SERVEUR

Le serveur est un serveur HTTP et a été développé en PHP version 5.3.3. par Cyril Py ([cyril.py@imerir.com](mailto:cyril.py@imerir.com)).

Il est composé de quatre fichiers:

* connexion\_bdd.php -> définit les paramètres de connexion à la base de données
* server\_arduino.php -> communique avec l’Arduino Yun et la base de données
* server\_synchro.php -> communique avec l’Arduino Yun et la base de données
* server\_web.php -> communique avec le client et la base de données

Pour ce qui est des requêtes HTTP, le serveur reçoit des requêtes GET de la part de l’Arduino, et des requêtes POST de la part du client.

Le choix du POST pour l’échange avec le client vient du fait qu’en GET on peut transmettre moins de données lors de la réponse. Et lorsque le client demande des données sur une grande durée, le nombre de données retournées peut être très important. C’est afin d’éviter tout problème de taille de réponse que les requête se font en POST.

Connexion\_bdd.php

Ce fichier contient les informations relatives à la connexion à la base de données. Les informations contenues sont :

* l’adresse du serveur MySQL
* le nom de la base de données
* le nom de l’utilisateur
* le mot de passe

server\_arduino.php

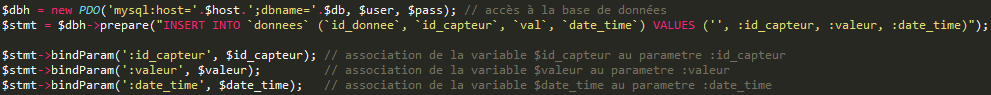
Ce fichier reçoit les données envoyées par l’Arduino via une requête GET et les insère dans la base de données. A ce jour, les données qui sont reçu sont :

* la température
* la pression atmosphérique
* l’hygrométrie
* la force et la direction du vent
* la pluviométrie
* le niveau de la batterie

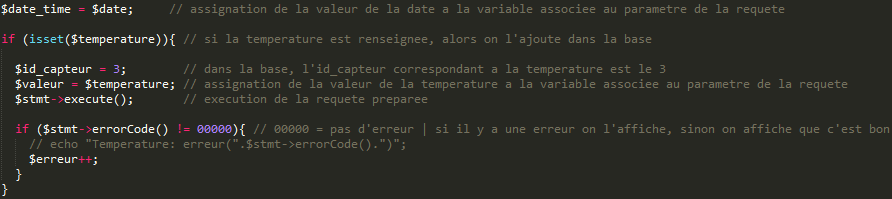
Le nom et le type des variables à envoyer sont listés dans le tableau ci-dessous.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type/Format | Capteur | Obligatoire |
| date | aaaa-mm-jj |  | X |
| heure | hh\_mm\_ss |  | X |
| temp | float | Température |  |
| fvent | float | Force du vent |  |
| dvent | float | Direction du vent |  |
| pluie | float | Pluviométrie |  |
| batterie | float | Tension en sortie de batterie |  |
| pression | float | Pression atmosphérique |  |
| humidite | float | Hygrométrie |  |

Les données sont insérées en base grâce à une requête préparée.



Cette requête est exécutée pour chaque capteur envoyé dans la requête HTTP.



Server\_synchro.php

Ce fichier sert à la synchronisation de la base de données de la station avec celle du serveur après une coupure internet. Les requêtes se font en GET.

Le nom et le type des variables à envoyer sont listés dans le tableau ci-dessous.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type/Format | Capteur | Obligatoire |
| date | aaaa-mm-jj\_hh-mm-ss |  | X |
| idcapteur | entier | Correspond à id\_capteur dans la base de données du serveur | X |
| temp | float | Température | X |

Server\_web.php

Ce fichier sert à renvoyer les données de la base de données au client au format JSON. Les requêtes http du client sont en POST.

Le nom et le type des variables à envoyer sont listés dans le tableau ci-dessous.

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type/Format | Description | Obligatoire |
| dateDebut | jj/mm/aaaa | Borne inférieure de l’intervalle de durée lors de la requête SQL | X |
| dateFin | jj/mm/aaaa | Borne supérieure de l’intervalle de durée lors de la requête SQL | X |
| capteur | string | Correspond au capteur dont les valeurs seront retournées au client | X |

L’accès à la base de données se fait via trois requêtes SQL.

La première sert à récupérer les 280 données les plus récentes de la base. 280 car il y a 7 capteurs en base, et pour avoir un affichage optimal sur le graphe, nous affichons 40 dates donc 7\*40=280 -> les 40 dernières données de chaque capteur. La requête imbriquée sert à trier les données dans l'ordre de la plus ancienne à la plus récente.

La seconde sert à récupérer la valeur la plus récente pour la batterie.

Et la troisième sert à récupérer toutes les données d’un capteur passé en paramètre par la variable « capteur », dans un intervalle de temps donné en paramètre par les variable « dateDebut » et « dateFin ».

Les données récupérées de la base sont mises dans un tableau de façon à ce que l’encodage du tableau en JSON donne le résultat attendu par le client. Soit : $json[date][capteur] = valeur pour donner le JSON : {« date » :{« capteur » : valeur }}.

La valeur de la direction du vent est convertie au moment de l’insertion dans le tableau.

La valeur de la tension en sortie de batterie est convertie en pourcent et arrondie à l’unité, afin d’améliorer la lisibilité par l’utilisateur final.

La tension maximale, à charge pleine, est de 13 et la tension minimale, à charge vide, est de 11.7. Le principe est donc :

* de ramener la tension entre 0 et 1.3 (13 – 11.7 = 1.3) en lui soustrayant la tension minimale (11.7).
* Et de calculer le pourcentage en prenant comme référence :
  + 0 → 0%
  + 1.3 → 100%.

Le calcul de cette conversion est le suivant :

( ( tensionAConvertir – tensionMin ) \* 100 ) / ( tensionMax – tensionMin )

Le résultat de ce calcul nous donne donc le niveau de charge de la batterie en pourcent.

BASE DE DONNÉES

La base de données a été réalisée par Mathias Da Costa (mathias.dacosta@imerir.com).

Capteurs

id\_capteur

nom

val\_min

val\_max

reference

unite

Donnees

id\_donnee

# id\_capteur

val

date\_time

0,n

1,1

Diagramme des échanges