****

Ordre : 2021/2022

Faculté des Sciences Kenitra

Département de physique

Licence Universitaire Spécialisée

**MECATRONIQUE**

Projet de fin d’études

Présenté par :

**FAOUZI Youssef**

Sous le thème :

|  |
| --- |
| **Station Météo**  **Arduino et Android APP**  **Projet IOT** |

Encadré par : Pr. **ECHCHELH Adil**

Soutenu le : 22/06/2022

Devant les membres du jury :

* Président : Pr. ECHCHELH Adil
* Examinateur : Pr. Adil Lasfar
* Examinateur : Pr. Hicham Ahchich

Contenu

1. Mon projet Arduino - Station météo 3
2. Technologies 3
3. Analyse et conception 6
4. Mise en œuvre et essais 13
5. L’application Mobile 17
6. Conclusions 19
7. Référence Web 19

1 . Projet IOT - Station Météo

J'ai l'intention de créer une station météo en utilisant le microcontrôleur Arduino Uno et des capteurs avec communication REALTIME Firebase. Cette station météo pourra mesurer et afficher la température, l'humidité, la pression atmosphérique.

Pour une fonctionnalité supplémentaire, j'ajouterai une application Android connectée pour afficher en temps réel toutes les mesures mentionnées ci-dessus ainsi qu'un rapport sur la météo.

2. Technologies

● Qu'est-ce qu'Arduino ?

Arduino est une entreprise de matériel et de logiciels open-source, projet et communauté d'utilisateurs qui conçoit et fabrique des microcontrôleurs monocartes et des kits de microcontrôleurs permettant de construire des dispositifs numériques.

Les conceptions de cartes Arduino utilisent une variété de microprocesseurs et de contrôleurs. Les cartes sont équipées d'ensembles d'entrées/sorties (E/S) numériques et d'entrées/sorties (E/S) numériques et analogiques qui peuvent être reliées à diverses cartes d'extension ou de circuits imprimés et autres circuits.

Les cartes sont dotées d'interfaces de communication série, y compris d'un série, y compris le bus série universel (USB) sur certains modèles, qui sont également utilisés pour le chargement de programmes à partir d'ordinateurs personnels. Charger des programmes à partir d'ordinateurs personnels. Les microcontrôleurs peuvent être programmés à l'aide des langages de programmation C et C++. en utilisant une API standard également connue sous le nom de "langage Arduino". Sur traditionnelles, le projet Arduino fournit un environnement de développement intégré (IDE) et une ligne de commande (arduino-cli) développé en Go.

● Firebase

Firebase est une plateforme développée par Google pour la création d'applications mobiles et web, qui nous offre des fonctionnalités comme l'analyse, les bases de données, la messagerie et les rapports de collision. Firebase fournit un moyen rapide de conserver les données sensorielles collectées au niveau de l'appareil, je vais donc utiliser cette plateforme pour stocker les données enregistrées par mes capteurs. Pour utiliser cette plateforme, j'ai dû créer un compte puis créer ma base de données en temps réel.

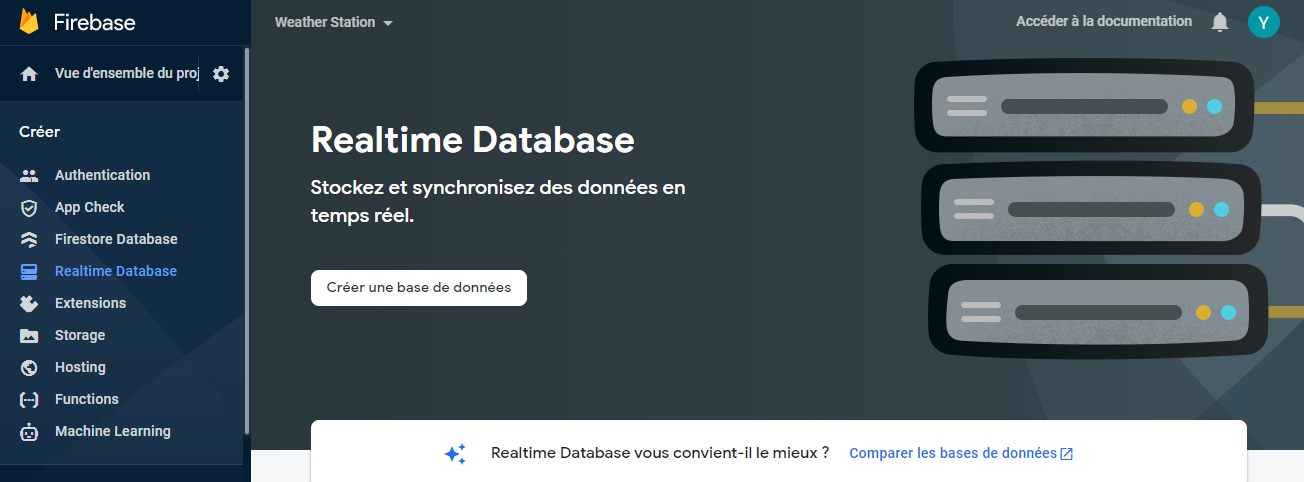


Figure 2 - Plate-forme Firebase

Le didacticiel vidéo fourni sur leur site Web lors de la première entrée dans la section Base de données en temps réel était très utile et convivial pour les débutants. J'ai créé une base de données simple qui sera le lieu de stockage de mon application.

● Android Studio

 Android Studio est l'environnement de développement intégré (IDE) officiel du système d'exploitation Android de Google. le système d'exploitation Android de Google, construit sur le logiciel IntelliJ IDEA de JetBrains et conçu spécifiquement pour le développement Android. J'ai choisi d'utiliser cet environnement de développement plutôt qu'Eclipse en raison de son interface qui est plus facile à comprendre et à utiliser. De plus, l'option de glisser-déposer rend la partie conception beaucoup plus agréable. beaucoup plus agréable. L'interface de l'application est présentée dans la figure 3.

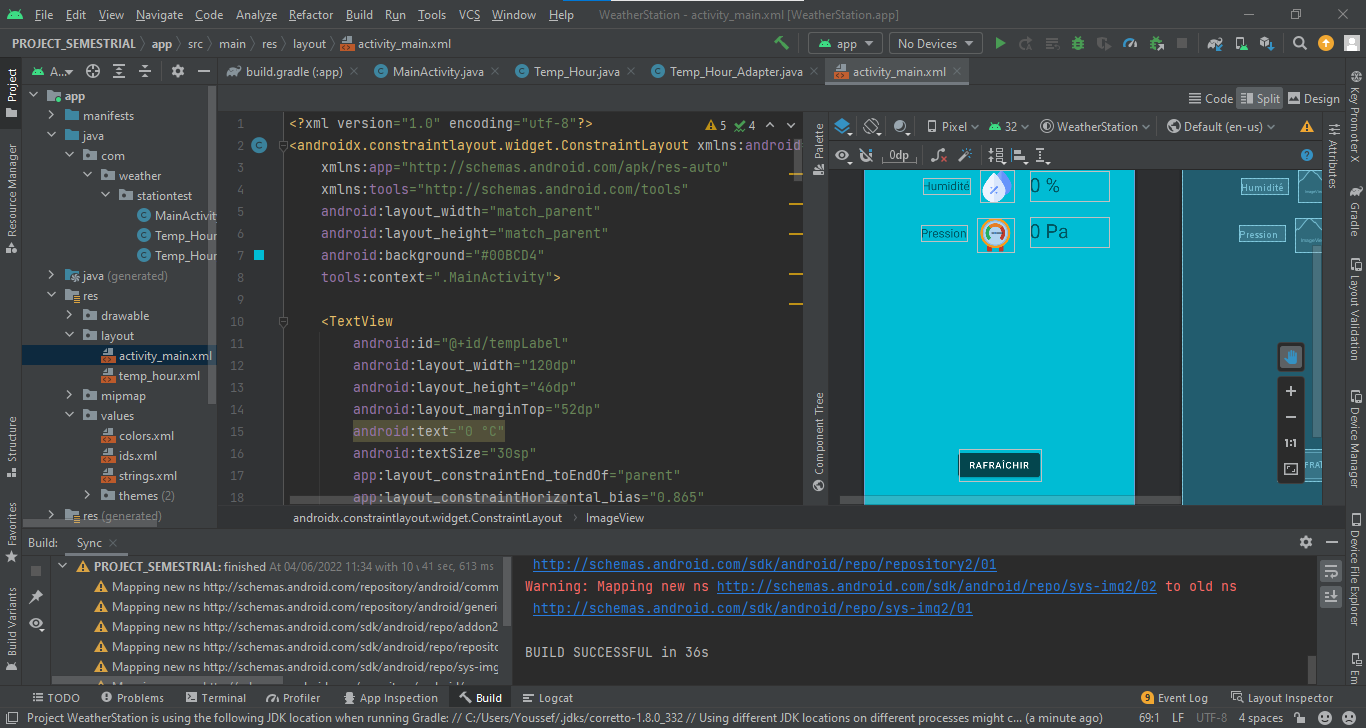


Figure 3 - Interface Android Studio

● Fonctionnement :

Les capteurs seront programmés dans Arduino IDE et le programme enverra des données via le module NodeMCU. (en utilisant une connexion au Wi-fi) à la base de données Firebase. De cette partie, la base de données recevra des informations sur la température, l'humidité et la pression atmosphérique.

Ensuite, l'application Android se connectera à la base de données Firebase, en extraira les données et les affichera sur l'appareil Android qui fonctionne. les afficher sur l'appareil Android qui exécute l'application.

3. Analyse et conception

Dans ce chapitre, nous allons passer en revue les composants utilisés pour créer la station météorologique, la façon dont nous connectons ces composants au NodeMCU, au module Wi-fi (et aussi à l'Arduino UNO), et le code qui est utilisé est nécessaire pour le bon fonctionnement de ces composants

● Capteur de température et d'humidité SNS-DH11 :

Le DHT11 est un capteur numérique de température et d'humidité de base et de faible coût. et d'humidité. Il utilise un capteur d'humidité capacitif d'humidité capacitif et une thermistance pour mesurer l'air ambiant et émet un signal numérique sur la broche de données (aucune broche d'entrée analogique n'est nécessaire). C'est assez simple à utiliser, mais nécessite un timing précis pour saisir les données. Le seul véritable inconvénient de ce capteur est que vous ne pouvez obtenir de nouvelles données que toutes les 2 secondes.

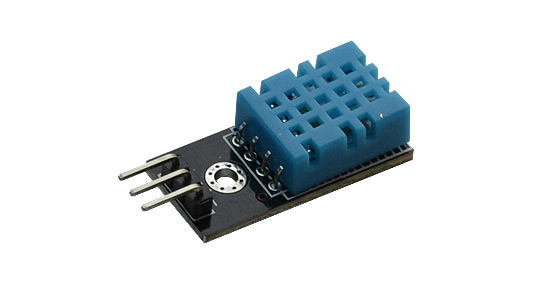


Figure 4 - Capteur de température et d'humidité SNS-DH11

Après avoir effectué le câblage présenté dans la Figure 5 et après avoir écrit le code de la Figure 6 dans l'IDE Arduino IDE, nous pouvons tester notre capteur dans le Serial Monitor (également affiché dans la Figure 6). J'ai testé mon capteur en le tenant entre mes doigts et en le regardant enregistrer une humidité et une température plus élevées. Une bibliothèque très utile bibliothèque très utile pour lire et traiter l'entrée de ce capteur est DHT11Lib (nous pouvons ajouter une bibliothèque directement en utilisant l'IDE Arduino en cliquant dans la barre de menu sur le bouton "Tool" et ensuite "Manage Libraries..." et une fenêtre de Library Manager apparaîtra à l'écran où nous pourrons rechercher la bibliothèque nécessaire).

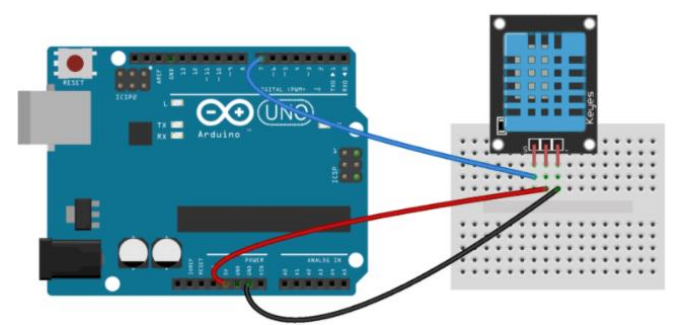


Figure 5 - Câblage du capteur de température et d'humidité SNS-DH11

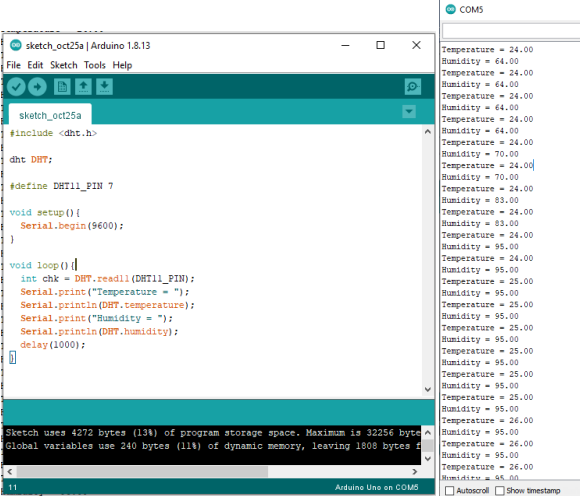


Figure 6 - Code de lecture du capteur SNS-DH11 et des données enregistrées pendant le test

● BMP180 Digital Barometric Pressure Sensor:

Les capteurs de pression barométrique mesurent la pression absolue de l'air autour du capteur. Cette pression varie en fonction des conditions météorologiques (température et humidité). Selon la façon dont nous interprétons les données, nous pouvons surveiller les changements météorologiques, ou toute autre tâche qui nécessite une lecture précise de la pression.

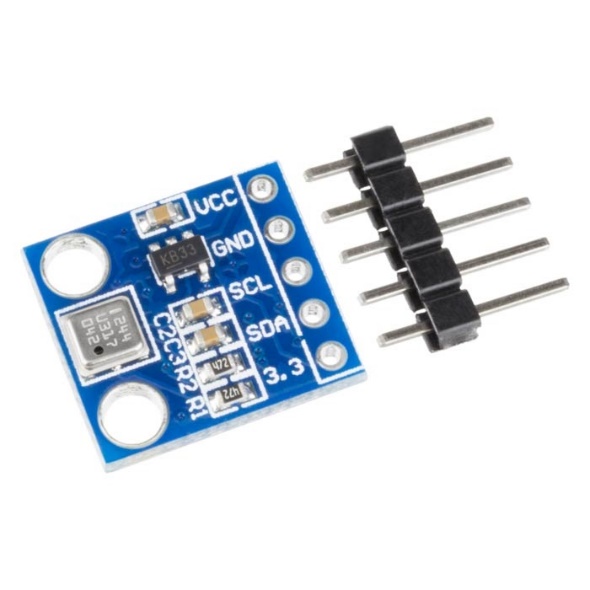


Figure 7 - Capteur numérique de pression barométrique BMP180

Pour traiter les données provenant de ce capteur, j'ai utilisé une bibliothèque BMP180 de Sparkfun.com appelée SFE\_BMP180 Library (lien [1] dans les références web). Cette bibliothèque est très utile car elle prend en charge tous les calculs de la température et de la pression réelles.

Après avoir téléchargé la librairie à partir du lien fourni à la fin de ma documentation, j'ai ouvert l'IDE Arduino et ajouté la bibliothèque en allant dans Sketch > Include Library> Add Library > sélectionner le fichier téléchargé.

Nous pouvons observer sur la Figure 8 le câblage que j'ai utilisé pour tester mon capteur de pression barométrique numérique BMP180.

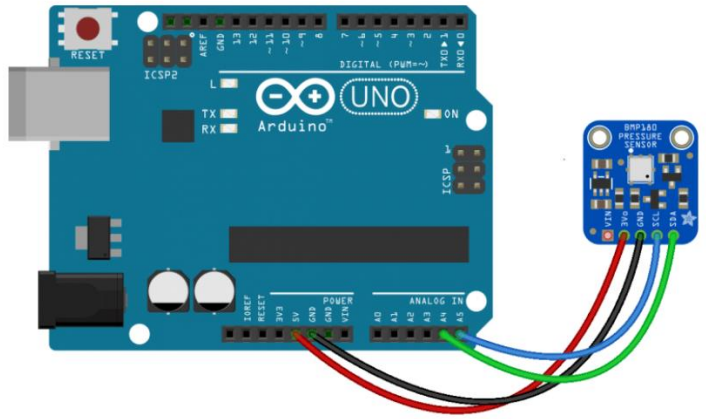


Figure 8 - câblage pour tester le capteur numérique de pression barométrique BMP180

Pour la partie codage du testeur, j'ai utilisé un exemple tiré de https://www.circuitbasics.com/. Le code est Figure 9 ainsi que le moniteur série où j'ai affiché les informations enregistrées par le capteur. Après avoir compilé le code, l'avoir téléchargé et avoir ouvert le Serial Monitor, j'ai commencé à déplacer le capteur à différentes altitudes pour tester la pression barométrique (différentes altitudes signifient différentes pressions barométriques). différente pression barométrique). J'ai découvert que le capteur est très sensible à tout changement dans l'altitude mais aussi aux changements de température - j'ai testé cette caractéristique de la même manière que l'autre capteur décrit précédemment - le SNS-DH. décrit précédemment - le capteur de température et d'humidité SNS-DH11.

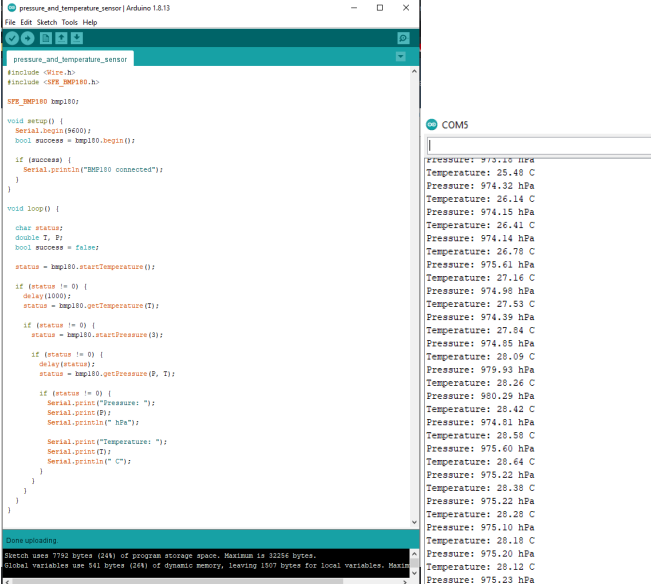


Figure 9 - code pour tester le capteur numérique de pression barométrique BMP180 et les résultats du moniteur série enregistrés lors de l'exécution du code dans l'Arduino IDE

● ESP8266 Wi-Fi microchip NodeMCU v2:



Cette micropuce fournit une solution complète de mise en réseau Wi-Fi, permettant aux utilisateurs de mettre en place un serveur web ou un client web avec un processeur séparé ou même autonome. Elle est également compatible avec Arduino ce qui signifie que vous pouvez la programmer à l'aide de l'Arduino

IDE.

Ce composant sera utilisé dans une étape ultérieure du développement de ma station météo. Développement de ma station météo et je vous expliquerai son utilisation et le code et le câblage dans les chapitres suivants.

Traduit avec www.DeepL.com/Translator (version gratuite)

Figure 10 - ESP8266 Wi-Fi microchip NodeMCU v2

Maintenant que nous en savons un peu plus sur tous les composants de base qui seront utilisés pour créer cette station de météo, nous pouvons passer à l'étape suivante : connecter tous les composants entre eux.

J'ai testé ce composant en prenant un exemple sur Internet. Le lien pour le tutoriel est le 3ème lien de référence web. lien de référence. Le code est présenté ci-dessous, ainsi que la configuration pour ce test.

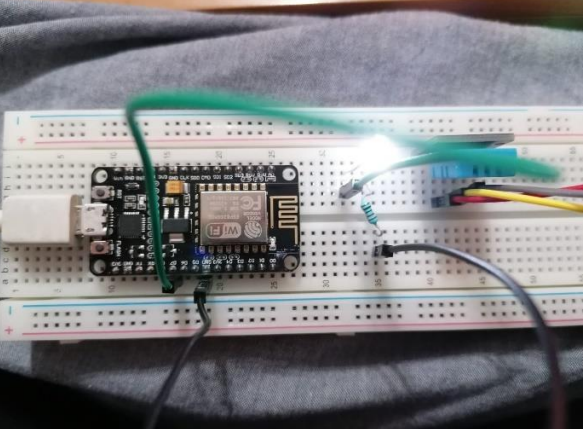
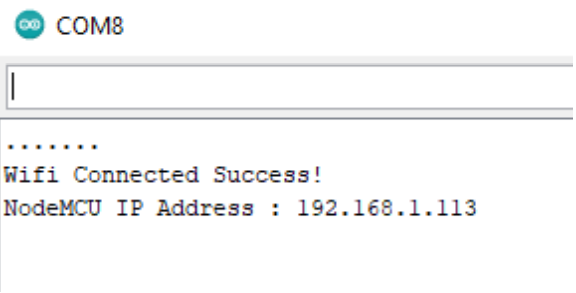
 

Figure 11 - Configuration pour tester le module wi-Fi Figure 12 - Réponse du moniteur série

Le code que j'ai utilisé pour les tests est celui présenté ci-dessous :

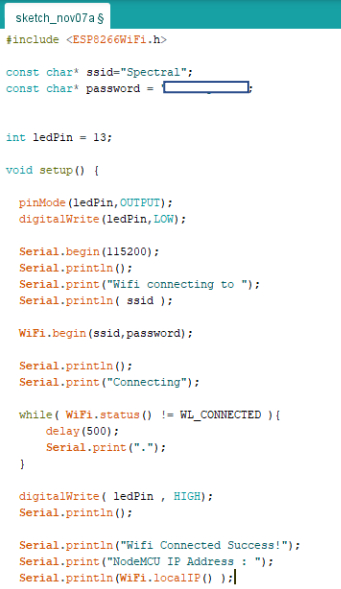


Figure 13 - Code pour tester le module wi-Fi de l'ESP8266

● La base de données :

Après avoir créé la base de données, j'ai connecté le module WI-Fi et testé la connexion en prenant un exemple trouvé au 4ème lien dans la section des références web de cette documentation.

J'ai testé la connexion à la base de données en utilisant ce code et nous pouvons observer que la valeur de RandomVal change simultanément avec la valeur du Serial Monitor. Comme nous pouvons voir dans la Figure 14, la communication fonctionne parfaitement

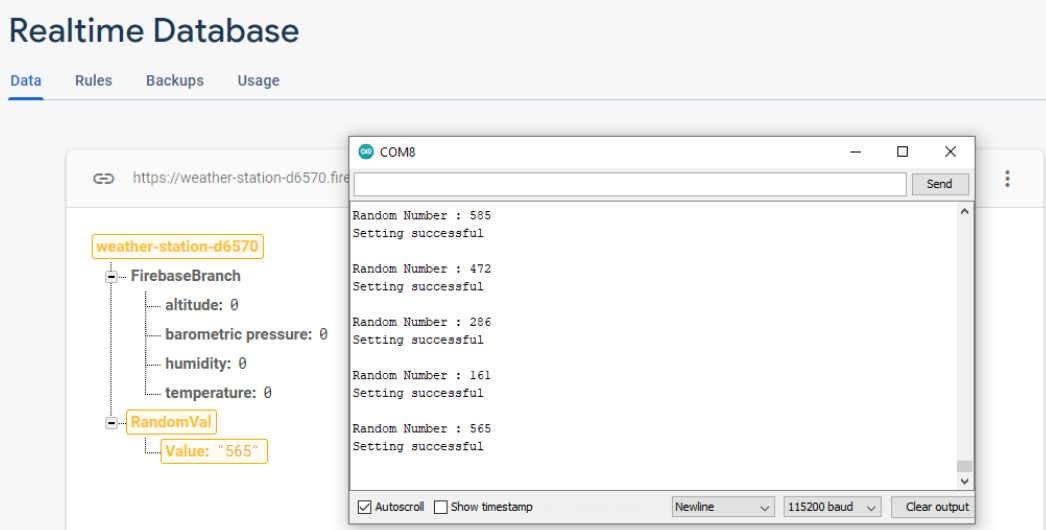


Figure 14 - Réponse du moniteur série et mise à jour enregistrée dans le fichier

4. Mise en œuvre et essais

Dans cette section, je vais assembler tous les composants pour créer une station météo fonctionnelle. Je commence par connecter le capteur de température et d'humidité DHT11 au module wi-Fi (la broche GND du capteur est connectée à la du capteur est connectée à la masse du module, la broche d'alimentation du capteur à la broche 3.3V du module et la broche de module et la broche de données du capteur est connectée à la broche D4 du module qui est une broche numérique), comme nous pouvons le voir sur la Figure 15.

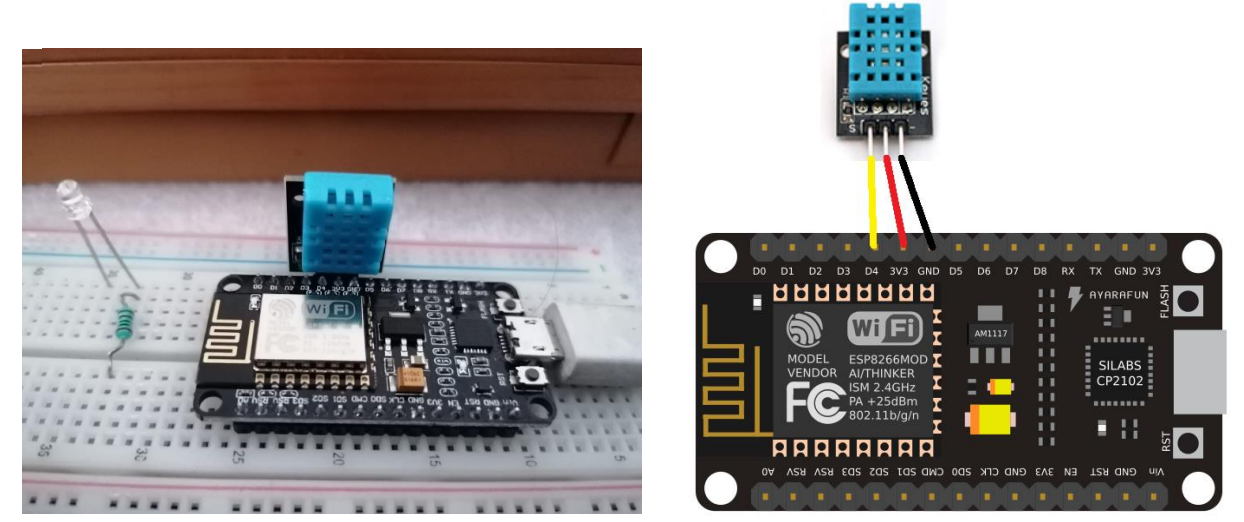


Figure 15 - Câblage simple pour le capteur DHT11 - Réel et schématique

Après avoir fait cette configuration, nous essayons d'envoyer les données reçues du capteur à notre base de données Firebase. Le code utilisé est listé ci-dessous. Le code a des commentaires inclus afin qu'il soit plus facile à comprendre et plus lisible.

Dans les figures suivantes (Figure 16 et 17), nous pouvons voir que le capteur envoie des données à la base de données et que la base de données est mise à jour toutes les 5 secondes. Le capteur est lent, la lecture de la température ou de l'humidité prend environ 250 millisecondes. humidité prend environ 250 millisecondes et les relevés du capteur peuvent également dater de 2 secondes (en raison du délai de transmission du capteur).du délai de transmission du capteur).

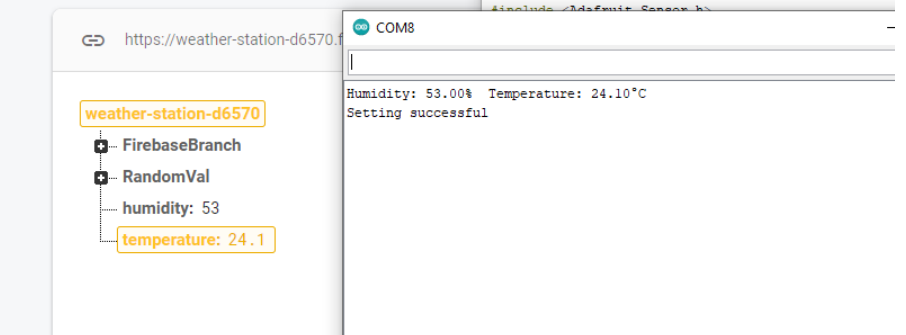


Figure 16 - Réponse de la base de données vue à côté de la fenêtre du moniteur série - température mise à jour

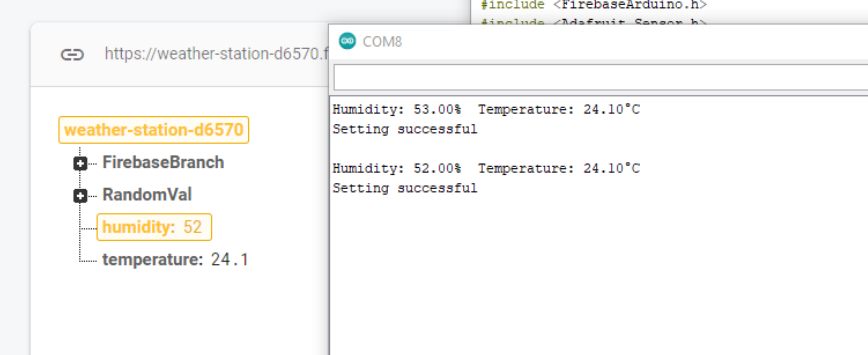


Figure 17 - Réponse de la base de données vue à côté de la fenêtre du moniteur série - humidité mise à jour

Dans la figure suivante (20), nous pouvons voir comment le capteur enregistre une température et une humidité plus élevées en le testant comme nous l'avons fait auparavant - en le tenant entre mes doigts. Les données sont transmises à la base de données, et les valeurs sont mises à jour en conséquence.

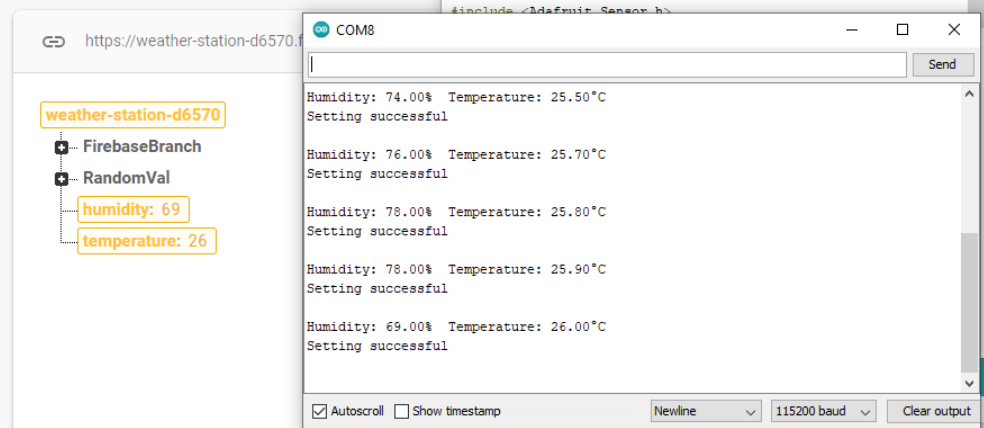


Figure 18 - Réponse de la base de données vue à côté de la fenêtre du moniteur série

Ensuite, nous devons connecter le capteur BMP180 à notre installation précédente. Le câblage est illustré à la Figure 21. J'ai choisi un schéma qui ne contient pas de breadboard pour plus de simplicité et une meilleure lisibilité.

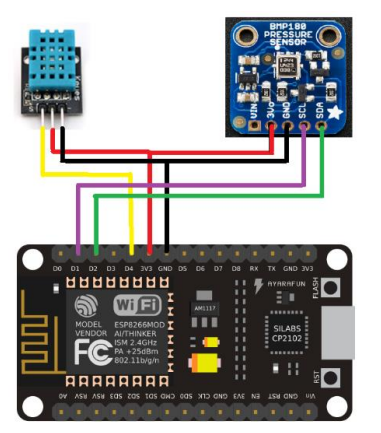


Figure 19 - Câblage des deux capteurs

5. L’application mobile

La façon dont j'ai choisi de traiter la grande quantité de données enregistrées par les capteurs (les capteurs enregistrent une valeur / 5 secondes) est la suivante : la valeur est enregistrée par le capteur. valeur / 5 secondes) est la suivante : J'ai créé une entrée pour chaque heure de la journée et tous les jours, la dernière valeur enregistrée par les capteurs pour cette heure spécifique est sauvegardée à jamais dans la base de données, mais nous pouvons toujours voir les valeurs actuelles des capteurs parce que la mise à jour se produit une fois toutes les 5 secondes ou à la pression sur le bouton Rafraîchir.

Maintenant, nous allons parler du développement de l'application Android.

Tout d'abord, je suis déjà un développeur des application mobile utilisation d'Android Studio. Après m'être familiarisé avec les contraintes, j'ai conçu l'interface de mon application. La première vue est la vue principale, où nous pouvons voir l'état actuel de nos capteurs. Le bouton d'actualisation met à jour le texte de la vue principale mais il n'est pas nécessaire de l'utiliser car l'application met à jour ces valeurs en permanence lorsque des changements apparaissent dans la base de données. valeurs constamment lorsque des changements apparaissent dans la base de données.

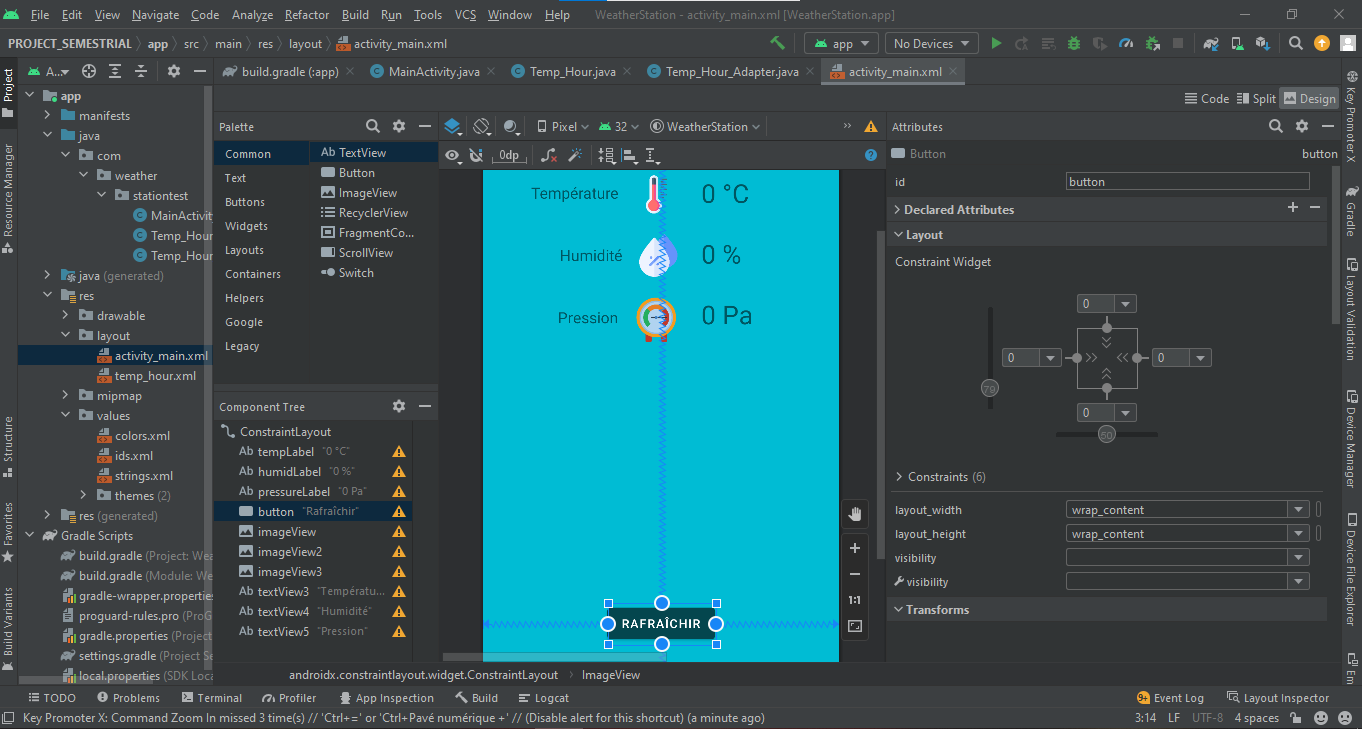


Figure 20 – Android Studio - activity\_main – Layout desing

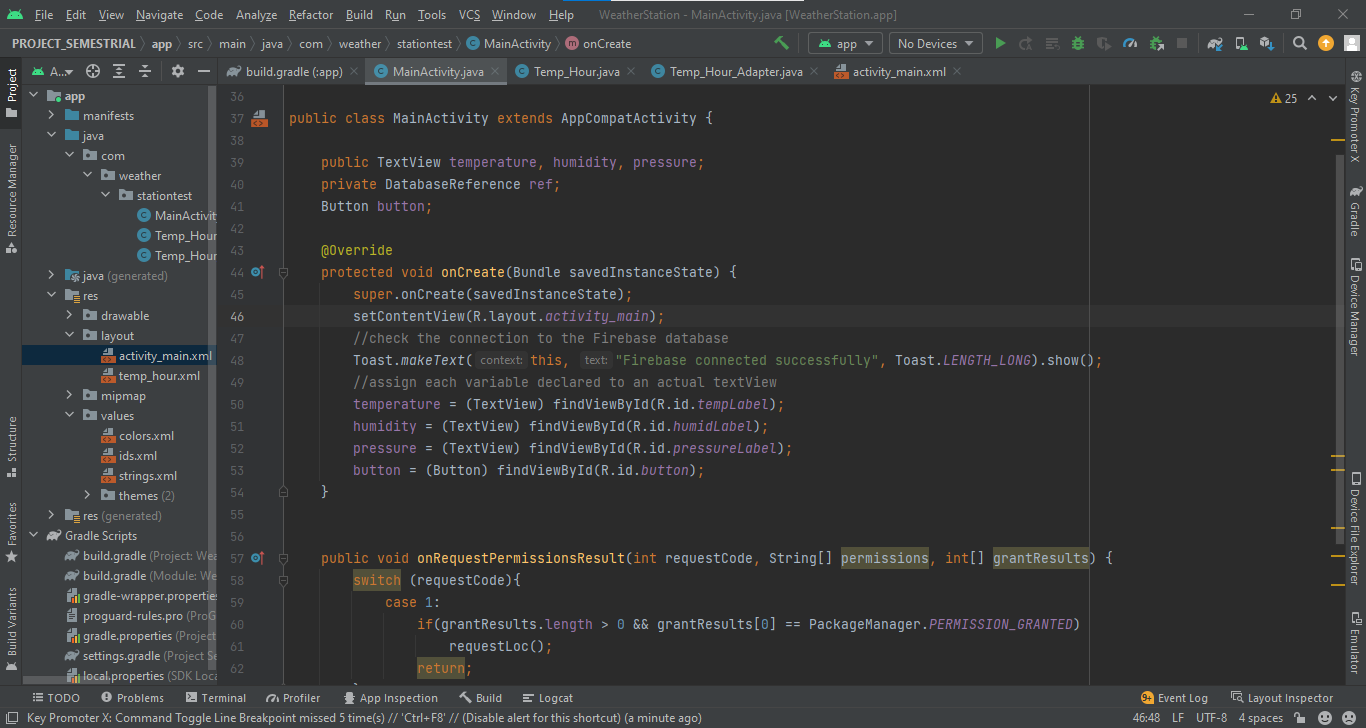


Figure 21 – Android Studio - MainActivity – Classe JAVA CODE

L'application est connectée à la base de données Firebase et doit en extraire des données. La mise à jour des informations se fait en temps réel se fait automatiquement ou en appuyant sur le bouton de rafraîchissement.

5. Conclusions

Le processus de création de cette application m'a permis d'améliorer mes connaissances en matière de programmation Arduino et développement mobile et IOT. concernant la programmation Arduino et la programmation Java. Le projet a été un succès car je sais maintenant où trouver les informations dont j'ai besoin pour réaliser des projets complexes. en combinant différents IDE pour réaliser une application utile.

6.Références Web

[1] - <https://learn.sparkfun.com/tutorials/bmp180-barometric-pressure-sensor-hookup-/all>

[2] - <https://github.com/FirebaseExtended/firebase-arduino>

[3] - <https://www.youtube.com/watch?v=TnWDlHpY56o>

[4] - <https://www.dfrobot.com/blog-910.html>

[5] - <https://www.instructables.com/Interface-DHT11-Humidity-Sensor-Using-NodeMCU/>

[6] - <https://desire.giesecke.tk/index.php/2018/01/30/esp32-dht11/>

[7] - <https://github.com/RobTillaart/Arduino/blob/master/libraries/DHTstable/dht.cpp>

[8] - <https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version1143054.pdf>

[9] - <https://lastminuteengineers.com/esp8266-ntp-server-date-time-tutorial/>

Mes Références

Lien de projet : <https://github.com/yousseffaouzi/Projet-IOT-Station-Meteo>

Lien de profile LinkedIn : <https://www.linkedin.com/in/youssef-faouzi-89710421a/>

Lien de profile Github : <https://github.com/yousseffaouzi>