

# Rapport de projet :

## Préambule

### Le projet

Faire un digicode à 2 input, communiquant à distance.

Le code d'accès sera modifiable à distance via l'application. Plusieurs codes d'accès seront disponible.

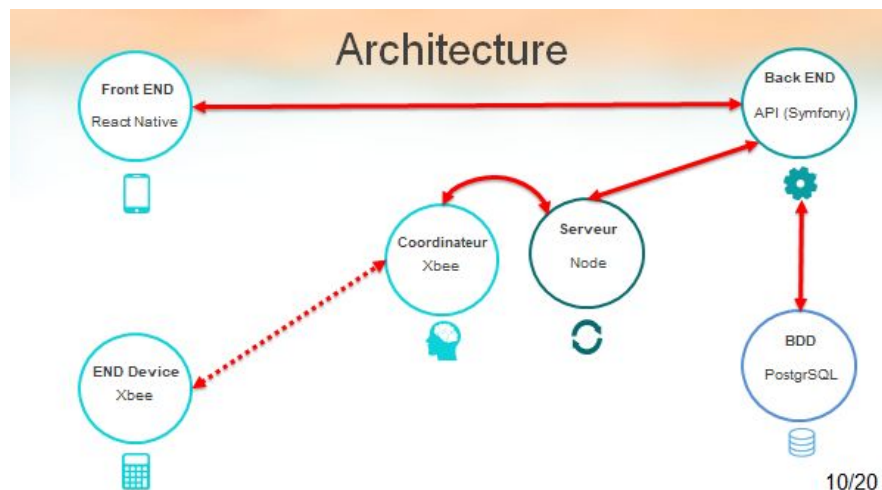
### Contexte d'utilisation

- Porte d'entrée
- Escape Game
- Code éphémères pour un livreur de colis
- Air BnB

### Pré-requis

- Logiciel XCTU
- Ordinateur (sous Windows or Mac)
- PhpStorm (+ PHP & Node.JS)
- Docker (+ docker-compose cli)
- NPM
- YARN
- Xbee (+ différent matériel électronique: LED, boutons poussoirs, fils, breadboard ..)

### Architecture



# 1 - Conception et installation du matériel

La conception de ce projet ai fais en 2 parties, une partie qui permet de recevoir les informations et un autre permettant de les envoyer.

## Partie récepteur

Commandes à faire :

AT ID 8888

AT CE 0

▼ Networking

Change networking settings

|                                   |              |            |  |
|-----------------------------------|--------------|------------|--|
| i ID PAN ID                       | 8888         |            |  |
| i SC Scan Channels                | 7FFF         | Bitfield   |  |
| i SD Scan Duration                | 3            | exponent   |  |
| i ZS ZigBee Stack Profile         | 0            |            |  |
| i NJ Node Join Time               | FF           | x 1 sec    |  |
| i NW Network Watchdog Timeout     | 0            | x 1 minute |  |
| i JV Channel Verification         | Disabled [0] |            |  |
| i JN Join Notification            | Disabled [0] |            |  |
| i OP Operating PAN ID             | 7777         |            |  |
| i OI Operating 16-bit PAN ID      | D00D         |            |  |
| i CH Operating Channel            | 12           |            |  |
| i NC Number of Remaining Children | 14           |            |  |
| i CE Coordinator Enable           | Disabled [0] |            |  |
| i DO Device Options               | 0            | Bitfield   |  |
| i DC Device Controls              | 0            | Bitfield   |  |

AT DH 0

AT DL 0

▼ Addressing

Change addressing settings

|   |          |          |
|---|----------|----------|
| i SH Serial Number High                   | 13A200   |          |
| i SL Serial Number Low                    | 41A7133C |          |
| i MY 16-bit Network Address               | C3C2     |          |
| i MP 16-bit Parent Address                | FFFE     |          |
| i DH Destination Address High             | 0        |          |
| i DL Destination Address Low              | 0        |          |
| i NI Node Identifier                      | Titi     |          |
| i NH Maximum Hops                         | 1E       |          |
| i BH Broadcast Radius                     | 0        |          |
| i AR Many-to-One Route Broadcast Time     | FF       | x 10 sec |
| i DD Device Type Identifier               | A0000    |          |
| i NT Node Discovery Backoff               | 3C       | x 100 ms |
| i NO Node Discovery Options               | 0        |          |
| i NP Maximum Number of Transmission Bytes | 54       |          |
| i CR PAN Conflict Threshold               | 3        |          |

## AT AP 0

### Serial Interfacing

Change modem interfacing options

|  |                       |  |
|--|-----------------------|--|
| <b>BD</b> Baud Rate                        | 9600 [3]              |  |
| <b>NB</b> Parity                           | No Parity [0]         |  |
| <b>SB</b> Stop Bits                        | One stop bit [0]      |  |
| <b>RO</b> Packetization Timeout            | 3 x character times   |  |
| <b>D6</b> Pin 16 - DIO6/nRTS Configuration | Disable [0]           |  |
| <b>D7</b> Pin 12 - DIO7/nCTS Configuration | nCTS flow control [1] |  |
| <b>AP</b> API Enable                       | Transparent mode [0]  |  |
| <b>AO</b> API Output Mode                  | Native [0]            |  |

## AT D0 5

## AT D1 5

## AT D2 5

## AT D3 5

### I/O Settings

Modify DIO and ADC options

|   |                            |  |
|---|----------------------------|--|
| <b>D0</b> Pin 20 - DIO0/AD0/CB Configuration        | Digital Out, High [5]      |  |
| <b>D1</b> Pin 19 - DIO1/AD1/nSPI_ATTN Configuration | Digital Out, High [5]      |  |
| <b>D2</b> Pin 18 - DIO2/AD2/SPL_SCLK Configuration  | Digital Out, High [5]      |  |
| <b>D3</b> Pin 17 - DIO3/AD3/nSPI_SSEL Configuration | Digital Out, High [5]      |  |
| <b>D4</b> Pin 11 - DIO4/SPI_MOSI Configuration      | Disabled [0]               |  |
| <b>D5</b> Pin 15 - DIO5/Associated Configuration    | Associated indicator [1]   |  |
| <b>D8</b> Pin 9 - DIO8/nDTR/Sleep_Rq Configuration  | Sleep_Rq [1]               |  |
| <b>D9</b> Pin 13 - DIO9/nOn_Sleep Configuration     | Awake/Asleep indicator [1] |  |
| <b>P0</b> Pin 6 - DIO10/RSSI PWM/PWM0 Configuration | RSSI PWM Output [1]        |  |
| <b>P1</b> Pin 7 - DIO11/PWM1 Configuration          | Disabled [0]               |  |
| <b>P2</b> Pin 4 - DIO12/SPI_MISO Configuration      | Disabled [0]               |  |
| <b>P3</b> Pin 2 - DIO13/DOUT Configuration          | DOUT [1]                   |  |
| <b>P4</b> Pin 3 - DIO14/DIN/nConfig Configuration   | DIN [1]                    |  |
| <b>PR</b> Pull-up Resistor Enable                   | 1FBP                       |  |
| <b>PD</b> Pull-up/down Direction                    | 1FFF                       |  |
| <b>LT</b> Associate LED Blink Time                  | 0 x10 ms                   |  |
| <b>RP</b> RSSI PWM Timer                            | 28 x 100 ms                |  |

## AT IR 0

## AT IC FFFF

### I/O Sampling

Configure IO sampling parameters

|   |          |  |
|---|----------|--|
| <b>IR</b> IO Sampling Rate              | 0 x 1 ms |  |
| <b>IC</b> Digital IO Change Detection   | FFFF     |  |
| <b>V+</b> Supply Voltage High Threshold | 0        |  |

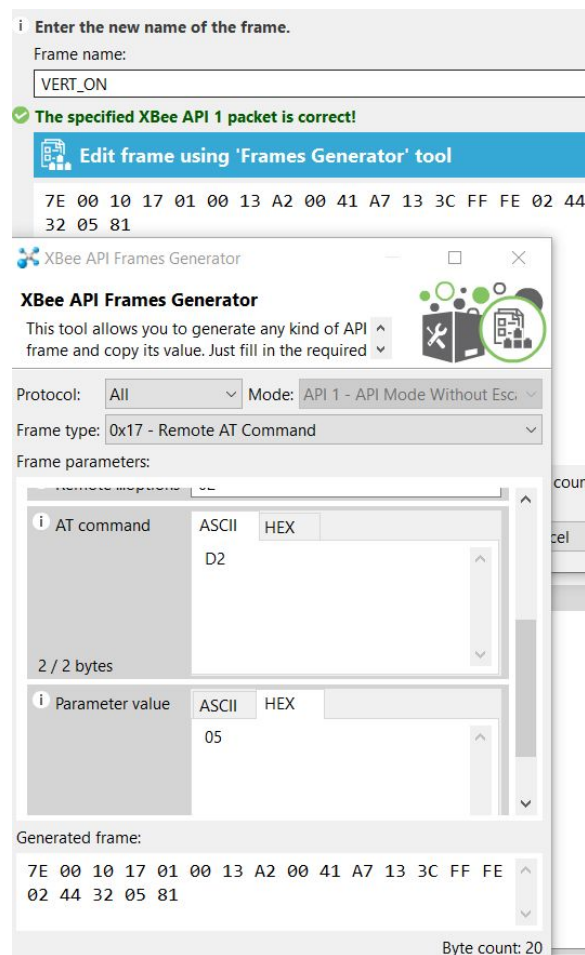
## Partie Coordinateur

Commandes à faire :

AT ID 8888  
AT CE 1  
AT DH 13A200  
AT DL 41A7133C  
AT AP 1

Frames envoyés par le Coordinateurs depuis XCTU :

Allumer Led Vert :



Pour éteindre je passe en parametre 00 (disable)

Je peux créer en suivant cette logique une séquence, dans l'exemple ci-dessous, Je fait clignoter la led vert puis la led bleu autant de fois que je le souhaite (je peux aussi paramétrer la vitesse d'exécution de la séquence dans le champ "Transmit interval").

Ici la liste des frames que j'utilise

Send frames

| Name      | Type                      |
|-----------|---------------------------|
| VERT_OFF  | Remote AT Command Request |
| VERT_ON   | Remote AT Command Request |
| ROUGE_OFF | Remote AT Command Request |
| ROUGE_ON  | Remote AT Command Request |

CloseRecordDetach

CTS | CD | DSR

DTR RTS BRK

Tx frames: 10761  
Rx frames: 13810

Frames log

| ID    | Time         | Length | Frame                             |
|-------|--------------|--------|-----------------------------------|
| 24560 | 15:20:57.441 | 98     | Explicit RX Indicator             |
| 24561 | 15:20:57.490 | 7      | Transmit Status                   |
| 24562 | 15:20:57.490 | 22     | Explicit Addressing Command Frame |
| 24563 | 15:20:57.696 | 98     | Explicit RX Indicator             |
| 24564 | 15:20:57.736 | 7      | Transmit Status                   |
| 24565 | 15:20:57.736 | 22     | Explicit Addressing Command Frame |
| 24566 | 15:20:57.963 | 48     | Explicit RX Indicator             |
| 24567 | 15:20:58.000 | 7      | Transmit Status                   |
| 24568 | 15:20:58.050 | 22     | Explicit Addressing Command Frame |
| 24569 | 15:20:58.227 | 45     | Explicit RX Indicator             |
| 24570 | 15:20:58.281 | 7      | Transmit Status                   |

Frame details

Transmit Status

(API 1)

7E 00 07 88 06 00 00 23 00 4B

Start delimiter

7E

Length

00 07 (7)

Frame type

88 (Transmit Status)

Frame ID

06 (6)

Send frames

| Name     | Type                      |
|----------|---------------------------|
| VERT_OFF | Remote AT Command Request |
| VERT_ON  | Remote AT Command Request |
| BLEU_OFF | Remote AT Command Request |
| BLEU_ON  | Remote AT Command Request |

Send a single frame

Send selected frame

Send sequence

Transmit interval (ms): 100

Repeat times: 1

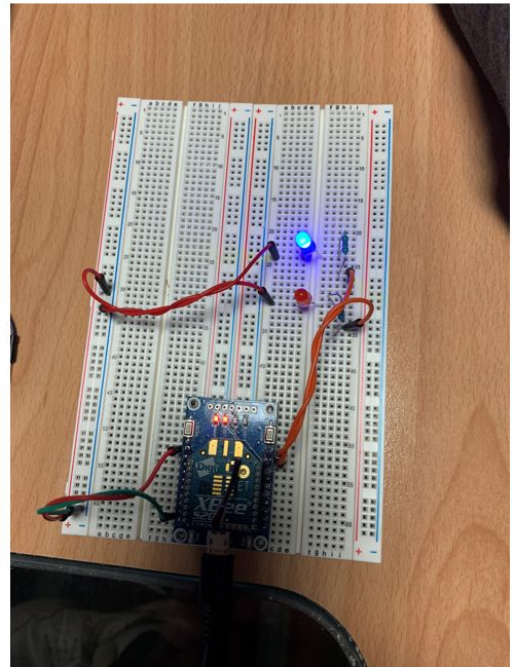
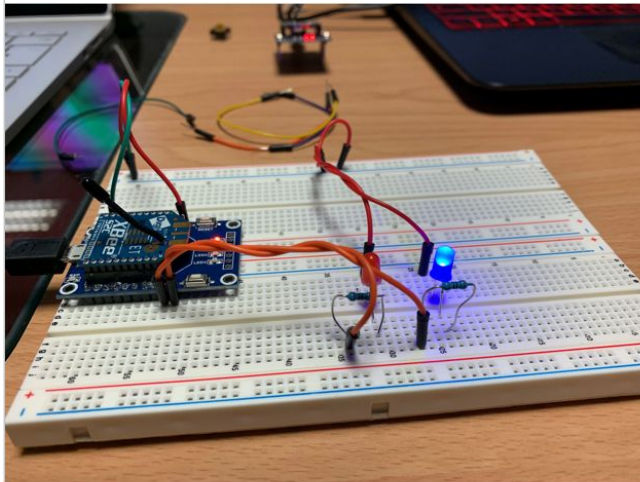
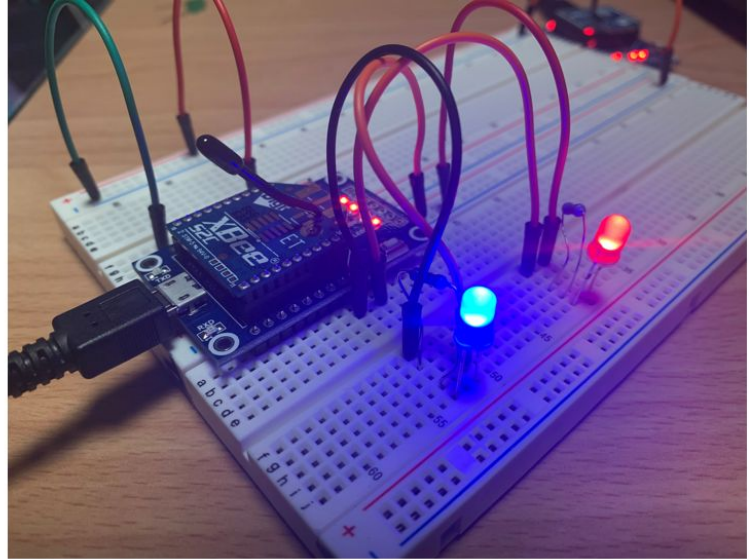
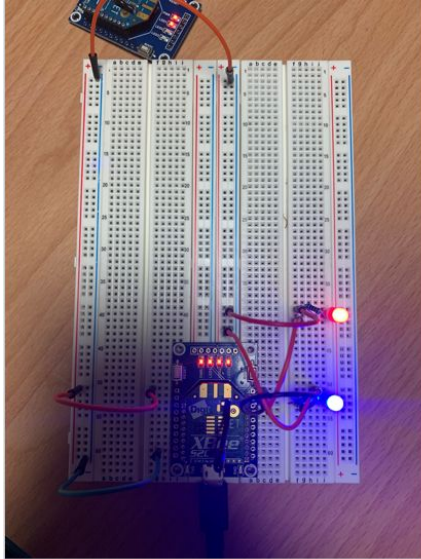
Loop infinitely

Start sequence

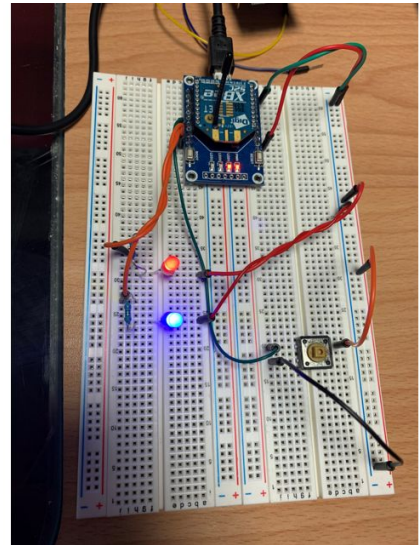
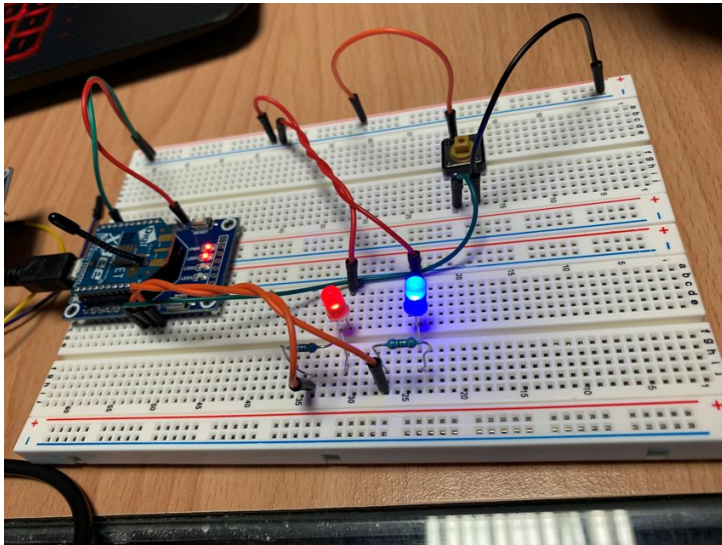


## 2 - Evolution du Montage

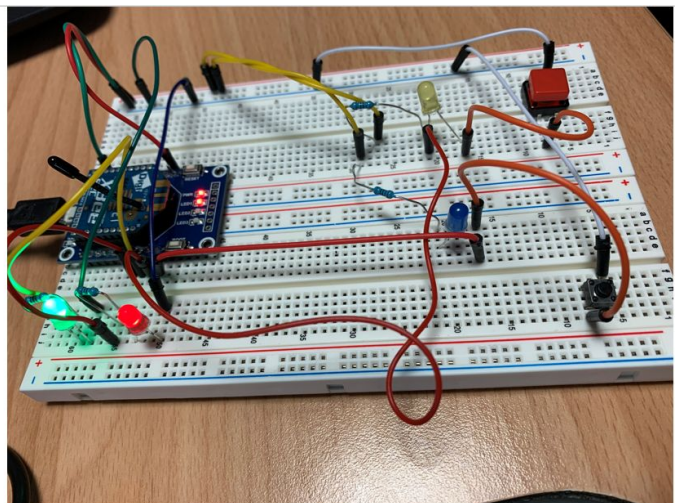
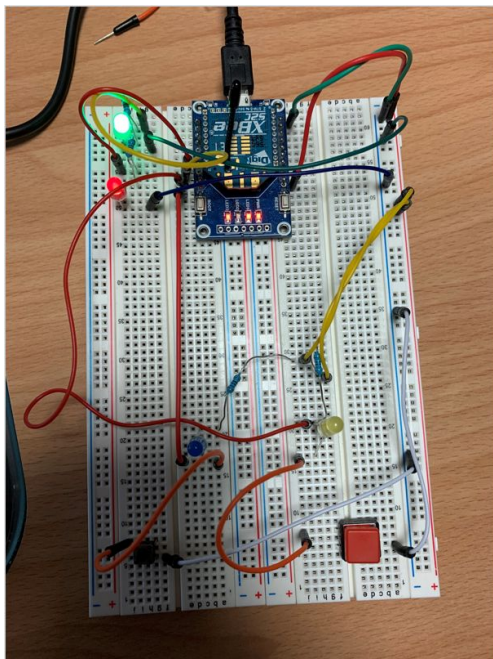
Montage sans bouton



Montage avec bouton

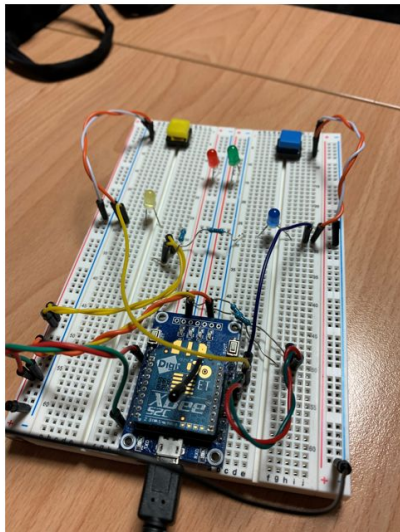
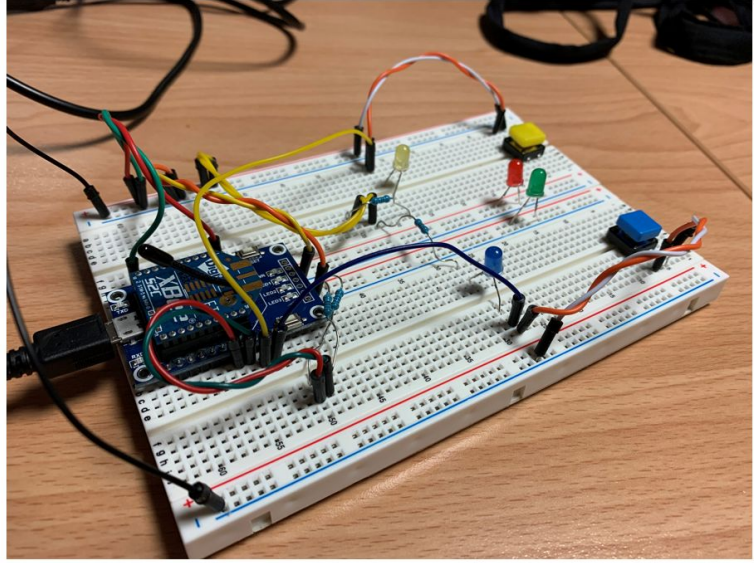
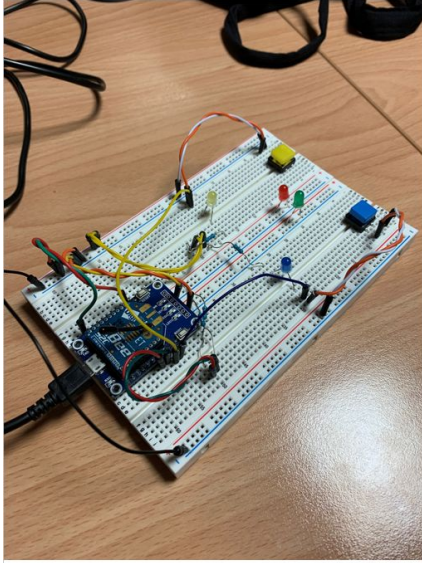


Montage 2 boutons & 4 leds



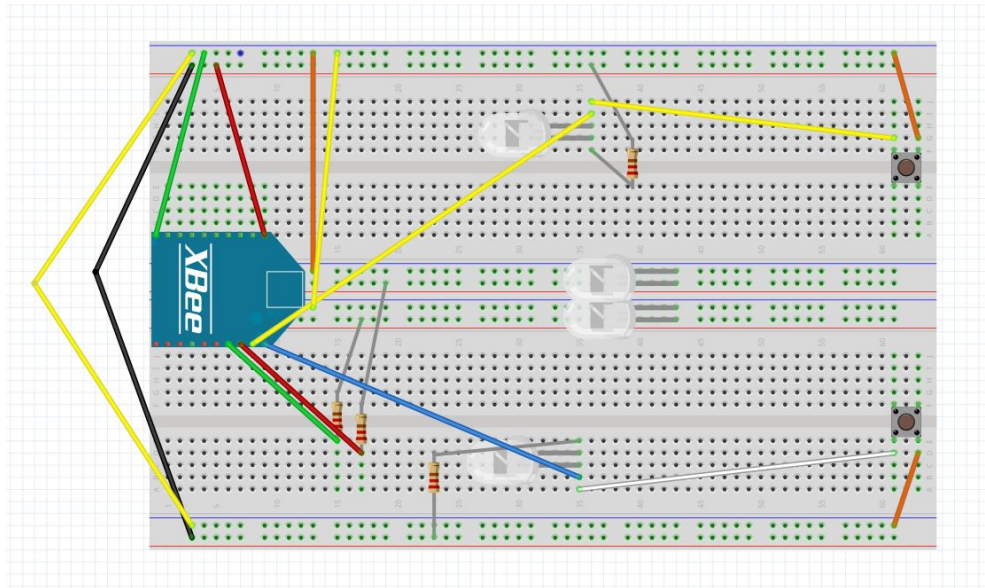


## Montage final optimisé





## Schéma



## 3 - Fonctionnement en détail

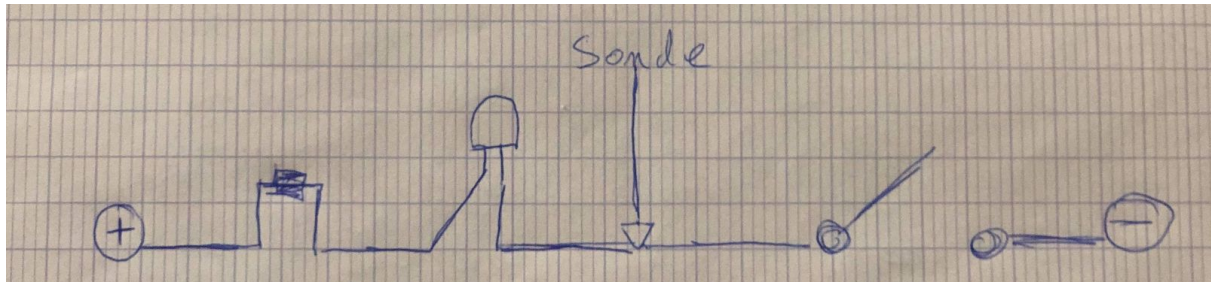
Pour envoyer les informations au Coordinateur :

| I/O Sampling                       |      |
|------------------------------------|------|
| Configure IO sampling parameters   |      |
| i IR IO Sampling Rate              | 0    |
| i IC Digital IO Change Detection   | FFFF |
| i V+ Supply Voltage High Threshold | 0    |

IC FFFF veut dire que à chaque détection de changement sur une parmi toutes les PIN du Xbee on vas envoyer notre Sample au coordinateur.

Dans notre cas à chaque fois qu'on appuie sur un bouton une trame est envoyé mais aussi lorsqu'on le relâche, il a fallu donc côté code trier l'information reçu en doublon.

## Schéma simplifié du fonctionnement du bouton :



Le bouton agit comme un coup circuit dans notre maquette, tant qu'il n'est pas actionné le circuit n'est pas connecté à la masse, de ce fait la led reste off. Cependant à partir du moment où l'on presse le bouton, le circuit est relié à la masse et donc la led s'allume, la sonde détecte donc un changement dans notre circuit et donc grâce à la configuration vu précédemment le xbee envoie un Sample contenant l'état de tous nos PIN.

## **4 - Développement serveur Node.JS**

### Documentation

Première étape afin de bien prendre en main le "squelette" de code fourni dans le projet fourni par l'enseignant, c'est la documentation sur le Xbee-api, la librairie utilisée par la partie "/socket" du projet.

### Tests

Ensuite, nous avons testé de voir ce que donne le code en lisant les frames reçues sur le port série branché au Xbee "coordinateur"

## Explications de quelques lignes de code :

Ces lignes de code permettent de récupérer les mots de passe qui sont enregistré en base de données et de les stocker dans une variable.

Elles permettent également de vérifier si le bouton pressé est **bleu** ou **jaune** et tri l'information afin de savoir si le bouton est pressé ou relâché.

Nous faisons une boucle afin de vérifier si la combinaison entrée actuellement correspond à un des mots de passe enregistré en base.

Enfin si la combinaison possède une correspondance, une led **verte** s'allume pendant 3000 ms puis s'éteint afin de laisser la led **rouge** s'allumer.

```
89 xbeeAPI.parser.on("data", function (frame) {
90   //on new device is joined, register it
91   RecoverPassword(function(data){
92     passwords=data;
93     console.log(passwords);
94   });
95   if(frame.digitalSamples !== undefined)
96   {
97     if(frame.digitalSamples['DIO0'] == 0){
98       combinationEnteredString += "B";
99     }
100    if(frame.digitalSamples['DIO1'] == 0){
101      combinationEnteredString += "J";
102    }
103    for(let i = 0; i < passwords.length; i++){
104      let password = passwords[i];
105      console.log(password);
106
107      if(!truePass && combinationEnteredString.match(password)){
108        truePass = true;
109
110        xbeeAPI.builder.write(enableGreenLight);
111        xbeeAPI.builder.write(disableRedLight);
112
113        console.log(combinationEnteredString + " : Combinaison correct !");
114
115        setTimeout(function(){
116          xbeeAPI.builder.write(disableGreenLight);
117          xbeeAPI.builder.write(enableRedLight);
118          combinationEnteredString = "";
119          truePass = false;
120        },3000);
121      }
122      else{
123        console.log(combinationEnteredString);
124      }
125    }
126    // Si la combinaison entré est la bonne, on allume la led verte pendant 10sec
127  }
```

Dans cette partie, la fonction est focalisée afin de faire un appel en base pour récupérer les mots de passe qui y sont stockés.

Les informations récupérées sont alors transcrites en JSON puis un parcours de cette transcription est faites pour récupérer les informations contenues dans **“hydra:member”**.

```

176 function RecoverPassword(callback){
177     var options = {
178         url: 'https://localhost:8443/access_passes',
179         strictSSL: false,
180         secureProtocol: 'TLSv1_method'
181     }
182
183     request.get(options, function(error, response, body) {
184         if (!error) {
185             let jsonData = JSON.parse(body);
186             let passwordArray = [];
187             for(let i = 0; i < jsonData["hydra:member"].length; i++){
188                 passwordArray.push(jsonData["hydra:member"][i]["pass"]);
189             }
190             callback(passwordArray);
191         }
192         else {
193             console.log(error);
194         }
195     });
196 }
197

```

## **5 - Développement React Native**

Grâce au composant “Client Generator React Native” proposé par Api-platform (et oui, il permet beaucoup de choses..), nous avons pu générer toute une appli mobile permettant de communiquer avec les différents endpoints de l’API.

En effet, l’application mobile est générée depuis le lien de la documentation de l’API qui représente toutes les routes/endpoints exposée par cette dernière.

Pour notre cas, nous avons générer l’application



## API Symphony

### AccessPass

|        |                     |   |
|--------|---------------------|---|
| GET    | /access_passes      | Retrieves the collection of AccessPass resources. |
| POST   | /access_passes      | Creates a AccessPass resource.                    |
| GET    | /access_passes/{id} | Retrieves a AccessPass resource.                  |
| DELETE | /access_passes/{id} | Removes the AccessPass resource.                  |
| PUT    | /access_passes/{id} | Replaces the AccessPass resource.                 |
| PATCH  | /access_passes/{id} | Updates the AccessPass resource.                  |

### Device

|        |               |   |
|--------|---------------|---|
| GET    | /devices      | Retrieves the collection of Device resources. |
| POST   | /devices      | Creates a Device resource.                    |
| GET    | /devices/{id} | Retrieves a Device resource.                  |
| DELETE | /devices/{id} | Removes the Device resource.                  |
| PUT    | /devices/{id} | Replaces the Device resource.                 |
| PATCH  | /devices/{id} | Updates the Device resource.                  |

## 6 - Problèmes rencontrés

### Driver / Pilote :

Il se peut que votre driver/pilote de votre XBEE ne soit pas installé ou alors il peut posséder une erreur dans l'installation. Vous pourrez trouver le pilote Windows à cette adresse :

<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

### API SSL :

L'api REST ("générée" grâce à "api-platform") est accessible seulement et uniquement en HTTPS, avec un certificat "self-signed"/"auto-signé".

Cela a posé des problèmes à tous les niveaux puisque pour chaque client qui a eu à requêter l'API, il fallait augmenter la flexibilité au moment de la requête afin de forcer l'accès à l'endpoint.

Par exemple pour le serveur Node.JS, dans les options de la requête, il a fallu ajouter ce paramètre là:

```
strictSSL: false,
```

### Erreur 502 API :

Exécuter la ligne de commande suivante dans le projet :

```
docker-compose exec php php-fpm -D
```

### Erreur d'exécution du container docker :

Il faut changer les paramètres "**DB**" du fichier docker-compose.yml qui se trouve à la racine du projet.

```
db:
  image: postgres:12-alpine
  environment:
    - POSTGRES_DB=api
    - POSTGRES_PASSWORD=!ChangeMe!
    - POSTGRES_USER=api-platform
  volumes:
    - db-data:/var/lib/postgresql/data:rw
    # you may use a bind-mounted host
    # directory instead, so that it is harder to
    # accidentally remove the volume and lose all
    # your data!
    # -
    ./api/docker/db/data:/var/lib/postgresql/data:rw
  ports:
    - target: 5432
      published: 5432
      protocol: tcp
```