RUBINO SEBASTIAN

RUBILAR CRISTIAN

Programmation d’applications en réseau

243-621-MA

**Projet Final**

**Rapport du projet**

**Application IoT sur Beaglebone Black**

Travail présenté à

*Karine Saladin*

Système Ordinés

Cégep Maisonneuve

10 Mai 2022

[**Description du projet**](#_cjvfyye5oxqu) **3**

[**Matériel**](#_h038up8kg7bm) **3**

[**Équipe et nom de l’application**](#_spyymbytll5z) **3**

[**Tâches semaine 1 à 2**](#_s9wiho8bjat8) **4**

[**Problèmes et solutions**](#_mq14vywp4b6q) **4**

[Problème 1) Accès au Beaglebone Black](#_2c2op7u0ue54) 4

[Solution 1) Travailler sur les tâches à venir](#_ta612cu885sw) 4

[Problème 2) Comment connecter le capteur au Beaglebone Black](#_qdexsk3rabw5) 5

[Solution 2) Trouver la fiche technique des composantes](#_16xhrp65d0z7) 5

[Problème 3) Affichage en temps réel](#_t29l617vxpr3) 5

[Solution 3) Chercher de l’aide auprès de nos enseignants](#_7bbd3su7lfm3) 5

[Problème 4) Restructuration des équipes](#_tbwkctkwd50s) 6

[Solution 4) Prendre le temps pour se mettre au travail](#_h019ov8is4y1) 6

[Problème 5) Connection du beagle bone au réseau](#_klb18y83b5qk) 6

[Solution 5) Changer de local](#_y3sjms1g9tqw) 6

[Problème 5) Changement d’adresse I2C](#_1kjhkfha5zx) 7

[Solution 5) Optimisation de nos connexions](#_g2opbepabwv) 7

[Problème 6) Implémentation du serveur APACHE](#_i7zjpfts9mrw) 7

[Solution 6) Copie d’une machine virtuelle](#_g1kra05o7ov8) 7

[**Tâches réalisées**](#_n7uzhjx27x6d) **8**

[**Prochaines étapes**](#_k5t45vs6nfu9) **8**

[**Problèmes et solutions**](#_xibfawgtwwqx) **9**

[Problème 1) Envoyer les variables vers HTML](#_82p1gt9n090c) 9

[Solution 1) Implémentation d’une fonction qui lit et écrit dans un fichier HTML](#_mf1z6bj05atp) 9

[Problème 2) Trouver le bon item à remplacer dans le HTML](#_haeacun81v5k) 10

[Solution 2) Utilisation d’un script externe pour trouver le numéro de nos items](#_kcfqeis3x6e9) 10

[Problème 3) Affichage décalé dans page HTML](#_ylq7bffcs0cc) 10

[Solution 3) Modification de paramètres](#_h7bnud8z2o9u) 10

[**Tâches réalisées**](#_ck8rmibsfuai) **11**

[**Conclusion**](#_uds63givxxvg) **11**

Semaine 1 et 2

# Description du projet

Dans le cours de Programmation d’applications en réseau, nous arrivons au projet final qui a pour but de créer une application Iot sur notre carte contrôleur Beaglebone Black. Cette application va avoir besoin de l'intégration d’un protocole de communication entre un serveur et un client. Le résultat devrait nous donner une page web dynamique (qui fait des mises-à-jour automatiques) qui affiche les valeurs d’un capteur de température, d’humidité et de pression en temps réel.

# Matériel

* Carte Beaglebone Black
* Carte capteur BME 280
* Fil Ethernet

# Équipe et nom de l’application

Comme convenu la semaine passée avec notre enseignante, nous allons travailler en équipe de 3 pour la première semaine. Du coup, pour la semaine 1, voici les membres de l’équipe:

* Rubilar Cristian
* Rubino Sebastian

Nom de l’application : Foggy Breezy

# Tâches semaine 1 à 2

* Former des équipes
* Trouver nom d’application
* Établir une communication en continu entre le Beaglebone Black et le serveur
* Documenter les tâches réalisées
* Planifier les tâches à venir
* Rédiger le journal (maintenir à jour)

# Problèmes et solutions

Voici une liste de problèmes rencontrés lors de notre première semaine de travail:

## Problème 1) Accès au Beaglebone Black

Durant la semaine 1 du projet, nous n’avions pas eu accès au contrôleur. Les cartes sont limitées à 1 par équipe de deux et un enseignant dans un autre cours les avait pris pour corriger un travail fait auparavant. Nous avons eu accès à ces cartes à partir de vendredi soir. Du coup, le travail qui devait être ciblé durant la semaine 1 a été impossible à faire.

## Solution 1) Travailler sur les tâches à venir

La solution la plus simple que nous avons trouvé est très logique. Nous avons décidé d’avancer ce qui allait devoir se faire éventuellement. Pour être plus clair, nous avons travaillé sur:

* Notre page html afin d’avoir un template organisé, propre et professionnel.
* Trouver comment afficher sur html des valeurs qui changent en temps réel.
* Trouver (théorique) comment récupérer les valeurs du capteur sur le Beaglebone.
* Afficher les valeurs du capteur sur le terminal

## Problème 2) Comment connecter le capteur au Beaglebone Black

C’est la première fois que nous travaillons avec cette carte de cette façon. Du coup, on n’avait aucune idée de comment on était censé connecter le capteur sur notre carte contrôleur. Il faut faire beaucoup d’attention pour éviter d' endommager les composantes électroniques.

## Solution 2) Trouver la fiche technique des composantes

Malgré que ce problème n’est pas si compliqué, il faut faire attention de ne pas se tromper car, une erreur de branchement pourrait nous coûter la vie des pièces. Du coup, il nous a suffit d’aller voir sur internet la fiche technique de notre carte Beaglebone Black et de notre capteur pour comprendre les branchements à faire.

## Problème 3) Affichage en temps réel

C’est la première fois que nous devons faire une page web dynamique. Toutes les fois qu’on a dû créer une page en html, c’était pour créer des pages avec des valeurs statiques. Du coup, on n’a aucune idée de comment on doit s'y prendre pour faire une page web qui va avoir des valeurs dynamiques, c’est-à-dire des valeurs qui vont se mettre à jour sans que l’usager soit obligé de rafraîchir le browser.

## Solution 3) Chercher de l’aide auprès de nos enseignants

Nous avons fait beaucoup de recherches sur différents forums, sur youtube, etc. Chaque personne avait des méthodes différentes. Quelques personnes nous conseillent d’utiliser javascript (celui-ci on l’a jamais vu de notre technique), d’autres nous ont conseillé d’utiliser Flask-Live Server (on l’a essayé sans avoir de succès). Du coup, la solution qui semble être plus prudente est celle d’attendre le moment pour demander à notre enseignante de mettre ça au clair car, nous sommes super perdus par rapport à cela.

## Problème 4) Restructuration des équipes

Au début de la semaine deux, Cristian est revenu en classe et on a dû changer les équipes. Au commencement l’équipe était formée par Antoine, Simon et Sebastian , mais avec l’arrivée de Cristian, Simon et Antoine se sont formé une équipe et Cristian et Sebastian ont formé une nouvelle équipe. Une bonne partie du travail étant déjà faite, il a fallu mettre à jour le nouveau coéquipier.

## Solution 4) Prendre le temps pour se mettre au travail

On a passé deux après-midis à parler du projet et du fonctionnement de celui-ci. Sebastian m’a montré comment il avait écrit le code et ce que chaque ligne avait pour but dans le fonctionnement du programme.

## Problème 5) Connection du beagle bone au réseau

Durant la semaine 2, nous sommes venus au local 309 pour tester notre serveur et notre client. Pour y arriver, il fallait établir un lien entre le Beaglebone Black et notre machine virtuelle via le réseau local. Nous avons essayé de faire les connexions nécessaires pour faire un ping request mais, on a pas été capables. En demandant à un enseignant, nous avons appris qu’il était possible que notre contrôleur soit bloqué par un firewall.

## Solution 5) Changer de local

Dans le cours de Systèmes d'exploitations, nous n'avons jamais eu de problème de ce genre. Du coup, Cristian a eu l’idée de changer de local et d'essayer une nouvelle prise ethernet. Ce fut une grande surprise de voir qu’il avait bien raison. Une fois ce problème réglé, ça a bien fonctionné pour le reste du travail.

## Problème 5) Changement d’adresse I2C

Pour communiquer avec notre capteur BME280 nous avons dû établir une connexion en I2C. Du coup, dans le code du programme nous devons lui indiquer une adresse à laquelle communiquer. Si cette adresse change, le code ne marche plus. Le problème qu’on avait, c’était que cette adresse n’arrêtait pas de changer constamment à chaque fois qu’on démarrait le système.

## Solution 5) Optimisation de nos connexions

En parlant avec Émile, nous avons appris qu’il est important de ne pas laisser des pins non connectées car, une pin flottante peut donner ce genre de phénomène. Du coup, en revoyant le schéma de connexion, nous avons réglé ce problème.

## Problème 6) Implémentation du serveur APACHE

Un problème rencontré, auquel nous n'avons jamais su son origine, se fut de créer un serveur 100% fonctionnel sur APACHE. Bien évidemment, en étant quelque chose de nouveau pour nous, le défi existait déjà à la base. Par contre, la machine virtuelle que j’avais installée (Sebastian) en début de session me donnait des problèmes pour ouvrir et modifier des fichiers sudo. Autrement dit, c’est comme si, lors de l’installation de virtualbox, j'avais manqué des étapes quelque part. Du coup, impossible pour moi et cristian de créer notre serveur APACHE car, on ne pouvait pas utiliser les pouvoirs sudo.

## Solution 6) Copie d’une machine virtuelle

Chez moi (Sebastian), j’avais une autre machine virtuelle que j’avais aussi installé en début de session, dans laquelle je n’avais jamais rencontré de problèmes de ce genre auparavant. Du coup, j’ai essayé de créer un serveur APACHE chez moi et cela a fonctionné à 100%. Il a seulement fallu que je crée une copie de ma machine virtuelle pour l’amener au cégep et travailler sur celle-là. De même, au cégep, nous n’avons plus jamais eu ce genre de problèmes et on a pu installer APACHE pour continuer notre projet.

# Tâches réalisées

En raison de la restructuration des équipes dans la semaine deux, nous allons mettre l’emphase sur les avancements faits lors de la deuxième semaine.

* Établir la connexion entre le client et le serveur. (Sebastian)
* Réussir à récupérer les valeurs données par le capteur BME280 et les traiter(Cristian)
* Installer et appliquer Apache Web Server (Sebastian)
* Rédaction du journal (Sebastian et Cristian)

# Prochaines étapes

Maintenant que nous avons accès au matériel, nous allons pouvoir travailler sur les tâches qui étaient censées se faire la semaine passée.

* Développer une bonne page html pour afficher les données
* Traiter les données pour créer des statistiques

Semaine 3

# Problèmes et solutions

## Problème 1) Envoyer les variables vers HTML

Le plus gros problème rencontré lors de cette semaine, ce fut de trouver un moyen pour envoyer les valeurs de notre capteur vers la page HTML. Nous avions bien essayé avec render\_templates (fonction qui appartient à Flask) ainsi que la solution proposée par Sebastian lors du TP2 (créer un fichier HTML directement dans le script python et envoyer les variables directement de façon local). Cependant, le fait d’ajouter une fonctionnalité de google charts, pour créer des graphiques de nos données, a fait en sorte que ces solutions (qui fonctionnaient à la base) ne soient plus fonctionnelles. Du coup, nous avons dû faire démonstration de créativité et on a trouvé une solution différente.

## Solution 1) Implémentation d’une fonction qui lit et écrit dans un fichier HTML

Effectivement, la solution trouvée démontre que nous avons quand même une bonne compréhension des outils disponibles sur python. Nous avons implémenté une fonction capable de lire un fichier HTML et de lire son contenu. Une fois ceci fait, on procède à la création d' une liste de chaque mot trouvé dans le HTML. En faisant cela, c’est un jeu d’enfants remplacer un item par la valeur d’une variable local (là où on enregistre les valeurs de notre capteur). Par exemple, si dans le HTML nous avons une variable de température comme ça:

**<P2> {Var\_temp} </P2>**

Alors dans notre fonction, on est capable de voir ceci:

**[<P2>]**

**[{Var\_temp}]**

**[</P2>]**

Pour ensuite changer l' item par notre variable:

**float X = 22.0**

**[{Var\_temp}] = str(X)**

Cela va finalement se traduire par: **<P2> 22.0 </P2>**

## Problème 2) Trouver le bon item à remplacer dans le HTML

Une fois que nous avons trouvé la solution du problème 1, nous sommes tout de suite tombés sur le deuxième problème. On ne savait pas quel numéro d’item nous étions censés changer. On les a compté et on a fait un test pour trouver que nous avions changé des trucs dans la structure du HTML que nous n’étions pas censés changer. Nous sommes revenus sur nos pas pour faire un brainstorming.

## Solution 2) Utilisation d’un script externe pour trouver le numéro de nos items

Après quelques minutes, Cristian est arrivé à une solution! Il a fait un script pour compter le nombre de chaque item. Par exemple: [abc] - 1, [def] - 2, [ghi] - 3, etc. Du coup, comme ça nous avons pu trouver exactement le numéro d’item que nous étions censés changer.

## 

## Problème 3) Affichage décalé dans page HTML

Finalement, tout fonctionnait, par contre quelque chose que nous avions pas prévu c'est la proportion du texte en lien avec les images que nous avons conçu. Autrement dit, le texte avec les valeurs était beaucoup trop large pour l’espace que nous avions pensé auparavant. Donc, niveau propreté et esthétique, la page web est devenue impossible à lire.

## Solution 3) Modification de paramètres

Vu que nous avions déjà eu de l'expérience avec ce genre de problème, nous avons tout de suite su quoi et où le régler. Il nous a suffit de jouer avec nos paramètres de margin left, margin top et la taille de texte pour que tout puisse rentrer dans son espace attribué. C’est ainsi que la page est redevenue lisible et propre comme vous pouvez le voir dans la section de Photos plus bas.

# Tâches réalisées

* Photos et fichiers HTML. (Sebastian)
* Script google charts (Sebastian)
* Script pour trouver des items (Cristian)
* Mise en forme des HTML (Cristian)
* Mise-à-jour du script python (Sebastian et Cristian)
* Présentation devant enseignante (Sebastian et Cristian)

# Conclusion

Ce projet était dû pour se faire en 4 semaines de cours. Cependant, notre équipe a pu le faire en 3 semaines car, nous avons vraiment mis nos cœurs et fait beaucoup d’efforts. Si bien, nous n’avons pas implémenté Flask dans notre projet (dû à une mauvaise compréhension du travail), nous avons quand même été les premiers à plonger dans l’inconnu pour rompre les barrières de nos connaissances et être les premiers à comprendre comment utiliser APACHE. Cela démontre notre compromis avec la matière. Cela peut aussi se voir dans le détail que nous avons mis dans l’esthétique de nos pages HTML, toutes les images ont été faites à la main pour donner une meilleure qualité visuelle à notre projet. Nous sommes très fiers de notre travail qui est une vraie source d’inspiration.

En conclusion, nous avons trouvé ce projet très enrichissant.

Merci et surtout n’oubliez pas de visiter les sections plus bas! On y voit des photos de nos résultats ainsi que nos sources.

À une prochaine occasion!

Photos

Image qui montre notre page web (www.foggybrezzy.com/accueil.html)

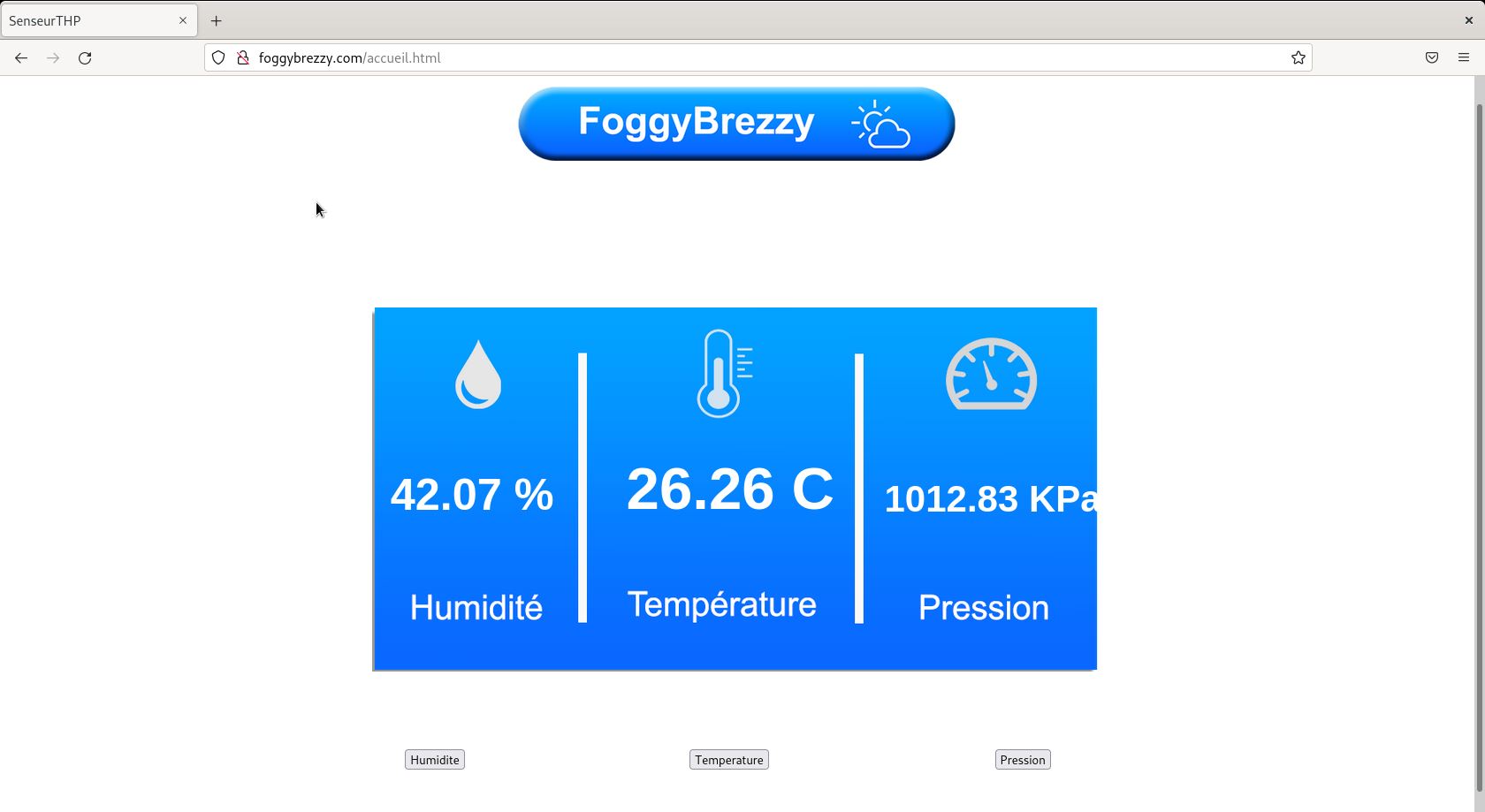


Image qui montre notre page web (www.foggybrezzy.com/historique\_hum.html)

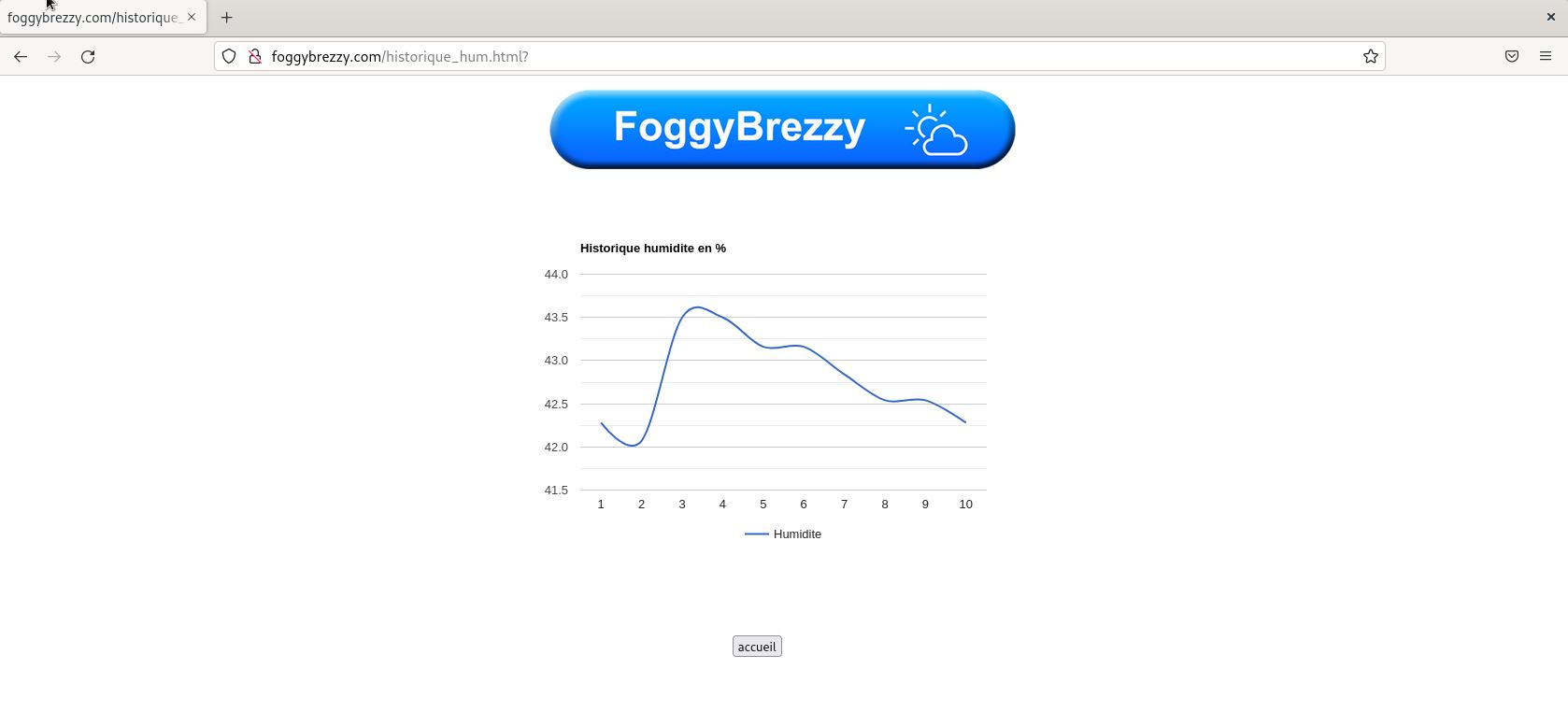


Image qui montre notre page web (www.foggybrezzy.com//historique\_press.html)

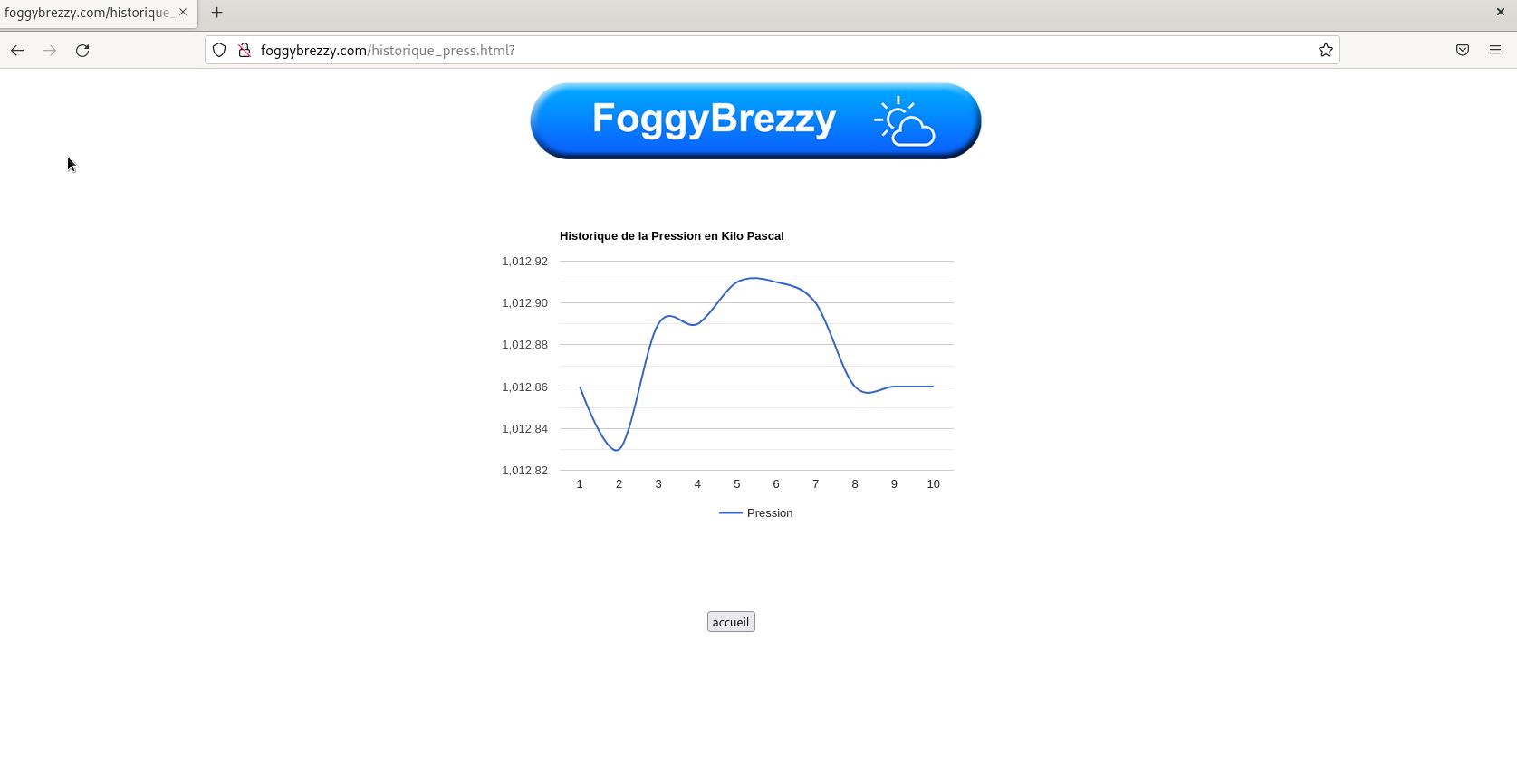
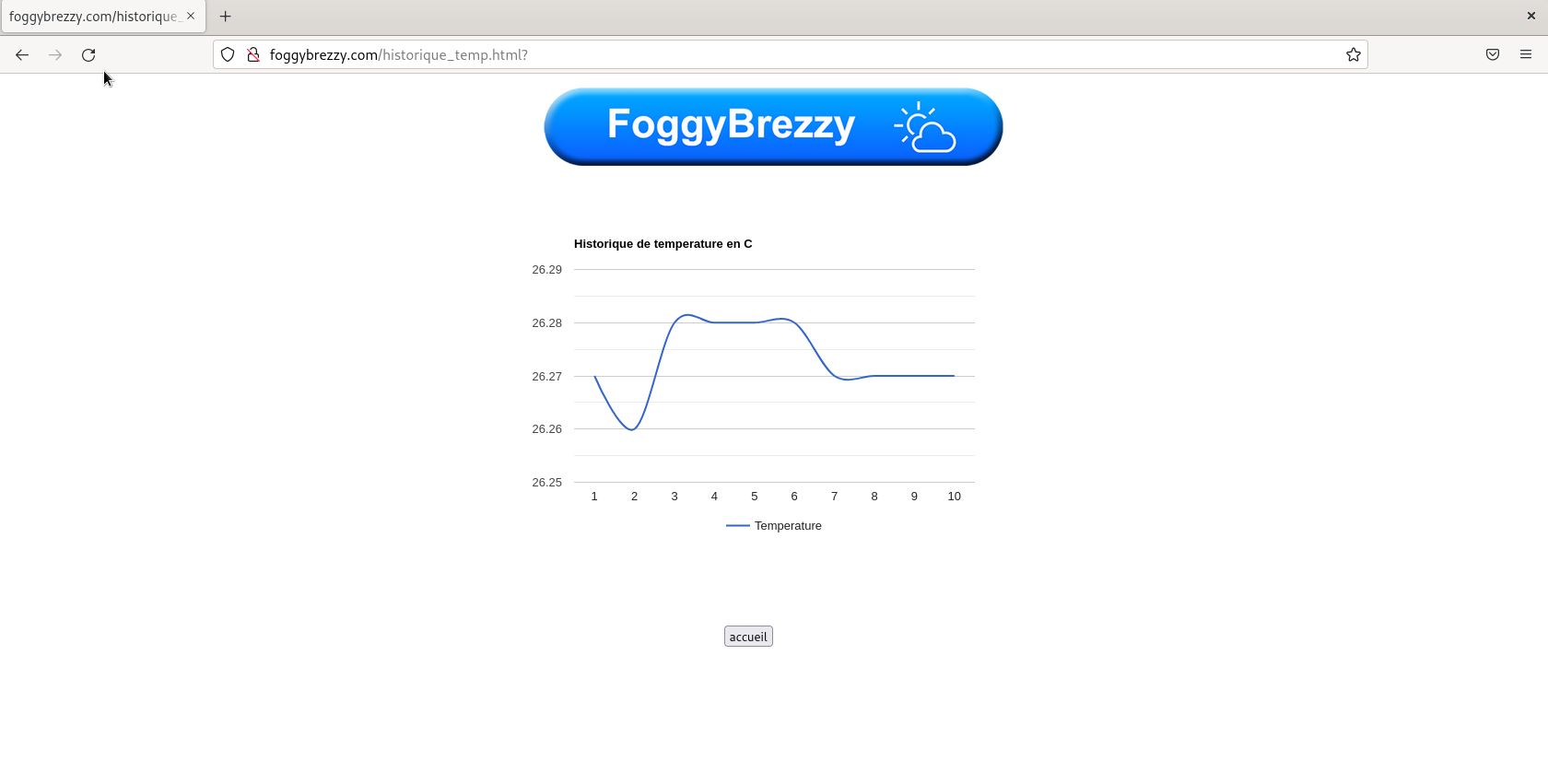


Image qui montre notre page web (www.foggybrezzy.com//historique\_temp.html)



Ressources

Dans notre travail nous avons cherché de l’aide auprès de plusieurs ressources.

* Serveur et client TCP : cours de Julien Regis Boriasse
* Serveur Apache : <https://www.youtube.com/watch?v=-q8Jj4aAWYw>
* Recupérer les valeurs du capteur: <https://www.instructables.com/Weather-Station-Using-Raspberry-Pi-With-BME280-in-/>
* Créer des graphiques sur HTML: https://www.w3schools.com/ai/ai\_google\_chart.asp