Nicolas Soulard-Bouchard

Sébastien Jean

**TP3 : Station Météo**

Travail présenté à messieurs Alain Parent & Pierre-François Léon

Applications mobiles et objets connectés.

420-W46-SF

Département de la formation continue

Programmation, bases de données et serveurs

Cégep de Sainte-Foy

1er Mars 2021

Contexte

Dans le cadre du cours d’applications mobiles et objets connectés, nous avons reçu le mandat de concevoir une station qui permettra de prélever et envoyer des données concernant diverses informations relatives à la météo et ce, en temps réel. Les informations qui devront être collectées sont la température, la pression atmosphérique ainsi que le taux d’humidité dans l’air. La station devra être réalisée à partir d’un microcontrôleur ESP32, et d’un module BME280.

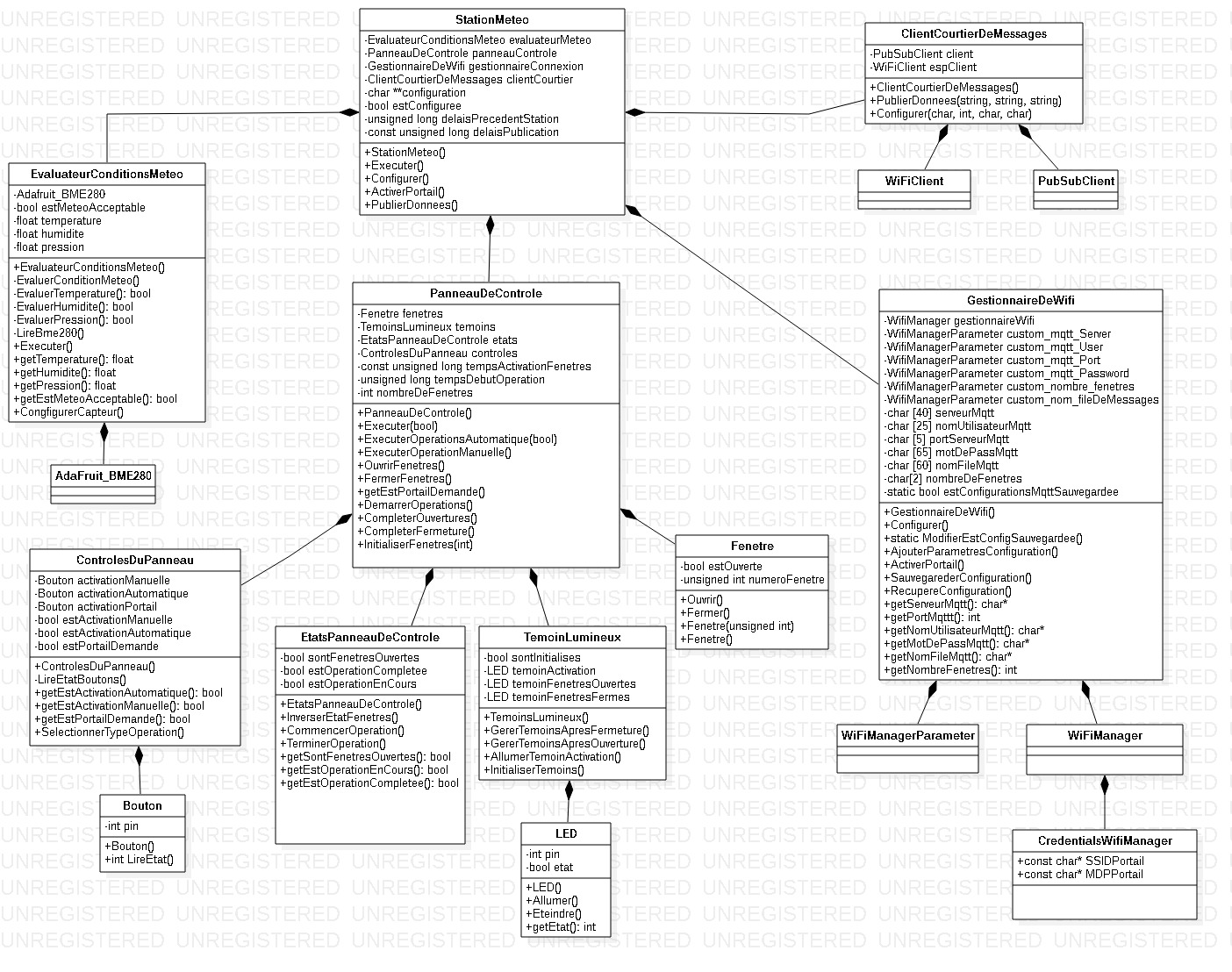
La station en question devra être capable de se connecter à un réseau Wifi afin d’envoyer les données qui seront collectées dans un courtier de messages, pour ensuite être récupérées et utilisées par un logiciel de domotique quelconque.

Des appareils devront ensuite être capable d’utiliser ces informations via le logiciel de domotique, afin de performer certaines actions ou tâches que nous pourrons nous-mêmes choisir.

Planification & attribution des tâches

|  |  |  |
| --- | --- | --- |
| **Planification et registre des heures\*** | | |
| **Tâche** | **Nom du membre de l’équipe** | **Durée de réalisation** |
| **Documentation** | | |
| Rédaction de la liste de tâches | Nicolas Soulard-Bouchard | 60 minutes |
| Rédaction de la page de présentation du document | Sébastien Jean | 5 minutes |
| Rédaction du contexte | Sébastien Jean | 30 minutes |
| Inventaire des pièces | Nicolas Soulard-Bouchard | 20 minutes |
| Réalisation du diagramme de classes | Sébastien Jean et Nicolas Soulard-Bouchard | 180 minutes |
| Réalisation et montage du vidéo explicatif | Sébastien Jean | 75 minutes |
| Réalisation de la publicité | Sébastien Jean | 60 minutes |
| Consultation de la documentation | Nicolas Soulard-Bouchard et Sébastien Jean | Au total pour les deux :900 minutes |
| Écriture du manuel d’utilisation pour configurer la station météo | Nicolas Soulard-Bouchard | 240 minutes |
| Explication du schéma | Nicolas Soulard-Bouchard | 60 minutes |
| **Montage schéma et plan** | | |
| Schéma sur Fritzing | Nicolas Soulard-Bouchard | 45 minutes |
| Plan sur Fritzing | Nicolas Soulard-Bouchard | 45 minutes |
| **Programmation** | | |
| Coder classe Bouton | Sébastien Jean | 15 minutes |
| Coder classe LED | Sébastien Jean | 15 minutes |
| Coder classe Fenêtre | Sébastien Jean | 15 minutes |
| Coder classe TemoinsLumineux | Sébastien Jean | 30 minutes |
| Coder classe ControlesDuPanneau | Sébastien Jean | 30 minutes |
| Coder classe EvaluateurConditionsMeteo | Sébastien Jean | 45 minutes |
| Coder classe EtatPanneauDeControle | Sébastien Jean | 15 minutes |
| Coder classe PanneauControle | Sébastien Jean | 120 minutes |
| Coder classe ClientCourtierDeMessage | Sébastien Jean | 15 minutes |
| Coder classe GestionnaireDeWifi | Sébastien Jean | 120 minutes |
| Coder classe StationMeteo | Sébastien Jean | 30 minutes |
| Écriture du code procédural | Nicolas Soulard-Bouchard | 120 minutes |
| Restructuration du code procédural afin de découper en méthodes | Nicolas Soulard-Bouchard | 30 minutes |
| Écriture du code (JSON, WifiParameter, Économie d’énergie, etc) | Sébastien Jean | 240 minutes |
| **Configuration des VM** | | |
| Installation et configuration de Home Assistant | Nicolas Soulard-Bouchard | 240 minutes \*\* |
| Installation et configuration de Home Assistant | Sébastien Jean | 90 minutes |
| Débugage | Nicolas Soulard-Bouchard | 60 minutes |
| Débugage et refactorisation | Sébastien Jean | 1000 minutes |
| \*Le registre des heures fait également office de liste de tâches et de registre des heures. Nous avons premièrement fait la liste des tâches et avons ajouté le temps nécessaire pour chacune d’elles lorsque celles-ci étaient complétées.  \*\* Après avoir essayé pendant environ 3 heures (seul et avec Alain Parent) à faire l’installation de Home Assistance sur un VMware Workstation Pro, j’ai finalement fait l’installation sur Oracle VM VirtualBox à la suite de conseil reçu de mes collègues de classe. | | |

Diagramme de classes



Inventaire des pièces

|  |  |  |
| --- | --- | --- |
| **Inventaire des pièces** | | |
| **Pièce** | **Quantité** | **Prix** |
| Microcontrôleur ESP32 | 1 | 13.99 $ |
| Capteur BME280 | 1 | 5.99 $ |
| Bouton poussoir | 3 | 0.25 $ |
| Résistances 1000Ω | 3 | 0.15 $ |
| Résistances 220Ω | 3 | 0.15 $ |
| LED (1x Rouge, 1x Verte, 1x Bleu) | 3 | 0.25 $ |
| Platine d’essaie | 1 | 15.00 $ |
| Câbles cavalier (longueurs & couleurs variées) | 20 | 0.25 $ / ch. |
| Total |  | 27.40 $ |

Consommation d’énergie

En ce qui concerne la consommation en énergie, nous avons deux composants principaux, soit le microcontrôleur et le capteur BME280. À cela s’ajoutent les composants visuels qui sont des DEL, des résistances et des boutons. Pour le calcul de la consommation des DEL, puisque la majorité du temps il n’y a qu’une DEL d’allumé et que la DEL bleu sera allumée un nombre de temps indéterminé, nous avons inclus une seul DEL dans nos calculs. En nous fiant à la documentation sur Internet et fournie par les professeurs, nous avons déterminé une intensité minimale de 160 mA et une intensité maximale de 260 mA. Dans le but de limiter la consommation énergétique, nous avons désactivé en continu le Bluetooth du microcontrôleur. Nous savons que l’autre composant qui consomme beaucoup d’énergie est le wifi. Nous voulions implanter un processus d’activation et de désactivation du wifi par intermittence (par exemple 30 secondes d’inactivité suivi par 10 secondes d’activité), mais par manque de temps, nous n’avons pas pu faire les modifications au code pour y arriver. Nous avons recherché la consommation du Bluetooth, mais malheureusement nous n’avons rien trouvé. La seule information que nous avons pu trouver est la consommation du Wifi, qui se situerait entre 80 et 180 mA. Sachant que le microcontrôleur consomme entre 27 et 44 mA, nous estimons la consommation du Bluetooth entre 36 et 53 mA.

Calcul de l’intensité des résistances :

Pour celles de 220 ohm :

**U = 3.3 volts R = 220 ohm I=? I = U/R 🡪 I = 3.3/220 = 0,015 A ou 15 mA**

**P = U \* I 🡪 P = 3.3 \* 0.015 P= 0.0495 W**

Pour celles de 1000 ohm :

**U = 5 volts R = 1000 ohm I=? I = U/R 🡪 I = 5/1000 = 0,0050 A ou 5 mA**

**P = U \* I 🡪 P = 5 \* 0,0050 P= 0.025 W**

Calcul de la consommation par jour:

Nombre de DELs : 1

**1 \* 0.0495 = 0.0495 W 0.0495 \* 24 = 1.188 W/jour 0.0000495 kWh**

Nombre de bouton : 3

**3\*0.025 = 0.075 W 0.075\*24 = 1.8 W/jour 0.000075 kWh**

Nombre de BME280 : 1

Intensité : 1.0 mA = 0.001 A

**P = U \* I 🡪 P = 3.3 \* 0.001 P= 0.0033 W**

**1\*0.0033 = 0.0033 W 0.0033 \*24 = 0.013200 W/jour 0.0033 KWh**

Nombre d’Arduino Esp32 : 1

Pour une intensité de 107 mA = 0.107 A pour une tension de 3.3 V

**P = U \* I 🡪 P = 3.3 \* 0.107 P= 0.353 W**

**1\*0.353 = 0.353 W 0.353 \*24 = 8.4744 W/jour 0.000353 KWh**

Pour une intensité de 224 mA = 0.224 A pour une tension de 3.3 V

**P = U \* I 🡪 P = 3.3 \* 0.224 P= 0.7392 W**

**1\*0.7392 = 0.7392 W 0.7392 \*24 = 17.7408 W/jour 0.0007392 KWh**

Calcul de la consommation annuelle:

Pour la DEL :

**((1.188 W/jour \* 365) / 1000) = 0.43362 KW / an**

Pour les boutons:

**((1.8 W/jour \* 365) / 1000) = 0.6570 KW / an**

Pour le capteur BME280 :

**(0.0132 W/jour \* 365) / 1000 = 0.004818 KW / an**

Pour le microcontrôleur d’Arduino Esp32 :

**Pour 3.3 V : (8.4744 W/jour \* 365) / 1000 = 3.0931 KW / an**

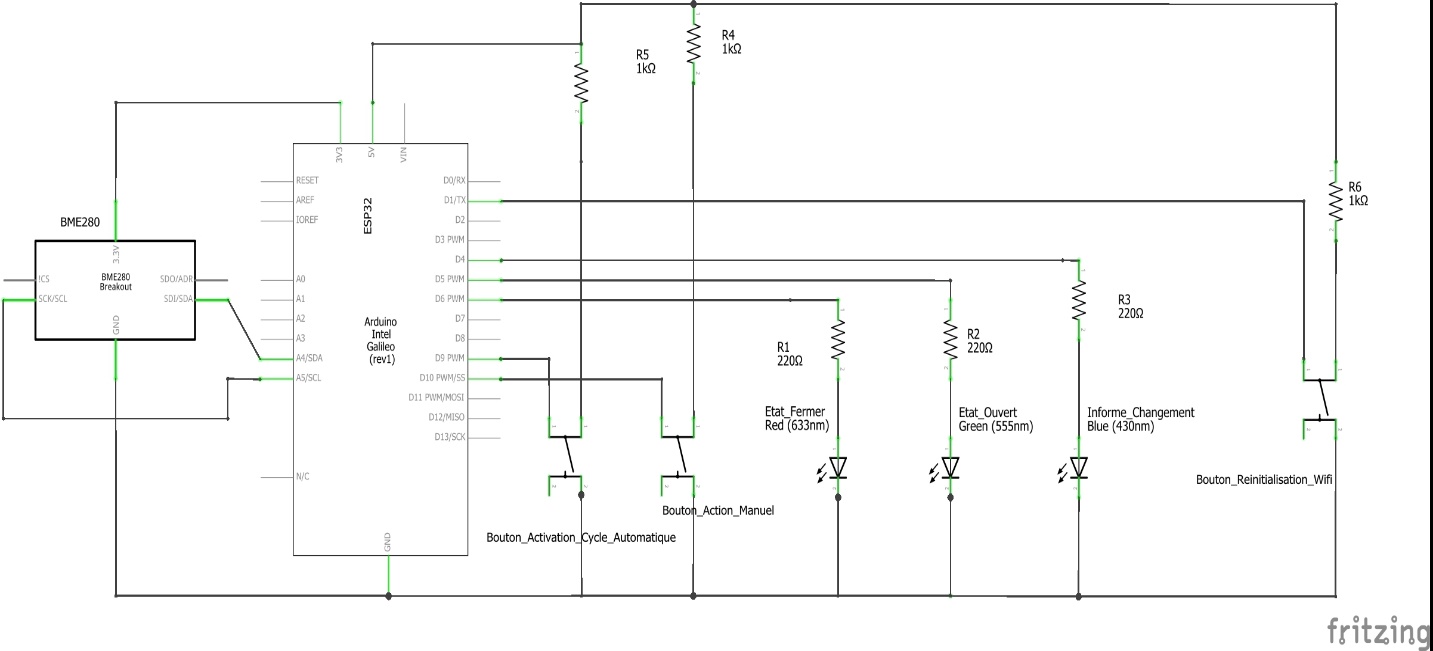
**Pour 3.3 V : (17.7408 W/jour \* 365) / 1000 = 6.7454 KW / an**

Tableau récapitulatif des consommations

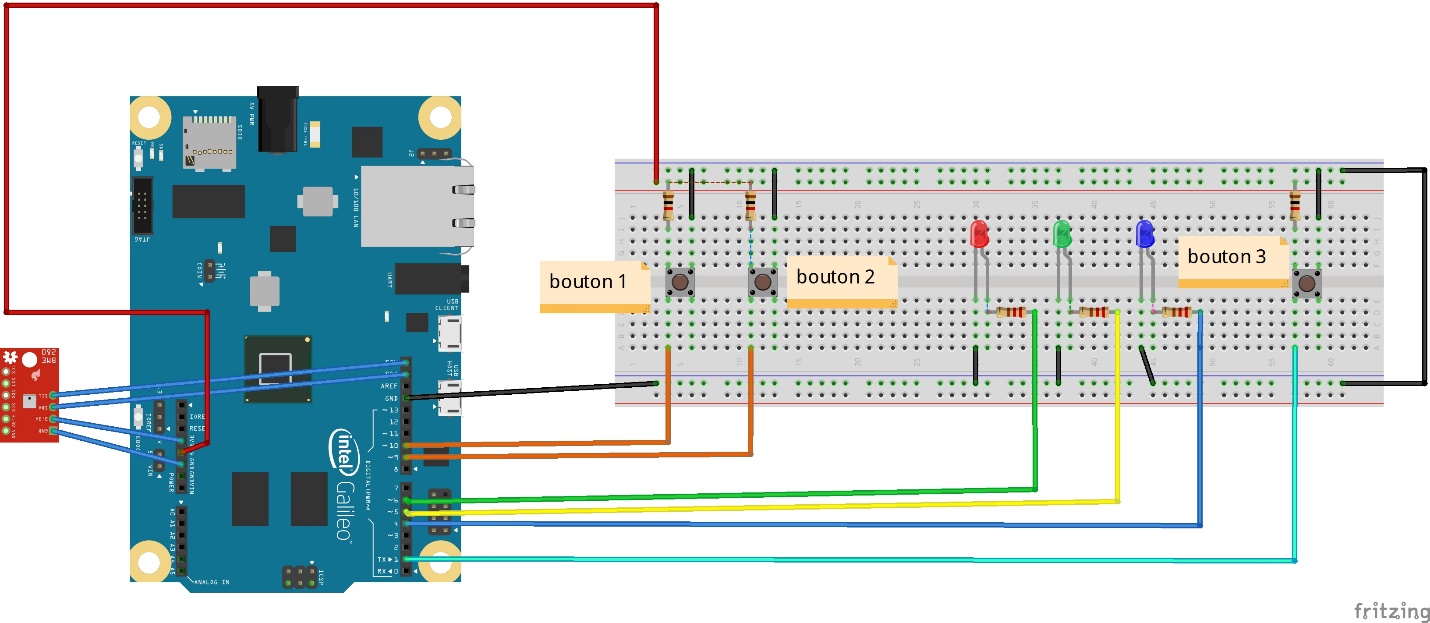
|  |  |  |  |
| --- | --- | --- | --- |
| **Consommation** | **Journalière (en Watt)** | **Mensuelle**  **(en watt)** | **Annuelle**  **(en watt)** |
| Pièce |  |  |  |
| LED | 1.188 | 35.64 | 433.62 |
| Bouton | 1.80 | 54 | 657 |
| BME280 | 0.0132 | 0.396 | 4.818 |
| ESP32 minimum | 8.4744 | 254.232 | 3093.156 |
| ESP32 maximum | 17.7408 | 532.224 | 6475.392 |
| **Total minimum** | **11.4756** | **344.268** | **4188.594** |
| **Total maximum** | **20.742** | **622.26** | **7570.83** |

Schéma technique et explication

Schéma :



Plan



Explication

En plus de la prise et l’envoi des informations demandées (température, pression et humidité) par le biais d’un messager pub-sub, nous avons décidé d’ajouter la fonctionnalité de la gestion des fenêtres (ouvrir/fermer). Nous avons un programme de base qui gère automatiquement les fenêtres selon des paramètres prédéterminés. Ainsi, l’ouverture des fenêtres s’effectue lorsque les conditions externes sont optimales. En cas de condition défavorable (temps trop chaud ou condition de pluie détectée), le contrôleur de fenêtres active la fermeture de celles-ci. Lorsque les fenêtres sont fermées, c’est le témoin lumineux rouge (DEL rouge) qui est allumé. Lorsqu’un changement est enclenché (soit ouverture ou fermeture), le témoin lumineux bleu (DEL bleu) est activé pendant 10 secondes afin d’informer d’un changement d’état des fenêtres. Enfin, lorsque les fenêtres sont ouvertes, c’est le témoin lumineux vert (DEL verte) qui est allumé. Notre station a trois boutons qui permettent une gestion de cette dernière. Le bouton complètement à gauche, bouton 1, est le bouton qui permet d’activer ou de désactiver le cycle automatique des fenêtres. Le bouton situé au centre gauche, bouton 2, permet de faire une gestion manuelle des fenêtres (ouvrir ou fermer) et désactive par le fait même le cycle automatique de gestion des fenêtres. Le dernier bouton situé complètement à droite, bouton 3, permet d’afficher le portail de WifiManager afin de permettre la modification de la connexion au réseau wifi. Nous savons que le microcontrôleur sur le plan et le schéma n’est pas le bon modèle, mais le modèle que nous utilisions n’est pas disponible sur Fritzing.

Manuel d’utilisation

Pour le manuel d’utilisation, nous avons fait un document à part puisqu’il est très détaillé.

Merci de consulter le document « Manuel d’utilisation ».

Références

<https://lastminuteengineers.com/bme280-esp32-weather-station/>

<https://randomnerdtutorials.com/cloud-weather-station-esp32-esp8266/>

<https://randomnerdtutorials.com/build-an-all-in-one-esp32-weather-station-shield/>

<https://custom-one.fr/station-meteo-qualite-de-lair-diy-arduino-mqtt/>

<https://forums.adafruit.com/viewtopic.php?f=19&t=138414>

<https://projetsdiy.fr/esp8266-dht22-mqtt-projet-objet-connecte/>

<https://projetsdiy.fr/esp32-test-librairie-wifimanager-gerer-connexions-wifi/>

<https://github.com/tzapu/WiFiManager>

<https://github.com/tzapu/WiFiManager/blob/master/examples/DEV/OnDemandConfigPortal/OnDemandConfigPortal.ino>

<https://www.instructables.com/ESP8266-and-ESP32-With-WiFiManager/>

<https://lastminuteengineers.com/creating-esp32-web-server-arduino-ide/>

<http://www.esp32learning.com/code/publishing-messages-to-mqtt-topic-using-an-esp32.php>

<https://iotdesignpro.com/projects/how-to-connect-esp32-mqtt-broker>

<https://randomnerdtutorials.com/esp32-mqtt-publish-subscribe-arduino-ide/>

<https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/protocols/mqtt.html>

<https://learn.sparkfun.com/tutorials/introduction-to-mqtt/all>

<https://techtutorialsx.com/2017/04/24/esp32-publishing-messages-to-mqtt-topic/>

<https://www.home-assistant.io/integrations/mqtt/>

<https://www.home-assistant.io/docs/configuration/devices/>

<https://www.home-assistant.io/getting-started/configuration/>

<https://www.home-assistant.io/docs/mqtt/broker>

<https://www.home-assistant.io/docs/configuration/yaml/>

<https://randomnerdtutorials.com/wifimanager-with-esp8266-autoconnect-custom-parameter-and-manage-your-ssid-and-password/>

<https://github.com/zhouhan0126/WIFIMANAGER-ESP32>

<https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/>

<https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/>

<https://randomnerdtutorials.com/esp32-mqtt-publish-bme280-arduino/>

<https://randomnerdtutorials.com/esp32-web-server-with-bme280-mini-weather-station/>

<https://github.com/bblanchon/ArduinoJson>

<https://projetsdiy.fr/esp32-code-arduino-sommeil-deep-sleep-reveils-timer-touch-pad-gpio/>

<https://letmeknow.fr/blog/category/wifi/>

<https://randomnerdtutorials.com/esp32-deep-sleep-arduino-ide-wake-up-sources/>

<https://phmarduino.wordpress.com/2018/07/18/station-de-mesures-v1/>

<https://gist.github.com/Schm1tz1/d5f4d34492509611846862cfdc786b66>

<https://forum.arduino.cc/index.php?topic=627063.0>

<https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/sleep_modes.html>

<https://diyi0t.com/reduce-the-esp32-power-consumption/>

<https://github.com/espressif/arduino-esp32/issues/882>

<https://randomnerdtutorials.com/esp32-timer-wake-up-deep-sleep/>

<https://lastminuteengineers.com/datasheets/esp32-datasheet-en.pdf>

<http://empa.com/empaiot/esp-wroom-32d_esp32-wroom-32u_datasheet_en.pdf>

<https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/sleep_modes.html>

<https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf>

<https://github.com/tzapu/WiFiManager/issues/1050>

<https://github.com/tzapu/WiFiManager/tree/master/examples>

<https://arduinojson.org/>

https://projetsdiy.fr/esp8266-webserveur-partie4-arduinojson-charger-enregistrer-fichiers-spiffs/

<https://byfeel.info/eeprom-ou-spiffs/>

<https://arduinojson.org/v6/example/config/>

<https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>

<https://www.mischianti.org/2020/01/26/manage-json-file-with-arduino-and-esp8266/>

<https://github.com/esp8266/Arduino/blob/master/libraries/esp8266/examples/ConfigFile/ConfigFile.ino>

<https://github.com/bblanchon/ArduinoJson/issues/1215>

<https://www.radioshuttle.de/fr/medias-fr/informations-techniques/esp32-alimente-par-batterie/>

<https://lastminuteengineers.com/bme280-arduino-tutorial/>