

IOT_frontend_mobile

Lancer le projet

Après avoir clone le projet, il vous faut premièrement le construire avec

```
npm install
```

Ensuite vous pouvez le lancer en mode développement avec la commande

```
react-native run-android
```

Note : Le projet a été testé avec Android, il n'a malheureusement pas pu être expérimenté sur iOS et il n'est pas certain qu'il soit opérationnel sur cette plateforme à cause des spécificités de la librairie react-native-nfc-manager.

Application

Travail effectué

L'application mobile semble fonctionnelle. En effet, la connexion avec les deux backends (le backend TTN ainsi que le backend vidéo) est fonctionnelle. Le peering d'un nouveau noeud Lora ou vidéo fonctionne bien, il est possible de récupérer les informations stockées dans ces noeuds grâce à NFC, de créer ensuite ce noeud sur le backend correspondant et pour finir de réécrire les informations nécessaires dans le noeud grâce à NFC.

Les processus de peering sont différents selon les noeuds :

Noeuds Lora

L'application lit l'UID du noeud grâce à NFC, le transmet au backend TTN qui lui renvoie l'APPKEY et l'APPEUI à l'application. Pour finir, l'application écrit ces 3 informations (l'APPKEY, l'APPEUI et le DEVUI dans le noeud via NFC). Il a été convenu avec le groupe s'occupant des noeuds Lora que ces informations seraient stockées selon le format suivant :

APPKEY;APPEUI;DEVUI

Donc en séparant ces informations par des point-virgules.

Noeuds Vidéos

Pour les noeuds vidéos, la démarche est sensiblement la même mais les données échangées diffèrent. L'application récupère l'adresse IP du noeud ainsi que sa clé publique via NFC et envoie ces données aux backend vidéo. Le backend vidéo répond avec sa clé publique et l'application écrit dans le noeud vidéo les données suivantes dans l'ordre séparées par un point-virgule :

adresseIPNoeud;clePubliqueNoeud;clePubliqueServer

L'application envoie ensuite l'information au backend que les informations ont bien été écrites dans le noeud vidéo et c'est le backend qui va rentrer en contact avec le noeud.

Amélioration possibles

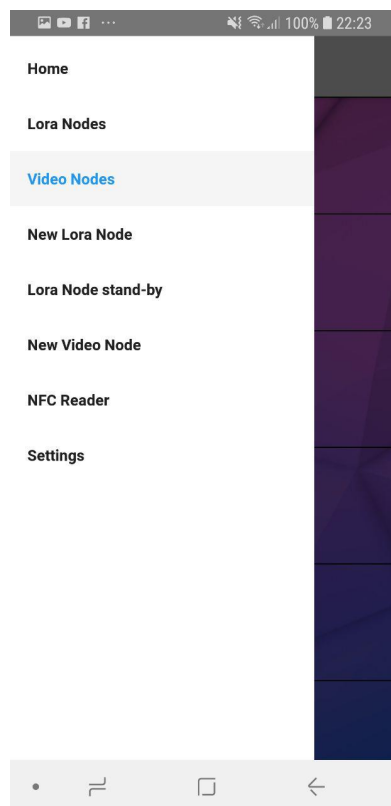
La chose la plus importante serait tout d'abord de tester l'application avec des noeuds Lora et Vidéo réels. Car même si l'application a été testée avec des puces NFC, il faudrait être sûr du format des données lues et écrites dans les noeuds. Après discussions avec les groupes responsables des noeuds et particulièrement des personnes responsables de la lecture/écriture NFC, il a été convenu que les informations étaient lues et écrites en texte pur avec des points-virgules entre les différentes informations.

Une autre amélioration importante serait la gestion des cas d'erreur. En effet, l'application fonctionne très bien lorsque les données lues correspondent bien au format attendu, ou que les backends répondent bien ce qu'ils sont sensés répondre. Mais si cela n'est pas le cas, l'application peut avoir des réactions étranges qui peuvent nécessiter de redémarrer l'application.

Il y aurait aussi un travail à faire au niveau de la refactorisation du code. En effet, plusieurs fichiers ont du code redondants qui pourrait être factorisé avec un peu de temps.

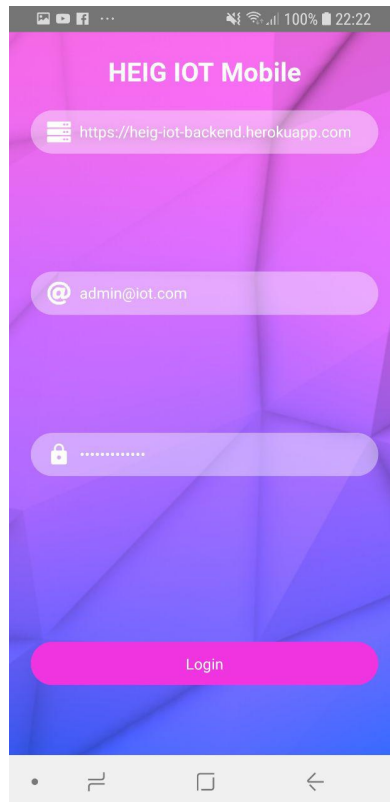
Architecture

L'application comporte différents menus :



Login

Simple menu pour se logger

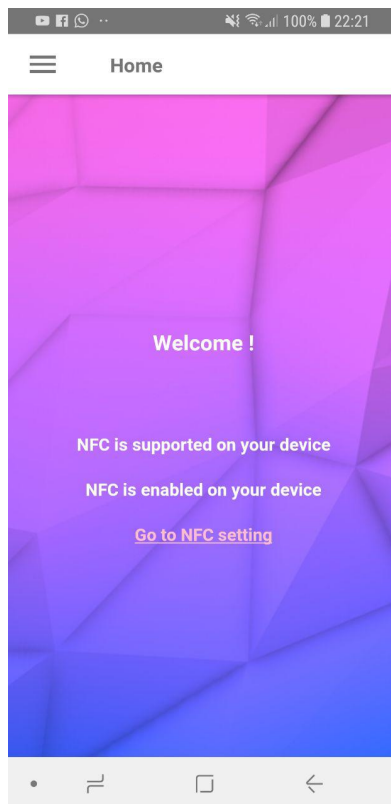


Accueil

Cette fenêtre indique si l'utilisateur possède un smartphone qui est compatible avec NFC.

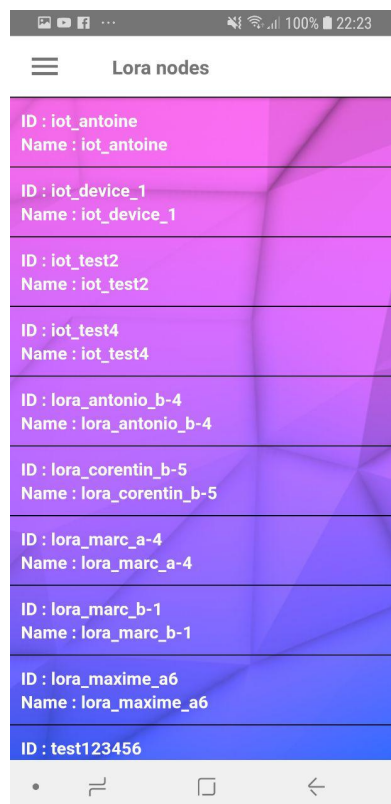
Pour Android :

- La fenêtrés indique si NFC est activé sur le smartphone
- La fenêtre propose un lien pour accéder aux réglages d'NFC sur le smartphone, ce qui permettrait d'activer NFC si cela n'est pas le cas.



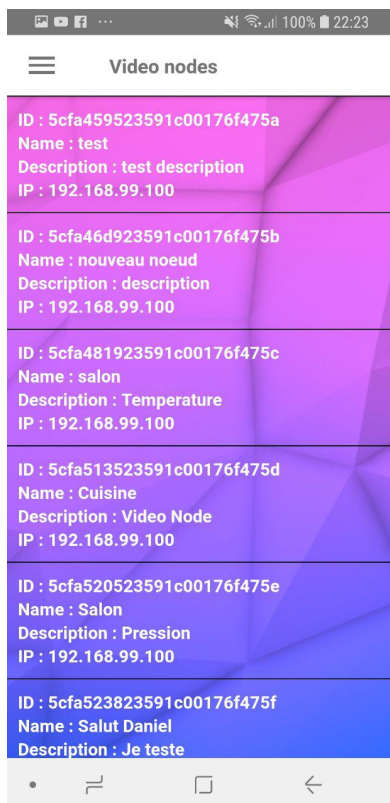
Noeuds Lora

Cette fenêtre affiche les noeuds Lora enregistrés sur le back-end TTN en fournissant leurs ids, leurs noms et leurs descriptions. On pourrait aussi afficher depuis quand ils ont été créés (le code est présent) mais cette informations n'a pas été stockée au niveau du backend TTN.



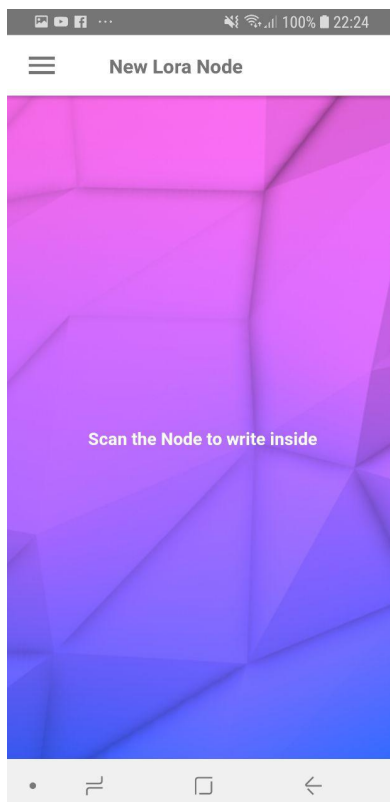
Noeuds Vidéo

Affiche les noeuds vidéo enregistrés sur le backend vidéo avec leurs ids, leurs noms, leurs descriptions ainsi que leurs adresses IP. On pourrait aussi afficher depuis quand ils ont été créés (le code est présent) mais cette informations n'a pas été stockée au niveau du backend vidéo.

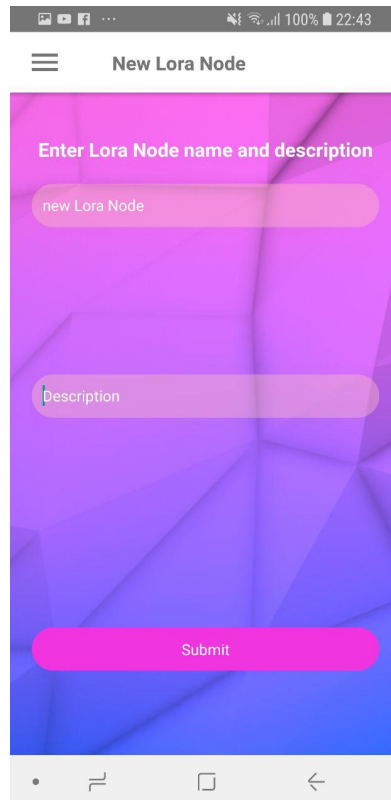


Nouveau noeud Lora

Cette page demande tout d'abord de scanner le noeud Lora afin de lire son UID.

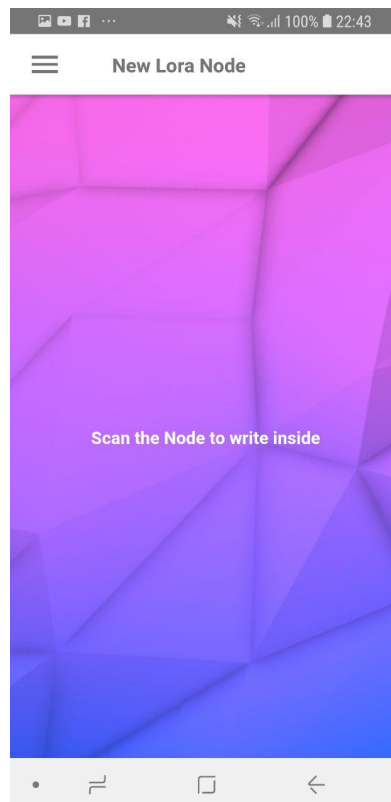


Une fois l'UID scanné, l'application demande le nom et la description du noeud Lora à l'utilisateur.

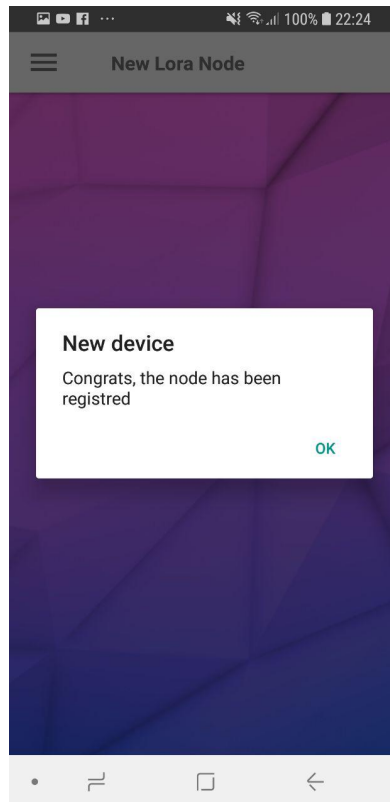


The screenshot shows a mobile application interface titled "New Lora Node". It features a form with two input fields: "new Lora Node" and "Description". Below the fields is a "Submit" button. The background is a blue and purple geometric pattern. The status bar at the top shows the time as 22:43 and 100% battery.

Lorsque ces deux informations sont validées et vérifiées comme étant non-nulles, l'application fait l'échange d'informations avec le backend TTN comme indiqué plus haut. L'application demande pour finir de scanner à nouveau le noeud Lora afin d'y écrire les informations nécessaires via NFC.



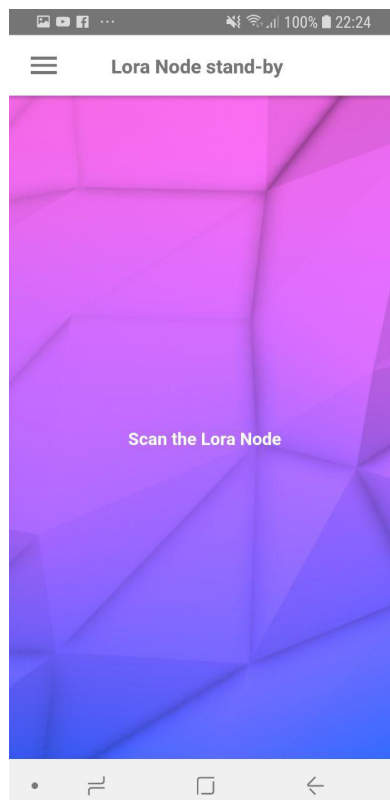
Un feedback indique ensuite à l'utilisateur que le noeud Lora a bien été configuré.



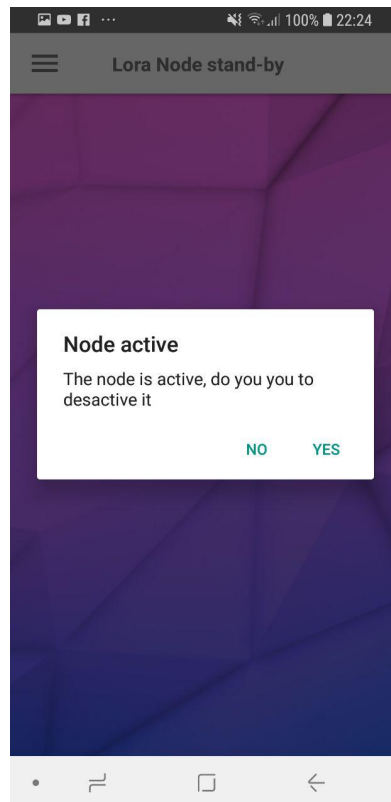
Stand-by noeud Lora

Cette page n'est pas reliée au backend TTN car celui-ci n'offre pas les possibilités d'activer ou desactiver le noeud Lora. Lorsque cela sera le cas, il faudra surement adapté un peu cette page pour la rendre compatible, mais le principe est présent.

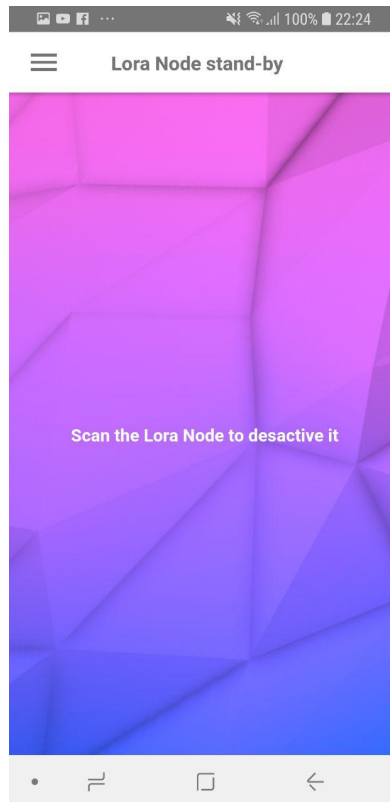
L'application demande tout d'abord de scanner le noeud Lora afin de lire son UID.



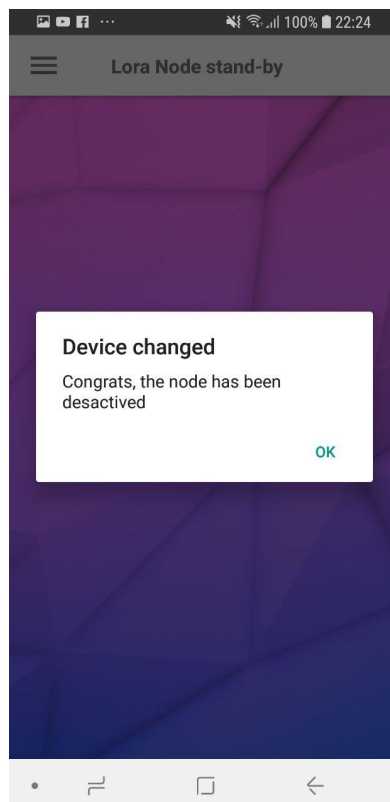
L'application demande ensuite au Backend TTN si ce noeud est en stand-by ou non (partie à implémenter avec la liaison avec le Backend TTN lorsque celui-ci aura implémenter le stand-by). En fonction de la réponse du Backend TTN, l'application demande si l'utilisateur veut activer (si le noeud est inactif) ou désactiver (si le noeud est actif) le noeud.



Si l'utilisateur appuie sur "YES" à l'étape précédente, l'application lui demande de scanner le noeud Lora afin d'y écrire la commande nécessaire à son activation/désactivation. Cette commande est à définir avec le groupe qui s'occupe du noeud Lora.

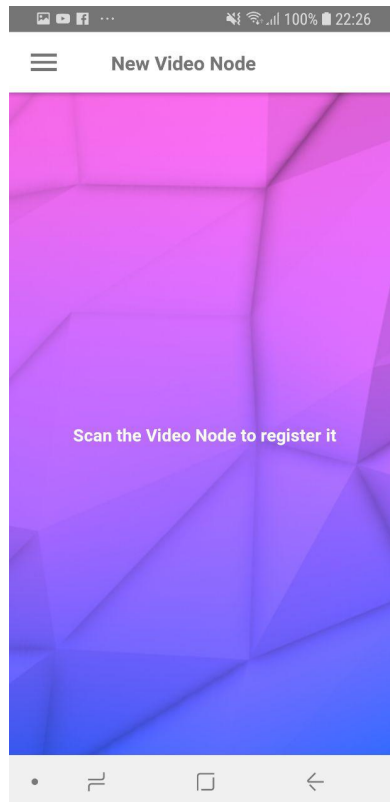


Une fois que l'utilisateur aura scanné le noeud afin de le désactiver/activer, il aura un feedback lui indiquant que ce noeud à bien été activé/désactivé.

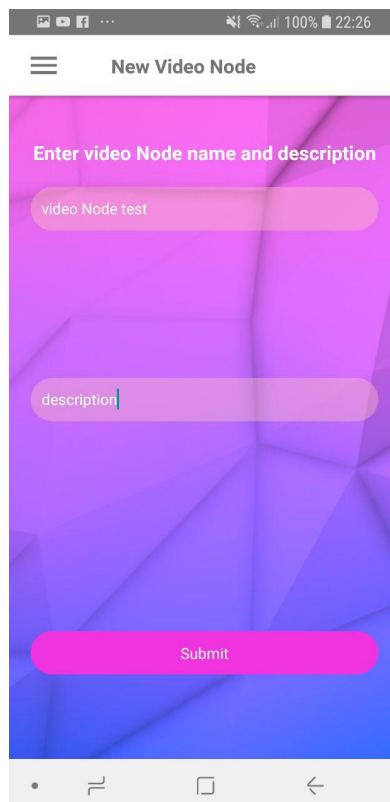


Nouveau noeud Lora

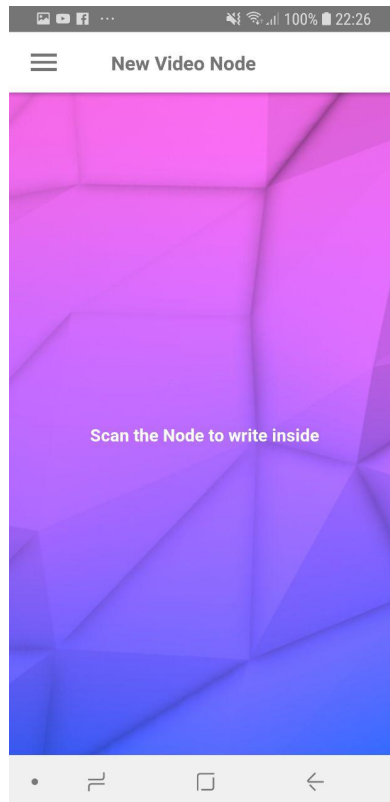
Cette page demande tout d'abord de scanner le noeud Vidéo afin de lire son adresse IP ainsi que sa clé publique.



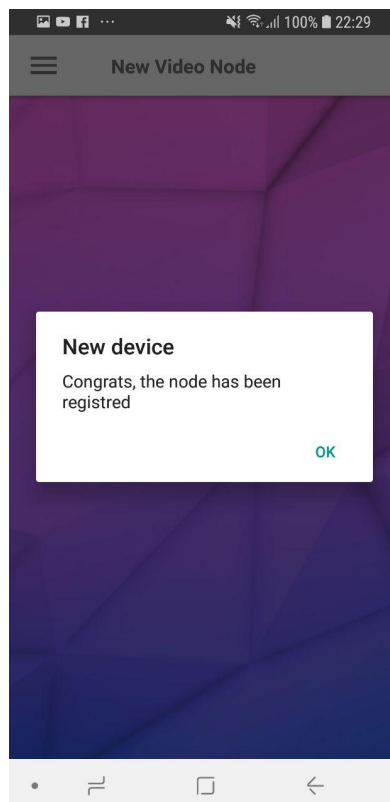
Une fois l'UID scanné, l'application demande le nom et la description du noeud Lora à l'utilisateur.



Lorsque ces deux informations sont validées et vérifiées comme étant non-nulles, l'application fait l'échange d'informations avec le backend Vidéo comme indiqué plus haut. L'application demande pour finir de scanner à nouveau le noeud Vidéo afin d'y écrire les informations nécessaires via NFC.

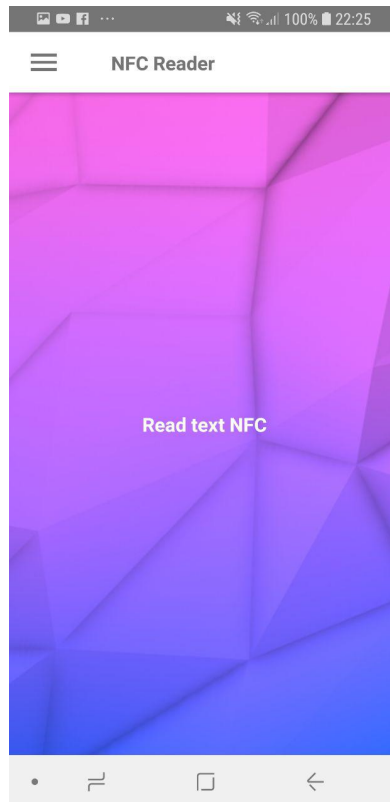


Un feedback indique ensuite à l'utilisateur que le noeud Vidéo a bien été configuré.

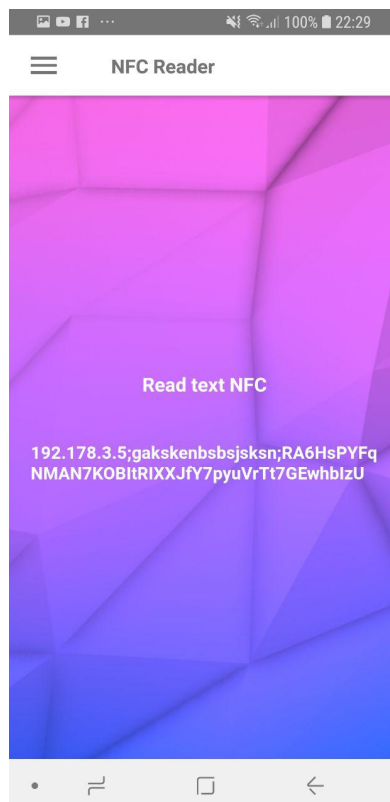


Lecteur NFC

Cette page permet simplement de lire les informations écrites dans le noeud Lora ou vidéo grâce à NFC. L'application demande de scanner le noeud voulu

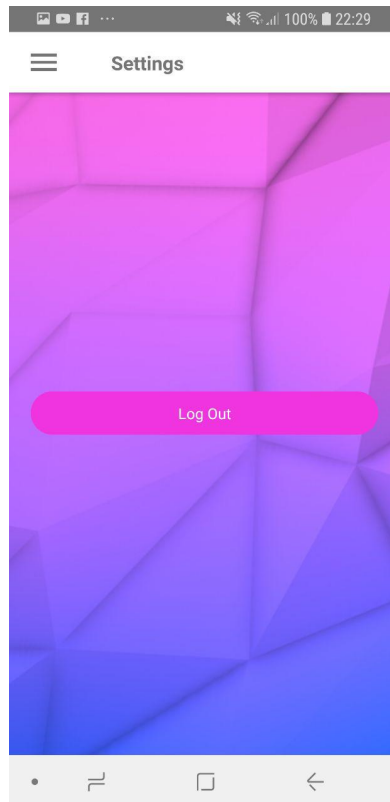


Une fois le tag NFC lu, le texte lu est affiché



Réglages

Cette page permet simplement de se déconnecter du compte



Librairies utilisées

@react-native-community/async-storage : Stockage clé-valeur asynchrone et persistant. Permet de stocker le token de l'utilisateur et donc de ne pas devoir se reconnecter à chaque fois que l'on relance l'application.

react-moment : moment.js adapté à react. Permet d'afficher proprement depuis combien de temps les noeuds Video et Lora ont été créés ("créé il y a ..."). Plus utilisé dans l'état actuel de l'application.

react-native-nfc-manager : Permet d'utiliser les fonctionnalités d'NFC avec React Native. Permet l'écriture et la lecture de tags NFC.

react-navigation : Permet le routing et la navigation entre les différentes pages de notre application.

lottie-react-native et *react-native-animated-loader* : permet d'afficher un loading facilement en utilisant les animations faites par la communauté de Lottie.

eslint : un linter pour avoir un code uniforme et qui respecte à une série de règles afin d'éviter les erreurs de syntaxe et d'assurer l'utilisation de bonnes pratiques de programmation.