# Remerciement

Avant tout développement, nous profitons de ce rapport pour remercier ceux qui nous ont beaucoup appris et ceux qui ont eu la gentillesse de faire de cette expérience un moment profitable.

Nous tenons tout d’abord à remercier Monsieur Aomar OSMANI, qui nous a aidé et encadré tout au long du projet avec beaucoup de patience et de pédagogie. Ainsi que Monsieur Hamidi MASSINISSA, qui est rester disponible pour toute questions et qui a su nous conseiller sur nos différents choix ainsi que sur la méthodologie à prendre.

# Présentation général

## Présentation de notre application

Le but de notre produit est une immersion dans une expérience multimédia s'approchant de la technologie de la 4DX à moindre coût. Pour cela nous devons créer un set-up adaptable sur toutes les structures (télévisions, ordinateurs...). Le set-up est composé de la façon suivante :

* LED : Il s'agit d'une bande LED connectée à votre smartphone en bluetooth que vous pourrez contrôler à votre guise. Mais la particularité de cette bande est son mode immersif : Elle s'adaptera automatiquement aux couleurs prédominantes, que ce soit pour un jeu video, un film ou juste une image, l'expérience est garantie.

## Les enjeux

L’enjeu du projet qui nous a donné la motivation de le réaliser est l’aspect humain.

En effet, cette interface alimentera notre base de données, qui nous permettra de créer un traducteur maya espagnol afin de venir en aide aux personnes qui en ont besoin au Mexique.

# Besoin de notre projet

Dans notre cahier de charges on a défini les besoins comme suite :

### **LEDs connectées**

|  |  |
| --- | --- |
| **Fonction** | S’accorder en fonction de la couleur de l’écran TV |
| **Objectif** | S’éclairer de la couleur dominante de l’écran |
| **Description** | La bande LED sera munie d’un capteur de lumière qui lui transmettra les couleurs de l’écran afin qu’elle change de couleur |
| **Contraintes** | Le capteur de lumière doit être placé correctement par rapport à l’écran |
| **Niveau de priorité** | Haut |

|  |  |
| --- | --- |
| **Fonction** | S’accorder en fonction des couleurs définies par l’utilisateur |
| **Objectif** | S’éclairer de la couleur choisie par l’utilisateur |
| **Description** | La bande LED sera connectée à l’application et devra changer de couleur en fonction de ce qui est sélectionné dans l’application. |
| **Contraintes** |  |
| **Niveau de priorité** | Haut |

### **Application mobile**

|  |  |
| --- | --- |
| **Fonction** | Se connecter au périphérique |
| **Objectif** | Pouvoir contrôler la bande LED |
| **Description** | Le téléphone sera connecté en Bluetooth à la bande LED |
| **Contraintes** | La distance entre les deux objets connectés doit être raisonnable |
| **Niveau de priorité** | Haut |

|  |  |
| --- | --- |
| **Fonction** | Contrôler la bande LED à distance |
| **Objectif** | Permettre à l’utilisateur de changer la couleur de la bande |
| **Description** | L’application permettra de changer de couleur, allumer et éteindre les LEDs |
| **Contraintes** | L’application doit avoir la priorité sur le capteur de lumière |
| **Niveau de priorité** | Haut |

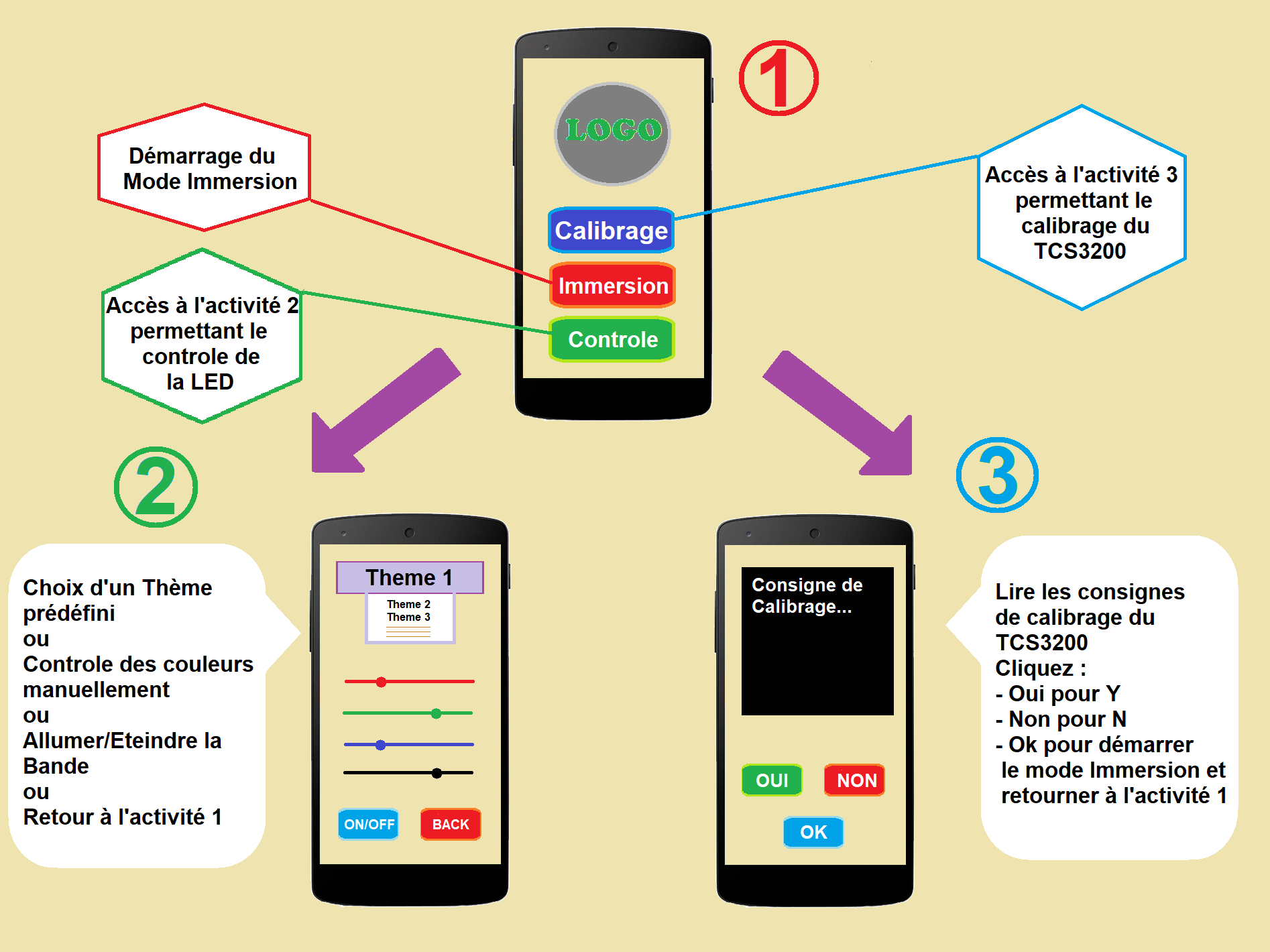
# Conception architectural

## Choix des langages de programmations

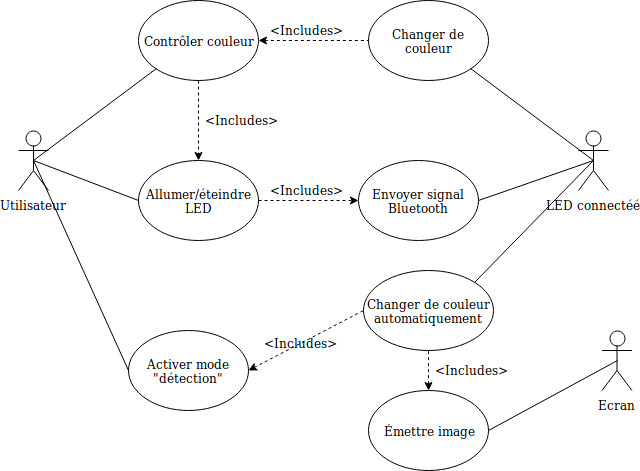
Notre application est mobile est codée en JAVA pour les activité et classe et XML pour le disign. Pour cela nous utilisons Android Studio

Pour l’arduino et le capteur nous utilisons du C++ (.ino) à l’aide de l’application Arduino IDE

## Maquette de notre application



## Diagramme des cas d’utilisation



Description des cas d’utilisation

# Gestion de projet

## Planification et environnement de travail

Afin de réussir le projet et respecter la date du rendu, il nous a fallu une très bonne gestion et pour cela nous avons opté pour deux méthodes :

La première est l’organisation des membres de l’équipe, d’où le choix d’un chef de projet (Mohamed BENSAAD), d’une équipe pour l’application mobile (Heba KADDOUH & Jessy COLOMBO) et d’une autre pour l’Arduino (Mohamed BENSAAD & Khalid BARAKAT). Pour faciliter la communication nous avons créé un groupe WhatsApp qui nous permettait d’organiser des réunions (le jeudi matin ainsi que les mardis à 17h00 après le cours d’IOT, sans oublier, les créneaux du module, les mardis après-midi), de s’entraider, de faire un échange d’informations et de faire une mise au point.

De plus, notre chargé de TD nous a conseillé de créer un compte Github afin de pouvoir suivre toutes les modifications du code et y avoir accès en cas de problème technique, que nous avons utilisé sans modération.

Nous avons mis en place sur GitHub dans l’onglet Projects, un planificateur de toute les taches que nous allons accomplir, que nous avons accomplis et que nous sommes en trains d’accomplir. Ce planificateur nous a permis de se retrouver et de voir ce qu’il restait à faire. On peut aussi y retrouver les liens utiles, les composants et les logicielles utilisé.

## Répartition des tâches

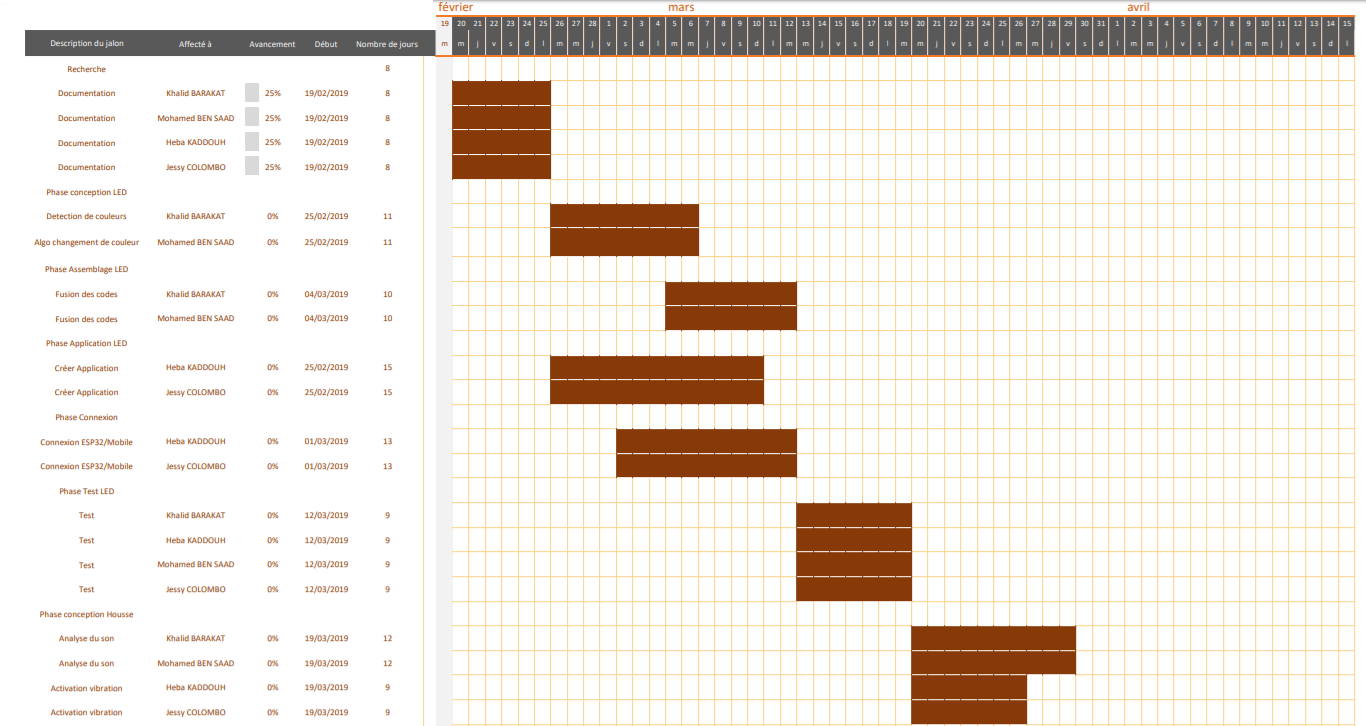
Comme dit plus haut nous avons décidé de repartir le travail en deux équipes.

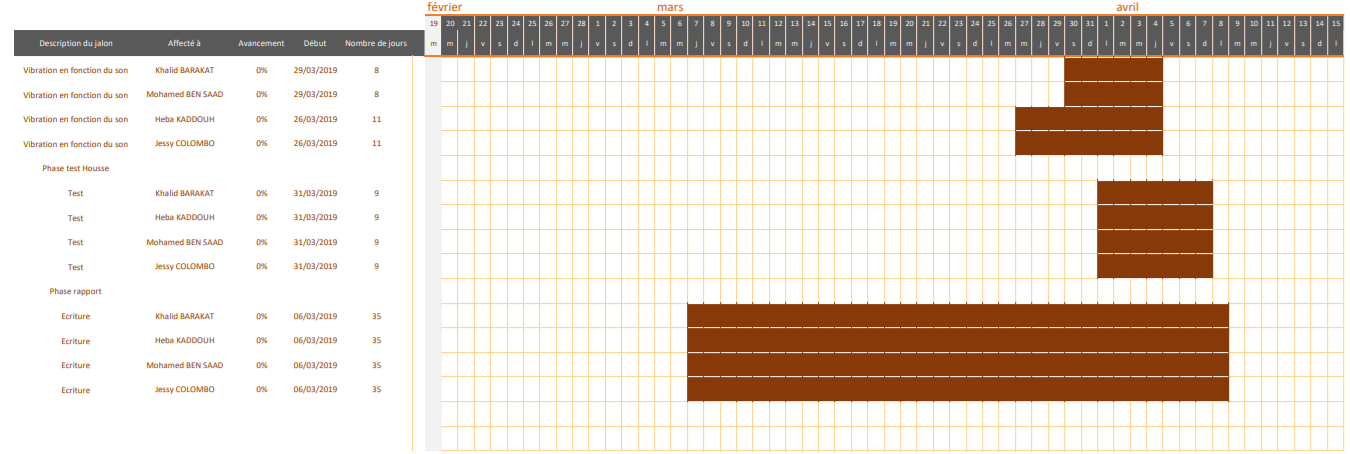
Nous avons tous contribué à la conception architecturale en réalisant le diagramme de Gant et du cas d’utilisation qu’on peut retrouver sur le GitHub. Ainsi qu’à la rédaction du cahier des charges, de l’étude de l’existant, du rapport et de la conception du diaporama de présentation.

Pour avoir un meilleur aperçu du temps accordé pour chaque tâche, nous avons réalisé un diagramme pour l’application mobile et pour l’arduino.

## Diagramme de Gant

Le diagramme de GANTT est un outil efficace permettant de générer une visualisation de l’avancement du projet. Il permet de donner une vue globale des tâches à réaliser et le du temps associé.





Nous avons réalisé le diagramme de Gant dans l’optique de l’accomplissement total du projet. Sauf que nous avions sous-estimer la complexité du projet. Nous détaillerons tous nos changements par la suite.

# Développement du Set-Up

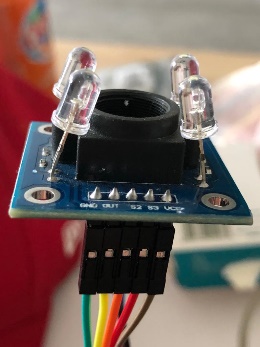
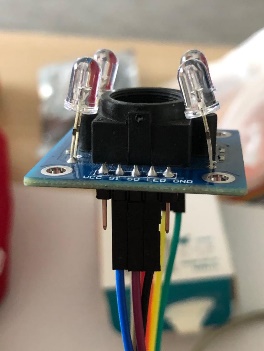
## Arduino

Au départ nous avons décidé d’utiliser une ESP32 pour notre projet. Malheureusement, après plusieurs essais de couplage de celle-ci avec notre capteur TCS3200, nous avons remarqué que l’architecture de l’ESP32 n’était pas compatible. Il aura fallu tout implémenter nous même pour l’adapter avec le capteur. Nous avons donc jugé plus judicieux de passer à un Arduino vu le temps qui nous était imparti. Ce choix n’a pas été anodin car nous avons trouvé plusieurs codes fonctionnels sur Arduino. Cela nous a permis de comprendre et de nous familiariser avec le TCS3200.

### Capteur TCS3200

Ce capteur est utilisé pour détecter les couleurs d’objets et non la couleur d’un écran. Nous avons fait un pari risqué en choisissant celui-ci car nous ne savions pas s’il répondrait à nos attentes. Son faible prix nous a également réconforté sur la perte qu’il aurait pu nous couter.

*Face droite Face Avant Face Gauche*

Notre premier objectif à été de faire fonctionner notre capteur avec l’Arduino. Le branchement a été rapidement maitriser grâce à quelques exemples trouver au préalable.

Voici le montage :

Un fois le montage opérationnel, nous avons tester notre capteur avec plusieurs codes comme celui fourni avec le TCS3200 et d’autres trouver sur le Web. Cependant, comme le capteur n’est pas destiné aux écrans, la plupart de nos tests ont malheureusement échoué. A ce stade, nous avons réussi à reconnaitre les couleurs primaires RVB avec des valeurs non conformes à nos attentes allant de -10000 à 10 000. Nous devions remédier à ce problème et trouvé une fonction qui nous permettrait de calibrer le capteur sur tout type d’écran ainsi que de ramené les valeurs reçues entre 0 et 255. Nous avons essayé plusieurs modèles mathématiques pour moduler nos valeurs mais les codes utilisé n’étaient toujours pas dans le même but que notre projet.

Après de nouvelles recherches, nous avons découvert une bibliothèque sur GitHub : <https://github.com/blascarr/TCS3200-ColorSensor> avec « TSC3200.h », implémenter spécialement pour notre capteur incluant une fonction de calibrage optimal. Nous avons analysé cette Library pour pouvoir la modifier par la suite et l’adapter à nos besoins. Une fois modifié à notre guise, le résultat a répondu à toute nos espérances.

Par la suite, nous avons intégré la bande LED « Pixel Strip » à l’Arduino. A l’aide de sa bibliothèque originale « Adafruit\_NeoPixel.h », nous avons pu lui envoyer les valeurs perçues par le capteur. Après quelques optimisations sur la synchronisation des deux composants électronique, le Mode Immersion venait de voir le jour.

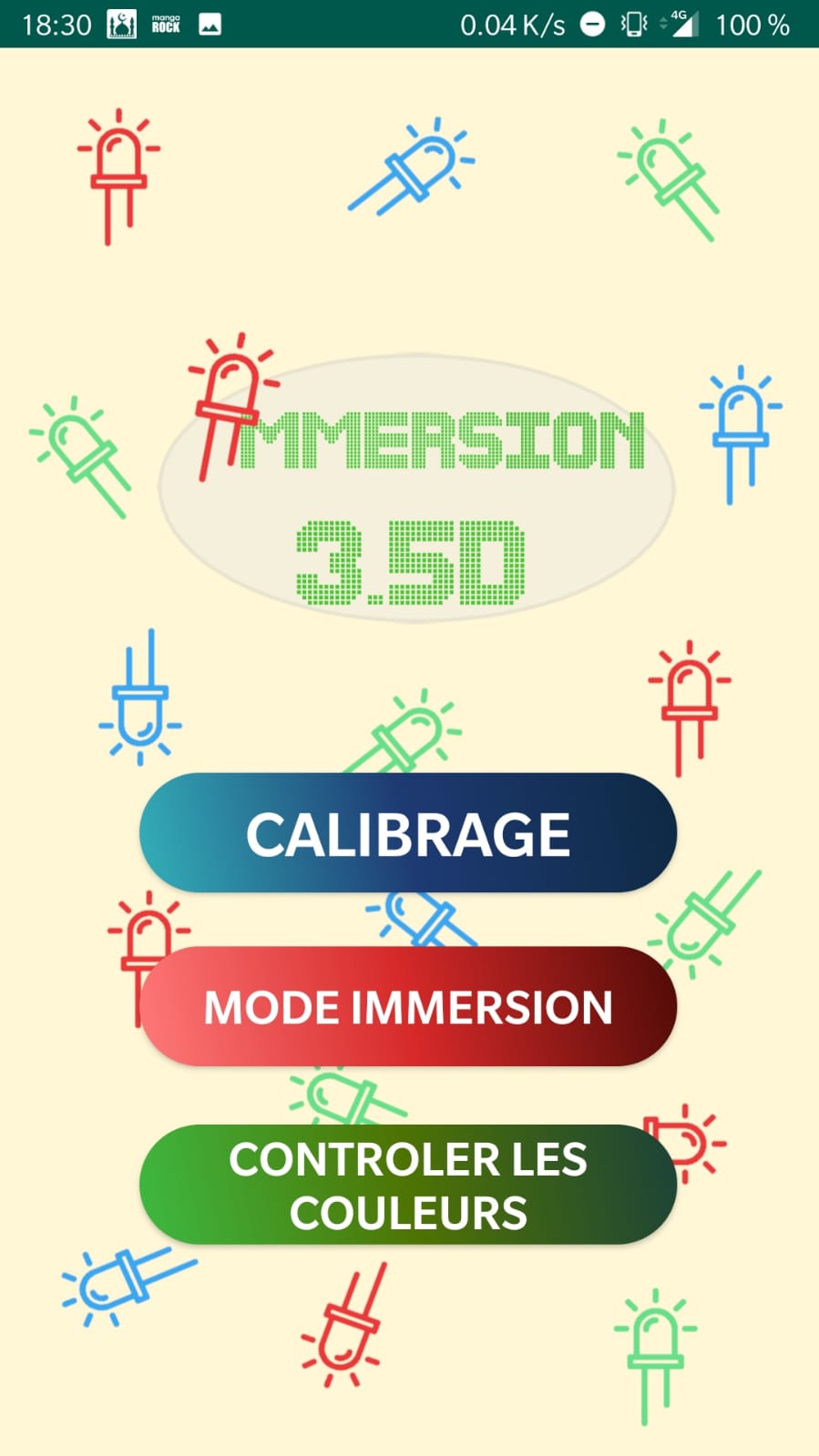
### Bluetooth HC-05

Le choix du HC-05 a été fait en fonction de son prix de sa popularité et de sa bonne notation.

### Logiciel Final

## Application mobile

### Page d’accueil



Lorsque l’utilisateur lance l’application, il se retrouve sur cette page et une connexion BluetoothSocket (bSocket) est lancé. La connexion Bluetooth est établi grâce à notre fonction « bluetooth\_connect\_device()». Si la connexion n’est pas établie une pop-up s’affiche « Vous n'êtes pas connecté au Bluetooth » et l’interface ne peux pas être utilisée.

Capture popup

Au lancement de l’application, on envoie la lettre « O » à l’Arduino qui permet de mettre l’Arduino en mode On/Off.

A partir de cette page il peut, au click :

* Calibrer le TCS3200, en envoyant la lettre « C » en Bluetooth à l’Arduino et passe à la page de calibration
* Entrer en mode immersion, en envoyant la lettre « I » en Bluetooth à l’Arduino et passe en mode immersion tout en restant sur cette page
* Contrôler les couleurs, en envoyant la lettre « T » en Bluetooth à l’Arduino et passe à la page de control

On envoie à l’Arduino en Bluetooth en utilisant la fonction « bSocket.getOutputStream().write("O".getBytes()); »

### Page de calibration

### Page de contrôle

### Mode dégradé

## Logo

# Contraintes/Difficultés

Une des premières difficultés que nous avons rencontrées est le choix de la carte électronique. Nous avons au début opté pour une ESP32 (Mohamed & khalid)……

# Bilan du projet

# Annexe

## Mode d’emploi

Pour utiliser l’application l’utilisateur doit d’abord être connecté en Bluetooth à l’Arduino. Une fois connecté il pourra effectuer les actions suivantes :

* Calibrer
* Contrôler les couleurs
* Lancer le mode automatique

Lorsque l’utilisateur click sur « Calibrage », on demande d’abord à l’utilisateur s’il veut calibrer ?

S’il ne veut pas calibrer il appuie sur « Mode immersion » sinon il appuie sur « Oui ».

En appuyant sur « Oui », le calibrage commencera et il faut suivre les instructions préciser sur le terminal en mettant le capteur sur la couleur demander. ………..

Lorsque l’utilisateur click sur « Mode immersion », la bande LED change de couleurs en fonction de la couleur captée par le capteur. L’utilisateur aura juste à placer le capteur devant un écran.

Lorsque l’utilisateur click sur « Contrôler couleur », il est amené sur une autre page qui lui permet de :

* Choisir des thèmes en cliquant sur le menu
* Changer les couleurs de la LED grâce au seekbar
* Eteindre ou allumer la LED en cliquant sur « Allumer/Eteindre »
* Retourner sur la première page en cliquant sur « Retour »
* Mode dégradé qui nous amènent sur une nouvelle page pour contrôler les couleurs