PABLO MATAS
INGÉNIEUR SYSTÈMES ET RESEAUX

19 chemin de la plaine - 69390 Vourles
+33 06 32 26 07 70
pablo.matas@cityn.com

RAPPORT DE STAGE

ADOUARD Camille Bachelor 2

4 Janvier au 26 Février 2016

Maître de stage : M. Pablo MATAS

Entreprise : Cityn

Table des matières

Introduction

J'ai effectué mon stage de deuxième année de Bachelor dans l'entreprise d'infogérance Cityn. L'entreprise Cityn est une SARL unipersonnelle ou EURL située 19, chemin de la plaine à Vourles dans le Rhône et fondée en 2012 par Pablo Matas.

EURL : société commerciale où la responsabilité de l'associé/gérant est limitée, soumis au même régime qu'une SARL. Cette disposition est appréciée des entrepreneurs individuels car elle permet de séparer leur patrimoine personnel et professionnel simplement.

Cette entreprise propose un service d'infogérance à de nombreux clients dans les alentours de Lyon et plus particulièrement aux alentours de Vourles. L'infogérance est un service dont le but est la gestion d'une partie ou de la totalité du système informatique d'une entreprise. Ce service peut aller de l'installation d'un réseau informatique jusqu'à l'hébergement de données en passant par la location de serveur.

Mon stage s'est déroulé du 4 Janvier au 26 Février. Durant ce stage de huit semaines j'ai pu réaliser diverses missions dont la mise en place de divers montages sur raspberry pi ou encore la mise en place d'un réseau domotique. En effet afin d'affiner mes connaissances en réseau et en environnement linux mon tuteur m'a confié au début du stage des taches en lien plus ou moins étroit avec l'infrastructure et de difficulté croissante.

Dans une première partie je vais vous présenter mes différents travaux sur les raspberry pi ainsi que mon travail au quotidien avec mon tuteur. En seconde partie je vous présenterais plus en détail le projet majeur qui m'a été confié, un réseau domotique.

I) Le travail de recherche sur les raspberry pi

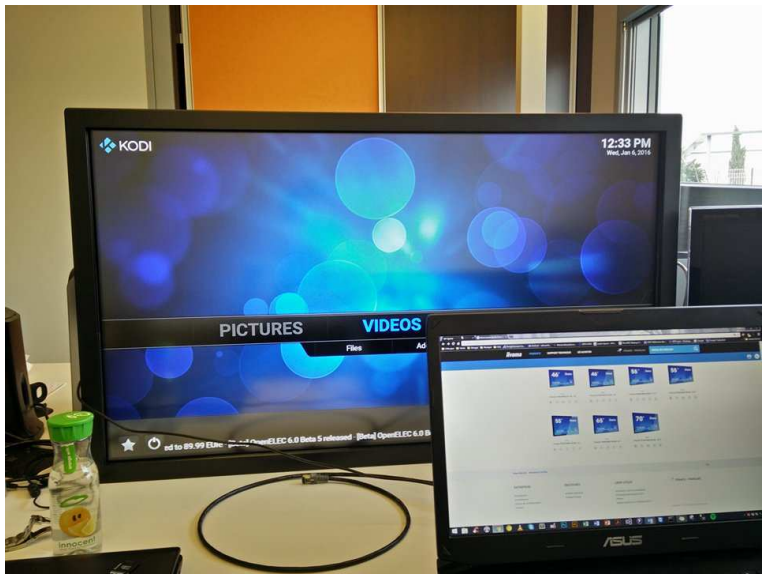
A) Les différentes configurations

Les raspberry pi sont des ordinateurs miniatures tenant dans la paume de la main. Ces mini-ordinateurs aux grandes possibilités fonctionnent de manière très efficace sous système Unix. En effet il existe même des distributions spécifiques destinées aux raspberry pi. Par exemple la distribution Raspbian est une adaptation de Debian spécifique pour raspberry.

J'ai travaillé sur un raspberry pi 2 B le modèle le plus récent et le plus puissant. Au niveau des systèmes Unix j'ai principalement travaillé sous la distribution Raspbian : Jessie et Wheezy qui en sont les deux dernières versions. La mise en place du système Unix se déroule sur une carte micro SD. Pour cela j'ai utilisé deux outils, Win32DiskImager permettant la mise en place d'un système d'exploitation et SDFormatter permettant comme son nom l'indique le formatage du périphérique sélectionné.

Mise en place de Kodi :

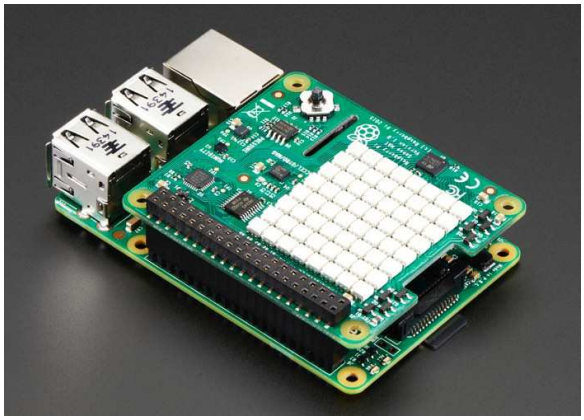
Le premier projet qui m'a été confié fut de mettre en place un media center, Kodi, sur le raspberry, ajouter Netflix, Twitch et enfin afficher cela sur un écran tactile géant iiyama. J'ai réalisé cette opération sur Raspbian Jessie. La plus grande difficulté lors de ce premier projet fut de trouver le driver permettant l'activation du tactile sur le raspberry. L'ajout de fonctionnalités sur Kodi se réalise via un dispositif intégré d'ajout de paquet via un url.



En premier lieu j'ai installé une image du media-center Kodi avec un démarrage automatique du logiciel. Puis dans le logiciel lui-même j'ai pu régler le clavier et la langue. Ensuite grâce au système d'ajout de plugin de Kodi par Url,

l'ajout de Netflix et Twitch a été possible. Enfin j'ai recherché sur le site du constructeur iiyama le driver permettant l'ajout de la commande tactile pour l'écran visible ci-dessus.

Création d'une mini station météo :



Le second projet fut de réaliser une mini station météo sur le raspberry à l'aide d'un Sense HAT. J'ai réalisé cette opération sur Raspbian Jessie. Le Sense HAT est une carte applicable sur les ports GPIO du raspberry possédant de nombreux capteurs dont un capteur de température, d'humidité, de pression, d'accélération ainsi qu'un écran soixante-quatre LED.

Afin de récupérer les données fournies par ces capteurs il m'a fallu consulter la documentation en ligne du Sense HAT et réaliser un script en python. Ce script m'a permis par la suite d'afficher sur l'écran du Sense HAT les diverses information récupérées par celui-ci.

```
#!/usr/bin/python3
from sense_hat import SenseHat

sense = SenseHat()
sense.clear()
import time
from time import sleep

X=[255, 0, 0]
O=[0, 0, 0]

Pixel_invader = [
O, O, O, X, X, O, O, O,
O, O, X, X, X, X, O, O,
O, X, X, X, X, X, X, O,
X, X, O, X, X, O, X, X,
X, X, X, X, X, X, X, X,
O, O, X, O, O, X, O, O,
O, X, O, X, X, O, X, O,
X, O, X, O, O, X, O, X,
]
sense.set_pixels(Pixel_invader)
sleep(3)

sense.show_message("Voici les dernieres informations", 0.05)
print("Station meteo: Voici les dernières infos recueillies par notre sonde")

temp = sense.get_temperature()
print("La température est de :",temp ," °C")
temperature= str(temp)
sense.show_message("La temperature est de :", 0.05)
sense.show_message(temperature, 0.05, [255, 0, 0])
sense.show_message("degres C", 0.05)

pres = sense.get_pressure()
print("La pression est de : ",pres ," Millibars")
pressure= str(pres)
sense.show_message("La pression est de :", 0.05 )
sense.show_message(pressure, 0.05, [255, 0, 0])
sense.show_message("Millibars", 0.05)

hum = sense.get_humidity()
print("L'humidité est de : ",hum ," %")
humidity= str(hum)
sense.show_message("L'humdite est de :", 0.05)
sense.show_message(humidity, 0.05, [255, 0, 0])
sense.show_message(" %", 0.05)

ori = sense.get_orientation_degrees()
print("L'orientation est de : ",ori ,"degrés")
orientation= str(ori)
sense.show_message("L'orientation est de :", 0.05)
sense.show_message(orientation, 0.05, [255, 0, 0])
sense.show_message("degres", 0.05)
```

Mon tuteur m'a ensuite demandé d'améliorer le script afin que les diverses informations soit accessible suivant la direction désignées par le joystick disponible sur le Sense HAT, ainsi que d'afficher une image correspondant au temps présumé selon la valeur de la pression ou de la température.

```
X=[255, 255, 0]
Y=[51, 0, 204]
Z=[204, 204, 255]
O=[0, 0, 0]

pixel_soleil= [
O, O, O, X, O, O, X, O,
O, X, O, X, O, X, O, X,
O, O, X, X, X, X, X, O,
X, X, X, X, X, X, O, O,
O, O, X, X, X, X, X, X,
O, O, X, X, X, X, O, O,
O, X, O, X, O, X, O, O,
O, O, O, X, O, O, X, O,
]
```

La partie du code visible ici permet la création d'images sur l'écran soixante-quatre LED du Sense Hat. En effet on peut observer les quatre variables que créent X, Y, Z et O avec leur définition en RGB permettant d'obtenir un grand nombre de couleurs.

On peut observer ici est la gestion du joystick. En effet on peut voir que si un évènement touche est lancé alors selon la touche qui sera activée le Sense Hat prendra la mesure de la température ambiante, de la pression ou encore de l'humidité. Ensuite selon la valeur obtenue il affichera une des images établies précédemment, par exemple si la température est supérieure à trente-cinq degrés Celsius alors l'écran LED affichera une image de soleil.

```
while running:
    for event in pygame.event.get():
        print(event)
        if event.type == KEYDOWN:
            if event.key == K_DOWN:
                temp = sense.get_temperature()
                temp = round(temp,2)
                if temp < 0:
                 sense.set_pixels(pixel_flocon)
                 sleep(2)
                if temp > 35:
                 sense.set_pixels(pixel_soleil)
                 sleep(2)
                print("La température est de :",temp ," °C")
                temperature= str(temp)
                sense.show_message("Temperature :", 0.05)
                sense.show_message(temperature, 0.05, [255, 0, 0])
                sense.show_message("degres C", 0.05)
            if event.key == K_UP:
                pres = sense.get_pressure()
                pres = round(pres,2)
                if pres > 1000:
                   sense.set_pixels(pixel_soleil)
                   sleep(2)
                if pres < 1000:
                   sense.set_pixels(pixel_nuage)
                   sleep(2)
                if pres < 700:
                   sense.set_pixels(pixel_pluie)
                   sleep(2)
                print("La pression est de : ",pres ," Millibars")
                pressure= str(pres)
                sense.show_message("La pression : ", 0.05)
                sense.show_message(pressure, 0.05, [255, 0, 0])
                sense.show_message("Millibars", 0.05)
            if event.key == K_LEFT:
                hum = sense.get_humidity()
                hum = round(hum,2)
                if hum > 90:
                   sense.set_pixels(pixel_pluie)
                   sleep(2)
                if hum > 60:
                   sense.set_pixels(pixel_nuage)
                   sleep(2)
                if hum < 30:
                   sense.set_pixels(pixel_soleil)
                   sleep(2)
                print("L'humidite est de : ",hum," %")
                humidity= str(hum)
                sense.show_message("L'humidite : ", 0.05)
                sense.show_message(humidity, 0.05, [255, 0, 0])
```

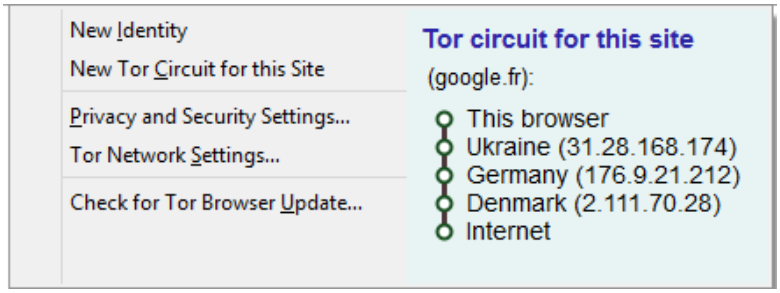
Ce projet m'a permis d'apprendre les bases du langage python. La plus grande difficulté fut de mettre en place la manœuvrabilité au joystick, car aucune information n'était disponible sur la documentation en ligne.

Mise en place d'un relai Tor :

Le troisième projet qui me fut confié était de mettre en place une plateforme wifi grâce à un raspberry. J'ai réalisé cela sur la distribution Raspbian Wheezy. J'ai également eut besoin de deux dongles wifi. Ce projet plutôt simple dans l'idée puisqu'il s'agit de se connecter au wifi de l'entreprise grâce à un premier dongle (ou en Ethernet) puis de mettre en place un point d'accès Wi-Fi grâce au second dongle et hostapd, s'est révélé plus difficile à mettre en place.

En effet il repose sur la gestion de l'interface réseau du raspberry et de divers éléments tel que le dhcp et le dnsmasq et le pare-feu iptable. Ensuite afin de sécuriser le réseau créé, mon tuteur m'a demandé d'obliger l'utilisateur à utiliser Tor lorsqu'il se connecte au réseau créé par le raspberry. Cette action a été réalisée via l'installation et des réglages nécessaires du paquet Tor.

Le fonctionnement de Tor est un sujet complexe avec le routage en oignions qui permet une certaine vie privé lors de l'utilisation d'internet. Cette communication est basée sur des nœuds aléatoires situés dans le monde entier. Chaque nœud connait l'identité du précèdent et du suivant mais pas plus. Ainsi le premier nœud connait l'adresse de la machine de l'utilisateur, mais ce n'est plus le cas dès le deuxième nœud. Par ce phénomène additionné à un double cryptage grâce à une clé publique et une clé privé, l'utilisateur devient quasi-indétectable.



On peut observer ci-dessus les différents nœuds mis en œuvre lors d'une utilisation du browser Tor.

Sécurité informatique et Pineapple pi :

Ensuite mon tuteur m'a demandé de me renseigner sur les distributions liées à la sécurité telle que Kali Linux et d'essayer de monter un Pineapple pi. Kali est une distribution de linux spécialisée dans l'analyse des réseaux et la sécurité informatique. La distribution de Kali orientée pour raspberry possède de nombreux outils bien qu'elle soit loin d'être aussi performante que la distribution classique. Elle possède notamment d'outils permettant l'analyse de réseaux wifi et le craquage de clé wifi, la mise en place d'honeypot.

Un honeypot est un mécanisme de défense active, c'est un leurre permettant d'attirer sur lui les attaques produites par des logiciels espion ou un hacker. Ce procédé permet de dévier les attaques voire de les neutralisées.

Ensuite mes recherches sur le Pineapple pi m'ont permis de mettre en place un Pineapple pi sur le raspberry grâce à la distribution Pwnie express. Cet outil particulièrement redouté des personnes travaillant dans la sécurité informatique est capable, grâce aux requêtes envoyées en permanence par les téléphones portables afin de vérifier la présence d'un réseau wifi enregistré, de recréer un réseau similaire en apparence à votre réseau domestique ou tout autre réseau enregistré sur votre téléphone. Le téléphone se connecte donc automatiquement en reconnaissant le réseau et sachant le Pineapple comprend une carte 3G la connexion internet est possible, cependant toutes les informations que vous entrez pendant cette connexion sont enregistrées sur le dispositif. Cette perspective est préoccupante d'autant plus qu'il n'existe pas vraiment de manière de le contrer à moins de supprimer les réseaux wifi préenregistrés sur son téléphone portable.

B) Les connaissances acquises :

Grâce aux raspberry pi :

Le premier projet raspbian/Kodi m'a familiarisé en premier lieu avec la gestion de l'interface réseau d'une machine fonctionnant avec un système UNIX. En effet le fait de configurer le wifi était nécessaire afin que je puisse accéder au raspberry grâce au protocole ssh. Par la même occasion j'ai pu utiliser le protocole ssh et donc en obtenir une plus grande compréhension, il fonctionne sur le port 22 et ne fonctionne qu'entre des machines appartenant au même réseau. Il existe un fichier sous Unix permettant la configuration des utilisateurs avec lesquels l'on peut se connecter grâce au protocole ssh. Les commandes basiques permettant de mettre à jour un système Unix après son installation sont apt-get update et upgrade. Tandis que les commandes permettant l'installation d'un paquet selon les deux voies disponibles : apt-get qui permet l'installation de paquets grâce à un dépôt APT. Mais également wget qui permet le téléchargement de paquets via les protocoles HTTP, HTTPS et FTP.

Grâce au second projet et l'utilisation du Sense-Hat j'ai appris à réaliser des scripts basiques en python, à me déplacer dans le système UNIX grâce aux commandes cd et pwd, mais également à gérer les droits d'utilisateurs liés à ces scripts à l'aide des commandes chmod et chown. Enfin à lancer des scripts ou programme dès le démarrage du raspberry. Cela se faisant grâce au placement du script dans le dossier /ets/init.d puis grâce à la commande update-rc.d. Cette manipulation permet notamment de disposer d'une station météo répondant au joystick du Sense Hat sans avoir besoin de le brancher à un écran, utile pour les présentations.

Le troisième projet m'a appris comment mettre en place le browser Tor sur un système d'exploitation Unix mais également de gérer deux interfaces Wi-Fi une recevant un réseau et l'autre servant de point d'accès. Une partie importante de ce projet a aussi reposée sur une compréhension des fonctions DNS et DHCP du logiciel DnsMasq (petit serveur peu gourmand utile pour les points d'accès WI-FI) et du pare-feu iptable (pare-feu en ligne de commande fonctionnant avec un système de règles permettant l'ouverture et la fermeture des ports). Enfin j'ai appris via ce projet l'importance des dépôts présent sur les systèmes Unix, ces dépôts permettent de télécharger et mettre à jour les logiciels compatibles.

Le quatrième projet m'a permis de découvrir la distribution Kali orienté vers la sécurité réseau et certains de ses outils. En effet j'ai pu m'initier au hacking basique d'un réseau wifi grâce aux logiciels airocrack-ng et airodump-ng permettant le crack d'une clé Wi-Fi. J'ai découvert nikto, un testeur de serveur web, kismet sniffer un détecteur de Wi-Fi, ce sont seulement quelques-uns des nombreux logiciels disponibles sur ces distributions. Ce projet m'a ouvert les yeux sur la facilité avec laquelle on peut pirater un réseau ou autre avec un peu de patience et de connaissances. De plus la possibilité de créer un Pineapple pi avec un budget aussi limité pose une sérieuse menace de sécurité liée aux réseaux Wi-Fi. Il n'existe de plus pas de réelle contre mesure, il faudra être prudent lors de l'utilisation de réseaux Wi-Fi sur téléphone portable notamment.

Grâce à mon tuteur:

Au cours de mon stage j'ai également aidé mon tuteur dans son travail en réalisant de nombreuses taches telles que le formatage d'ordinateurs portables pour des clients, la mise en place de serveur NAS, l'achat de matériel chez des revendeurs. J'ai également pu rédiger des notices sur l'utilisation de logiciels tels que Skype, ou de matériel tel que la caméra de conférence cc3000e appartenant au catalogue Logitech. J'ai pu également prendre part à la mise en place d'infrastructure chez des professionnels. Notamment le montage et la mise en place de NUC chez une professionnelle ainsi que divers écrans.

Ces diverses taches ont été formatrices. En effet j'ai notamment pu apprendre l'existence du protocole PXE qui permet de réaliser le formatage d'un ordinateur à partir d'un OS disponible sur un serveur. Il faut pour cela que l'ordinateur soit relié en filaire vers le serveur. Les divers formatages que j'ai pu réaliser m'ont également appris comment trouver les driver des constructeurs et les installer, mais également quelques connaissances sur le bios ou UEFI. Afin d'optimiser la mise en place du first boot d'un ordinateur j'ai dû apprendre notamment l'ordre d'installation des drivers: Chipset, LAN, Wi-Fi, Bluetooth, autres. Ce qui permet d'éviter un grand nombre d'erreurs. La réinitialisation du serveur NAS m'a permis de comprendre son utilité quant au partage d'information ainsi que l'intérêt de mettre en place un système de RAID pour la sécurité des données et l'efficacité des transferts.

J'ai pu apprendre des informations sur quelques-unes des normes Wi-Fi les plus utilisées, Wi-Fi ac : wifi très performant avec un débit quasiment dix fois supérieur sur un téléphone portable par rapport à un Wi-Fi 802.11g. Ainsi que l'intérêt d'utiliser un routeur dès la mise en place d'un réseau au lieu de la box fournie par le revendeur.

Découverte de logiciels variés tels que Visio 2013 afin de réaliser des schémas pour les présentations et ZMap afin de scanner un réseau.

Il m'a été également donné l'opportunité de travailler à un niveau administratif de l'entreprise en remplissant et joignant le service concerné afin d'obtenir un numéro EORI pour l'entreprise. Ce numéro permet le passage de commandes dans des pays étrangers par l'entreprise.

J'ai pu également avoir accès à l'active directory, service d'annuaire LDAP pour les services d'exploitation windows, de certaines entreprises travaillant avec Cityn. J'ai eu pour mission de mettre à jour les diverses informations présentes sur le personnel dans l'active directory.

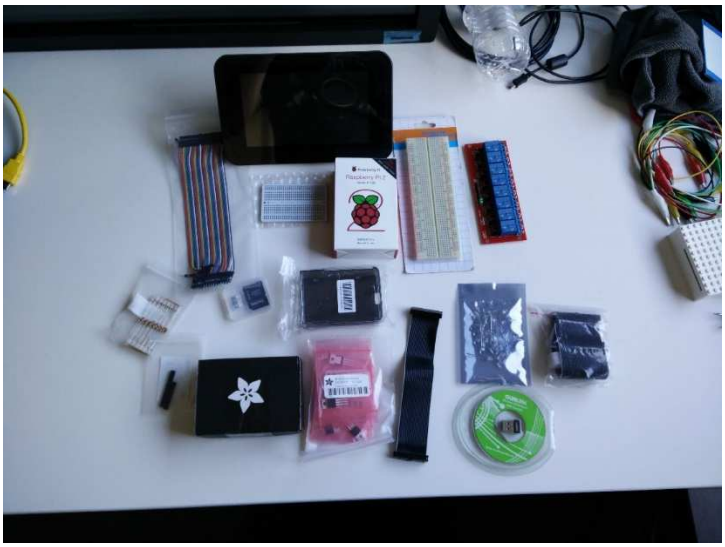
Enfin j'ai pu intervenir à plusieurs reprises avec des employés des entreprises se situant à proximité, en effet j'ai participé au maintien de leur environnement de travail au travers de la mise en place de nouveaux logiciels, la mise à jour. Mais également un soutient en cas de questions lors de l'utilisation d'outils tels que la suite office, TeamViewer, Skype.

II) Projet Domotique :

Le but de ce projet était de mettre en place une solution matérielle visant à remplacer le système permettant de gérer une salle de conférence appartenant à une entreprise locale. Mon tuteur m'a demandé de choisir le matériel nécessaire à la mise en place de ce réseau domotique en se basant sur des raspberry pi. Ce réseau devait être capable de gérer l'allumage de quatre groupes de lampes ainsi que l'ouverture ou la fermeture de deux stores roulants. Ceci devait de plus pourvoir être contrôlé à distance et ce n'importe où dans la salle de conférence. J'ai réalisé ce projet en collaboration avec le second stagiaire de l'entreprise en troisième année de Bachelor à SUPINFO.

A) Le choix du matériel et du système :

Apres plusieurs recherches nous nous sommes tournés vers l'utilisation de raspberry pi 2 modèle B, ainsi que du matériel électronique afin de mettre en place ce système. Il m'a paru indispensable d'utiliser deux raspberry pi, le premier étant relié au système électrique grâce à une carte huit relais. Alors que le second raspberry servirais de télécommande afin de réaliser les diverses actions du système. Il existe un système de domotique en accès libre appelé Domoticz grâce auquel il sera possible de mettre en place un serveur local web qui prendra en compte les scripts nécessaire à l'exécution des différentes actions du système. Le second raspberry munis d'un écran tactile sera dès le lancement orienté vers un browser web tel que Epiphany ainsi grâce à un compte sécurisé il pourra passer aux commandes sur serveur web local et ainsi commander à distance le système domotique.



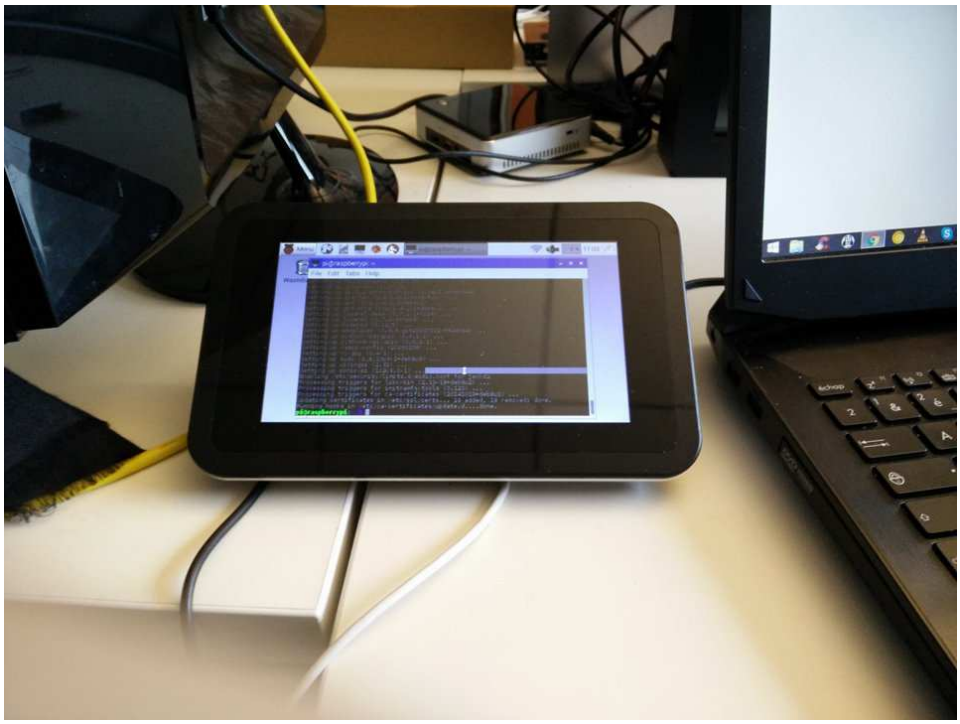
Matériel :

- Deux raspberry pi 2 B
- Deux cartes micro SD 16 et 32 Gb
- Une « breadboard »
- Plusieurs résistances 220 ohms
- Plusieurs mosfets, transistors permettant de moduler le courant
- LED
- Cable nappe GPIO
- Deux dongles wifi,
- Des cables GPIO (mâle/mâle, mâle/femelle)
- Deux servo moteurs,
- Ecran tactile 7 pouces
- Coque adaptée pour le raspberry avec écran tactile
- « Assembled pi cobbler » adaptateur permettant le branchement du cable nappe à la « breadboard »
- Une carte 8 relais, permettant le branchement du système domotique au réseau électrique.

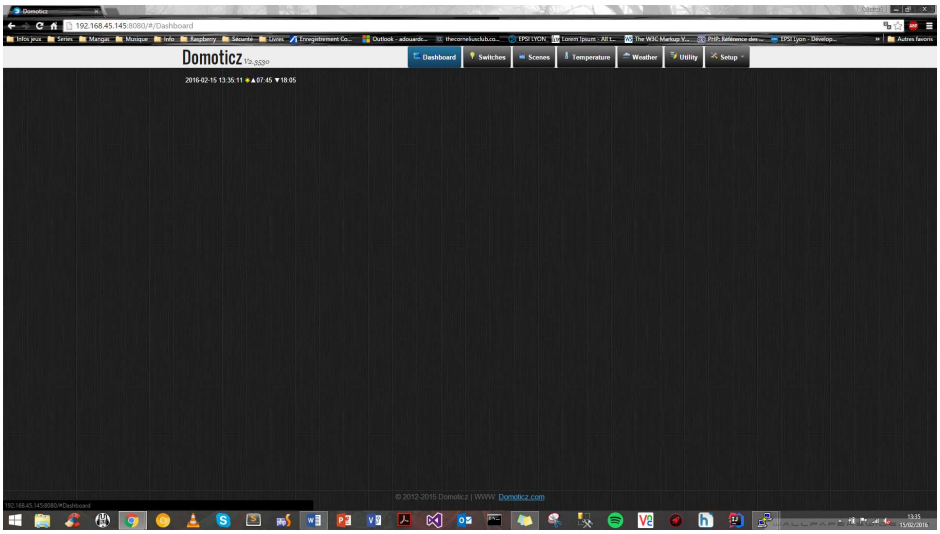
La mise en place du réseau :

Le premier raspberry ou hôte du serveur web a reçu une carte SD de 32 Go contenant un Os Raspbian Jessie ainsi qu'un dongle Wi-Fi afin de le connecter au réseau de l'entreprise. Puis il a fallu mettre en place le serveur web grâce à Apache. Ensuite j'ai ajouté le paquet Domoticz.

Le second raspberry c'est vu doté d'un écran tactile sept pouces ainsi que d'un boitier adapté afin de maintenir l'écran tactile. Il a été doté également d'un dongle Wi-Fi et d'une carte SD de 16Go. Pour ce raspberry l'installation de Epiphany et d'un script afin que Epiphany se lance au démarrage et en plein écran à suffit.



La mise en place en raspbian jessie sur le raspberry « remote » a été la première étape, puis il a fallu modifier les configurations du browser epiphany disponible avec raspbian jessy. Ces modifications ont été faites dans le but de permettre à epiphany de se lancer à chaque démarrage du raspberry, en plein écran mais également sur la page que nous lui avons indiqué dans le fichier de configuration à savoir le serveur local du second raspberry 192.168.45.145 :8080.



Sur le second raspberry j'ai installé en premier lieu raspbian jessie puis le package domoticz. Domoticz est un système automatisé, open source, permettant de gérer des installations domotiques. Apres de nombreuses difficultés lié aux intéractions entre le package disponible de domoticz et de raspbian jessie, j'ai décidé d'utiliser une image préchargée avec Domoticz. Ceci m'a permis d'obtenir une installation viable de Domoticz disponible sur l'ensemble du réseau à l'adresse 192.168.45.145:8080.

Voici une partie du fichier de configuration de domoticz :

```
# Do NOT "set -e"
/usr/local/bin/gpio export 14 out
/usr/local/bin/gpio export 7 out
PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/bin
DESC="Domoticz Home Automation System"
NAME=domoticz
USERNAME=pi
PIDFILE=/var/run/$NAME.pid
SCRIPTNAME=/etc/init.d/$NAME

DAEMON=/home/$USERNAME/domoticz/$NAME
DAEMON_ARGS="-daemon"
#DAEMON_ARGS="$DAEMON_ARGS -daemonname $NAME -pidfile $PIDFILE"
DAEMON_ARGS="$DAEMON_ARGS -www 8080"
DAEMON_ARGS="$DAEMON_ARGS -sslwww 443"
#DAEMON_ARGS="$DAEMON_ARGS -log /tmp/domoticz.txt"
#DAEMON_ARGS="$DAEMON_ARGS -syslog"

# Exit if the package is not installed
[ -x "$DAEMON" ] || exit 0

# Load the VERBOSE setting and other rcS variables
. /lib/init/vars.sh

# Define LSB log_* functions.
# Depend on lsb-base (>= 3.2-14) to ensure that this file is present
# and status_of_proc is working.
. /lib/lsb/init-functions
```

On peut observer dans ce fichier les ports GPIO que l'on va utiliser (ports 7 et 14), mais également le nom de l'utilisateur qui contrôle domoticz, ici pi, le port d'écoute : 8080, le chemin du fichier log.

Ensuite la phase de test a débuté afin de voir la mise en œuvre nécessaire pour gérer les lampes ainsi que les stores présents dans la salle de conférence.

Les ports GPIO test et fonctionnement :

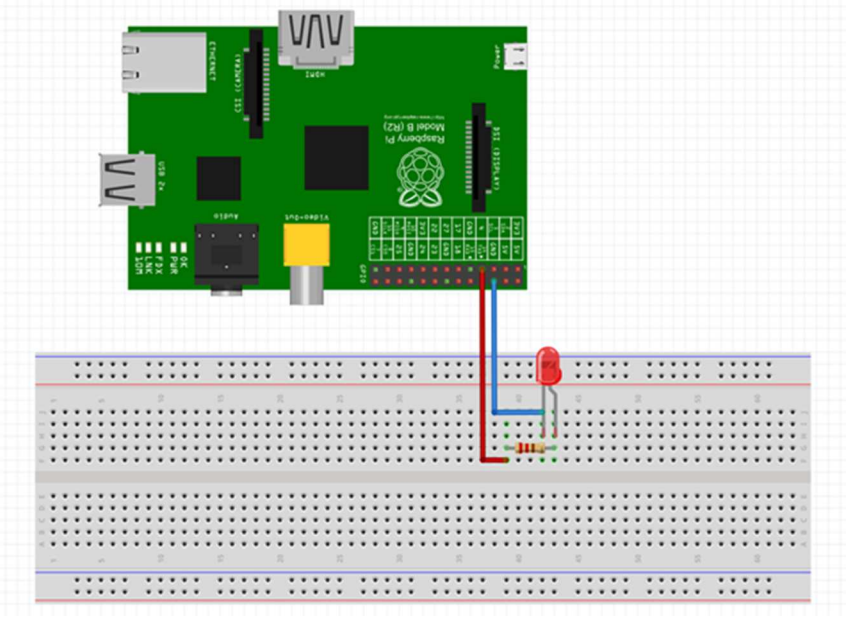
Pin#	NAME		NAME	Pin#
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I2C)		DC Power 5v	04
05	GPIO03 (SCL1 , I2C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I2C ID EEPROM)		(I2C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

Le raspberry pi 2 peut être utilisé dans la mise en place de système domotique grâce à ces ports GPIO, ces ports d'entrée/sortie permettent à toute carte dont ils sont équipés de communiquer avec d'autres circuits électroniques. Il existe des bibliothèques permettant de les paramétrer et e les contrôler suivant le langage.

On peut observer sur ce schéma trois grands types de port :

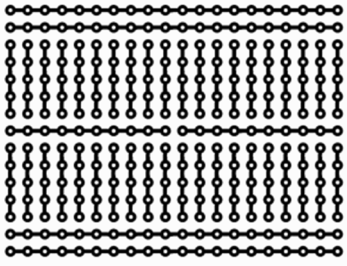
- Les ports GPIO
- Les ports Ground
- Les ports d'alimentation

Pour tester la mise en place des lampes nous avons eu besoin de créer un circuit électrique comprenant une LED, une résistance ainsi que deux fils électriques.



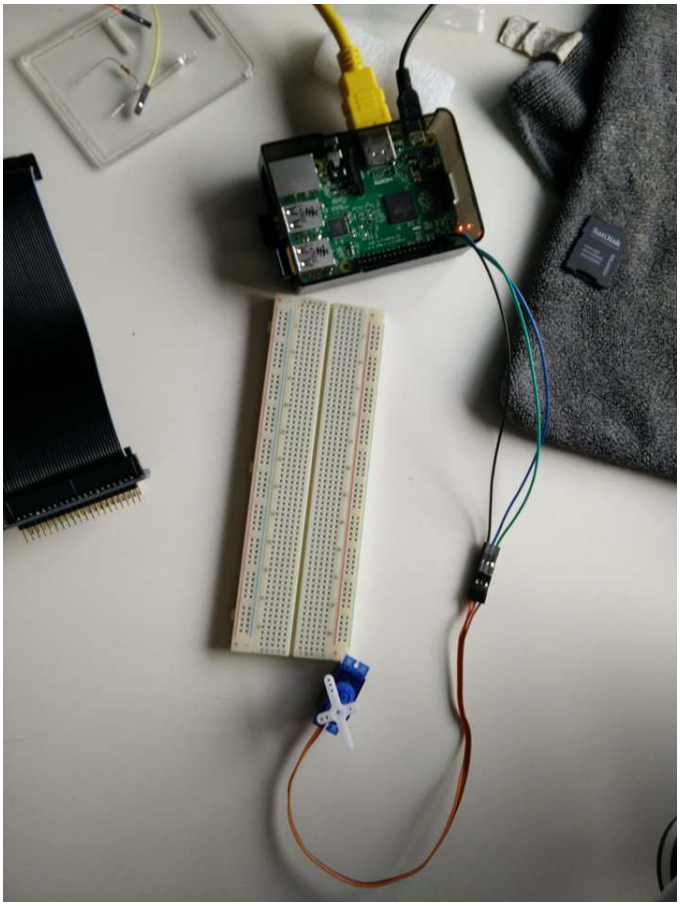
On peut observer dans le schéma un fil rouge partant d'un port GPIO, ce port est le GPIO 04, il est relié à la résistance qui est elle-même reliée à un fil bleu rejoignant un port Ground.

Le test c'est déroulé sur une « Breadbord », très utilisée pour tester des circuits électriques. En effet elle ne nécessite aucune soudure et est donc entièrement réutilisable.



Ce schéma indique les circuits internes de la « Breadboard ». Les cercles indiquent les points d'entrée et les traits les trajets possibles pour le courant électrique.

Pour simuler le fonctionnement des stores, nous avons relié un servomoteur FS90R de marque Fitec au raspberry à l'aide de trois câbles (Un relié à un GPIO, un relié à un port « ground » et un relié à un port d'alimentation 5V).



Cette étape a représenté de nombreuses difficultés liées à sa complexité, en effet un servo moteur est bien plus complexe qu'une LED qui peut être réglée de manière binaire. Le servo moteur a besoin pour sa part d'un script afin de démarrer. La principale difficulté que nous avons rencontrée a été de pouvoir arrêter le script à n'importe quel moment afin que les stores ne soit pas disponibles uniquement en position ouverte ou fermées mais également pour toutes les autres positions intermédiaires.

Domoticz permet de prendre en compte des scripts dans de nombreux langages tels que le python, le lua, le bash et bien d'autres. Le problème que nous avons alors rencontré était que lors de l'exécution d'un premier script notre moteur démarrait et était ensuite incapable de s'arrêter via les signaux que lui envoyait Domoticz. Ce fait était d'autant plus étonnant que lors des tests des scripts sur le raspberry le script en bash que nous avions confectionné arrêtaient bien le script en python permettant la rotation du moteur.

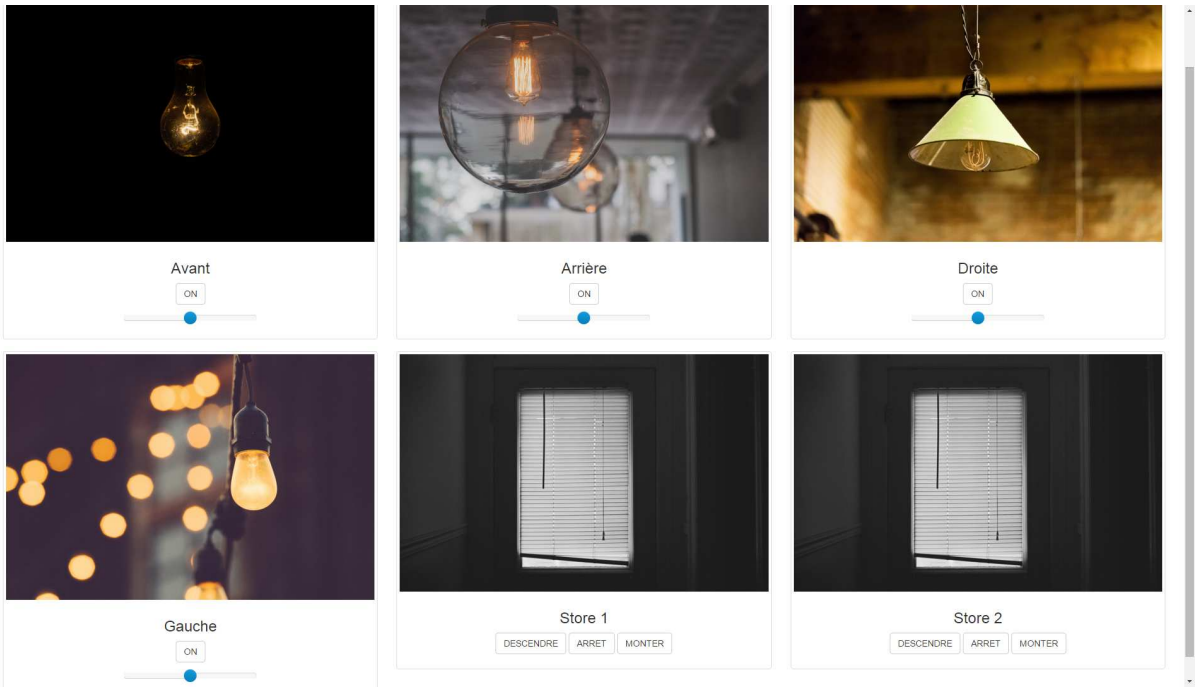
Après de nombreux problèmes lié aux interfaces pré-faites open sources nous nous sommes résolus à créer notre propre interface. Nous avons réalisé ceci avec l'aide de node.js, plateforme logicielle libre, et nginx, logiciel libre de serveur web alternative d'apache. Après avoir mis à jour notre système et installé les paquets précédemment cités nous avons mis en place le fichier de configuration nginx : domotic_central (/etc/nginx/sites-available). Dans le fichier de configuration nginx on définit la localisation du dossier de l'interface, le port d'écoute. Par la suite nous avons édité le fichier etc/host mise en place du host domotic_central pour qu'il corresponde à l'adresse ip du raspberry 192.168.45.145.

Ces diverses étapes nous ont alors permis de mettre en place les fichiers de la solution /var/www/domotic_central. Ensuite nous avons Installé express, Framework web pour node.js et socket.io une bibliothèque en javascript permettant des communications en temps réel entre un client et un serveur. L'étape suivante a été la mise en place du reverse proxy, Nginx est paramétré afin de recevoir les requêtes HTTP en écoutant sur le port 80 puis les redirige à node.js sur le port 3000.

La mise en place étant terminée à cette étape et nous avons pu démarrer le code de l'interface. Nous avons donc créé une page index.html et une page app.js, puis redirigé app.js vers l'index.html. L'installation de wiringPI a été nécessaire afin de nous permettre de contrôler les ports GPIO du raspberry. Par la suite nous avons mis en place un script javascript permettant de « dimmer » la LED.

Nous nous sommes retrouvés confronté à un problème technique que nous n'avions pas anticipé. Le raspberry ne dispose que d'un port GPIO PWM, ce type de ports permet de contrôler l'intensité d'une lampe, il n'est donc pas possible de contrôler les quatre groupes de lampes indépendamment si on veut qu'elles puissent varier leur tension. Il existe des cartes appelées 'hat' permettant d'ajouter au raspberry des ports GPIO PWM.

B) Mise en place de notre interface :



Mise en place du code :

Notre interface web est réalisée dans une page index en langage html et java script avec l'aide du Framework Bootstrap, la bibliothèque jQuery, la bibliothèque socket.io et la bibliothèque bootstrap-slider.

```
<script src="http://code.jquery.com/jquery-2.2.0.min.js">
</script>
<script src="/node_modules/bootstrap/dist/js/bootstrap.min.js">
</script>
<script src="/node_modules/bootstrap-slider/dist/bootstrap-slider.min.js">
</script>
<script src="/node_modules/socket.io-client/socket.io.js">
</script>
<script>
```

Du côté du fichier javascript app.js il nous faut également faire appel à de nombreux plugins :

```
var express = require('express');
var app = express();
var http = require('http').Server(app);
var io = require('socket.io')(http);
var GPIO = require('wiring-pi');
```

- express est un Framework web pour node.js
- La requête http permet la création d'un serveur http ce qui est nécessaire afin d'utiliser socket.io sur la même instance.
- La bibliothèque socket.io

- La bibliothèque wiring-pi permet le contrôle des ports GPIO du raspberry pi

La fonction apparaissant ici permet de gérer divers ports GPIO dont nous aurons besoin. On peut observer que les différents ports sont réglés sur OUTPUT car ils vont envoyer des informations et non en recevoir. Le port GPIO 18 est différent car c'est celui permettant de « dimmer » la LED.

Ensuite la fonction http.listen permet de définir le port sur lequel le serveur web va fonctionner.

```
// GPIO MAIN

GPIO.wiringPiSetupGpio();
GPIO.pinMode(18, GPIO.PWM_OUTPUT);
GPIO.pinMode(21, GPIO.OUTPUT);
GPIO.pinMode(16, GPIO.OUTPUT);
GPIO.pinMode(12, GPIO.OUTPUT);
GPIO.pinMode(25, GPIO.OUTPUT);


http.listen(3000, function(){
  console.log('listening on *:3000');
});
```

```
<div class="col-xs-12">
  <div class="col-xs-4 text-center">
    <div class="thumbnail">
      <img class="img-responsive" src="/images/gala_photo.png" alt="Test 1">
      <div class="caption">
        <h3>Avant <span id="status1"></span></h3>
        <p>
          <a id="Zone_1_On" type="button" class="btn btn-default LedSwitchOn"> ON </a>
          <a id="Zone_1_Off" type="button" class="btn btn-default hide LedSwitchOff"> OFF </a>
        </p>
        <p><input id="Dimmer_1" data-slider-id="dimmerSlider1" type="text" data-slider-min="0" data-slider-max="1024" data-slider-step="1" data-slider-value="512"/></p>
      </div>
    </div>
  </div>
  <div class="col-xs-4 text-center">
    <div class="thumbnail">
      <img class="img-responsive" src="/images/gala_photo1.png" alt="Test 2">
      <div class="caption">
        <h3>Arrière <span id="status2"></span></h3>
        <p>
          <a id="Zone_2_On" type="button" class="btn btn-default LedSwitchOn"> ON </a>
          <a id="Zone_2_Off" type="button" class="btn btn-default hide LedSwitchOff"> OFF </a>
        </p>
        <p><input id="Dimmer_2" data-slider-id="dimmerSlider2" type="text" data-slider-min="0" data-slider-max="1024" data-slider-step="1" data-slider-value="512"/></p>
      </div>
    </div>
  </div>
```

L'organisation visuelle, dans la page index.html, des six éléments se fait via la grille de Bootstrap : deux lignes possédant douze unités de large, donc chacun des six éléments en possède quatre de large. Les identifiants et classes apposés pour chacun des boutons sont également importants à noter car ils vont être essentiels dans la gestion des évènements en java script utilisés ensuite.

Pour réaliser ce projet nous avons décidé d'utiliser la bibliothèque java script socket.io pour sa grande vitesse de réponse entre clients et serveur web. Cette bibliothèque possède deux aspects, un aspect client dans le browser et un aspect serveur disponible pour node.js.

Voici un des switch gérant les évènements du côté de la page index.html, lorsque l'utilisateur clique sur un des boutons « on » des quatre groupes de lampe l'identifiant correspondant au bouton est saisi et une requête adaptée est envoyé au fichier app.js.

```
// ON PRESS
$('.LedSwitchOn').click(function(){
  switch (this.id) {
    case "Zone_1_On":
    zone_state1 = !zone_state1;
    socket.emit('Zone 1 Change State', zone_state1);
    break;
    case "Zone_2_On":
    zone_state2 = !zone_state2;
    socket.emit('Zone 2 Change State', zone_state2);
    break;
    case "Zone_3_On":
    zone_state3 = !zone_state3;
    socket.emit('Zone 3 Change State', zone_state3);
    break;
    case "Zone_4_On":
    zone_state4 = !zone_state4;
    socket.emit('Zone 4 Change State', zone_state4);
    break;
    default:
    break;
```

```
24    // User active / unactive LED
25    socket.on('Zone 1 Change State', function(data){
26      if(data == true) {
27        GPIO.pwmWrite(18, 512);
28        socket.emit('Zone 1 Status', 'Allumée');
29      } else {
30        GPIO.pwmWrite(18, 0);
31        socket.emit('Zone 1 Status', 'Eteinte');
32      }
33    });
```

Du côté du fichier app.js la requête a été reçue et l'action se met en place. Grâce à la fonction GPIO.pwmWrite la LED relié au GPIO 18 va s'allumée avec une intensité de 512 sur 1024.

Le code observable permet la mise en place du « slider » afin de « dimmer » les LED. La fonction formatter permet l'observation en temps réel de la valeur du « slider ».

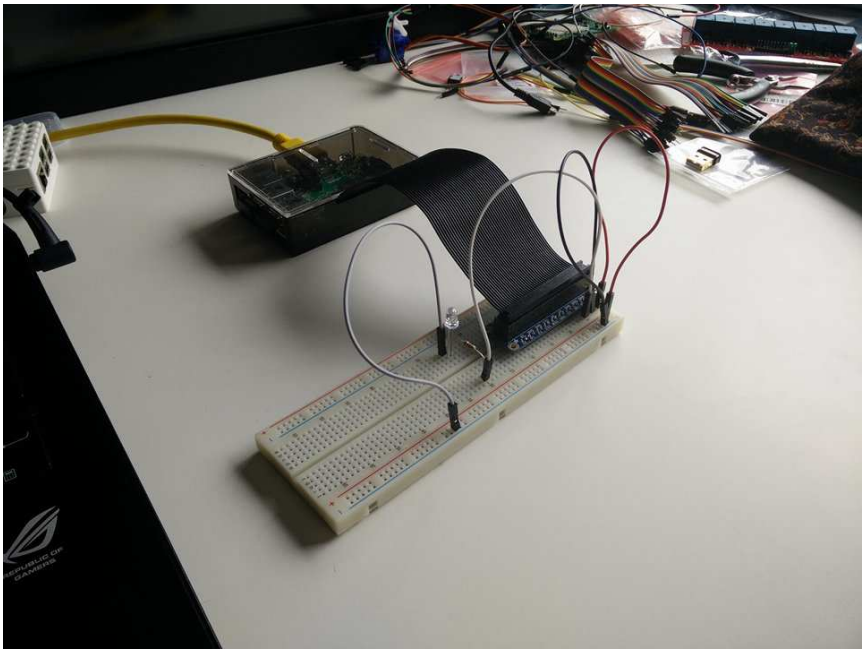
```
231   var slider1 = new Slider('#Dimmer_1', {
232     tooltip: 'hide',
233     formatter: function(value) {
234       if(zone_state1 == true) {
235         return socket.emit('Dimmer 1 Change State', value);
236       }
237     }
238   });
```

```
// DIMMERS
socket.on('Dimmer 1 Change State', function(value){
  GPIO.pwmWrite(18, value);
});
```

Du côté d'app.js la requête est reçue et on applique la valeur du « slider » à la LED avec la fonction GPIO.pwnWrite.

Pour les stores le fonctionnement est quasiment identique : seule la fonction GPIO change puisqu'il n'y a pas besoin de faire varier l'intensité. Il suffit lors du clic utilisateur sur descendre ou monter d'envoyer une impulsion, si une action est déjà en cours, il faut alors stopper en premier lieu l'impulsion du GPIO de l'action opposé. En clair si l'utilisateur clique sur le bouton descendre du premier store alors la première action à réaliser est de couper l'impulsion du GPIO correspondant à la montée du store correspondant. Enfin lors du clic sur le bouton arrêt, l'action coupe les deux ports GPIO correspondant au store.

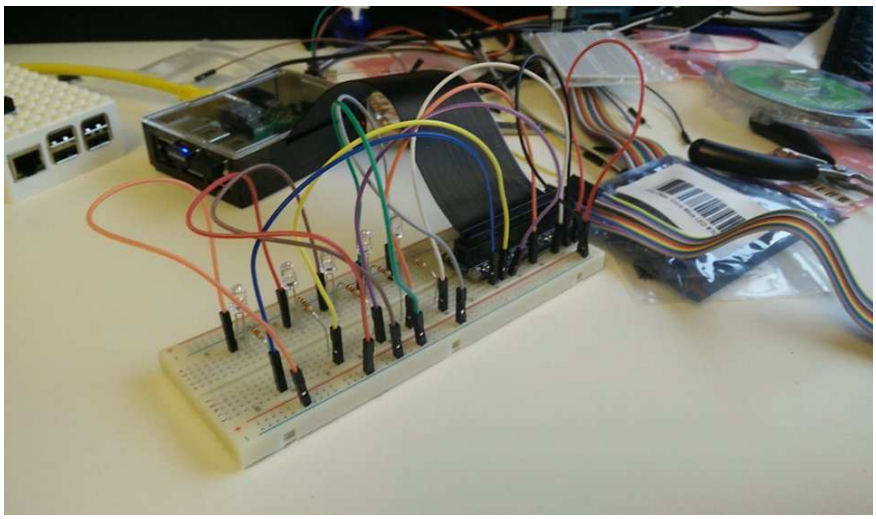
Tests de notre interface :



Le premier essai avait pour but d'allumer éteindre la lampe sur demande grâce au bouton allumer/éteindre disponible sur l'interface mais également la possibilité de « dimmer » la lampe. Ce qui signifit faire varier l'intensité de son éclairage.

Ce phénomène est possible grâce à une fonction de la bibliothèque wiring-pi qui intéragit sur le port PWM (Pulse With Modulation). Cette fonction fait varier la période entre les pulsations, plus la période diminue et plus la LED semble s'allumer intensément.

Le second test visait à vérifier le fonctionnement du code pour l'action sur les stores. Le fonctionnement des stores est simple ils possèdent chacun deux entrées une pour la montée et une pour la descente du store.



Il nous suffit donc d'envoyer une impulsion dans l'entrée correspondante suivant le bouton actionné et ce pendant une vingtaine de secondes qui est le temps maximum que met le store pour monter ou descendre entièrement. Nous avons de plus ajouté un bouton « Arrêt » permettant comme son nom l'indique l'arrêt du store à n'importe quel moment, celui-ci fonctionne en coupant l'apport des ports GPIO concerné par le volet que l'on souhaite arrêter.

Lors de mon dernier jour de stage l'interface était fonctionnelle et les tests sur les diverses LED étaient concluant. Malheureusement je n'ai pas pu participer ou observer la mise en place du système de domotique que nous avons créé sur le réseau électrique de l'entreprise et ce en raison du problème de port GPIO de type PWM cité précédemment. Le matériel permettant la mise en place du système complet arrivera après la fin de mon stage et sera donc mis en place sans moi.

Conclusion :

Ce stage a été très enrichissant sur de nombreux aspects.

Le fait que le domaine d'activité de l'entreprise pour laquelle j'ai effectué le stage soit l'infrastructure réseau. En effet cet aspect de l'informatique que nous n'avons pas encore approfondis en cours est très intéressant et mène à des carrières variées.

L'approfondissement de mes connaissances des systèmes Unix, ces systèmes d'exploitation dont la compréhension est indispensable pour de nombreux aspect de l'informatique tel que l'infrastructure réseau.

La découverte des raspberry pi et le grand nombre d'applications qui leur sont liés. En effet les possibilités semblent infinies allant du serveur web au super ordinateur en passant par le serveur NAS.

Le travail en équipe avec le second stagiaire de l'entreprise. Actuellement en troisième année de Bachelor Noé a été d'une grande aide lors de mes balbutiements au niveau du JavaScript mais également dans la compréhension de notions essentielles de l'infrastructure.

L'apprentissage de notion en électronique. Mes connaissances en électronique quasi inexistante avant le début de ce stage ont été grandement améliorées grâce aux nombreux tests et manipulations sur le projet domotique.

Augmentation de mes connaissances matérielle grâce aux manipulations sur les divers NUC, raspberry, ordinateurs, NAS, routeurs mais également logicielle avec la découverte de Zmap, Filezilla et de nombreux autres sous système Unix.

En programmation j'ai pu m'essayer à plusieurs nouveaux langages : Python, Bash, JavaScript.

Le relationnel avec les clients ; les connaissances en informatique de ces derniers étaient très variables. Les questions posées en l'absence de mon tuteur allaient du support à l'utilisation de TeamViewer, à la mise en place d'une ip fixe sur un poste de travail en passant par des domaines pointus de sécurité informatiques.