CAHIER DES CHARGES

Description du projet HODOR

Groupe GUIOT – MEZARD

Table des matières

[1 – Description générale du projet 2](#_Toc34982455)

[2 – Décomposition des fonctionnalités 2](#_Toc34982456)

[3 – Schéma d’architecture 2](#_Toc34982457)

[A/ Logiciel 2](#_Toc34982458)

[B/ Matériel 2](#_Toc34982459)

[4 – Scénario/acteurs/rôles 2](#_Toc34982460)

# 1 – Description générale du projet

Le projet consiste en l’ouverture d’une porte sans clé. Cela consiste en l’utilisation d’un jeu de carte/badge et d’un lecteur RFID ainsi que la fonctionnalité d’utiliser une application mobile permettant à l’utilisateur de s’identifier.

# 2 – Décomposition des fonctionnalités

On peut identifier plusieurs fonctionnalités :

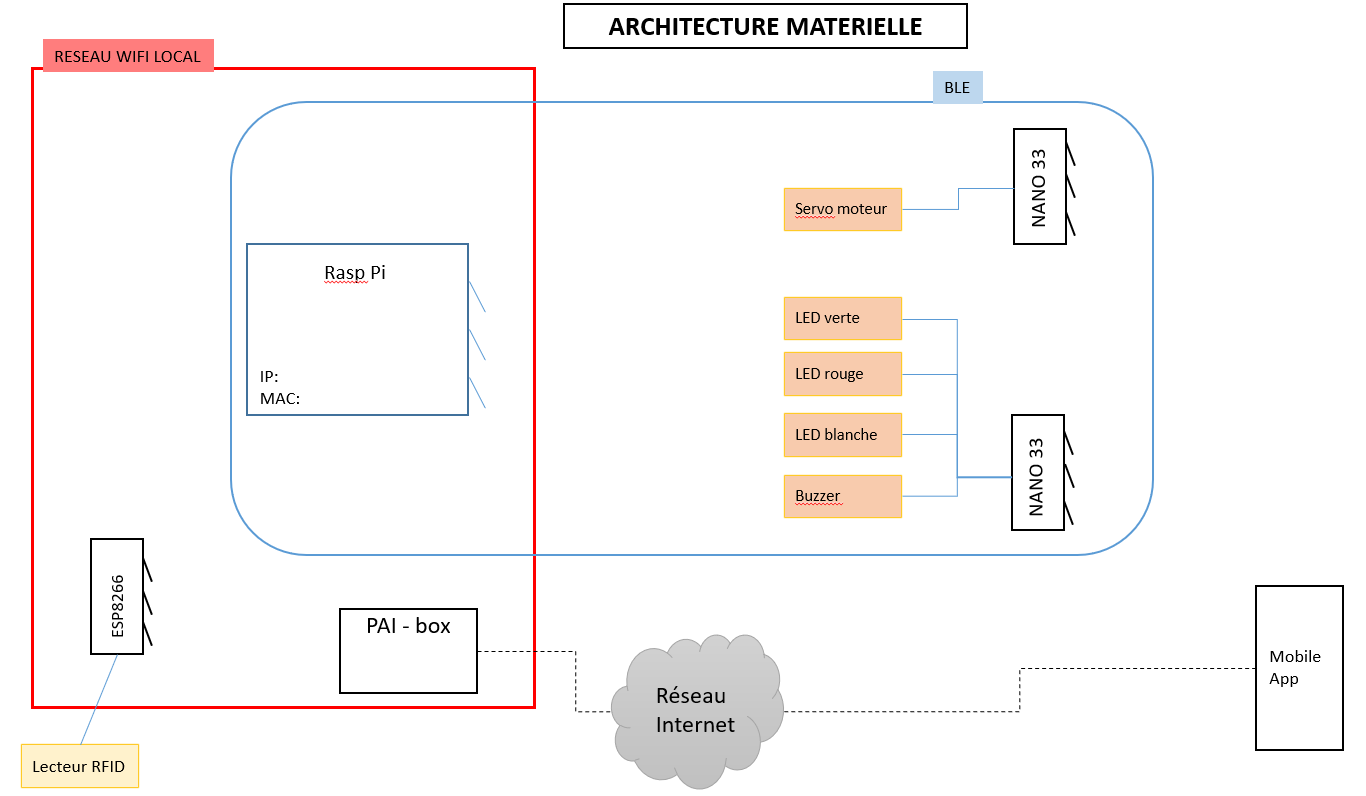
* F1 : Ouverture au contact via du RFID,
* F2 : Ouverture au contact soit avec les paramètres @mac
* F3 : Ouverture via connexion sur serveur avec authentification
* F4 : Exécution de F3 via internet en https
* F5 : Définition de plages d’ouverture pour un objet connecté donné (optionnel)
  + F5.1 : Droit d’accès sur plage horaire (ouverture via RFID ou login-mot de passe via l’application web)
* F6 : Faire de logs de l’activité de la porte (ouverture/fermeture, autorisations …)

On peut identifier des scénarios

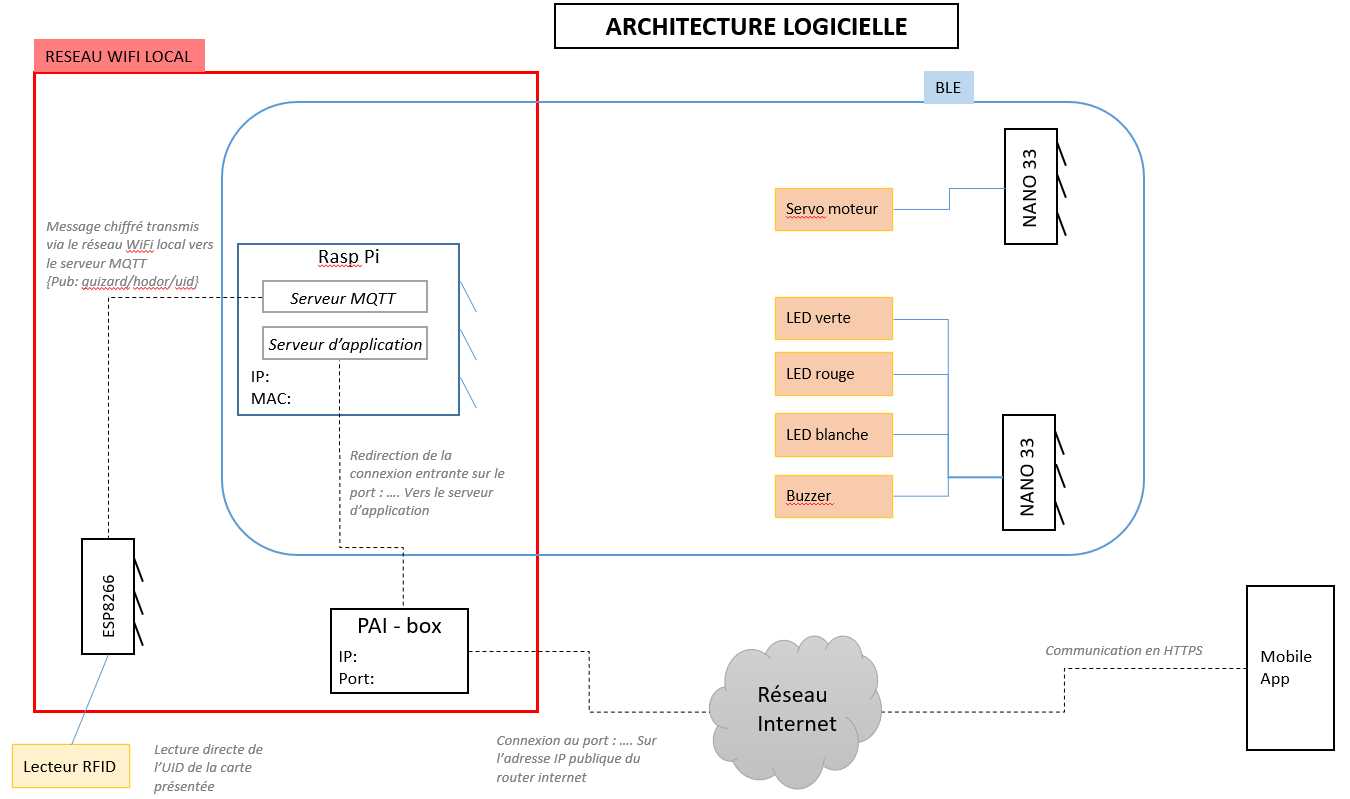
* Propriétaire/Invité utilise la fonctionnalité F1 :
* Propriétaire/ Invité utilise F2
* Propriétaire utilise F3
* Propriétaire utilise F4
* S’identifie via le lecteur RFID et doit donc posséder une carte ou un badge qui lui permet d’ouvrir ou non la porte d’accès.
* F2, si l’utilisateur ne possède pas de carte ni de badge mais a quand même l’autorisation d’activer l’ouverture de la porte après authentification via une application mobile qui le permet.

# 3 – Schéma d’architecture

## A/ Matériel



## B/ Logiciel



# 4 – Scénario/acteurs/rôles

Premier cas – Identification via RFID

L’utilisateur présente sa carte/badge au lecteur RFID.

L’ESP8266 transmet via le réseau WiFi l’UID de la carte au serveur MQTT (*Publish*)

Un script python permet de récupérer et traiter l’UID (*Subscribe*)

Après traitement, le Rasp Pi va transmettre en BLE les ordres aux actionneurs.

En cas de succès :

* Ouverture de la porte
* Buzzer
* Led Verte

En cas d’échec :

* Buzzer
* Led rouge

Second cas – Identification via mobile

*(A définir plus en détail en fonction d’un accès via le réseau WiFi local ou non)*

Troisième cas – Identification via mobile depuis internet avec accès sécurisé en https

Prérequis : client dispose d’un certificat valide sur le portable

Le client se connecte en https

Il peut ouvrir ou fermer la porte

Quatrième cas – Donner des accès avec des plages horaires (étant donné un id de téléphone (@mac ou @BLE) un locataire a accès à l’ouverture selon es plages horaires définies préalablement

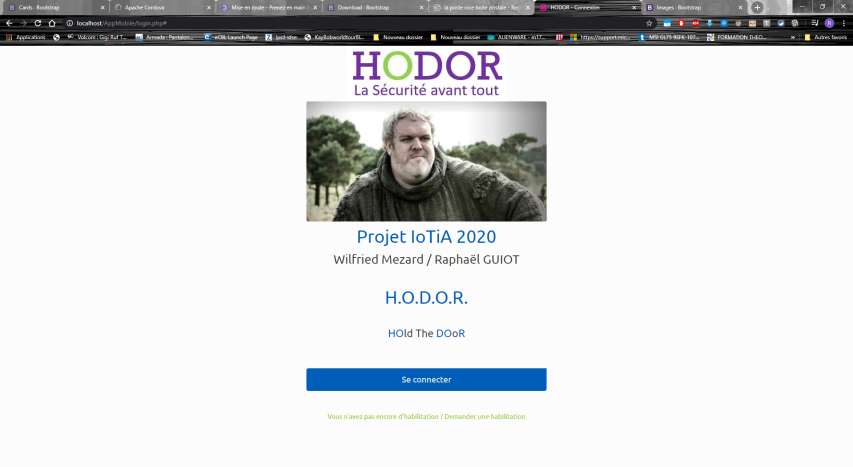
*NOTA : En termes de sécurité, l’adresse MAC étant falsifiable, peut-on considérer ce cas d’utilisation viable ?*

# 5 Qui fait quoi & Calendrier

Cette semaine 26/03/2020:

Raphael : appli mobile avec deux pages :

* Jeedom ? et regarder l’accès en https
* Développement application mobile pour authentification



* Redirection de ports de la Livebox au Raspberry Pi pour mettre le Rasp à disposition sur le web
* Ouverture de la porte via un panel d’accès (page de control de l’ouverture)
* Mettre en place la fonctionnalité de définition de plages horaires selon les différents cas :
  + Droit d’accès à un identifiant RFID
  + Droit d’accès via login-Mot de passe
  + Droit d’accès selon reconnaissance en BLE

Wilfrid : partie entre ESP et Raspberry Pi.

* Encryption sur l’ESP de la date et l’id du RFID (Utilisation de de la librairie ArduinoJson pour la récupération d’un datetime)
* Transmission vers le Raspberry Pi en WIFI (La publication via le broker MQTT est faite)
* Transmission BLE entre RPI et actionneurs en BLE – (je n’ai pas en ma possession de Raspberry)

Point la semaine prochaine sur ces deux aspects

Semaine du 08/04/2020

* Mise en place de l’utilisation d’une API externe livrant la date courante
* Transmission des ID lus via RFID en WiFi
* Communication BLE entre RPI et Arduino Nano
  + Transmission d’ordres en clair (pas de chiffrage de l’échange)
  + Réflexion sur le chiffrement de l’échange en BLE entre le RPI et les Arduino Nano

Recensement des services et caractéristiques BLE - <https://www.bluetooth.com/specifications/gatt/services/>

Je pense utiliser le service « Alert Notification » (Assigned number : 0x1811) - <https://www.bluetooth.com/wp-content/uploads/Sitecore-Media-Library/Gatt/Xml/Services/org.bluetooth.service.alert_notification.xml>

Ce service va permettre d’exposer une information à un device, en l’occurrence notre Arduino Nano BLE. L’arbre XML proposé par la spécification présente trois types d’information échangée :

* Le type d’alerte et le message texte court associé
* Le décompte des nouvelles alertes
* Le décompte des alertes non lues

On dénombre également cinq caractéristiques

* « Supported New Alert Category », qui expose les catégories d’alerte supportées par le serveur BLE - Obligatoire
* « New Alert », qui expose le décompte de nouvelles alertes pour une catégorie donnée - Obligatoire
* « Supported Unread Alert Category », qui expose les catégories d’alerte non lues qui sont supportées par le serveur - Obligatoire
* « Unread Alert Status », qui expose le décompte de nouvelles alertes non lues existantes pour le serveur - Obligatoire
* « Alert Notification Control Point », permet au device connecté d’activer/désactiver les alertes le notifiant d’évènement.