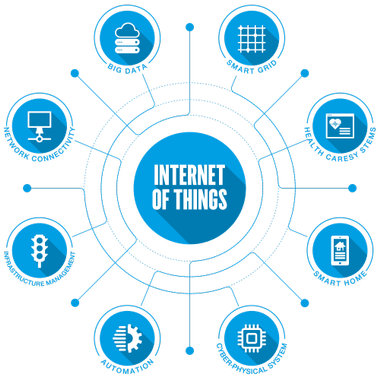
**Notice technique de conception du projet « IOT data environnement »**

****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| VISAS | | |  |  |
| A | BPO | 22/03/21 | Cédric GUILLEMIN | Mohamed BADAOUI |
| **Rév.** | **Etat** | **Date** | **Rédacteur** | **Vérificateur** |

Table des matières :

[1. Enonce du besoin 3](#_Toc71903284)

[1.1 données d’entrée 3](#_Toc71903285)

[1.2 contexte 3](#_Toc71903286)

[2. schema general du projet 4](#_Toc71903287)

[3. glossaire 5](#_Toc71903288)

[4. platine IOT 5](#_Toc71903289)

[4.1 définition et commande du matériel arduino 5](#_Toc71903290)

[4.1.1 Définition et commande des capteurs arduino 5](#_Toc71903291)

[4.1.2 Définition et commande de la carte processeur arduino 11](#_Toc71903292)

[4.1.3 Définition et commande de la carte Ethernet pour connexion au réseau internet 15](#_Toc71903293)

[4.1.4 Définition et commande du matériel électronique divers pour câblage de l’ensemble. 17](#_Toc71903294)

[4.2 platine de test 19](#_Toc71903295)

[5. application 21](#_Toc71903296)

[5.1 base de données 21](#_Toc71903297)

[5.2 api 25](#_Toc71903298)

[5.3 Authentification : 28](#_Toc71903299)

[6. planning realisé 31](#_Toc71903300)

[7. Cout total du projet : 32](#_Toc71903301)

# Enonce du besoin

## données d’entrée

Le projet consiste à cartographier les données environnementales d’une ville de façon détaillée. Pour cela, à l'aide de modules IOT, je dois remonter à mon API diverses données en temps réel qu’il me faudra définir (la pollution sonore, la qualité de l'air, la température, l’hygrométrie).

Le projet devra être présenté le 17 mai 2020 et conforme aux exigences de la fiche FICHE PROJET UF B3 20203.DOCX.

Il n’y pas de budget alloué, mais je ferais tout de même une évaluation financière du projet.

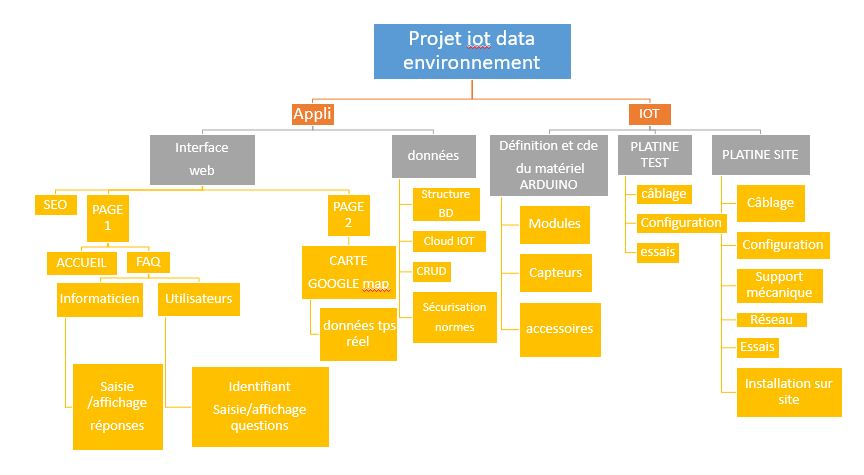
## contexte

Mon projet s’inscrit dans une démarche environnementale par le biais de la technologie IOT. Grace à la baisse du coût des capteurs, les entreprises connectent de plus en plus d’objets à internet pour améliorer leurs processus ou créer de nouveaux services personnalisés pour leurs clients. Mon application pourra être utilisée par exemple par les localités (habitants, mairie …).

Mon projet sera mis en œuvre dans le village de Saint-Martin-de-la-Brasque (84760).

Pour toutes ces raisons, je vous propose un outil qui permet aux collectivités et aux habitants d’être informés sur les paramètres environnementaux de leurs villes grâce aux données remontées par les modules IOT. Elles seront accessibles depuis une page web spécifique.

# schema general du projet



Le projet IOT data environnement est clairement scindé en deux parties :

* La réalisation de l’application,
* la réalisation de la partie électronique IOT.

Ces deux parties sont ensuite reliées par une interface permettant le fonctionnement de l’ensemble

Je vais décrire dans cette notice technique les deux ensembles et leurs éléments qui composent le projet IOT data environnement.

# glossaire

| **Termes / Sigles** | **Définitions** |
| --- | --- |
| IOT | L'Internet des objets ou IdO (en anglais (the) Internet of Things ou IoT) est l'interconnexion entre l'Internet et des objets, des lieux et des environnements physiques. |
| APTE | **AP**plication aux **T**echniques d' **E**ntreprise |
| IDE | Integrated développent environement |
| MTBF | Middle Time Between Failure |
| UML |  |
| FAST |  |
|  |  |

# platine IOT

## définition et commande du matériel arduino

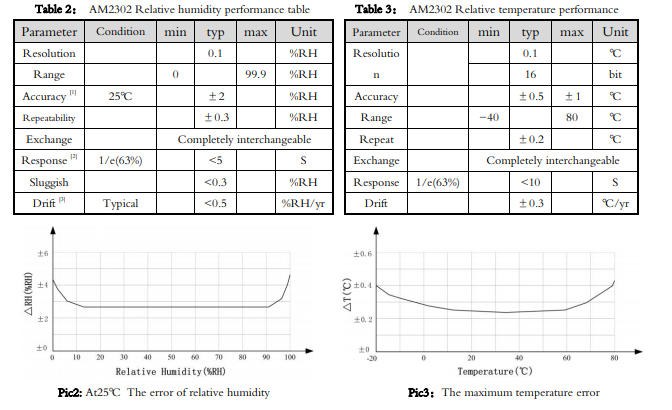
### Définition et commande des capteurs arduino

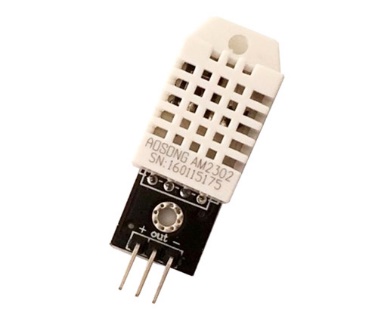
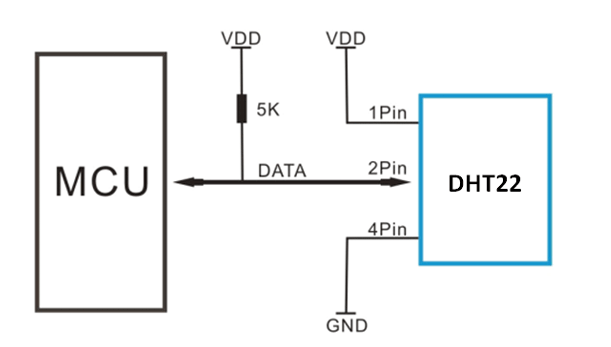
Les mesures environnementales proposées concerneront la température, l’hygrométrie, la qualité de l’air et la pollution sonore.

#### La température et l’hygrométrie seront mesurées par un capteur de type DHT22 qui réalise ces deux mesures.

Spécifications techniques du capteur DHT22 :

* Operating Voltage: 3.5V to 5.5V
* Operating current: 0.3mA (measuring) 60uA (standby)
* Output: Serial data
* Température : -40°C to 80°C
* Humidité Range: 0% to 100%
* Résolution: Température and Humidité both are 16-bit
* Précision: ±0.5°C and ±1%



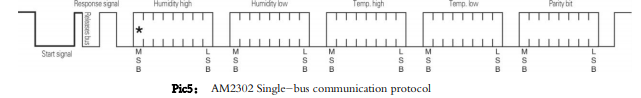
Protocole de communication :

Le circuit donne 16 bits pour l'humidité relative, exprimée en dixièmes et codé en binaire.

Par exemple, si les 2 premiers bytes sont 0000 0010 1001 0010 = 0x0292 La conversion en décimal donne 658, l'humidité est 65.8%.

Un Serial.print () Arduino affichera 658 et c'est toujours comme cela que cela se passe avec les langages "évolués": les variables sont en binaires en mémoire, mais on les montre en décimal. Si les 2 bytes suivant pour la température sont 1000 0000 0110 0101, le 1er bit à un indique que la température est négative et sa valeur absolue suit: 0000 0000 0110 0101 = 0x0065 La conversion en décimal donne 101, donc la température est de -10,1 degrés

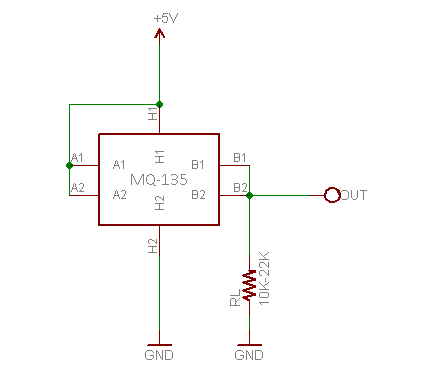
Il semble important de vérifier que la somme des 4 premières mesures (en ignorant les dépassements) est égale à la somme de contrôle. Il faut alors recommencer la mesure.

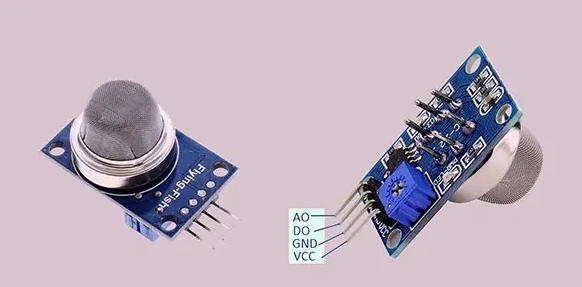


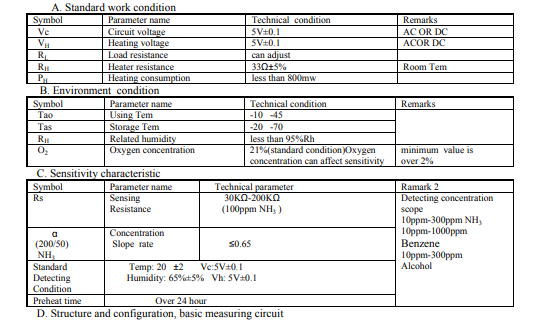
#### La qualité de l’air sera mesuré par un capteur MQ135

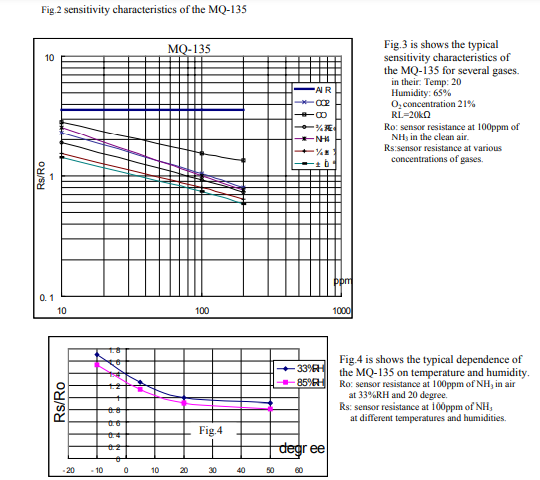
Spécifications techniques du capteur MQ135 :

* Tension nominale: 5V
* 2 Sorties: Analogiques et numérique tout ou rien selon un seuil réglable
* Le module possède un indicateur LED de l'état de la sortie numérique
* Détecte NH3, NOx, alcool, benzène, fumée et CO2
* Haute sensibilité: 10 - 300 ppm NH3, 10 - 1000 ppm Benzène, 10 - 300 Alcool
* Plus la concentration est importante plus la sortie analogique est élevée
* Stable
* Longue durée de vie
* Dimensions: 18mm diamètre, 17mm hauteur sans pin, Pins - 6mm de haut

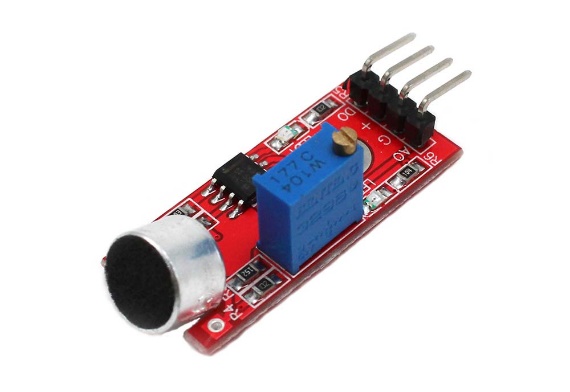
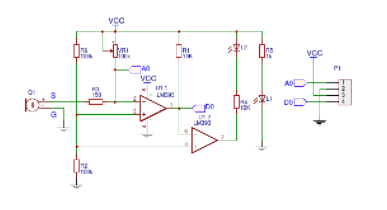






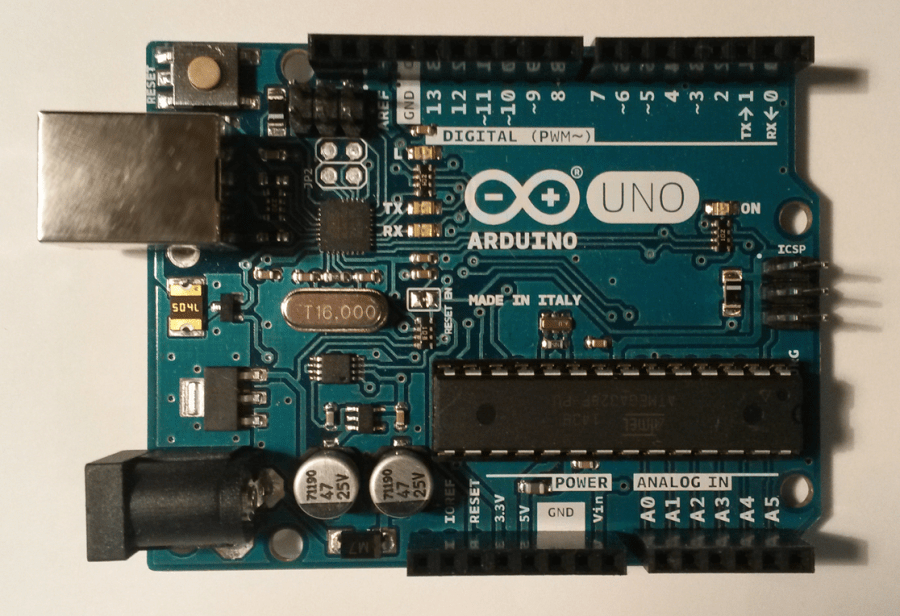
#### La pollution sonore sera réalisée aec le capteur KY038

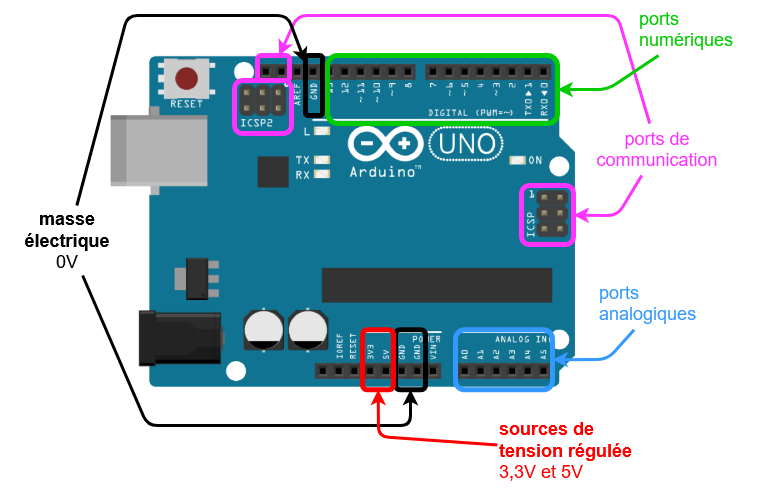
 

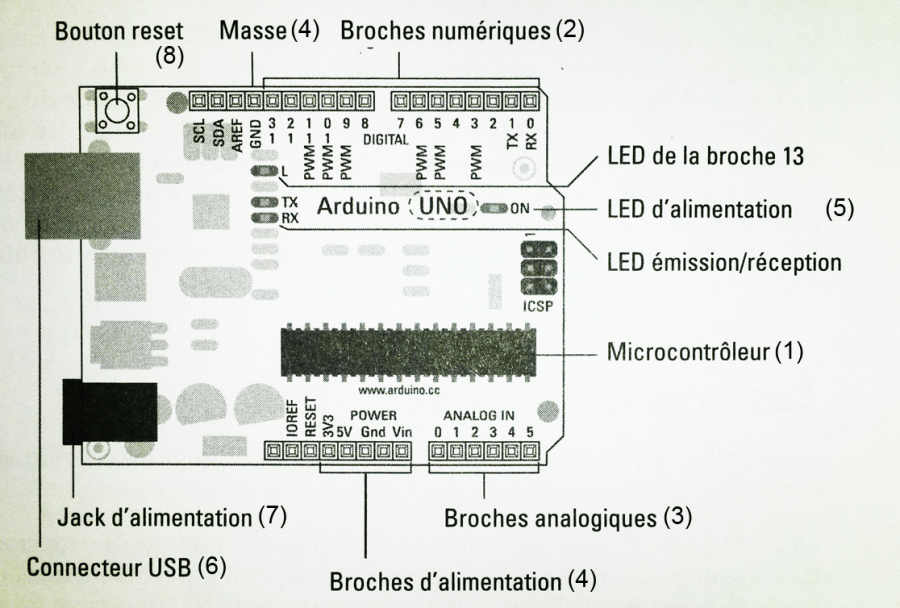
J’utilserais dans le cadre de mon pojet, la sortie analogique qui servira à transmettre integralement le signal recu par le micro à electrect du KY037.

Ce signal permettra de visualiser le niveau sonore de l’environnement mesuré.

### Définition et commande de la carte processeur arduino







**(1) La puce ATmega328**

La puce ATmega328 est le cerveau d’Arduino. C’est un microcontrôleur de la série AVR produite par la société Atmel et supporté par l’application Arduino pour son rapport puissance/prix très intéressant. Ne vous affolez pas non plus, on reste dans le raisonnable avec les 16 MHz d’horloge et 32 Ko de mémoire, mais pour 5 $ et ce que vous allez faire avec, c’est parfait. Par chance, l’ATmega328 de l’Arduino est fixé sur un support plutôt que soudé directement sur la carte, ce qui vous permet de le changer facilement s’il vous arrivait de le griller.

**Les entrées/sorties**

Comme vous pouvez le voir sur la photo et le plan plus haut, la carte est équipée sur ses bords d’une foule de connecteurs qui lui permettent d’interagir avec le monde extérieur. Certains d’entre eux sont des entrées/sorties numériques, d’autres sont des entrées/sorties analogiques, enfin les broches restantes ont un rôle en rapport avec l’alimentation électrique du circuit. Sur le modèle Uno, comme sur la plupart de ce genre de cartes, Arduino utilise des connecteurs de type HE (les barres de plastique noir avec les trous « carrés »), qui sont très pratiques pour le prototypage puisque leur format est le même que celui des platines d’essai (breadboards). Les connecteurs HE permettent également de facilement réaliser des « étages » avec d’autres cartes. On appelle ces cartes supplémentaires, montées au-dessus de la carte Arduino, des « shields ». On reparlera de ça plus en détail dans les tutoriels à venir.

**(2) Entrées/sorties numériques**

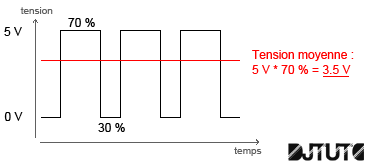
Il y a 14 entrées/sorties numériques notées de 0 à 13 sur l’Arduino Uno. Elles sont situées sur la grande rangée du haut de la carte. Ces connecteurs sont numériques car le signal ne connaît que deux états : haut ou bas. Électriquement, cela se traduit par une tension de 5 volts ou 0 volt, respectivement.

**(3) Entrées analogiques**

Il y a six entrées analogiques notées de 0 à 5 en bas à droite de la carte. Ces connecteurs transmettent des valeurs analogiques, c’est à dire que, du point de vue électrique, il s’agira d’une possibilité infinie de valeurs situées entre 0 et 5 volts.

**(2-PWM) Sorties analogiques**

Comme vous avez pu le remarquer, il n’y a pas de sortie analogique à proprement parler. Par contre, six des connecteurs numériques (les connecteurs 3, 5, 6, 9, 10 et 11) sont capables de simuler des sorties analogiques et fonctionnent donc comme telles. Elles sont marquées du sigle PWM, qui veut dire Pulse Width Modulation (Modulation de Largeur d’Impulsion en français). Le PWM fonctionne ainsi : comme il n’est possible que d’envoyer des informations binaires (haut ou bas, c’est à dire 5 V ou 0 V), Arduino va faire varier le temps pendant lequel ces deux valeurs vont être employées afin d’obtenir le résultat attendu. En gros, le circuit va alterner ultra rapidement entre « haut » et « bas ». Si, par exemple, « haut » occupe 70% du temps pendant les alternances, alors votre LED va être à 70% de son éclairage maximum, ou votre moteur va tourner à 70% de sa vitesse. C’est beau la technologie.

**[](http://i2.wp.com/www.djtuto.fr/wp-content/uploads/2015/06/pwm.png)**

**(4) Les broches d’alimentation**

Les broches d’alimentation sont utilisées pour alimenter en courant les entrées et sorties du circuit selon les besoins. La broche « 5V » sert à fournir 5 volts de courant tandis que la broche « 3.3V » sert à fournir 3.3 volts. Logique. Les broches « GND », qui signifie « ground » (« terre », en français), servent à relier le courant à la masse, c’est à dire au 0 V. Enfin, la broche « Vin », qui signifie « Voltage in » (« tension d’entrée » en français), sert à faire venir du courant d’une source externe (entre 7 et 12 V), comme pour le port USB (6) et le jack d’alimentation (7) qu’on aborde un peu plus loin.

**(5) les LED**

La carte Arduino Uno possède quatre LED qui servent à donner des indications sur son fonctionnement. « ON » s’allume lorsque la carte Arduino est alimentée. « RX » et « TX » s’allument lorsque la carte reçoit et transmet des données, respectivement. « L » est connectée à la broche 13. Elle s’allume lorsque celle-ci est sur « haut », ce qui fait de cette broche un connecteur essentiellement utilisé pour réaliser des tests.

**(6) Le port USB**

Le connecteur USB sert à, comme vous vous en doutez, connecter la carte à l’ordinateur ou éventuellement tout autre appareil disposant également d’un port USB. Celui-ci sert donc à l’alimentation en courant (5 V) autant qu’au transfert de données. Sur l’Arduino Uno, c’est évidemment le connecteur USB que l’on utilise lorsqu’on veut transférer un programme écrit dans le logiciel Arduino pour faire fonctionner un montage réalisé avec la carte.

**(7) Le jack d’alimentation supplémentaire**

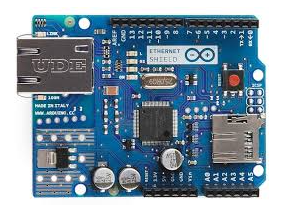
Le jack d’alimentation est un autre moyen de fournir du courant à la carte (de 7 à 12 V). Cela pourrait être un adaptateur AC-DC relié à une prise électrique, ou une source portable comme une batterie électrique ou un panneau solaire. En fait, ce connecteur est souvent utilisé lorsqu’on souhaite que notre circuit s’alimente de manière autonome, comme lorsqu’il sert à piloter un drone ou un robot.

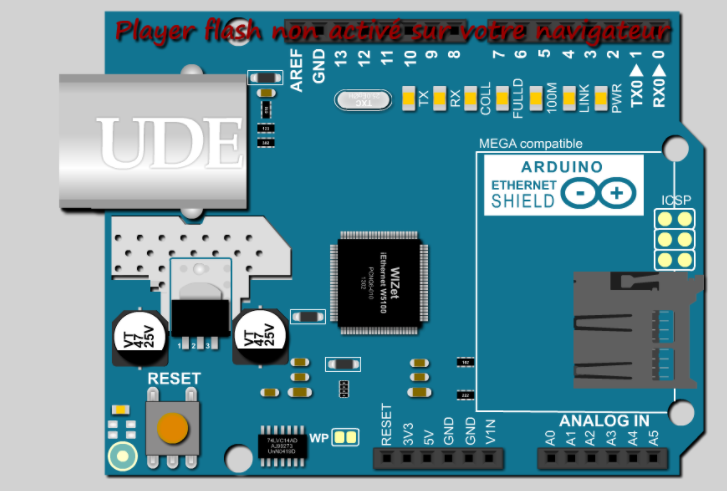
**(8) Le bouton reset**

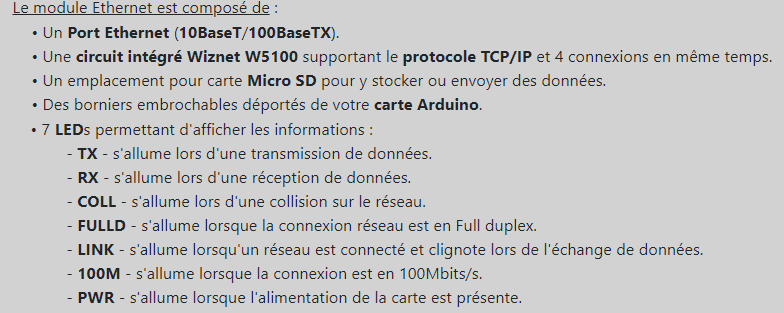
Le bouton reset est un petit interrupteur qui vous permet de réinitialiser la carte ou de l’arrêter lorsqu’il est maintenu enfoncé quelques secondes. On peut, par ailleurs, obtenir le même résultat en connectant la broche « reset », située à côté de la broche « 3.3V », à la masse (une des broches « GND »).

### Définition et commande de la carte Ethernet pour connexion au réseau internet

Le Lien entre la carte arduino uno et ma base de données sera assurée par une carte Ethernet d’extension shield internet W5100. Cette carte s’enfiche sur la carte arduino uno (shield).







La communication utilise le réseau Ethernet qu’il est possible de connecter au réseau Wifi. Pour pouvoir accéder à la carte Arduino sans avoir à la connecter au routeur internet, il est nécessaire de créer un pont entre la connexion wifi et la connexion Ethernet sur votre ordinateur. Pour cela, il faut:

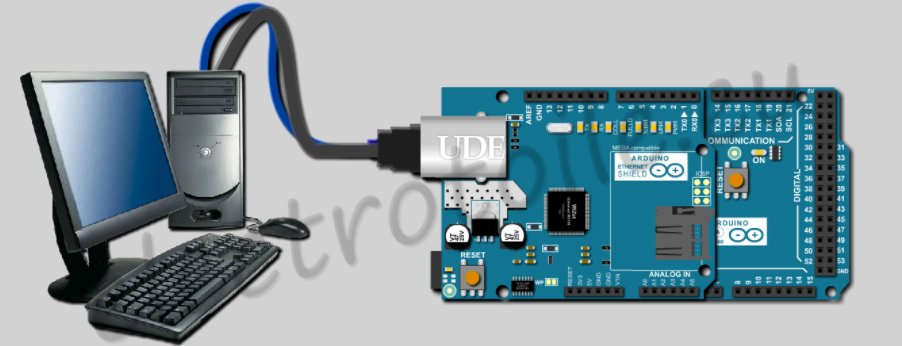
* Allez dans le Centre Réseau et Partage
* Allez dans « Modifier les paramètres de la carte »
* Sélectionnez Ethernet et Wifi /(internet source) et clique droit
* Cliquez sur « Créer pont »

Le Shield Ethernet W5100 est compatible avec les microcontrôleurs Arduino UNO et Mega.

La puce W5100 et la carte SD utilisent le SPI bus pour communiquer avec la carte Arduino. La broche 10 est utilisé pour sélectionner le contrôleur Ethernet W5100, la broche 9 comme RESET pour le W5100 et la broche 4 pour sélectionner le module de carte SD.

En résumé, les broches utilisées sont:

* 4 pour la sélection de la carte SD (SD\_CS)
* 9 pour le reset de la micro puce W5100 (W5100\_RST)
* 10 pour la sélection de la micro puce W5100 (W5100\_CS)
* le bus SPI : broches 11,12,13 sur une carte Arduino UNO/Duelmilanove. Les broches 50, 51 et 52 pour la Mega.





### Définition et commande du matériel électronique divers pour câblage de l’ensemble.

De manière à réaliser le câblage électronique, il m’a fallu commander aussi deux platines de tests, des fils de connexion, un jeu de résistances et potentiomètres, un jeu de led et un afficheur 20X4 (DSD TECH SH-D2004 20x4 2004 Ecran LCD Bleu avec Adaptateur d'interface de Module IIC pour Arduino UNO R3 Mega2560 Nano Raspberry Pi) permettant de visualiser directement les mesures réalisées et de pouvoir ensuite comparer ces valeurs avec les valeurs utilisées dans la base de données.



L’afficheur sera connecté par la liaison série I2C à la carte arduino uno.

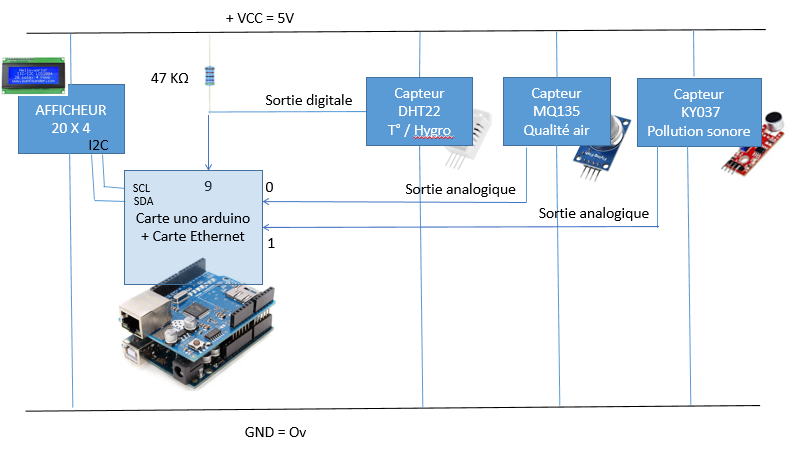
 

Tous ces accessoires électroniques m’ont permis de réaliser le câblage de la platine de test définie ci-après.

## platine de test

Après avoir commandé l’ensemble des composants électroniques, j’ai câblé la platine de test et je l’ai connecté à mon ordinateur par le biais du câble USB pour alimenter l’ensemble et configurer le processeur de la carte arduino uno de manière à visualiser l’ensemble des paramètres environnementaux sur l’afficheur LCD.

Schéma de la platine de test :



Une image contenant texte

Description générée automatiquement

# application

API = (Application Programming Interface) un service à disposition d’un ou plusieurs développeurs.

Une image contenant texte, signe, capture d’écran

Description générée automatiquement

Ici l’application passe directement par l’API et l’API comunique avec la base de données.

Code televerser sur le module IOT par l’interface arduino pour la connection a la base de donnée :

Une image contenant table

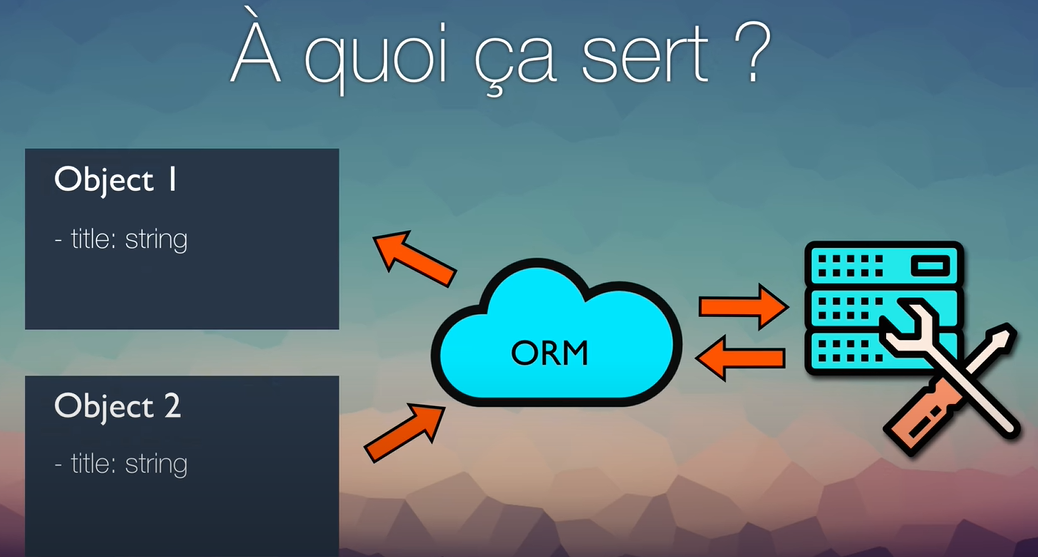
Description générée automatiquement

Ce code recupère les données des capteurs pour les envoyer sur la base de donnée via une requete PHP :

Une image contenant texte

Description générée automatiquement

## base de données



Ici la réalisation de l’ORM m’a permis de créer un ensemble de classes qui vont représenter un objet dans le but de pouvoir les manipuler (ajout, suppression, MAJ) et communique automatiquement avec notre base de données SQL.

ORM = Object Relational Mapping

Utilisation du CLI Sequelize l’ORM.

Une image contenant texte

Description générée automatiquement

Création de model exemple :

Une image contenant texte, intérieur, capture d’écran, portable

Description générée automatiquement

Model en code pour User :

Une image contenant texte

Description générée automatiquement

Fichier de migration :

Une image contenant texte

Description générée automatiquement.

Migration sequelize (grâce au db :migrate) :

Une image contenant texte

Description générée automatiquement

## api

Pour la réalisation de l’API j’ai choisi comme langage NodeJS comme runtime combiné au Framework Express.

Création du projet grâce au npm init :

Une image contenant texte

Description générée automatiquement

Dépendances ajoutés :

Express :

Une image contenant texte

Description générée automatiquement

Nodemon :

Une image contenant texte

Description générée automatiquement

Server.JS :

Instancie la connexion au server, création de route ici la route ‘/’ est la racine du server :

Une image contenant texte

Description générée automatiquement

A la ligne 17 le «(200) » correspond au status de la réponses j’ai ajouté ci-dessous la liste de tous les codes.

Liste des codes http : https://fr.wikipedia.org/wiki/Liste\_des\_codes\_HTTP

La ligne 24 permet de mettre en écoute le server sur le port 8080.

## Authentification :

Une image contenant texte, signe, ciel nocturne

Description générée automatiquement

J’ai réalisé une authentification par token JWT afin de fiabilisé et sécuriser la connexion.

Le Token est un contrôle qui sécurise l’acces a mon API.

Une image contenant texte

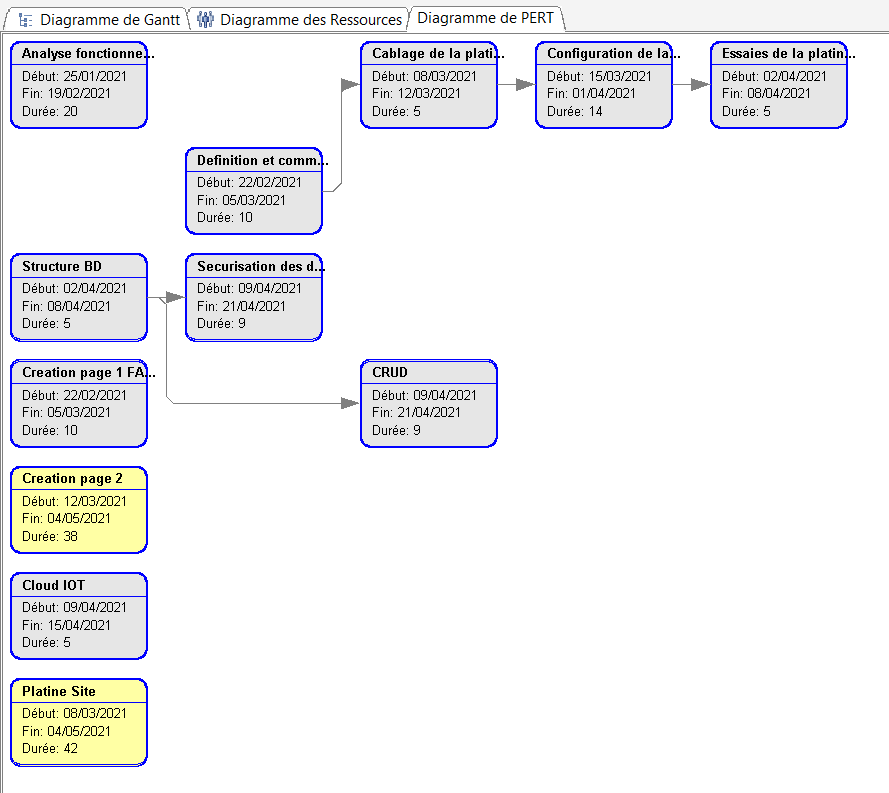
Description générée automatiquement

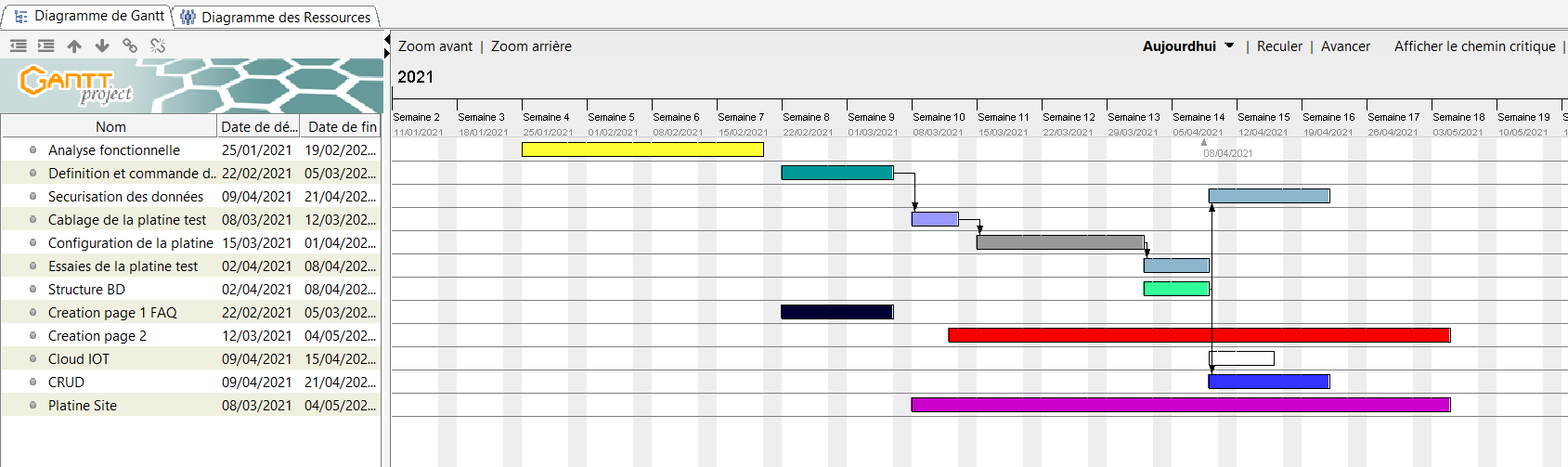
J’ai créé un user Controller qui va contenir mes fonction (register, login …) :

Une image contenant texte

Description générée automatiquement

# planning realisé





# Cout total du projet :

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement



+ Coffret PVC 38e

Total :110e