La Veilleuse pour enfant connectée

RAPPORT DE PROJET IMA4

Eleves ingénieurs:

Romain RUET

Julien BIELLE

Année 2015/2016

SOMMAIRE

Introduction.....2

I.Présentation générale du projet.....3

I.1 Objectif et description du projet.....3

I.2 Choix matériels et logiciels.....3

II.Conception du prototype.....5

II.1.Développement des fonctionnalités.....5

II.1.1.Projection des image.....5

II.1.2.Diffusion des berceuses.....6

II.1.3.Webcam.....7

II.1.4.Application Web.....7

II.2 Modélisation et établissement de la structure extérieure.....8

III. Limites de notre projet.....10

Conclusion.....11

Annexes.....12

INTRODUCTION

Nous connaissons tous les petits accessoires de nuit indispensables aux enfants dès leur plus jeune âge, et à leurs parents dans le but d'assurer leur confort. Pour l'enfant, cela va de la petite veilleuse, destinée à éclairer faiblement la chambre de l'enfant, à la berceuse. Pour les parents, le Babyphone pour écouter ce qu'il se passe dans la chambre et savoir si le bébé est réveillé ou non.

Dans une ère où les objets connectés se multiplient de plus en plus, le sujet de notre projet paraissait évident. Dans le cadre de notre quatrième année à Polytech Lille, nous nous sommes donc lancé dans la conception d'une veilleuse pour enfant connectée ,qui intégrerait la plupart des fonctionnalités nécessaires au confort d'un bébé.

Dans une première partie, nous ferons une présentation générale du projet, puis nous expliciterons la phase conception. Enfin, nous parlerons des problèmes que nous avons rencontré et nous ferons un point sur notre ressenti à l'établissement du projet.

I. Présentation générale du projet

I.1 Objectif et description du projet

L'objectif général de ce projet est de créer un prototype de veilleuse pouvant être contrôlé à distance par le biais d'une application accessible sur tout type de support connecté, comme un ordinateur, un smartphone ou une tablette.

L'application permettra de contrôler les fonctionnalités de la veilleuse suivantes:

- Importer des images et les projeter au plafond de la chambre de l'enfant
- Importer des musiques et les diffuser
- Lancer la surveillance via une webcam et accéder à la vidéo

Il devra donc être possible de se connecter à la veilleuse à distance et, une fois la connection établie, d'accéder à l'application web dédiée permettant de la contrôler. Nous avons donc choisi de configurer la veilleuse comme un point d'accès wifi, et d'y implanter un serveur web auquel on accédera depuis l'application. L'application est codée en HTML, php et javascript à la manière d'un site internet.

I.2 Choix matériels et logiciels

En ce qui concerne la gestion des différentes fonctionnalités, nous avons opté pour une Raspberry Pi 2B. La distribution linux Raspbian étant installée dessus, nous aurons accès à tous les outils nécessaires pour gérer nos fonctionnalités.

La projection d'image se fera par le biais du picoprojecteur ASUS S1 qui sera branché sur le port HDMI de la Raspberry. Ce projecteur ayant un port USB externe, la Raspberry pourra être directement alimentée, et ne nécessitera pas de prise externe à la veilleuse supplémentaire (en plus de celle nécessaire à l'alimentation du projecteur). La diffusion de musique se fera également par le biais de la sortie HDMI car le projecteur dispose d'un haut parleur intégré.

Nous utiliserons également un dongle Wi-Fi, le Wi-Pi dont nous inverserons le rôle dans le but de permettre une connection d'un ou plusieurs appareils externes vers la Raspberry, et non plus de connecter la Raspberry à Internet.

Pour gérer la surveillance à distance, nous utiliserons le module RaspiCam, qui permet de prendre des photos et des vidéos et de les enregistrer directement sur la Raspberry.

Coté logiciel, nous aurons à installer plusieurs paquets sur la Raspberry dans le but de gérer chaque fonctionnalité.

Pour la surveillance vidéo, nous utiliserons le paquet Raspistill qui permet de gérer directement la RaspiCam.

Pour la projection d'images, nous utiliserons FBI (pour FrameBuffer Imageviewer) et notamment sa version plus récente FIM (pour Framebuffer imageviewer IMproved), qui permet de monopoliser la sortie vidéo de la Raspberry dans le but d'afficher une image.

Pour la diffusion de musique, nous avons cherché un lecteur musical qui ne nécessite pas d'interface graphique pour fonctionner. Nous avons trouver le logiciel MOC (pour Music On Console) qui permet de jouer de la musique directement avec quelques commandes à rentrer dans le terminal.

Enfin, l'application sera hébergée sur un serveur apache. Nous attribuerons ensuite une adresse ip statique à la Raspberry. Pour accéder à l'application, il suffira seulement de se connecter à la Raspberry et d'aller à l'adresse 192.168.100.1.

II. Conception du prototype

II.1.Développement des fonctionnalités

II.1.1 Projection des images

Comme nous l'avons expliqué plus haut, la projection des images se fera par le biais du picoprojecteur ASUS S1. On peut voir sur l'image suivante qu'il est vraiment très compact, ce qui facilitera son intégration dans notre prototype final.

ASUS S1

Une autre fonction que la veilleuse doit proposer, c'est de laisser le choix aux utilisateurs des images qu'ils veulent diffuser. Nous avons donc intégré à notre application un formulaire d'upload de fichier (voir annexes) qui permettra à l'utilisateur d'aller parcourir ses fichiers locaux (sur son ordinateur ou son smartphone) et de choisir l'image qu'il souhaite diffuser. Le fichier sélectionné sera alors traité par un fichier php qui vérifiera si c'est bien une image, et l'enregistrera dans un dossier spécifique dans le serveur sur la Raspberry.

Une fois l'image enregistrée, l'application lui permettra de choisir, parmi toutes celles qui ont été enregistrées, celle qu'il veut diffuser sur la veilleuse.

Coté Raspberry, la commande qui permettra d'afficher cette même image est la suivante :

sudo fim -a <chemin_vers_image>

L'option *-a* permet à l'image de s'adapter automatiquement aux dimensions de l'écran projeté. Cette commande sera lancé dans un script php via un *shell_exec(<commande>);* après avoir récupéré en variable le chemin vers l'image choisie sur la page de sélection précédente. Pour faire passer une variable d'un script à l'autre, il suffit de lier les deux comme suit :

<a href="<chemin_vers_page_suivante>?variable">lien

Pour ensuite récupérer cette variable dans la page suivante, il faut rajouter la ligne suivante dans le script php :

$var=$_GET[variable];

Dans le cas où l'utilisateur souhaite arrêter la projection, il suffit de récupérer le PID associé au processus *fim* grâce à la commande *pgrep*, qui retourne directement le numero du processus qui correspond à la commande placée en paramètre. On intègre directement cette commande dans un script php (avec un *shell_exec*). Une fois le pid récupéré, il suffit de le killer, et l'image cesse d'être projetée.

II.1.2 Diffusion des berceuses

Pour la diffusion des berceuses, nous avons utilisé l'outil MOC. C'est un lecteur audio qui peut s'utiliser sans interface graphique. L'application propose, comme pour la diffusion des images, de télécharger des musiques et de les enregistrer directement sur le serveur local de la Raspberry.

Le raisonnement est précisément analogue à celui pour la gestion des images, seul les commandes pour exploiter les titres musicaux est différente. Pour utiliser l'outil MOC sans interface graphique, il faut préalablement lancer un serveur musical. Pour cela, nous avons modifié le fichier */etc/rc.local* pour qu'il lance la commande :

sudo mocp -S

Qui lance le serveur musical de l'outil MOC.

Une fois le serveur lancé, il faut spécifier au lecteur que le dossier où toutes les musiques sont enregistrée sera le dossier dans lequel il faudra aller chercher les fichiers à jouer. Pour cela, on utilise la commande :

mocp -a <chemin_vers_le_dossier>

A partir de la, pour lancer la lecture, on utilise la commande :

mocp -p

Qui lance la lecture de la musique sélectionnée dans l'application.

Pour ce qui est de faire sortir un son de la veilleuse, nous avons redirigé directement la sortie audio de la raspberry sur sa sortie HDMI. De ce fait, le picoprojecteur proposant des haut-parleurs intégrés, la musique sera diffusée à travers ces derniers.

Dans le cas ou les parents souhaitent arrêter la diffusion de la musique, on lance, de la même manière que pour la projection d'images, un script php qui effectue un *shell_exec* de la commande *mocp -s* , destinée à arrêter le serveur musical.

II.1.3 Webcam

Pour la Webcam, nous avons donc utilisé le module RaspiCam côté materiel ,et raspistill côté logiciel. Raspistill permet de prendre une image via le module RaspiCam par simple écriture d'une ligne de commande. Le principe de notre Webcam est d'activer raspistill à intervalles réguliers.

Pour cela, nous utilisons le paquet "webcam" qui permet ,une fois lancé, de prendre une image avec raspistill à la fréquence que nous voulons (Maximum 2 images par seconde) et de l'enregistrer dans le dossier de notre choix (Dans notre cas un dossier nommé "Videos"). Webcam prend, de plus, peu de place ,car il ne stock pas toutes les images prisent par la caméra, mais remplace l'ancienne image par la plus récente.

Afin que les parents puissent voir leur enfant, nous avons crée dans l'application une page à cet effet. La page est un fichier php qui récupère et affiche l'image contenu dans le dossier "Videos". Nous rafraîchissons l'image toutes les 2 secondes grâce à une fonction javascript.

Au niveau de l'activation et de la désactivation de "webcam", un script shell se lance à l'ouverture de la page php et celui-ci lance webcam via la commande :

./webcam webcam.conf

webcam.conf étant le fichier de configuration de webcam.

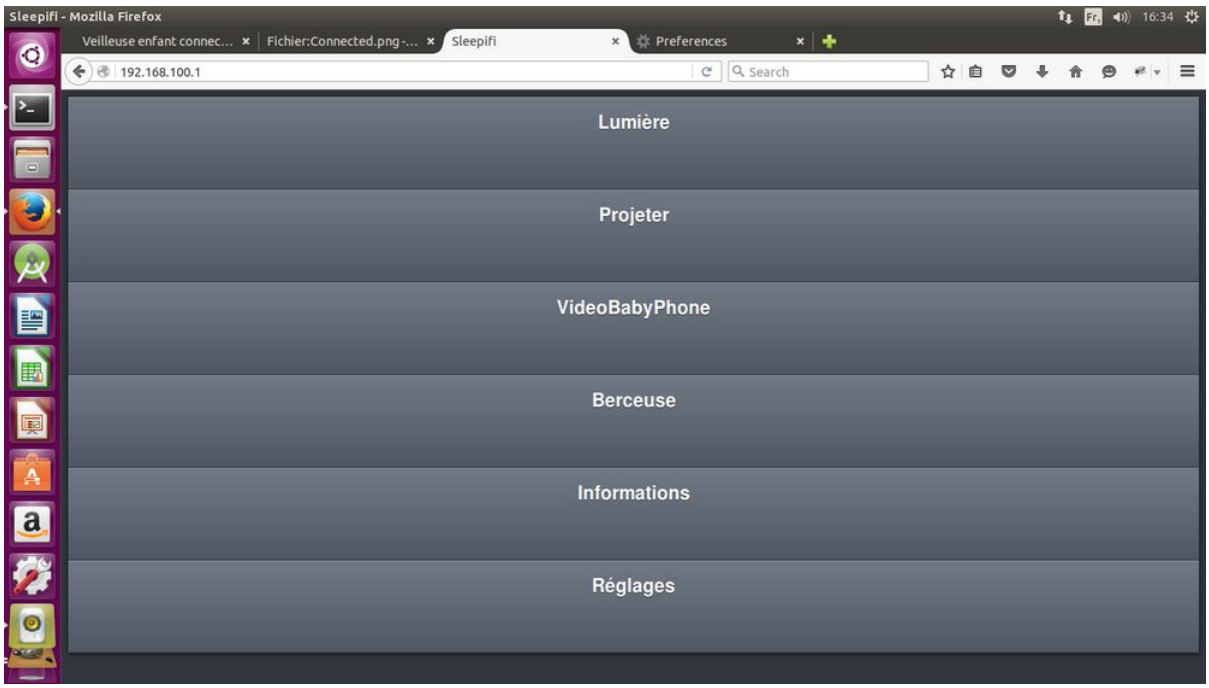
Ensuite, pour éteindre "webcam", on kill le processus webcam de la même façon que pour la projection et la berceuse.

II.1.4 Application Web

L'application web a entièrement été développée avec les langages HTML, php, javascript, et CSS pour la mise en forme.

Elle se présente comme un menu avec plusieurs choix, qui lorsqu'on clique dessus, laissent apparaître différents choix, comme "importer nouvelle image" ou "choisir image à diffuser", etc.

le rendu de la page est le suivant :



Rendu de l'application

Comme on peut le voir, on a bien accès à toutes les fonctionnalités nécessaires au contrôle de la veilleuse à distance.

Le rendu sera également le même sur un smartphone ou une tablette.

Chaque section sera ensuite liée à sa page de traitement HTML ou php dédiée.

II.2 Modélisation et établissement de la structure extérieure

Du point de vue utilisateur, il est important que la veilleuse soit jolie et peu encombrante.
Nous avons donc conçu l'enveloppe extérieure de la veilleuse selon ces deux critères.

Pour l'aspect esthétique, nous avions opté pour une structure de forme circulaire se rétrécissant en haut et imprimée en 3D (voir annexes)

Pour l'aspect ergonomique, nous avons mesuré la totalité des composants de la veilleuse (Raspberry, picoprojecteur, etc) afin d'étudier la disposition la plus optimale possible de ceux-ci.

Au finale, la veilleuse mesure 20x20 centimètres environ ce qui est acceptable. Par soucis technique, la structure a été réalisée en contre-plaqué découpé à la découpeuse laser (voir Annexes).

III. Limites de notre projet

Nous avons réussi durant ce projet à creer une veilleuse pour enfant dont les fonctionnalités sont des plus confortables pour les parents et aide l'enfant a s'endormir.

Cependant, avec un peu plus de temps nous aurions pu rajouter quelques options comme un gestionnaire d'horaire d'allumage ou encore une lampe contrôlable à distance.

Nous voulions également ajouter des capteurs dont les informations seraient récupérées par la Raspberry et visible sur l'application (bruits, température)mais les capteurs que nous avions acheté étaient analogiques alors que les ports de la Raspberry sont digitaux.
Ils nous auraient fallu un convertisseur analogique numérique pour régler ce problème ou bien creer un circuit électronique à base d'amplificateur opérationnel (AOP) qui permettrait de détecter si le bébé pleure ou alors si il fait trop chaud ou trop froid avec une simple valeur seuil de tension.

Également, la structure a un rendu satisfaisant, mais l'allumage du picoprojecteur et de la Raspberry n'est pas optimal.
En effet, le Asus S1 a son bouton marche/arrêt sur sa face arrière et il est difficile d'y accéder lorsque le picoprojecteur est dans la veilleuse.
Pour pallier à ce problème et placer un bouton sur l'extérieur de la veilleuse, nous devions démonter le picoprojecteur ce qui nous était défendu dans le cadre de ce projet.

Des modifications et des optimisations auraient donc pu être apportées à notre veilleuse avec du temps, néanmoins, ses fonctionnalités actuelles sont satisfaisantes et répondent au besoin des parents.

Conclusion

En conclusion, ce projet à été pour nous le moyen d'enrichir nos connaissances sur la Raspberry Pi et sur le codage en html, php, javascript, etc.

Ce fût notre premier projet IMA effectué totalement de façon autonome. Au cours de ces douze semaines, nous avons donc pu mettre en pratique les connaissances apprises durant notre cursus ingénieur à Polytech.

Cette expérience nous a familiariser avec des situations auxquelles nous pourrions très probablement être confrontés dans nos prochaines expériences professionnelles. Comme par exemple, la répartition du travail en équipe, le respect des échéances fixées, la tenue hebdomadaire d'un compte-rendu d'activités (wiki), la prise de responsabilité, la recherche d'alternatives faisant suite à certains problèmes.

Au final, nous avons réussi à proposer un prototype totalement opérationnel pour les fonctions qu'il propose, même si des améliorations pourraient être à prévoir.

Annexes

Raspberry Pi 2B

Dongle Wi-Pi

RaspiCam

ASUS S1

Rendu Final de la Veilleuse

```
iface lo inet loopback
iface eth0 inet dhcp

allow-hotplug wlan0
iface wlan0 inet static
        address 192.168.100.1
        netmask 255.255.255.0
```

/etc/network/interfaces

```
interface=wlan0
driver=nl80211
ssid=sleepi_fi
hw_mode=g
channel=6
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=sleepipi
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

hostapd.conf (config du pointd'accès)

```
1  <!doctype html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <title>Envoyer l'image sur la veilleuse</title>
6      <link rel="stylesheet" href="style.css">
7    </head>
8    <body>
9      <form id ="monform" action="upload_img2.php" method="post" enctype="multipart/form-data">
10       <input style="height:100px;" type="file" name="img">
11       <input style="height:100px;" type="submit" name="upload" value="envoyer l'image">
12     </form>
13   </body>
14 </html>
```

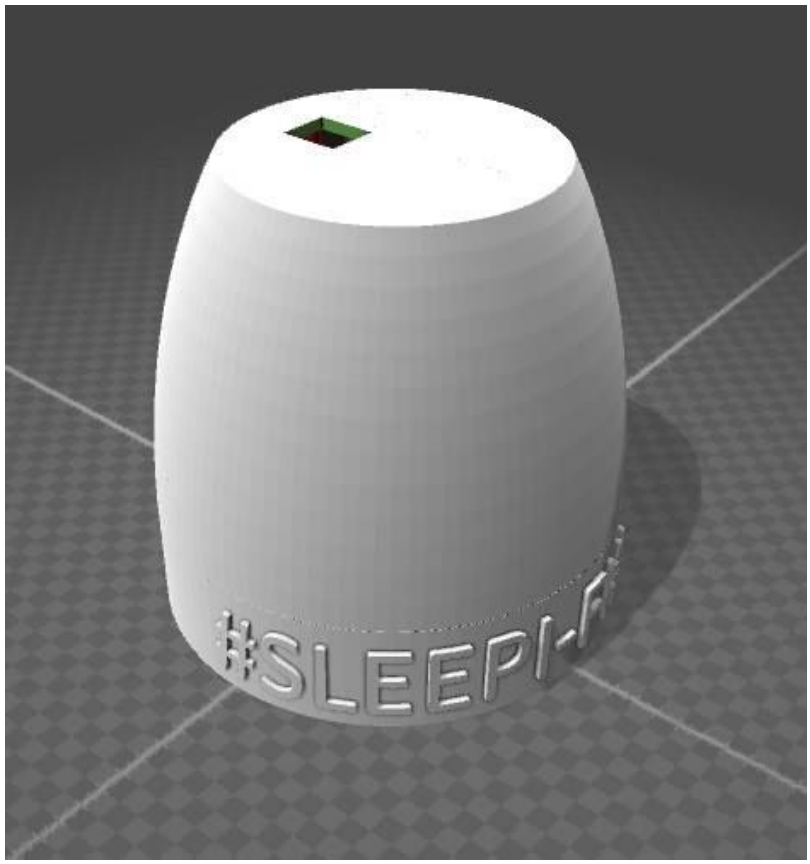
exemple de formulaire d'upload de fichier

```php
<?php

if( isset($_POST['upload']) ) // si formulaire soumis
{

    error_reporting(E_ALL);

    $content_dir = 'uploads/'; // dossier où sera déplacé le fichier

    if(!is_writable($content_dir))
        die('Impossible d\'écrire dans le répertoire cible.');

    $tmp_file = $_FILES['img']['tmp_name'];

    if( !is_uploaded_file($tmp_file) )
    {
        exit("Le fichier est introuvable");
    }

    // on vérifie maintenant l'extension
    $type_file = $_FILES['img']['type'];

    if( !strstr($type_file, 'jpg') && !strstr($type_file, 'jpeg') && !strstr($type_file, 'bmp') && !strstr($type_file, 'gif') )
    {
        exit("Le fichier n'est pas une image");
    }

    // on copie le fichier dans le dossier de destination
    $name_file = $_FILES['img']['name'];

    if( !move_uploaded_file($tmp_file, $content_dir . $name_file) )
    {
        exit("Impossible de copier le fichier dans $content_dir");
    }

    echo "Le fichier a bien été upload&eacute;";
    echo $content_dir.$name_file;
    $output = shell_exec('sudo fim -a '.escapeshellarg($content_dir.$name_file));
    echo "<pre>$output</pre>";
}
```

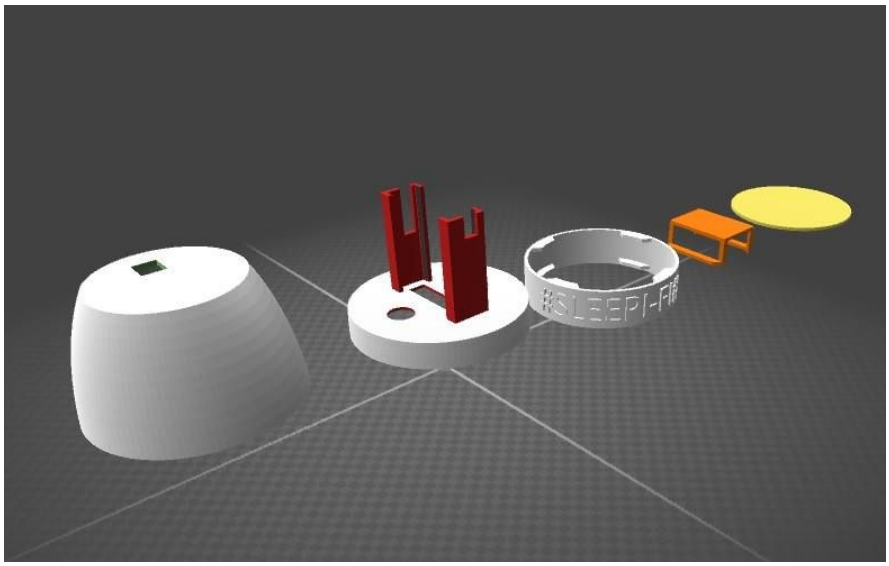
Exemple de script php qui traite l'upload d'une image (analogue pour l'upload d'une musique)

```
[grab]
device = /dev/video0
text = "Sleepi-Fi cam"
width = 640
height = 480
quality = 50
trigger = 0
delay = 0.5
[ftp]
dir = /var/www/videos
file = webcam.jpg
local = 1
```

Fichier webcam.conf



première modélisation de la veilleuse



Vue éclatée des pièces que nous avions prévues d'imprimer

Structure finalement retenue