

Baptiste Chevallier, Thomas Goward, Stéphane Ho Sik Chuen, Mikaël Petit, Gaël Pommé

**Rapport d’AWAIE**

TAHITI

Date de publication

15/01/2019

Adresse

3 rue de la Chocolaterie, 41000 Blois

Table des matières

[Introduction / Cahier des Charges 2](#_Toc535344538)

[Entreprise / Contexte 3](#_Toc535344539)

[Organisation de travail 3](#_Toc535344540)

[I. Architecture envisagée 3](#_Toc535344541)

[II. Planning prévisionnel 4](#_Toc535344542)

[III. Fonctionnalités souhaitées 4](#_Toc535344543)

[Réalisation 5](#_Toc535344544)

[I. Charte graphique 5](#_Toc535344545)

[II. Design retenu 5](#_Toc535344546)

[III. Intérêt et utilisation de Bootstrap 6](#_Toc535344547)

[IV. Architecture de programmation 7](#_Toc535344548)

[V. Base de données 8](#_Toc535344549)

[MCD 8](#_Toc535344550)

[MLD 8](#_Toc535344551)

[Création de la base de données 9](#_Toc535344552)

[Déclencheur automatique 9](#_Toc535344553)

[VI. Réalisation des pages dynamique 10](#_Toc535344554)

[Connexion à la base de données 10](#_Toc535344555)

[Récupération des données de température (Php) + Affichage (Javascript) 11](#_Toc535344556)

[Pour aller plus loin : création de jeton d’authentification 12](#_Toc535344557)

[Réalisation de la connexion avec les sondes de température 13](#_Toc535344558)

[I. PHPModbus 13](#_Toc535344559)

[II. Configuration du réseau local 13](#_Toc535344560)

[III. Récupération des températures 13](#_Toc535344561)

[Ajout d’un serveur local type Raspberry Pi 15](#_Toc535344562)

[I. Nouvelle architecture 15](#_Toc535344563)

[II. Installation du serveur sous Raspbian 15](#_Toc535344564)

[III. Connexion ssh / VNC viewer / sftp 16](#_Toc535344565)

[Configuration ssh : 16](#_Toc535344566)

[Connexion via VNC Viewer : 18](#_Toc535344567)

[Connexion sftp : 19](#_Toc535344568)

[IV. Connexion aux sondes de température 20](#_Toc535344569)

[V. Récupération des données de température en temps réel 21](#_Toc535344570)

[Application Web 22](#_Toc535344571)

[I. Application Android 22](#_Toc535344572)

[II. Application IOS 22](#_Toc535344573)

[Perspectives d’amélioration / Conclusion 22](#_Toc535344574)

[ANNEXE 23](file:///E:\wamp64\www\AWAI\Livrable\Rapport\Rapport_final.docx#_Toc535344575)

# Introduction / Cahier des Charges

Dans le cadre du cours d’Architecture Web et Automate Intranet Extranet, nous devons réaliser un projet de visualisation de température par le biais de sonde. Le but de ce projet est de pouvoir visualiser ses températures via une interface web. Nous avons à notre disposition un boitier contenant :

* 4 sondes de température
* Un module de communication ModBus (C0800 de AIM-PLC)

<http://www.aim-plc.com/pdf/Fiche%20Tech%20C0800.pdf>

* Un automate programmable (AT3200 de AIM-PLC)

<http://www.aim-plc.com/wp-content/uploads/2016/09/RV-AT3200-AT3300-BD-1.pdf>

* Un switch/hub

L’architecture du boitier se présente sous la forme suivante :

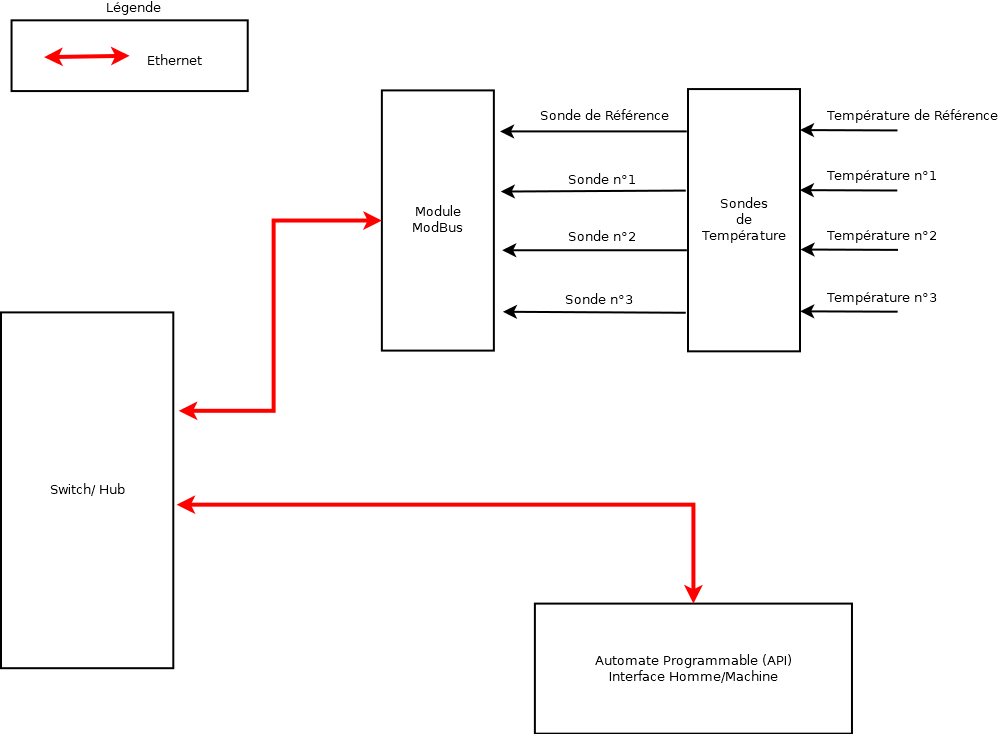


Figure 1: Architecture du boitier au commencement

Le but étant de se connecter en Ethernet sur le switch/hub afin de pouvoir récupérer les informations de température.

Afin d’amener à bien ce projet, nous avons créé une entreprise TAHITI dont l’intérêt majeur est de fournir à ses clients des données de température.

# Entreprise / Contexte

Notre entreprise s’appelle TAHITI pour Transfert Automate HTML Intégration et Traitement de l’Information.

Notre entreprise a pour but de fournir à ses clients une solution clé en main en matière de capteur atmosphérique tel que la température, pression atmosphérique, humidité de l’air (ou taux d’hydrométrie). Bien entendu nos principaux clients sont des entreprises de prévisions météorologique mais pas que, nous avons aussi dans notre panel des industriels conscient que la température notamment est un aspect crucial dans leur secteur d’activité tel que l’agroalimentaire ou divers secteurs industrielles nécessitant une attention accrue sur la température ou la pression d’enceinte.

Nous sommes 5 développeurs au sein de l’entreprise :

* Baptiste Chevallier, le développeur le plus expérimenté et pilote du projet
* Thomas Goward, développeur spécialisé dans l’IoT
* Stéphane Ho Sik Chuen, développeur web spécialisé
* Mikaël Petit, développeur web débutant
* Gaël Pommé, développeur web junior

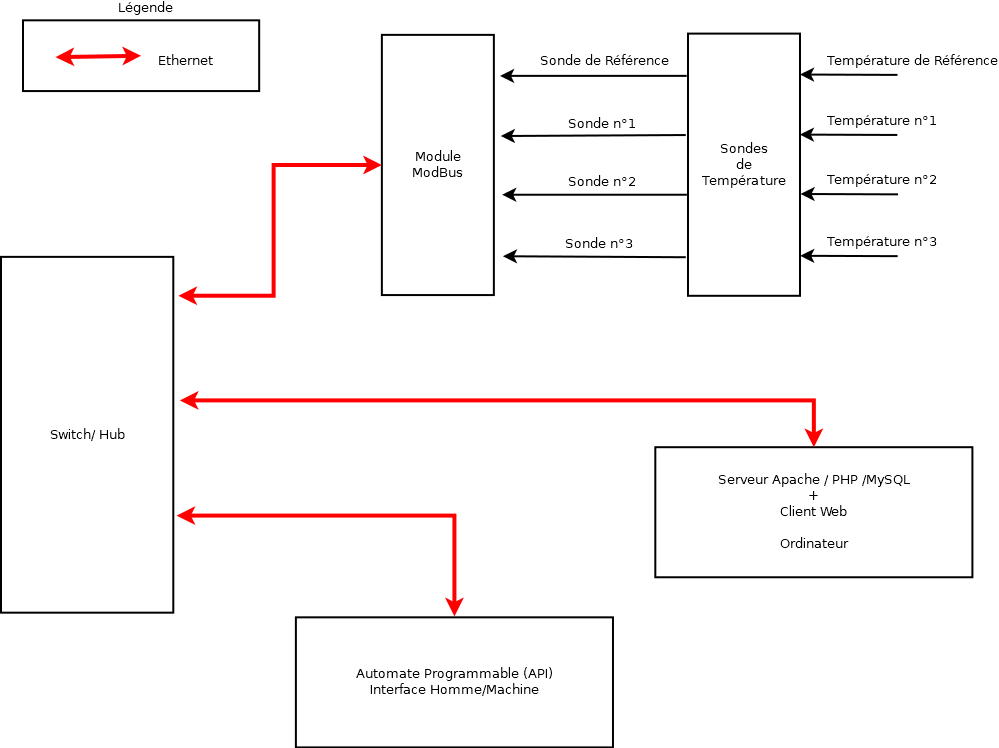
# Organisation de travail

## Architecture envisagée

Dans un premier temps, nous avons envisagé la solution la plus simple afin de visualiser ces données de température, c’est-à-dire en branchant directement l’ordinateur sur le switch/hub. Cette solution est pratique dans un premier temps car elle permet un prototypage rapide d’une solution cependant son intérêt reste limité car l’ordinateur reste relativement proche des sondes de température. Cela ne nous convenait pas car la supervision reste limitée. Nous souhaitions pouvoir superviser depuis un seul ordinateur plusieurs sondes de température situées à différents lieux sur le globe. Nous avons donc rajouté un server tampon qui récolte les données de température à intervalle régulier. Le pc se connecte toujours par Ethernet sur le boitier, cependant il communique avec le server et non le module de communication ModBus des sondes.

Etant limité qu’un unique boitier il n’est pas nécessaire d’avoir un server maître supplémentaire afin de contrôler les autres sondes.

L’architecture à minima est donc la suivante :

Figure 2 : Architecture souhaitée a minima

## Planning prévisionnel

Nous avons réalisé un planning prévisionnel pour mener à bien ce projet. Pour cela nous avons réalisé un diagramme de Gantt. (cf Annexe 1)

## Fonctionnalités souhaitées

Lors de la première réunion, nous avons défini les fonctionnalités principales que nous voulions intégrer à notre application web tel que :

* La possibilité de s’enregistrer directement
* De pouvoir s’identifier
* De sécuriser la transmission des données entre l’utilisateur et le serveur Web
* De relier un utilisateur à une entreprise qui fait appel à nos services
* D’indiquer à l’utilisateur, la position d’une sonde précise
* De visualiser à l’utilisateur toutes les sondes dont il a accès à travers son affiliation à une entreprise
* De pouvoir visualiser tous les capteurs reliés à celle-ci
* Que les administrateurs de l’application web accèdent à l’ensembles des données quel que soit l’entreprise
* De pouvoir extraire dans un fichier Excel toutes les données de température relative à un capteur particulier.
  + - * De pouvoir visualiser des températures incohérentes ou dépassant un seuil réglable par l’utilisateur
      * L’utilisateur pourra ignorer une alerte
      * De laisser à l’utilisateur la possibilité de modifier des données physiques liés au capteur comme sa localisation ou les seuils de déclanchement des alertes
      * Les alertes devront prévenir l’utilisateur et les administrateurs de la non réception de données de température durant un temps prédéfini (possible coupure, disfonctionnement, ou extinction de la sonde)
      * Que l’utilisateur puisse envoyer des messages directement au service client de notre entreprise
      * D’intégrer une application Android de notre application Web (BONUS)

# Réalisation

## Charte graphique

Nous avons déterminé une charte graphique pour tous les documents et page web afin de standardiser nos différents travaux quel que soit la personne qui la rédige. Nous avons donc rédigé un document regroupant toutes les informations nécessaires pour la réalisation visuel du site et documents, comme les cartes de visites, en-tête de mail et lettre, etc.

La charte est disponible à l’annexe 2.

## Design retenu

Ainsi nous avons un site ressemblant à ceci :



Figure 3 : Page d'accueil du site



Figure 4 : Page d'un capteur

## Intérêt et utilisation de Bootstrap

Bootstrap est une collection d’outils dédié à la création du design (graphisme, animation et interaction avec la page) de sites et d’application web (Définition Wikipédia).

Bootstrap a été développé par deux ingénieurs du réseau social Twitter. C’est une bibliothèque Libre sous licence MIT (licence libre de droits, open source et non copyleft). Cette licence donne à toute personne recevant le logiciel le droit illimité de l'utiliser, le copier, le modifier, le fusionner, le publier, le distribuer, le vendre et de changer sa licence. La seule obligation est de mettre le nom des auteurs avec la notice de copyright.). Cette collection permet donc d’inclure des modifications sous d’autres licences y compris non libre de droits.

L’avantage majeur de l’utilisation de bootstrap réside dans le fait qu’il est relativement simple de prototyper un design d’un site web car toutes les fonctionnalités d’affichage et d’interactions sont déjà programmé. Bootstrap est un outil majeur dans la programmation de site web, et est devenu avec le temps un framework CSS de référence.

Le prototypage d’application web est relativement facile puisque Bootstrap propose une panoplie de code HTML et de framework CSS prêts à l’emploi. Cette bibliothèque intègre aussi des composant JavaScript utilisant la bibliothèque jQuery, une autre référence dans le développement web.

Le second avantage de Bootstrap est qu’il est web-responsive, c’est-à-dire qu’il sait s’adapter à tout type d’affichage que ce soit téléphone, tablette, pc ou TV connectée, dans le but d’améliorer et de facilité la navigation sur le site web.

Bootstrap s’organise alors de manière modulaire, ou chaque module occupe une place prédéfinie, ce qui rends l’intégration de modules tel qu’un calendrier, d’une messagerie où d’alertes relativement facilement puisque l’emplacement est déjà réalisé. De plus si au cours du développement du site, il manque un module sur la page, il est plus facile d’en ajouter un que de modifier le site « en dur » au risque de modifier drastiquement l’organisation de la page.

Bootstrap étant très bien documenter et proposant des pré-solutions clé en main, où l’on peut piocher des composants qui nous intéresse tels que les tableaux, formulaires ou boutons de navigation.

Les points faibles de bootstrap sont relativement mineurs, on peut s’apercevoir que de nombreux site développer sous bootstrap se ressemble dans leurs formes car ils utilisent tous les mêmes composants.

## Architecture de programmation

Nous avons opté pour une architecture de programmation s’appuyant sur une conception MVC (modèle-vue-contrôleur). Ce motif est répandu dans le domaine de la programmation web. Il est composé comme son nom l’indique de 3 modules :

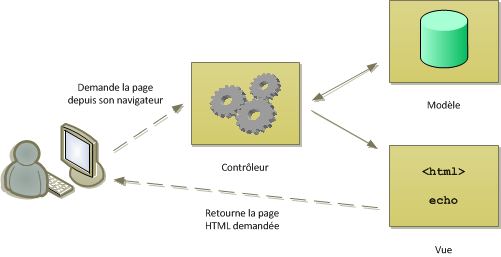
* Modèle : contient les données à afficher ; il gère la base de données. Son rôle est d’aller récupérer les données brutes dans la base de données, de les organiser et de les assemblés afin qu’elles puissent être traitées par le module *contrôleur* :
* Vue : cette partie se concentre uniquement sur l’affichage. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour l’affichage. On y trouve essentiellement du code HTML, et quelques boucles et condition PHP.
* Contrôleur : cette partie gère la logique du code et c’est cette partie qui prend les décisions. Le contrôleur va demander au modèle les données, puis il va les analysées et renvoyer les variables d’affichage à la vue. Le contrôleur contient exclusivement du PHP. C'est notamment lui qui détermine si le visiteur a le droit de voir la page ou non (gestion des droits d'accès).

Figure 5: Illustration du modèle MVC

## Base de données

### MCD

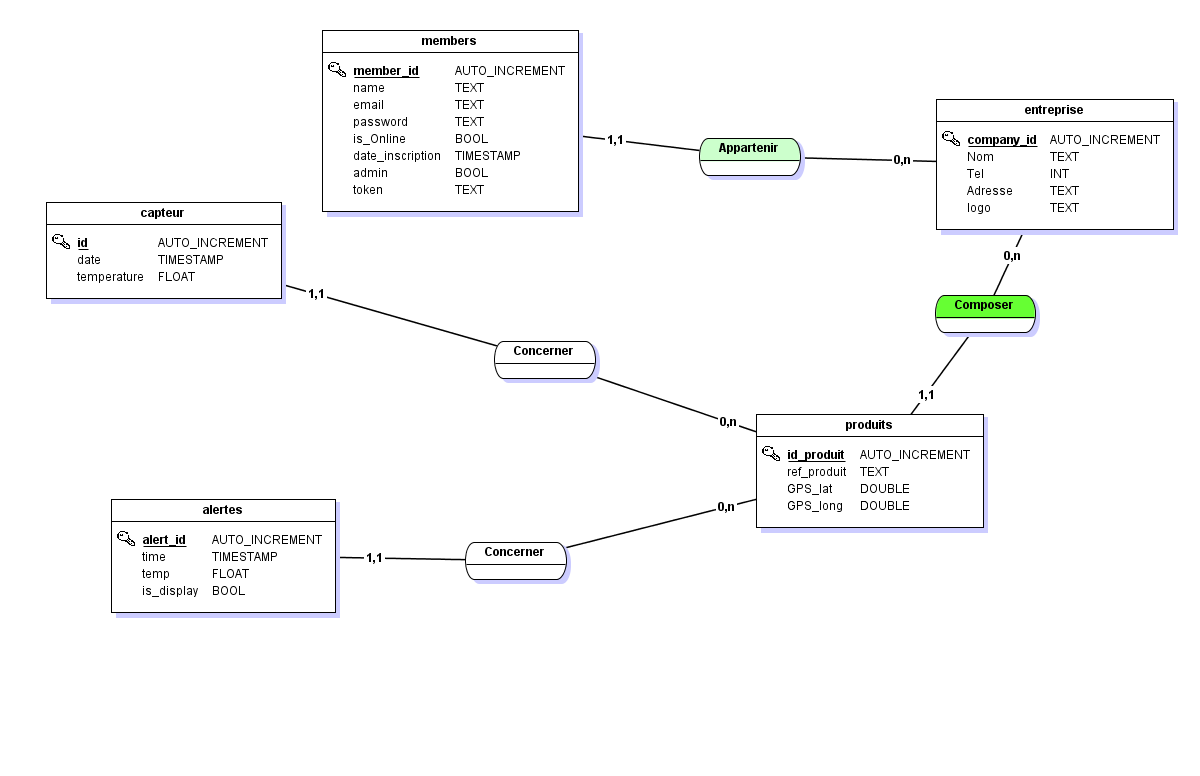
Lors de la première réunion nous avons aussi échanger autour de l’architecture de la base de données. Nous voulions avoir une base de données le plus simple possible pour notre besoin tout en répondant à toutes les fonctionnalités que nous voulions implémenter. Nous avons obtenu le Modèle Conceptuel de Données suivant :

Figure 6 : Architecture MCD de la Base de Données

### MLD

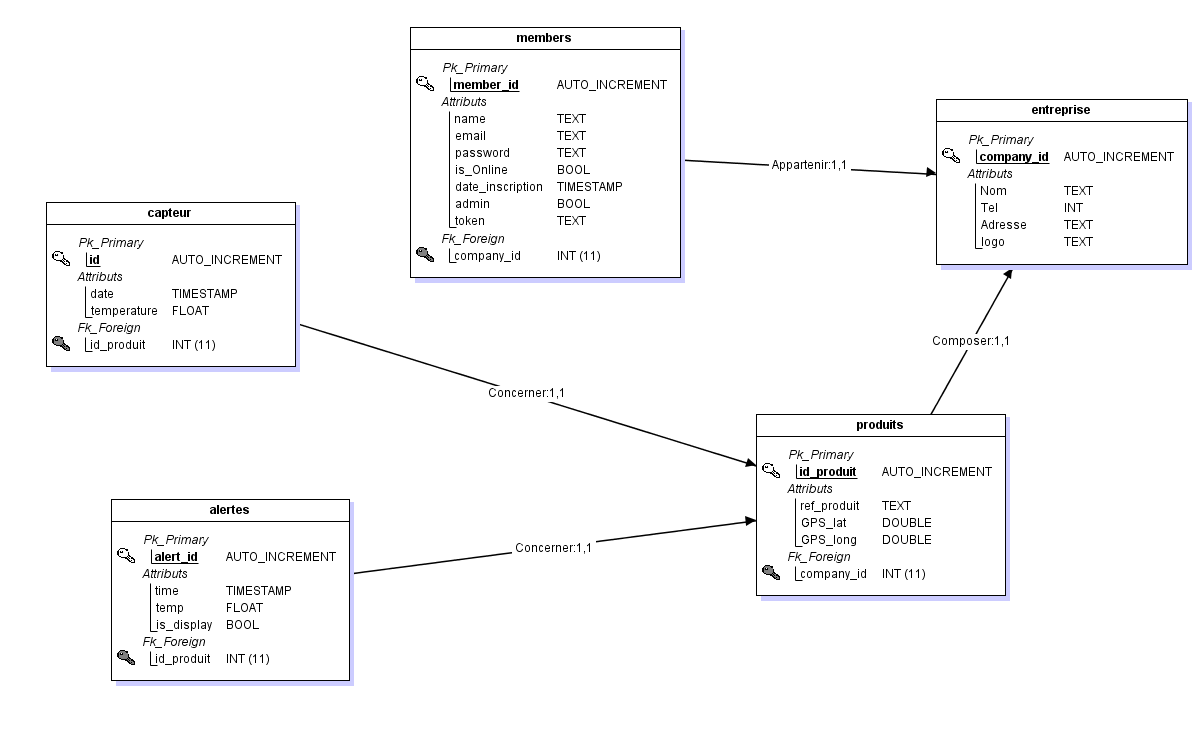
Une fois le MCD obtenu, nous pouvons réaliser le MLD en vue d’une implémentation dans une base de données fonctionnelle.

Figure 7 : Architecture MLD de la base de données

### Création de la base de données

Nous avons choisi phpmyadmin pour la création et la supervision de la base de données car cette interface est intégrée à WampServer (et sur la plupart des logiciels de création de serveur local). Phpmyadmin est facile d’utilisation, différentes fonctionnalités sont présentes comme le fait de pouvoir importer et exporter une base de données, de créer des événements et des déclencheurs automatiques.

Le choix du type de serveur est MySQL, interface de phpmyadmin le plus couramment utilisé.

Le moteur de stockage de la base de données est InnoDB qui permet la gestion des données sous MySQL, de plus ce moteur permet de créer le modèle conceptuel ce qui permet de créer rapidement les clés étrangères des tables.

### Déclencheur automatique

Nous avons une table ‘alertes’ qui contient toutes les valeurs des sondes de température dont leur valeur semble être incohérente (supérieur ou inférieur à un seuil). Afin de faciliter l’ajout des données au sein de cette table, nous avons créé un déclencheur automatique. Cela permet que dès qu’une valeur est insérée dans la table ‘capteurs’ le moteur de stockage, InnoDB, recopie cette entrée si la valeur dépasse les seuils de déclanchement. La programmation est plutôt facile, il suffit de renseigner des champs prédéterminer :

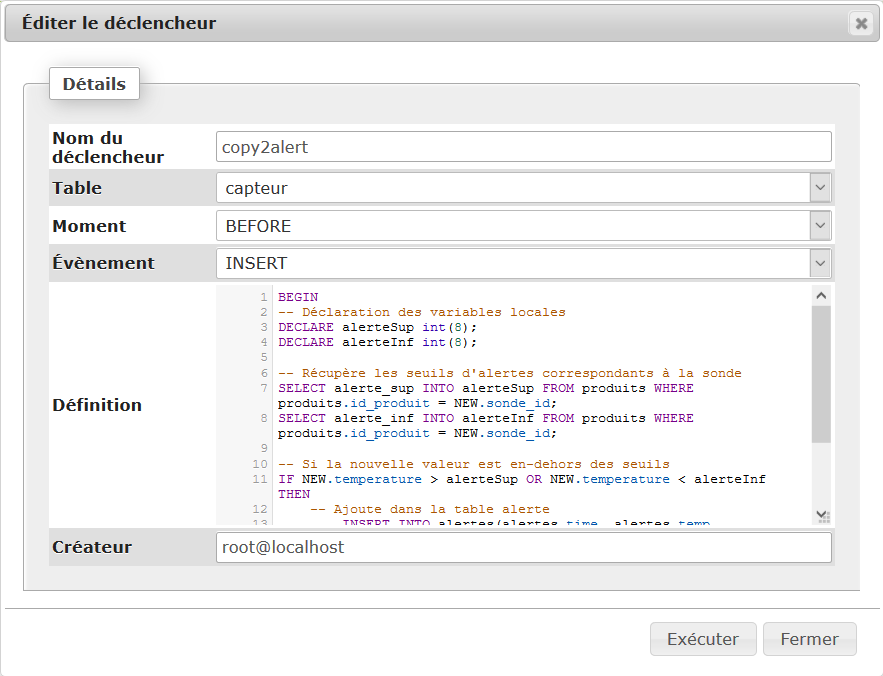


Figure 8 : Edition d'un déclencheur sous phpmyadmin

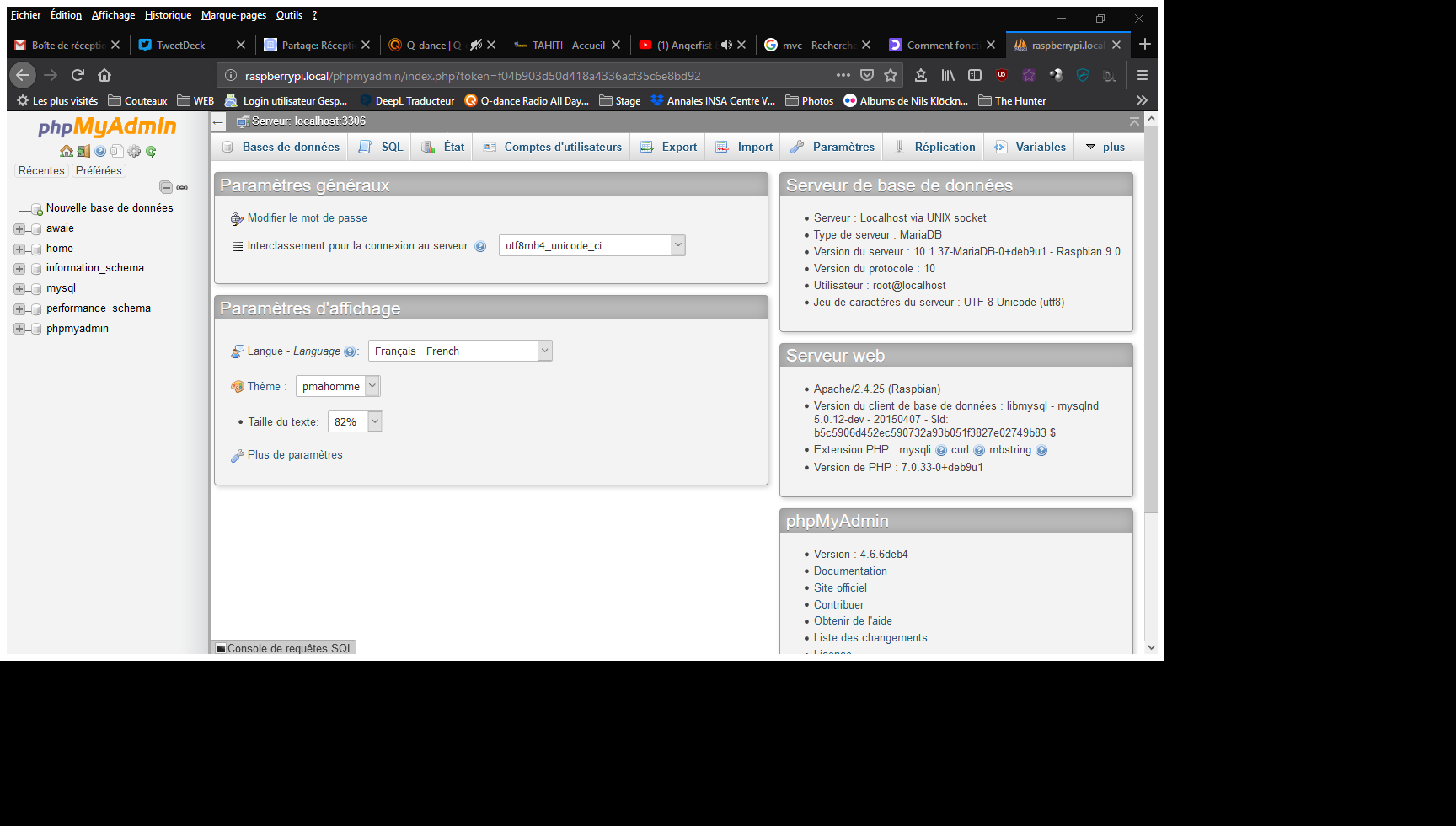
Ce qui peut poser problème se trouve au niveau de ‘Définition’, en effet c’est ici que nous programmons réellement le déclencheur. Ici, le déclencheur est activé lors de l’appel d’insertion de nouvelle valeur dans la table ‘capteur’ mais avant que ces données ne soient écrites. Le programme regarde si la valeur dépasse les seuils d’alerte de la sonde, si oui alors on insert la nouvelle valeur dans la table ‘alertes’

## Réalisation des pages dynamique

### Connexion à la base de données

Afin de se connecter à la base de données, il faut au préalable en créer une et importer celle existante.

Pour créer une nouvelle base de données depuis phpmyadmin rien de plus simple il suffit de cliquer sur ‘nouvelle base de données’ :

Figure 9 : Page d'accueil phpmyadmin

Il faut ensuite déterminer le nom de la base de données et son mode d’interclassement, dans notre cas les valeurs à saisir sont respectivement : ‘awaie’ et ‘utf8-general’.

Ensuite il faut importer la base de données existante, pour cela rendez-vous dans l’onglet ‘Importer’ et cliquez sur parcourir puis sélectionnez le fichier sql situé dans le répertoire sql du projet :

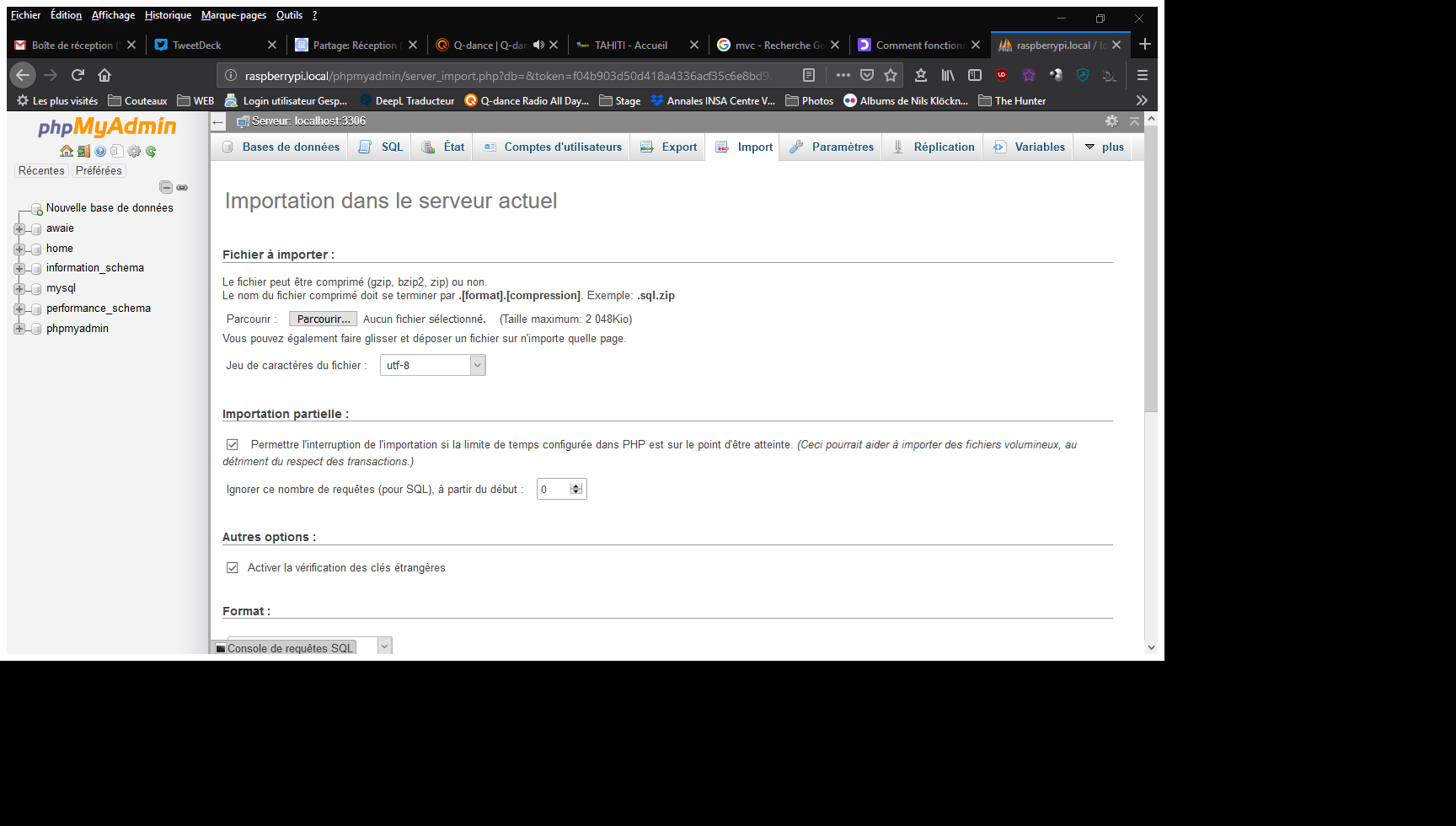


Figure 10 : Onglet d'importation de la base de données

Il ne vous reste plus qu’à cliquer sur le boutton valider. Et voilà la base de données est prête à être utiliser.

Par défaut, les logins administrateurs sont :

* Mail : [admin@admin.com](mailto:admin@admin.com)
* Mdp : admin

### Récupération des données de température (Php) + Affichage (Javascript)

Les données relatives aux sondes de température sont stockées dans une table appelée ‘capteur’. Nous y stockons l’identifiant de la sonde, la température ainsi qu’un timestamp (date & heure) à laquelle la donnée a été récupérée.

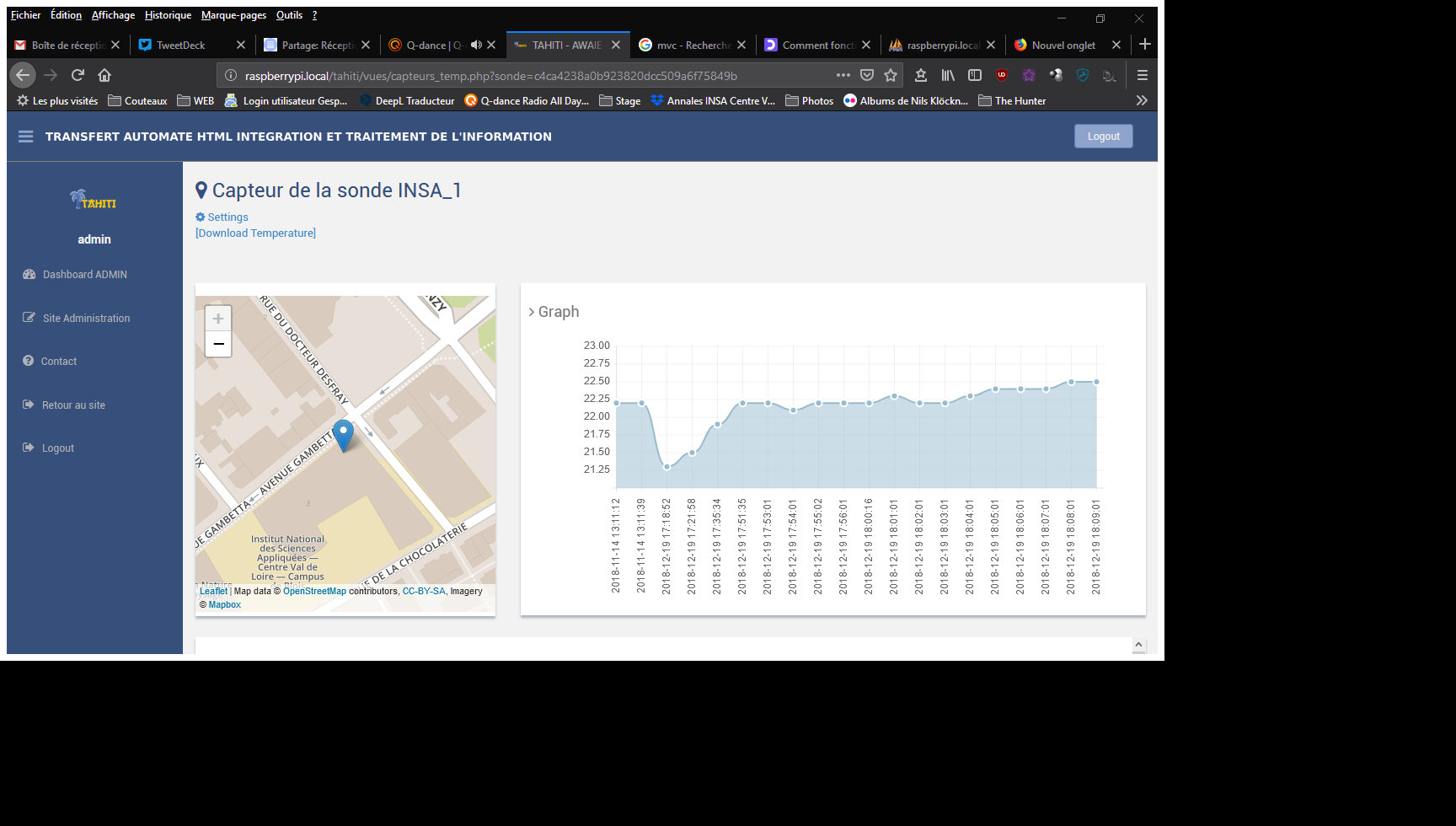
Grâce à l’utilisation de javascript et d’une libraire, nous sommes alors en mesure d’afficher un relevé de température propre à chaque sonde sous forme de graphique :

Figure 11 : Exemple de relevé d'une sonde

Ainsi l’utilisateur peut suivre l’évolution de la température de manière simple et efficase.

### Pour aller plus loin : création de jeton d’authentification

Afin de garantir une sécurité parfaite à nos clients, nous avons implémenté une technologie de token (ou jeton en d’authentification en français). Cela permet de garantir une sécurité optimale d’une part pour l’utilisateur mais aussi pour nous, car grâce à ce jeton, il n’est plus nécessaire de stocker les données relatives à l’utilisateur en session PHP ou en cookie pour le client. Il n’est stocké que le jeton d’authentification qui lui est unique puisse qu’il est créé lors du login du client. De plus ce token est copié dans la base de données afin de s’assurer que le server s’adresse au bon client. Ce token est chiffré avec la dernière technologie de chiffrage des données, BCrypt, bien plus résistant au MD5 ou RSA.

# Réalisation de la connexion avec les sondes de température

## PHPModbus

PHPModus est in plugin php permettant de communiquer avec les systèmes utilisant ce système de communication tel que nos sondes de température. Plus particulièrement dans notre cas nous utilisons le protocole Modbus TCP. Cette bibliothèque comprend plusieurs fonctions tel que l’établissement d’une communication en Modbus, des requêtes de lecture/écriture de données.

Dans notre cas nous utiliserons uniquement la lecture de registre multiple puisque nous ne voulons que lire la température des sondes.

## Configuration du réseau local

Dans un premier temps, il est nécessaire de vous connecter au réseau local. Pour cela rendez-vous dans votre gestionnaire réseau : « Centre Réseau et Partage », puis modifiez l’adresse IPv4 de votre ordinateur en paramétrant le réseau Ethernet ou Wifi. Changez l’adresse IPv4 en indiquant la suivante : 192.168.1.101 et le masque de sous-réseau : 255.255.255.0.

## Récupération des températures

Afin de récupérer les données, il est nécessaire dans un premier temps d’établir la communication avec le système gérant les sondes.

Pour cela nous utilisons :

$modbus = new ModbusMaster("192.168.1.99", "TCP");

Cette ligne permet d’effectuer une connexion en TCP sur l’adresse IP 192.168.1.99 qui correspond à l’adresse IP du module de communication Modbus des sondes de température.

Une fois la connexion établie, il faut envoyer une requête permettant de récupérer les données stockées dans les registres du module. Pour cela nous utilisons la fonction :

$recData = $modbus->readMultipleRegisters(0, 0, 32);

Le prototype de la fonction étant : readMultipleRegisters(int slaveAddr,int startRef,Array regArr). Le slaveAddr représente l’adresse de l’esclave, ici nous n’avons qu’un périphérique connecté sur cette adresse IP d’où le 0. Le startRef correpond à l’adresse de départ d’où nous voulons lire les données stockées sur la module. Le regArr correspond au nombre de valeurs que nous voulons lire sur le registre du module.

Nous pouvons alors voir les informations reçues en affichant les données :

// Print status information

//echo "</br>Status:</br>" . $modbus;

// Print read data

//echo "</br>Data:</br>";

//print\_r($recData);

//echo "</br><br><br>";

Status :

readMultipleRegisters : START

Connected

Packet: 7747000000060000300000040

Send

Data received

Packet : 77470000008300038000030003…

Modbus response error code : NOERROR

Disconnected

readMultipleRegisters: DONE

Data:

Array( [0] => 0 …. [33] => 241 [34] => 0 [35] => 230 …)

Les données de températures sont stockées dans les registres suivants :

Sonde de référence : dans les registres 32 et 33

Sonde n°1 : registres 34 et 35

Sonde n°2 : registres 36 et 37

Sonde n°3 : registres 38 et 39

Etant donné que les données sont stockées sous 8bits, le maximum est 255. De plus les températures sont codées de manière décimal, c’est-à-dire que la valeur, par exemple 24.3°C est écrite comme 243. On arrive donc à un overflow au-delà de 25,5°C. Il faut donc un 2 octet pour stocker la température. Si la température est au-delà de 25.5°C les données sont stockés sous la forme 16bits (2 octets successifs). Pour avoir la température décodée il faut donc multiplier le 1er octet par 256 et y ajouter le 2ème octet. D’où la formule suivante :

1er registre \* 256 + 2ème registre

# Ajout d’un serveur local type Raspberry Pi

## Nouvelle architecture

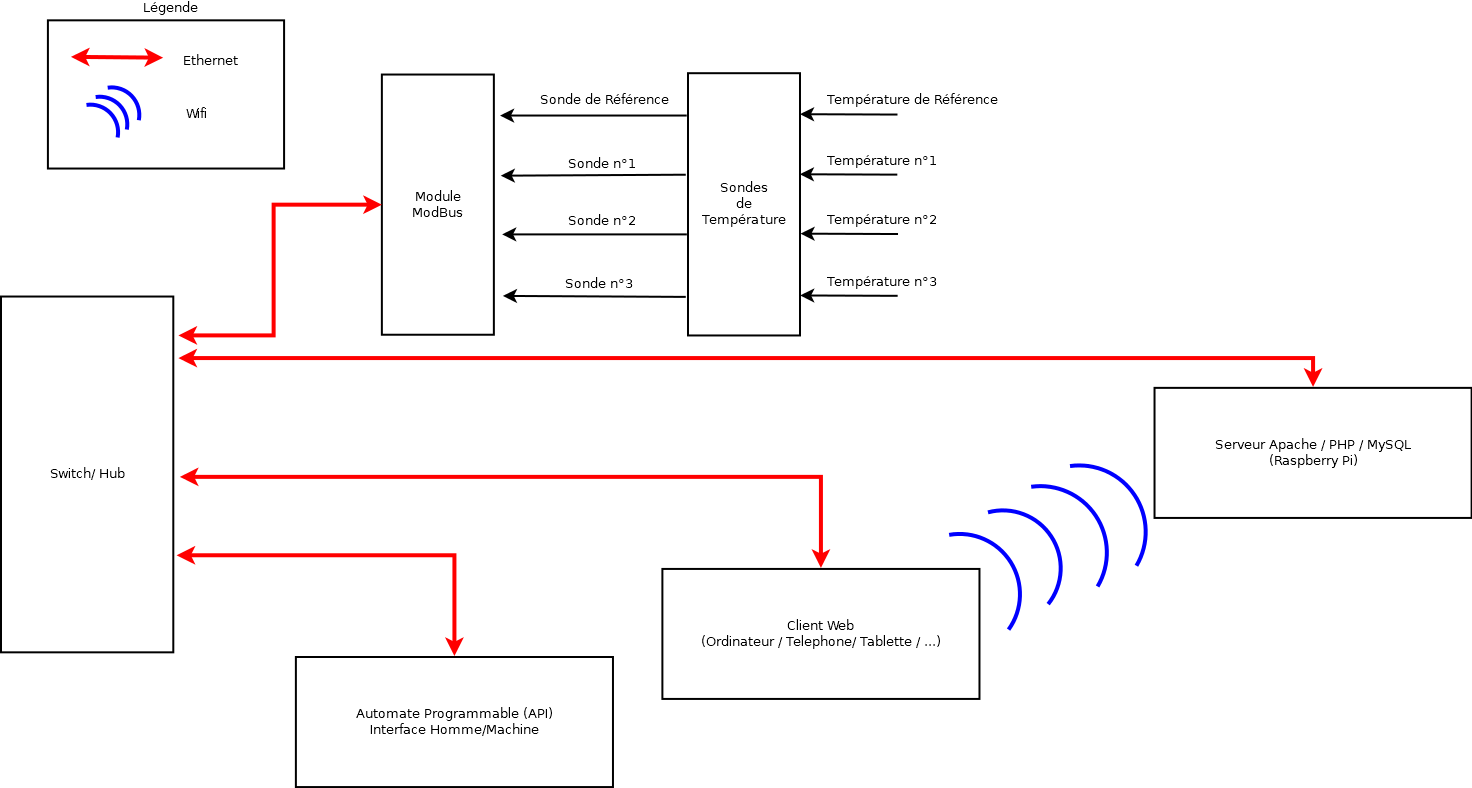
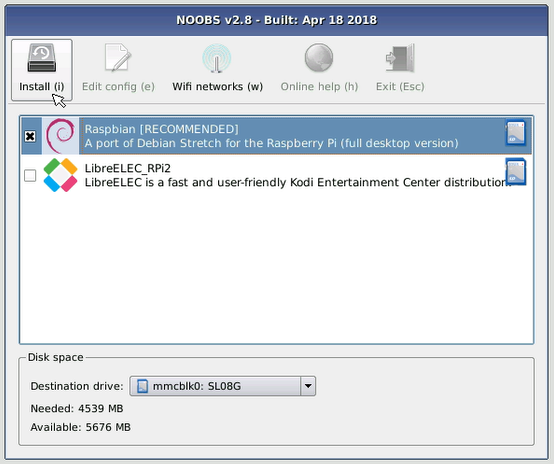
Avec la déportation de la base de données ainsi que du serveur local php sur un véritable serveur, il est nécessaire de modifier l’architecture afin d’y intégrer ce serveur. D’où la nouvelle architecture :

Figure 12 : Nouvelle architecture matérielle

## Installation du serveur sous Raspbian

Dans un premier temps il faut installer l’OS sur le Raspberry Pi, pour cela il faut télécharger l’OS via l’adresse url : <https://www.raspberrypi.org/downloads/noobs/>. Il suffit de suivre les instructions pour l’installer sur la carte SD. Ensuite il faut insérer la carte cd dans le Raspberry et le mettre sous tension, le Raspberry s’allume alors automatiquement. Veillez à avoir brancher les périphériques nécessaires, comme un écran, une souris et un clavier.

Il suffit alors de cliquer sur Raspbian, comme l’image ci-contre :

Figure 13 : Installation de Raspbian

L’installation de l’OS s’effectue alors, une fois arrivé sur le bureau. Il reste à mettre Raspbian ainsi que tous ces packages à jour. Pour cela ouvrez un terminal et exécuter les commandes suivantes :

sudo apt-get update

sudo apt-get upgrade

Maintenant que notre Raspberry est à jour il reste à installer le serveur Apache ainsi que les plugin PHP / MySQL et les configurer pour cela nous avons suivi les tutoriels suivants : <https://raspbian-france.fr/installer-serveur-web-raspberry-lamp/>.

Nous avons aussi besoin de configurer les certificats afin de pouvoir utiliser le module ssl (Secure Socket Layer) pour que la communication avec le serveur Gmail pour l’envoi de mail (formulaire de contact). Nous avons suivi le tutoriel :

<https://variax.wordpress.com/2017/03/18/adding-https-to-the-raspberry-pi-apache-web-server/comment-page-1/>

## Connexion ssh / VNC viewer / sftp

### Configuration ssh :

SSH est à la fois un programme informatique mais aussi un protocole de communication sécurisé. Ce protocole permet de communiquer par le réseau local à un appareil distant et de le piloté. Principalement il sert à exécuter des commandes sur un autre appareil de celui d’où on écrit cette commande.

Ce protocole est très pratique sur un Raspberry Pi car nous n’avons pas besoin d’écran pour exécuter des commandes. Une simple connexion sur le même réseau local entre le Raspberry et l’ordinateur suffit.

Cette fonctionnalité est désactivée par défaut sur le Raspberry Pi pour des raisons de sécurité. Pour l’activer rien de plus simple il suffit de se rendre dans le panneau de configuration du Raspberry par le biais de cette commande dans le terminal :

sudo raspi-config

Une liste des configurations apparaît alors. Il suffit de se rendre dans « Interfacing Options » et d’activé le ssh.

Afin de pouvoir savoir sur quelle adresse IPv4, le Raspberry est configuré, nous exécutons la commande suivante :

sudo ifconfig

Il faut regarder l’adresse IPv4 comme indiqué ci-dessous.

pi@raspberrypi:~ $ ifconfig

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500

inet 169.254.29.155 netmask 255.255.0.0 broadcast 169.254.255.255

inet6 fe80::dda2:fcaa:edbe:7e28 prefixlen 64 scopeid 0x20<link>

ether b8:27:eb:79:cf:da txqueuelen 1000 (Ethernet)

RX packets 1664 bytes 123924 (121.0 KiB)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 1848 bytes 535443 (522.8 KiB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

Il faut alors redémarrer le Raspberry Pi pour activer la nouvelle configuration.

On lance alors le logiciel PuTTY afin de se connecter par ssh sur le Raspberry, et on inscrit l’adresse IPv4 précédente dans le Host Name :

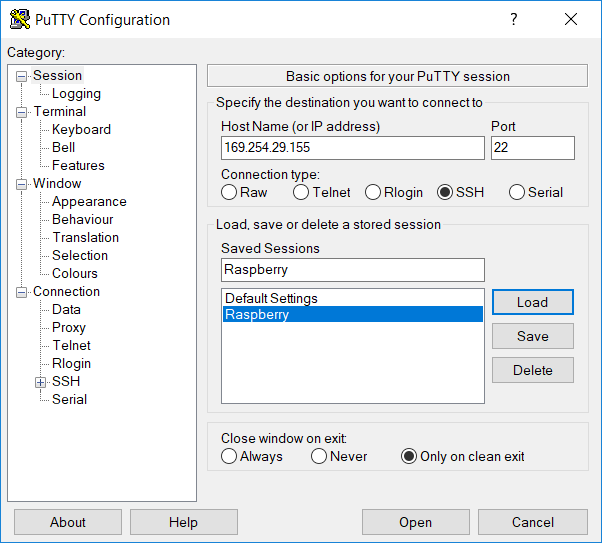
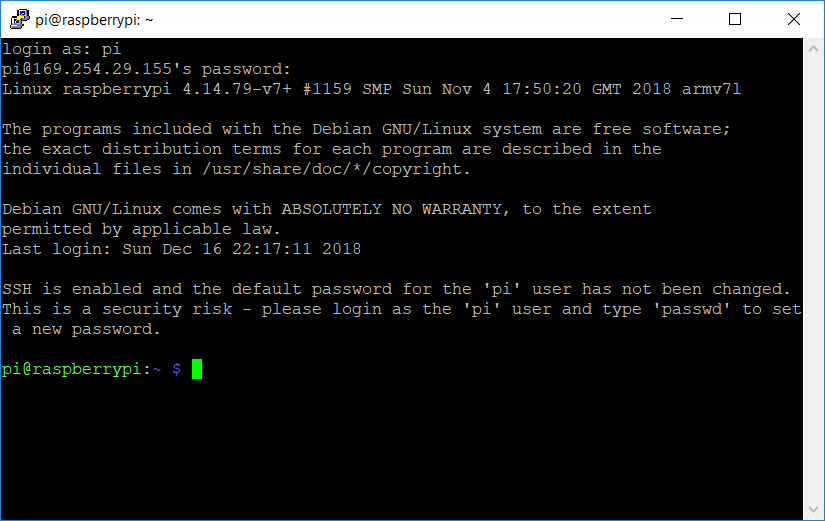


Figure 14 : Interface PuTTY

Il vous est maintenant demander de saisir votre login. Si le Raspberry est dans sa configuration par défaut, le login est ‘pi’ et le mot de passe est ‘raspberry’.

Une fenêtre comme celle-ci devrait apparaitre :

Figure 15 : Interface du raspberry sous PuTTY

Il ne vous reste plus qu’à écrire les commandes que vous voulez dans ce terminal comme ci c’était sur le Raspberry.

### Connexion via VNC Viewer :

Le protocole SSH est très utile mais pas très conviviale, surtout si l’on est habitué à Windows. Pour cela on peut utiliser VNC Viewer, installé par défaut avec Raspbian, qui copie l’écran du Raspberry directement sur votre ordinateur.

Pour l’activer il faut aller modifier le fichier de configuration comme pour l’activation du protocole SSH :

sudo raspi-config

Une liste des configurations apparaît alors. Il suffit de se rendre dans « Interfacing Options » et d’activé ‘VNC’ ou ‘VNC Viewer’.

Il est nécessaire de redémarrer le Raspberry. Puis via le logiciel VNC Viewer sur l’ordinateur on renseigne l’adresse IPv4 dans le champ encadré :

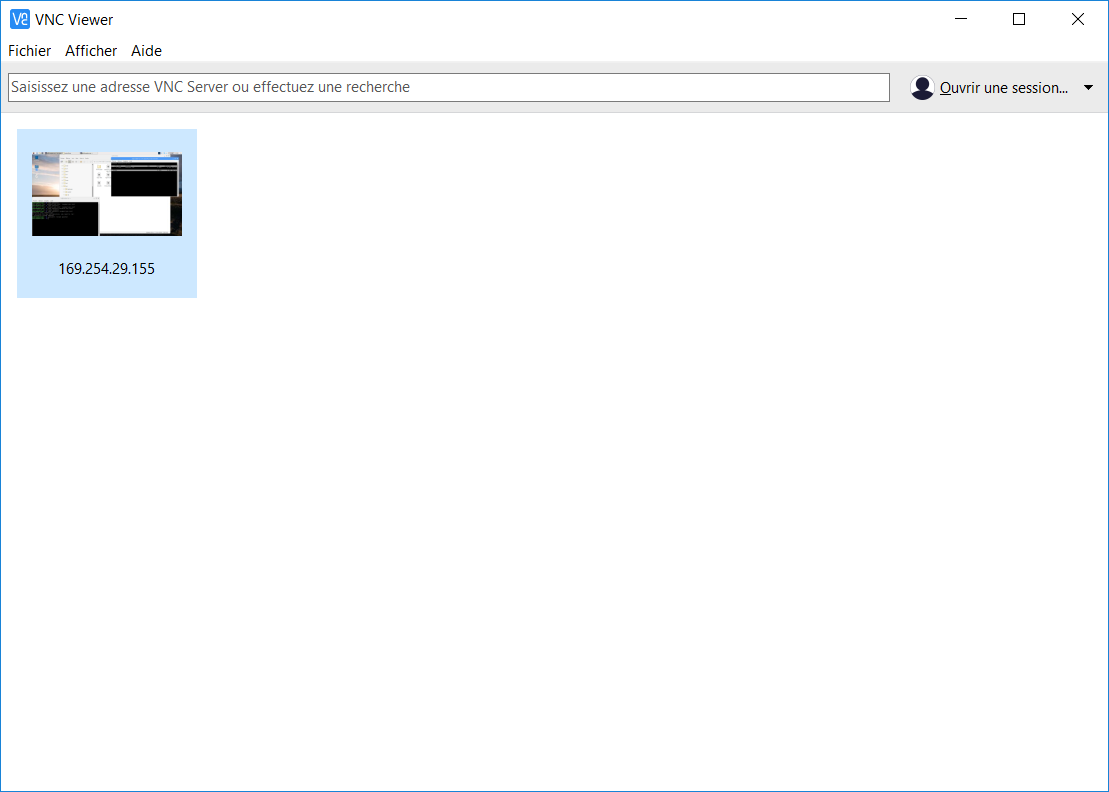


Figure 16 : Interface de VNC Viewer

Si tout cela se passe bien vous devriez avoir une fenêtre avec l’écran du Raspberry.

### Connexion sftp :

Pour se connecter au Raspberry Pi via sftp afin de transférer des fichiers par Ethernet, rien de plus simple il suffit d’ouvrir un logiciel permettant le protocole sftp comme FileZilla. Pour cela il suffit de remplir les champs suivant comme le montre l’image ci-dessous.

Le serveur est : sftp://raspberrypi.local

L’identifiant est par défaut : pi ; et le mot de passe : raspberry.

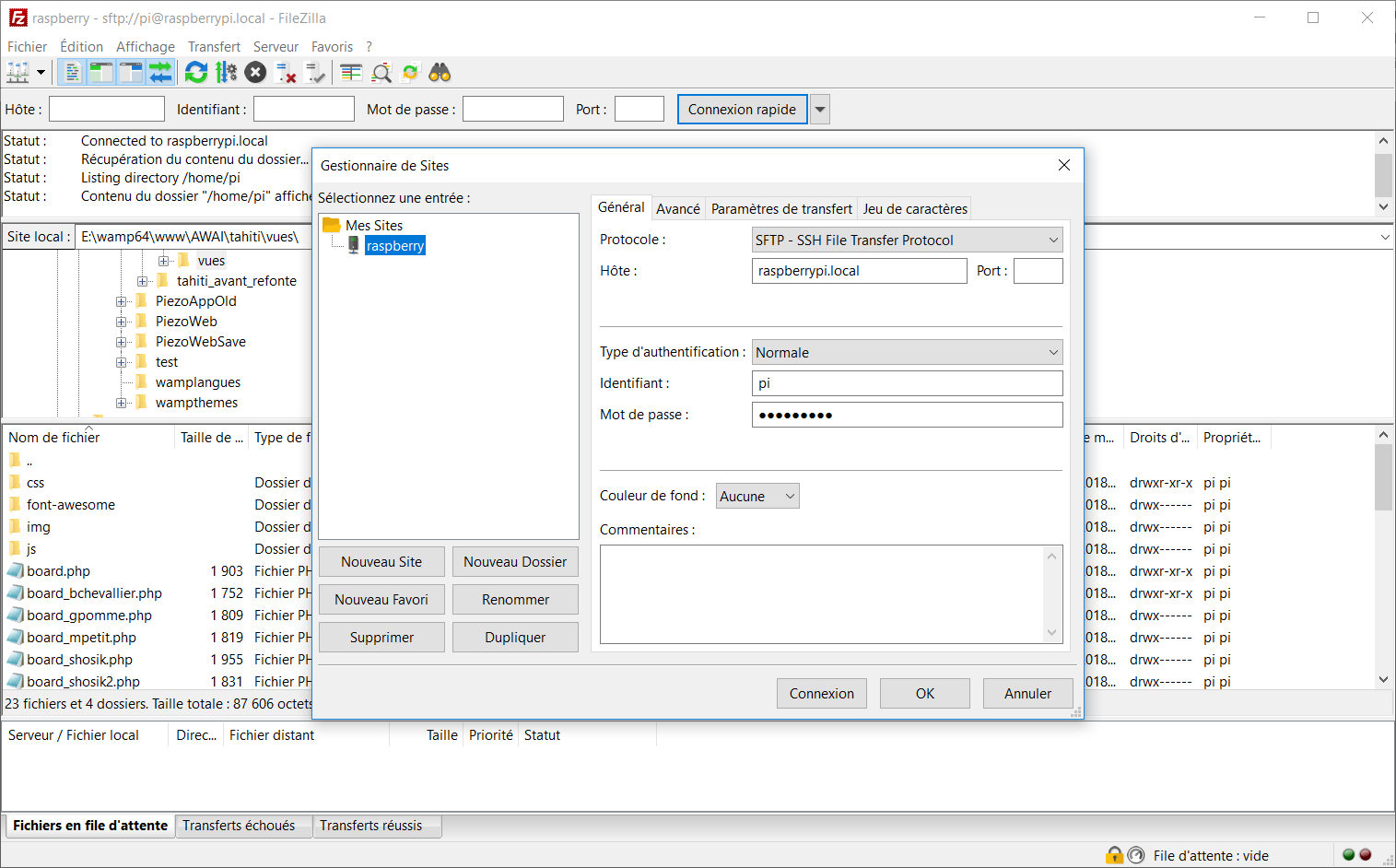


Figure 17 : Interface de FileZilla

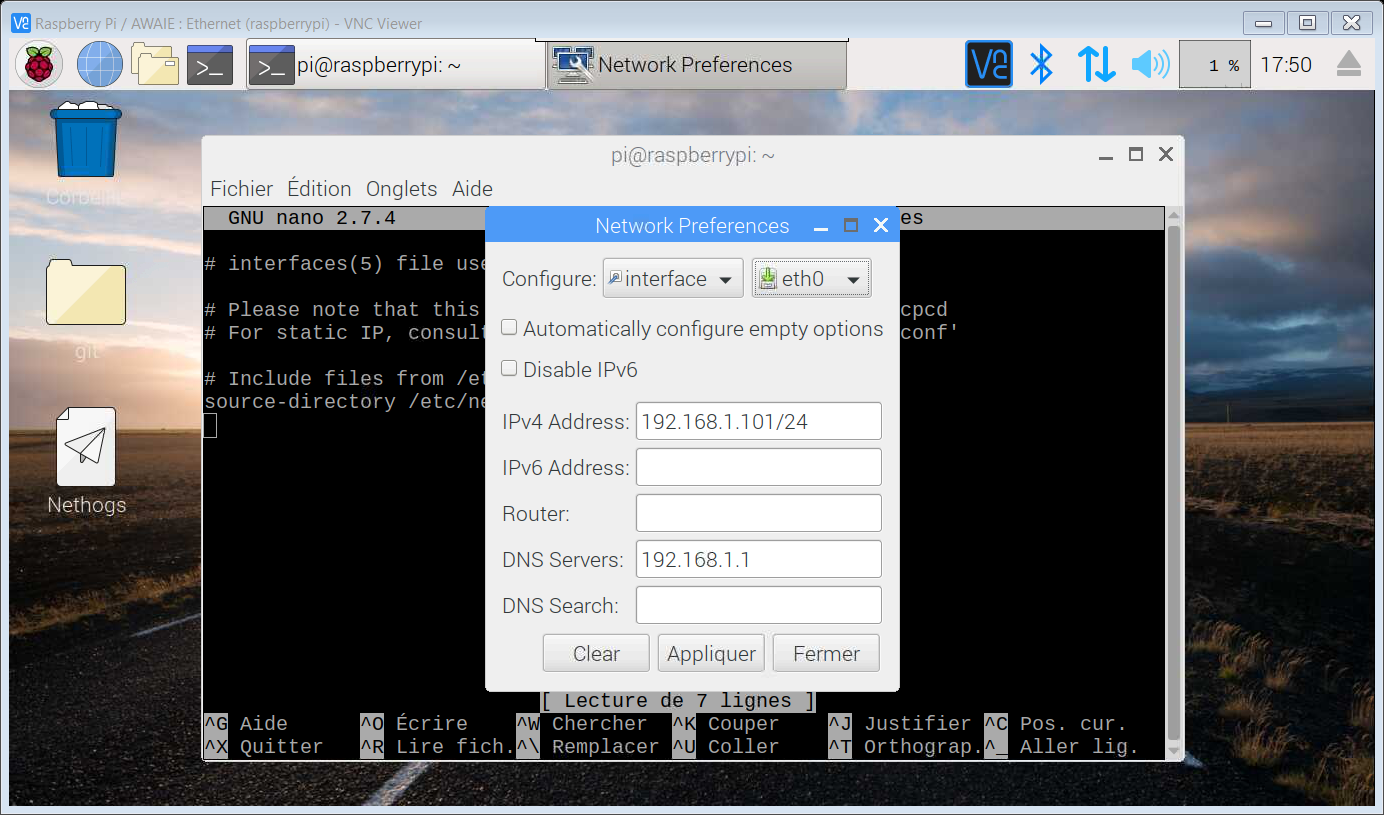
## Connexion aux sondes de température

Avec la nouvelle architecture, le Raspberry Pi est connecté sur le hub qui permet de connecter en simultané les sondes de température, l’automate programmable, notre serveur (Raspberry Pi) ainsi qu’un éventuel ordinateur. Il faut que le l’ensemble de ses éléments puisse communiquer entre eux. Or l’architecture déjà existante (Sondes + Automate) communique déjà en via Ethernet et le réseau local en 192.168.1.x.

L’adresse IPv4 : 192.168.1.99 étant déjà réservé la carte de communication Modbus des sondes de température, et l’adresse 192.168.1.100 par l’automate. Nous avons décidé d’attribuer l’adresse IPv4 192.168.1.101 au Raspberry Pi.

Or l’adresse IPv4 de base de notre Raspberry est 169.254.29.155. Avec une telle adresse IPv4, la communication entre le Raspberry et le système actuel est impossible. Pour pallier à ce problème nous avons dû changer l’adresse IPv4 fixe du Raspberry.

Pour cela il suffit de faire un clic droit sur le symbole à gauche de celui du volume, et de cliquer sur ‘Wireless & Wired Network Settings’, puis de sélectionner interface et eth0, et finalement de renseigner les champs comme le montre l’illustration ci-dessous.

Figure 18 : Configuration Ethernet du raspberry

Il ne reste plus qu’à reboot le Raspberry. Attention si les modifications ont été effectué via SSH ou via le logiciel VNC Viewer depuis votre PC, il faudra configurer le réseau local Ethernet de l’ordinateur en modifiant l’adresse IP sur 192.168.1.102 par exemple (cf Chapitre : Configuration du réseau local). La reconnexion via SSH / VNC Viewer se fera alors avec la nouvelle adresse IPv4 : 192.168.1.101.

## Récupération des données de température en temps réel

L’avantage majeure d’avoir un Raspberry Pi à disposition, outre le fait de déporter le serveur et la base de données sur un autre appareil et de permettre la connexion simultanée de plusieurs clients, est le fait de pouvoir exécuter des scripts à intervalle régulier.

Pour cela il est fourni dans les dernières versions de Raspbian, un logiciel s’occupant de cette tâche : crontab. Ce logiciel est relativement facile à configurer il suffit d’exécuter la commande ci-dessous afin de modifier le fichier de configuration du logiciel. C’est dans ce fichier que l’on va dire au logiciel quels script il doit exécuter tous les x temps.

sudo crontab -e

Toutes les explications de la configuration d’un nouveau script à exécuter est expliqué à l’interieur.

Dans notre cas, nous voulons exécuter un script php toutes les minutes afin de récupérer les données de températures des sondes d’où l’ajout de la ligne suivante :

\*/1 \* \* \* \* /usr/bin/php /var/www/Tahiti/models/get\_temp.php

La configuration s’effectue de telle sorte :

\*/1 \* \* \* \* /usr/bin/php /var/www/Tahiti/models/get\_temp.php

Minutes Jour Mois Chemin du fichier à exécuter

Heures Semaine Application à lancer

# Application Web

## Application Android

La création d’une application Android est un plus pour la commercialisation de notre produit, en effet le futur client pourra accéder au suivi des températures partout dans le monde, moyennant une connexion internet.

Nous avons donc décider de lancer dans un premier temps une application mobile Android permettant d’afficher le contenu de la page web, comme si le client était sur son ordinateur. Grâce à l’utilisation de Bootstrap et de web-responsivité, l’intégration a été d’autant plus simple car l’affichage mobile et tablette a déjà été prise en compte.

Pour la conception à proprement parler de l’application, nous l’avons développée sous Android Studio. Une fois le programme compilé, l’interface de programmation génère un fichier d’installation compatible avec la plupart des téléphones récent tournant sous Android (version 5.0 et plus récente). Il suffit alors de le télécharger depuis son smartphone et de l’installer.

Pour l’utilisation, elle reste identique à la navigation sur ordinateur traditionnel.

## Application IOS

Afin de couvrir la plus grande part du marché mobile, noua avons aussi développé une application pour iOS. Cette application a été créer à l’aide de l’outil de développement dédié à iOS intégré à macOS, XCode. Cela permet une plus grande simplicité de programmation mais aussi une parfaite compatibilité entre iOS et macOS.

Identiquement à l’application sous Android, son utilisation reste la même que sur navigateur traditionnel.

# Perspectives d’amélioration / Conclusion

Nous aurions aimé avoir plus de temps à consacré à ce projet afin d’y intégrer d’autre capteurs : pression atmosphérique, lumière ambiante, détecteur de mouvement ou une caméra IP.

Ce projet nous a permis pour certains de découvrir la programmation web et de nouveau langage de programmation telle que PHP ou JS, pour d’autres de consolidé leurs acquis de développer de nouvelles compétences : utilisation différents Framework, librairies, méthode de programmation, utilisation de Git, …

# ANNEXEs

Une image contenant capture d’écran

Description générée avec un niveau de confiance très élevé

## Annexe 1 : Gantt CHART



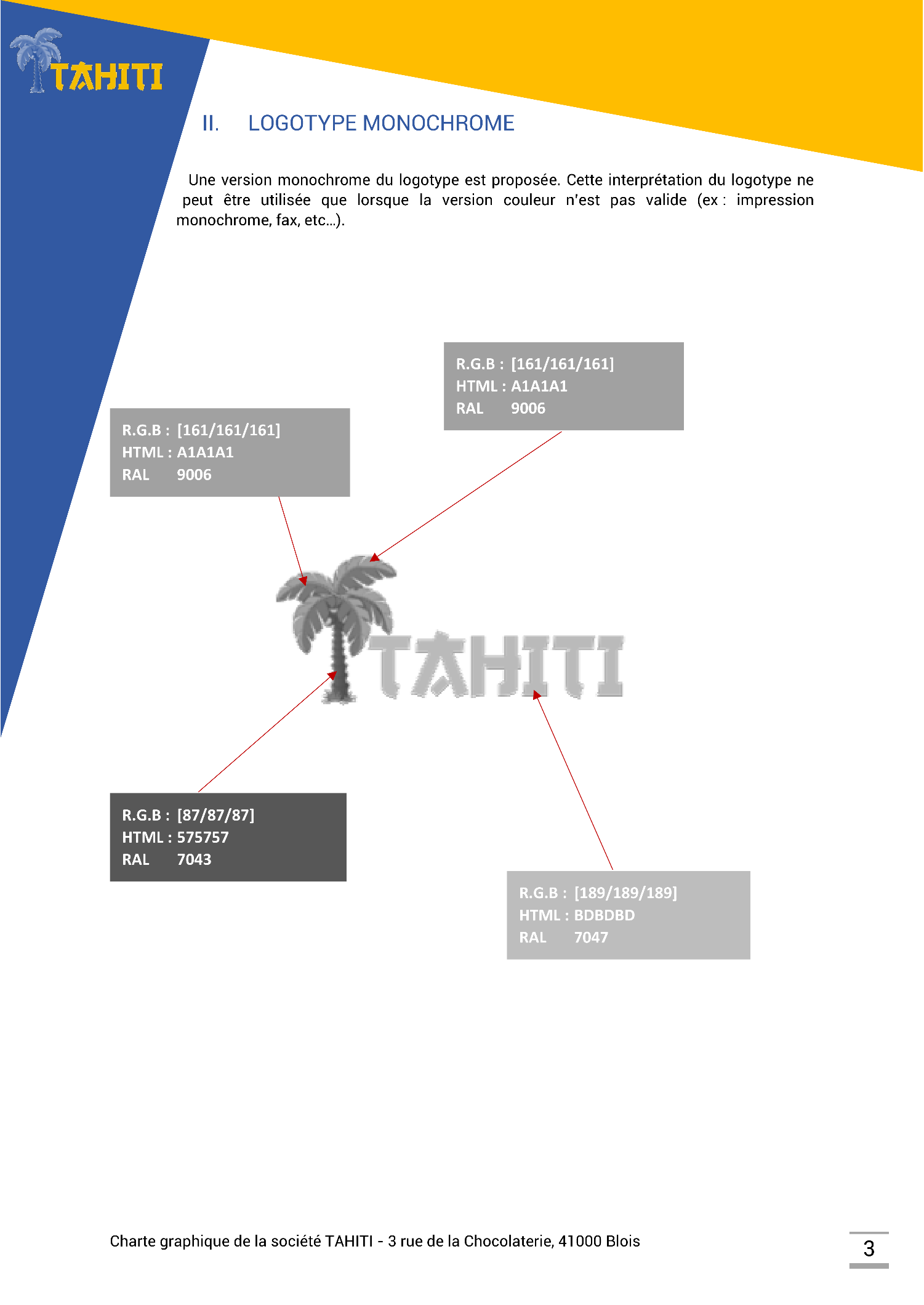
## Annexe 2 : Charte GRAPHIQUE

Une image contenant texte

Description générée avec un niveau de confiance très élevé

Une image contenant texte, carte

Description générée avec un niveau de confiance très élevé



Une image contenant capture d’écran, texte

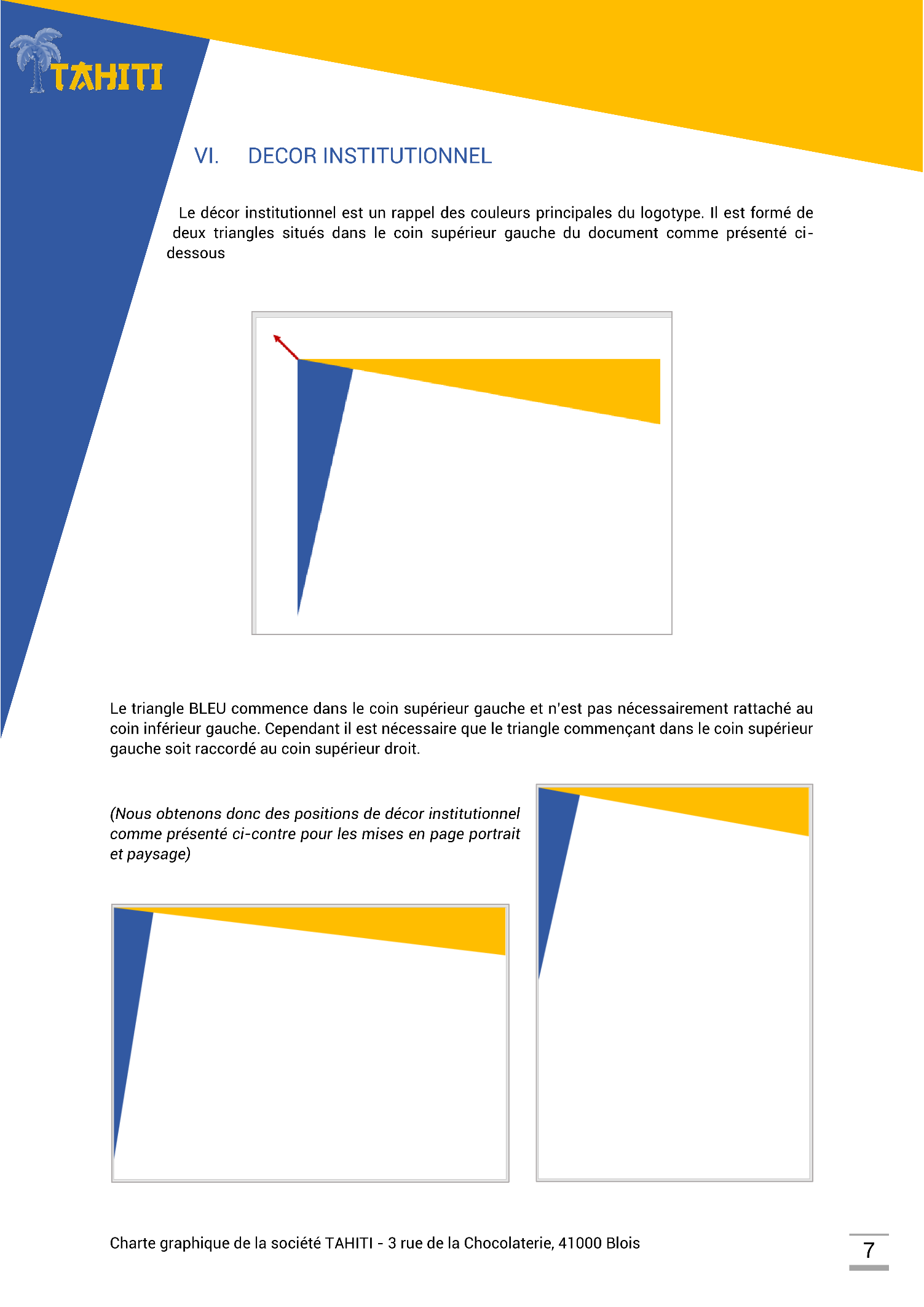
Description générée avec un niveau de confiance élevé

Une image contenant texte

Description générée avec un niveau de confiance élevé

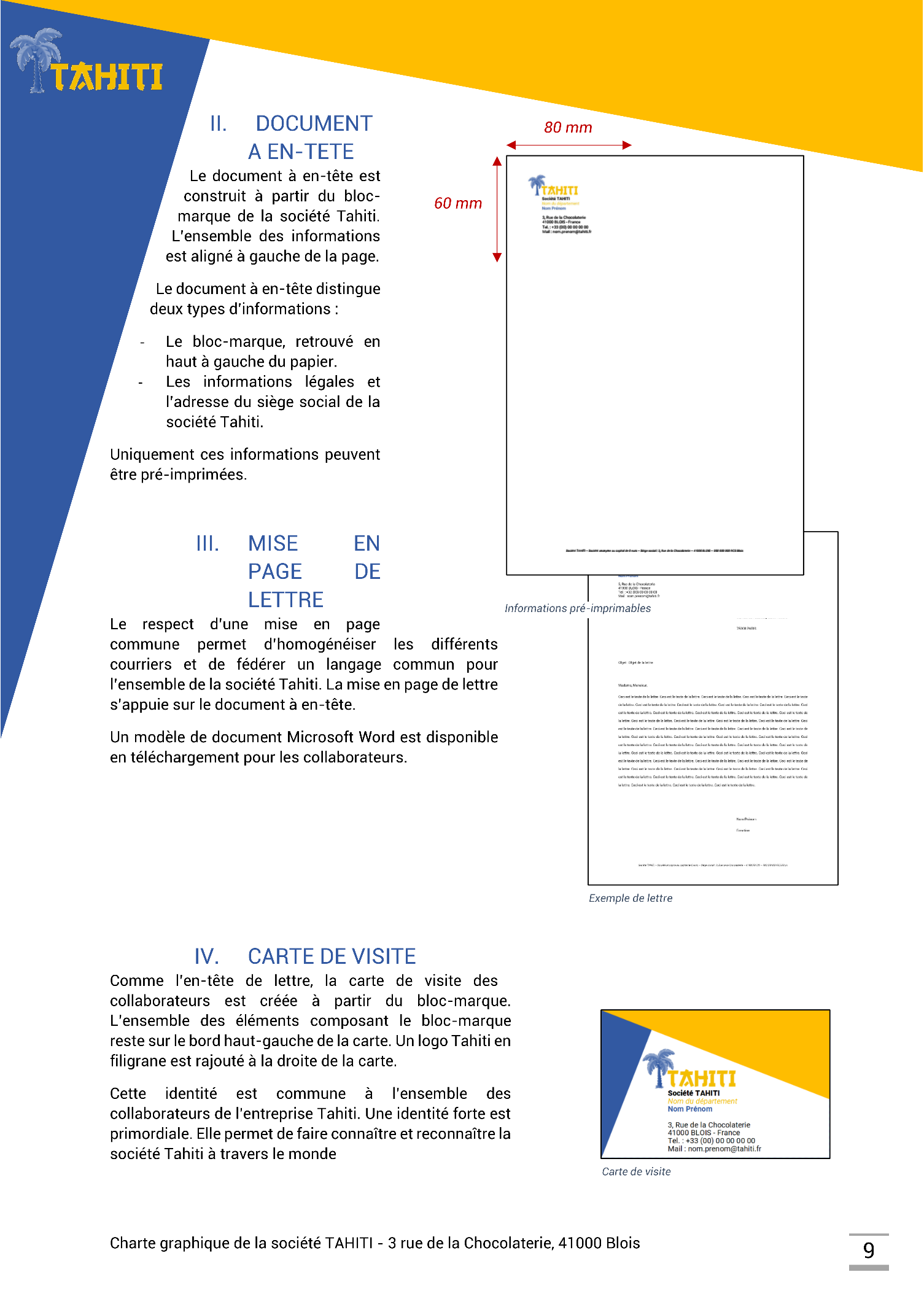
Une image contenant texte

Description générée avec un niveau de confiance très élevé



Une image contenant texte, capture d’écran

Description générée avec un niveau de confiance très élevé



Une image contenant texte

Description générée avec un niveau de confiance très élevé

Une image contenant texte, capture d’écran

Description générée avec un niveau de confiance très élevé

