DOCUMENTATION IOT

Par BRANCOURT Dimitri, GENDRON Adrien, GAMBEY Noa, LAMINE Valentin

APRIL 18, 2024
YNOV CAMPUS
Aix-En-Provence

Table of Contents

Préambule :

Introduction :

Bienvenue sur la documentation pour le projet d'IOT. Nous avons choisi de réaliser le projet suivant : YnovContactTracer.

Ce projet a pour but de permettre dans un contexte d'épidémie/pandémie de suivre les différentes rencontres que nous avons faites, de les sauvegarder en tant que « contacts » et de suivre leur état de santé afin de déterminer (si ces derniers s'avèrent positif) si nous sommes des cas contacts.

Ce projet est entièrement basé sur la communication entre deux cartes Arduino ESP32 couplé à un système centralisé pour permettre la bonne communication et l'intégrité des données.

Prérequis :

Pour utiliser notre code il est nécessaire de disposer de :

- IDE Arduino (veiller à télécharger les différentes librairies)
- ESP32

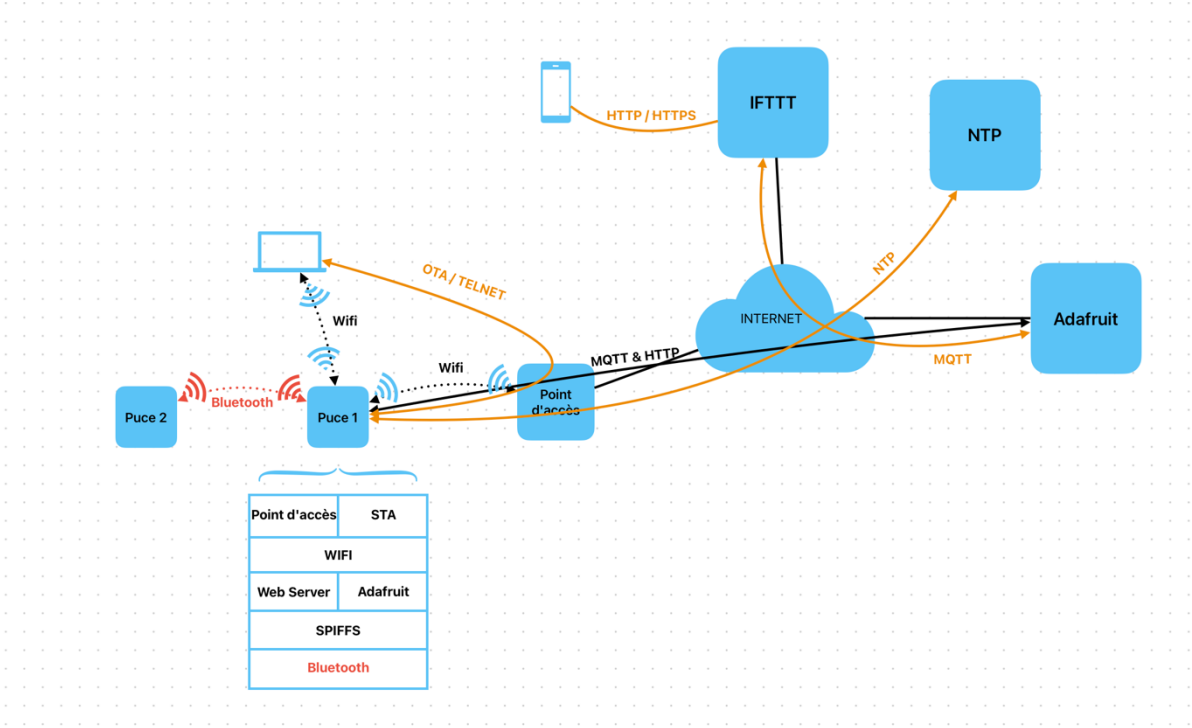
Sommaire :

Notre documentation contiendra les éléments suivants :

- Notre projet expliquer de manière fonctionnelle avec des schémas simple pour accompagner nos explications
- L'architecture du projet avec également un schéma
- La liste des fonctionnalités développées et démontrer

Documentation :

Architectures des communications :



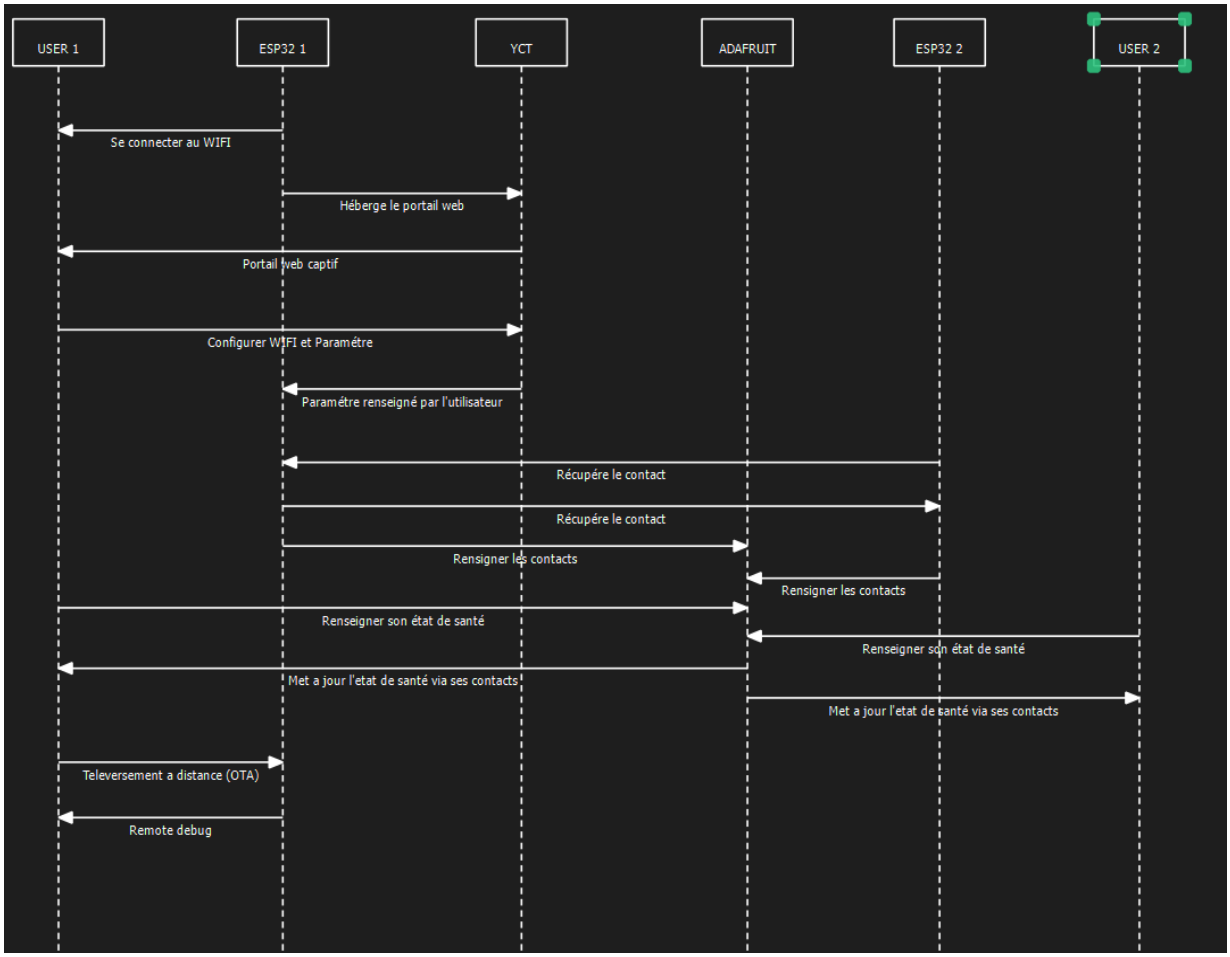
Le diagramme ci-dessus représente l'architectures des différents types de communication et protocoles pour le projet YCT. Il montre un exemple de communication typiques entre deux cartes et internet.

Tout d'abord une carte peu importe laquelle démarre en Access Point afin d'être configurée, si elle est déjà configurée elle se connecte au réseau Wifi.

Avec un ordinateur on peut choisir de se connecter directement à la carte, ou via réseau wifi pour pouvoir accéder au web server. Le but étant de pouvoir trouver les cartes à proximité, on va utiliser le moyen le plus évident, le Bluetooth. Malheureusement s'il y a trop d'appareils aux alentours, la carte va crasher lors d'un scan Bluetooth, c'est la raison pour laquelle cette feature est en rouge sur le schéma car elle comporte des bugs.

La carte a besoin du Wifi pour trois raisons. La première raison est de communiquer avec un serveur ntp pour récupérer l'heure, utile afin de gérer les logs. La deuxième raison est la centralisation des données sur Adafruit notamment sur la donnée des personnes se déclarant positives. Et enfin la dernière raison est de relier le téléphone à la carte avec IFTT.

Diagramme de séquence & exemple de cas d'usage :



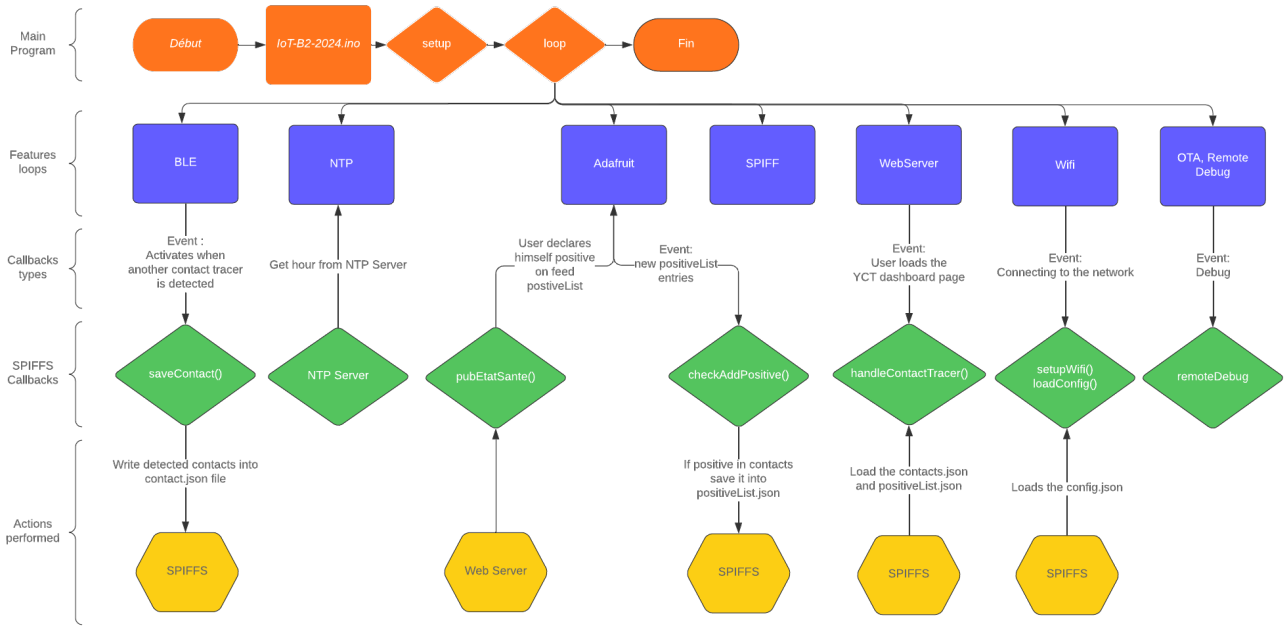
Le diagramme de séquence ci-dessus représente un cas d'usage de YnovContactTracer, c'est à dire l'interaction entre deux cartes Arduino ESP32, leur BDD locale et des feeds Adafruit MQTT afin de centraliser et de transmettre certaines informations.

Tout d'abord, les deux cartes vont se connecter au Wifi et vont lancer l'hébergement du serveur Web. Ce serveur Web permettra aux utilisateurs respectifs de configurer les paramètres de la carte ainsi que la configuration du Wifi (notamment pour changer de connexion Wifi).

Les deux ESP32 seront aussi capables de se repérer et de s'ajouter via Adafruit. Grâce à la possibilité de renseigner son état sur Adafruit, cela permettra de vérifier entre les personnes positives et sa propre liste de contacts, les redondances afin de déterminer si on doit devenir cas contact.

D'un point de vue plus administrateur il est également possible de téléverser le code à distance et de récupérer des logs à distance vie respectivement l'OTA et le Remote Debug.

Architecture de projet :



Ce schéma explique la structure de nos fichiers et de notre code. La première ligne en orange représente le fichier Arduino principal avec l'initialisation du setup et la loop de base.

La seconde ligne sont les fichier Headers de chaque technologie. La troisième ligne montre nos fonctions et la dernière est le résultat de nos fonctions.

Le code YTC a pour but de s'appuyer sur de l'événementiel, c'est à dire on veut le moins possible se reposer sur les boucles et ainsi gagner en optimisation. Pour ce faire notre rôle dans ce projet est donc d'ajouter nos propres fonctions et de les appeler aux endroits les plus pertinents.

Technologies et Fonctionnalités :

Liste des technologies :

- WiFi - 3 points
- MQTT / Adafruit - 5 points
- Serveur Web - 3 points
- Système de fichier + JSON - 3 points
- Network Transport Protocol - 2 points
- Bluetooth - 3 points
- Interaction type IFTTT - 3 points
- Over The Air - 4 points
- Remote Debug - 4 points

Liste des Fonctionnalités :

- Charger et enregistrer une configuration
- Ajout de contacts et des contacts positifs
- Formatage de la carte et de la BDD interne (3 SPIFFS JSON)
- Suppression automatique des contacts au bout de 30 jours
- Nettoyage de la BDD automatique à chaque ajout de contacts
- Utilisation sans fil de la carte possible via OTA et Remote Debug
- Paramétrage du point d'accès et du Wifi
- Possibilité de publier son état de santé, les autres contacts récupère cet état
- Calcul de l'état de santé (négatif, cas contact, positif) en fonction de la BDD locale