sortantes de l'Arduino via l'interface USB.

Enfin, il y a aussi un module à part en C++ destiné uniquement à l'Arduino
*"aqropolNUC/Arduino/sensor/novaPMsensor/Module/Arduino"* : permettant de lire les données du capteurs SDS011, et de les envoyer via l'interface USB en json. Cela permet également d'écrire sur un écran LCD les données du capteur en temps réel.

## II. Mise en place de l'environnement

Afin de lancer les modules du projet, celui-ci nécessite quelques prérequis. Assurez vous d'avoir le JDK Java 1.8 minimum ainsi que Maven 4.0. Un serveur postgre ainsi qu'un broker RabbitMQ sera nécessaire. Le reste des dépendances sera téléchargé grâce à Maven.

## A. Base de donnée : Postgre

Si ce n'est pas déjà fait, démarrer un serveur postgres. Créez une base de données (ici 'aqropol', mais cela est modifiable dans les fichiers properties des modules). Pour instancier la base de données, le schéma se trouve à la racine du projet aqropolNUC init.sql.

```
CREATE USER api PASSWORD 'password';
GRANT UPDATE, SELECT, INSERT, DELETE ON sensor, measure, nuc TO api;
GRANT SELECT ON measure TO PUBLIC;
GRANT SELECT, UPDATE, INSERT, DELETE ON measure, sensor, nuc TO api;
```

L'utilisateur créé est 'api', si vous souhaitez changer le nom et le mot de passe de cet utilisateur, les changements devront être répercutés dans le pom parent du projet dans le fichier *aqropolNUC/pom.xml* aux lignes suivantes :

```
<properties>
    <aqropol.datasource.username>api</aqropol.datasource.username>
    <aqropol.datasource.password>password</aqropol.datasource.password>
    <aqropol.datasource.database>aqropol</aqropol.datasource.database>
    <aqropol.datasource.host>localhost</aqropol.datasource.host>
    <aqropol.datasource.port>5432</aqropol.datasource.port>
    …
</properties>
```

À la fin du fichier init.sql se trouve les entrées nécessaires au fonctionnement du projet dans la base de données :

```
insert into nuc (token) values ('token_de_mon_nuc_dev_001');
insert into sensor (name, type, unity) values ('SDS011', 'pm10', 'ug/m3');
insert into sensor (name, type, unity) values ('SDS011', 'pm2.5', 'ug/m3');
```

La première ligne est l'identifiant du Nuc, il doit être unique par rapport au serveur principal. Les deux lignes suivantes sont deux valeurs différentes pour un même capteur, le capteur Nova PM (SDS011). Si le capteur n'est pas présent dans la table sensor alors il ne sera pas possible d'entrer des mesures pour celui-ci.

## B. Broker AMQP : RabbitMQ

Lancer un serveur rabbitMQ avec les paramètres suivants :

```
default_vhost=aqropol
default_user=aqropol
default_pass=64GbL3k7uc33QCtc
management.listener.port=8081
listeners.tcp.default=5672
```

Ou modifiez les paramètres dans le "pom" parent du projet : "*aqropolNUC/pom.xml*" .
Les paramètres de connexion au broker rabbitMQ :

```
<properties>
<aqropol.amqp.host>10.42.0.1</aqropol.amqp.host>
<aqropol.amqp.port>5672</aqropol.amqp.port>
<aqropol.amqp.vhost>aqropol</aqropol.amqp.vhost>
<aqropol.amqp.user>aqropol</aqropol.amqp.user>
<aqropol.amqp.pass>64GbL3k7uc33QCtc</aqropol.amqp.pass>
...
</properties>
```

Les paramètres communs au producteur et au consommateur (même exchange, même queue, même route) :

```
<aqropol.amqp.channel.exchanger>aqropol_sensors</aqropol.amqp.channel.exchanger>
<aqropol.amqp.channel.route>aqropol_sensors_data</aqropol.amqp.channel.route>
<aqropol.amqp.queue>sensors</aqropol.amqp.queue>
```

Et les paramètres de la queue, propre au producer :

```
<aqropol.ampq.queue.durable>true</aqropol.ampq.queue.durable>
<aqropol.amqp.queue.exclusive>false</aqropol.amqp.queue.exclusive>
<aqropol.amqp.queue.autodelete>false</aqropol.amqp.queue.autodelete>
```

# III. Exécution des modules

L'exécution des modules peut être faite dans un ordre quelconque, à l'exception du module "*measurepmsensor*" qui doit être exécuté seulement après avoir branché l'arduino à un port USB de votre machine. Une fois le serveur *postgre* en route on peut lancer le service web "Istinlineaqropolrest" puisque celui-ci se contente de mettre en ligne les ressources stockées dans la base. Lancer le producteur avant le consommateur *AMQP* importe peu puisque les messages seront stockés dans une queue tant qu'ils ne sont pas consommés. Avant de lancer les modules il faut demander à *Maven* de les construire. Placez vous à la racine du projet AqropolNUC (le pom parent). et exécutez la commande suivante :

```
mvn clean package
```

## A. Service REST

Pour démarrer le serveur web, rien de plus simple, il suffit d'éxecuter le jar suivant : aqropolNUC/aqropolrest/target/aqropol-rest-1.0-SNAPSHOT.jar :

```
java -jar aqropolrest/target/aqropol-rest-1.0-SNAPSHOT.jar.
```

Ensuite rendez vous à l'adresse suivante grâce à n'importe quel navigateur internet : *http://localhost:8080/* :


```
{
    "_links" : {
        "sensors" : {
            "href" : "http://localhost:8080/sensors{?page,size,sort,projection}",
            "templated" : true
        },
        "measures" : {
            "href" : "http://localhost:8080/measures{?page,size,sort,projection}",
            "templated" : true
        },
        "nucs" : {
            "href" : "http://localhost:8080/nucs{?page,size,sort}",
            "templated" : true
        },
        "profile" : {
            "href" : "http://localhost:8080/profile"
        }
    }
}
```


Spring Data Rest utilise le modèle HAL (Hypertext Application Language) pour faire des liens entre toutes les ressources proposées par le service REST. Le paramètre "projection" n'est pas utile ici car il n'en existe qu'un seul et c'est également celui par défaut, le paramètre est donc implicite. Vous pouvez également vous rendre à l'adresse suivante : *http://localhost:8080/admin* pour accéder à diverses fonctionnalités. Les identifiants pour se connecter aux ressources d'administration se trouvent dans la classe : *aqropolNUC/aqropolrest/src/main/java/config/WebSecurityConfig.java*

```
@Bean
@Override
public UserDetailsService userDetailsService() {
UserDetails user = User.withDefaultPasswordEncoder()
                      .username("admin")
                      .password("admin")
                      .roles("ADMIN", "USER")
                      .build();
return new InMemoryUserDetailsManager(user);
}
```

## B. RabbitMQ : Producteur

Le module "*aqropolmomProducer*" sert à instancier un nouveau producteur pour le broker rabbitMQ, avec des paramètres par défaut instancié grâce à maven à la compilation. Si les bons paramètres sont entrés dans le *pom* du projet parent *aqropolNUC* alors le producteur et le consomateur seront correctement réglés et il ne restera qu'à instancier un nouveau producteur comme ceci :

```
RoutingProducer.RoutingProducerFactory fact = new RoutingProducer.RoutingProducerFactory();
RoutingProducer producer = fact.build();
producer.send(jsonMessage);
```

Si certains paramètres doivent être changés, la "*factory*" offre quelques méthodes :

```
public void setExchanger(String exchanger);
public void setRoute(String route);
public void setEncoding(String encoding);
```

Ce module doit donc être instancié par tout nouveau capteur souhaitant publier des données (récoltées par le consommateur, et stockées dans la base de donnée).

## C. RabbitMQ : Consommateur

Le module *aqropolmomConsumer* a pour rôle de récolter les données du ou des producteurs. Comme pour le producteur, si les paramètres par défaut du pom parent sont réglés comme il faut, il n'y aura pas besoin de toucher à quoi que ce soit, simplement le démarrer et il fera persister dans la base de données toutes les mesures envoyées par le producteur.

*Voir partie 2.1 et 2.2 pour les paramètres de la base de données et du broker rabbitMQ*

Pour démarrer le consommateur il suffit donc d'exécuter le jar correspondant : aqropolNUC/aqropolmomConsumer/target/mom_consumer-0.0.1-SNAPSHOT.jar

```
java -jar aqropolmomConsumer/target/mom_consumer-0.0.1-SNAPSHOT.jar
```

## D. Capteur : Measure PM Sensor

Le module "*measurepmsensor*" est un module implémentant un producteur rabbitMQ et qui lit les données sortantes de l'Arduino via l'interface USB pour les envoyer au producteur. Avant d'exécuter ce module il vous faut donc brancher en USB l'Arduino et s'assurer que les paramètres du module "*aqropolmomProducer*" soient correctes. Pour l'exécution le module a besoin d'au moins un paramètre. (Le module parse les paramètres grâce à "*commons-cli*" de apache). Vous pouvez par exemple afficher l'aide comme ceci :

```
$java -jar target/measure-pmsensor-1.0-SNAPSHOT-jar-with-dependencies.jar -h
```

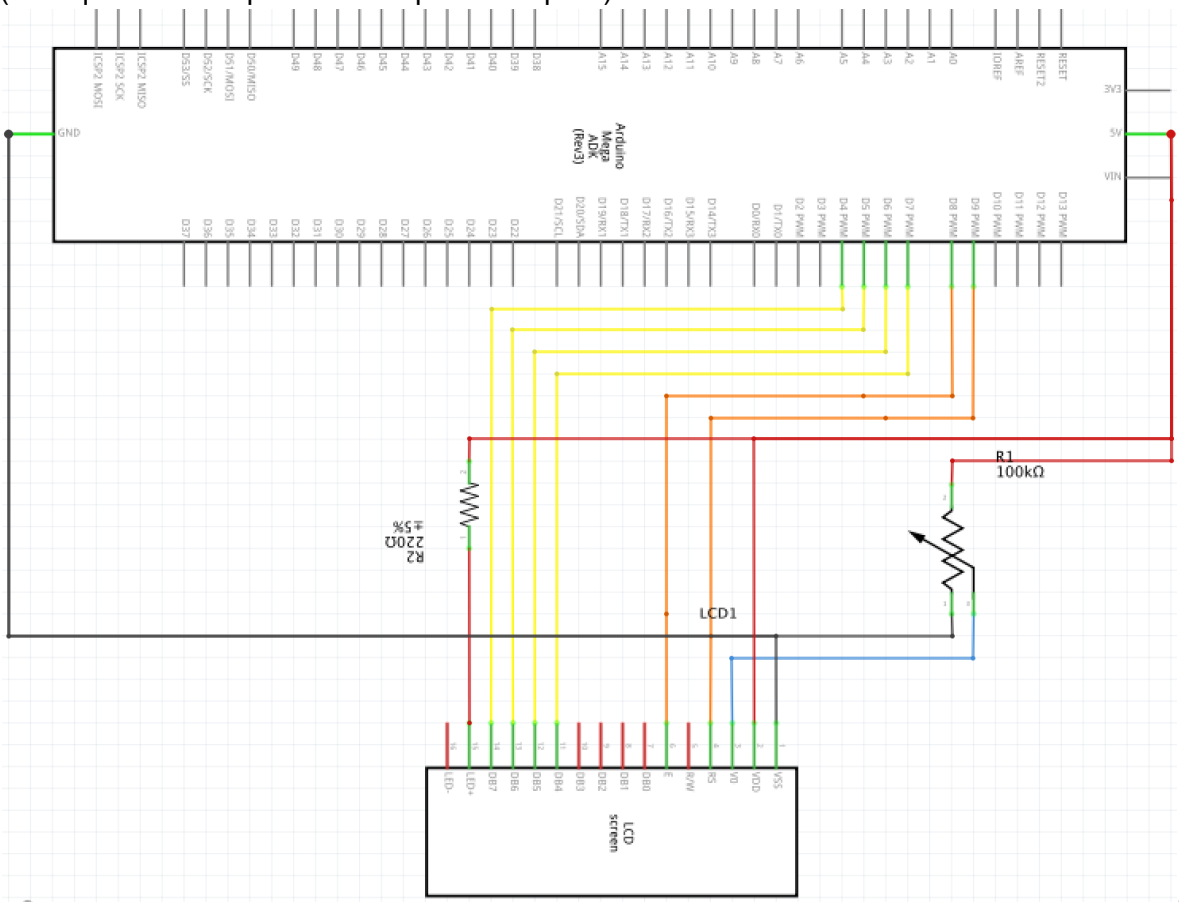
```
$usage: [-h] | -l | -i <name>
    -h,--help                print this message.
    -i,--interface <name> Ouvre l'interface USB spécifié. Par exemple :
                             /dev/tty.usbmodem1421
    -l,--list                Liste les interfaces USB disponibles.
```

L'aide étant assez explicite et exhaustive sur les options existantes, il n'y a pas beaucoup d'explication à rajouter. Il vous suffit donc de chercher l'interface USB correspondant à Arduino. Le nom de l'interface et le répertoire où les interfaces USB sont situées, dépendent de l'environnement.

## E. Arduino : Le capteur

Le code nécessaire à la partie électronique (Arduino) du projet se trouve à cet endroit du projet : *aqropolNUC/Arduino/sensor/novaPMsensor/novaPMsensor.ino*
Ce projet utilise un *ArduinoMega ADK* mais peut être adapté à tout type d'Arduino. Dans le même dossier se trouve un *README* qui liste les librairies utilisées dans le code ainsi que tous les liens utiles à la compréhension du code.

Pour aider à reconstituer le circuit Arduino voici comment était cablé l'écran LCD (bien que non indispensable en phase de prod) :




*Fi*

*g.1: Schéma écran LCD*

Ces pins devraient correspondre à une majorité d'Arduino. Néanmoins si vous

désirez les changer afin de faire votre propre câblage cela devra être répercuté dans le code à cette ligne :

```
LiquidCrystal lcd(9, 8, 4, 5, 6, 7);
```

La partie capteur *SDS011* (Nova PM Sensor) est configurée dans le code à cet endroit :

```
SDS011 my_sds;
...
void setup() {
    my_sds.begin(11, 10);
```

Les paramètres de la méthode sont définis ainsi :

```
begin(uint8_t pin_rx, uint8_t pin_tx);
```

Il suffit donc de câbler le pin *rx* du *SDS011* au pin *11* puis le pin *tx* au pin *10* de l'Arduino ou alors de modifier le code en conséquence. Il faudra également relier le pin *Vin* (5V) au pin *Vin* du capteur et le GND du capteur au GND de l'arduino. Pour plus d'informations, vous pouvez vous référer à la documentation du capteur située à cet endroit : *aqropolNUC/Arduino/sensor/novaPMsensor/DS011_laser_PM2.5_sensor_specification-V1.3.pdf*

# Prise en main de l'application Android

## I. Lancement de l'application

### 1. Installation requise

Pour pouvoir installer l'application sur votre smartphone android il faut activer le fait d'installer une application d'une source inconnue. C'est une option à cocher qui se trouve en général dans Configuration → Sécurité → Gestion de l'appareil.
Le fichier d'installation se trouve sur GitHub à l'adresse suivante https://github.com/AQROPOL/AQROPOL_App/blob/master/AQROPOL_Appli.apk

### 2. Lancement de l'application

Lors du premier lancement de l'application il faut être connecté à Internet pour pouvoir récupérer un identifiant venant du serveur. Une fois fait, il faut se mettre physiquement à proximité d'un NUC pour commencer à utiliser l'application.

## II. Comment exécuter l' application

### 1. Connexion au NUC

Pour pouvoir interagir avec le NUC, il faut d'abord s'y connecter en wifi en appuyant sur le bouton **CONNEXION**.

### 2. Connexion établie

Une fois la connexion établie, on peut récupérer les données mesurées par le capteur en appuyant sur **RECUPERATION DES MESURES.**

### 3. Connexion au Serveur

Il faut appuyer sur le bouton **ENVOIE DES MESURES** pour envoyer le fichier des données mesurées par le capteur tout en disposant d'une connexion internet soit 4G soit Wifi. En même temps, le smartphone récupère un fichier qui sera envoyé par la suite aux NUC pour leur permettre de supprimer les mesures enregistrées sur le serveur.

### 4. Arrêter la communication avec le NUC

Il faut appuyer sur le bouton **DECONNEXION** ce qui coupera la connexion entre le NUC et le smartphone. À éviter au cours d'un transfert de données au risque d'avoir des données corrompues.

## III. Comment quitter l'application

Pour quitter l'application il suffit tout simplement de faire retour sur le menu principal.

# Visualisation Web

La visualisation web a pour but d'observer les relevés de pollution fournis par les utilisateurs de l'application. Elle se veut simple et ergonomique afin que toute personne puisse étudier les mesures des particules fines.

## I. Prérequis

Un navigateur internet: Internet Explorer, Safari, Edge, Opera, Firefox, Chrome, Tor Browser. Les prérequis de chacun de ces navigateur ne sont pas de notre ressort, et nous ne sommes responsables d'aucune difficultée que vous pourriez rencontrer en les utilisant.

## II. Fonctionnalités

Sur votre navigateur accédez au domaine hébergeur de la visualisation (Provisoirement: http://pilic27.irisa.fr). Vous tomberez alors directement sur une carte interactive de Rennes avec quelques fonctionnalitées et informations.

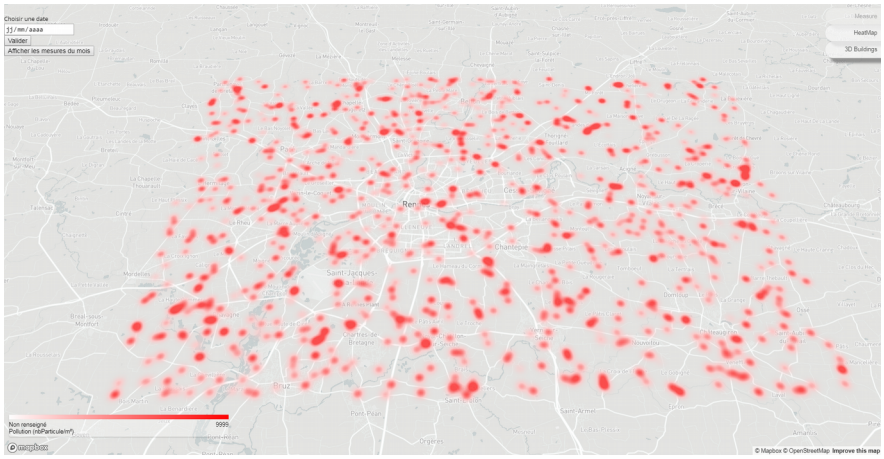
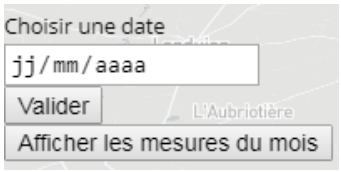


*Fig.2: Carte a l'arrivée sur le site web*

## 1. Datation

Vous trouverez premièrement en haut à gauche un champ vous permettant de




sélectionner une plage de données en fonction de leurs dates.

*Fig.3: Menu de sélection de Dates*

Cette dernière permet de demander l'affichage des mesures géolocalisées mesurées à la date renseignée.

## 2. Filtres

Vous trouverez en haut à droite divers boutons vous permettant de passer entre différents filtres. Ces derniers changent l'aspect de la carte ainsi que l'affichage des


données, afin de les apprécier de manières différentes.

*Fig.4: Boutons de sélections des filtres*

Les filtres actifs sont définis par un blanc plus vif tandis que les désactivés sont grisés.

- Le filtre **Measure** permet d'afficher les mesures de particules géolocalisées sélectionnée via la date et géolocalisée sur la carte, avec leurs valeurs respectives.
- Le filtre **Heatmap** est une carte thermique de la pollution. Cette dernière montre des tâches ou la pollution semble se regrouper.

- Le filtre **3D Building** est un filtre permettant une visualisation 3D des bâtiments de Rennes. Cette modélisation est basée sur les actuels relevés openstreetmap, ou si non disponible, sur le cadastre de la ville.

## 3. Légende

En bas à gauche vous trouverez la légende contenant le gradient de couleur pour les mesures en fonction du filtre actif.


*Fig.5: Légende de couleur sur l'affichage courant.*