**Push Notification via FireBase**

**Steps**

1)Création de compte FireBase puis création d’un nouveau projet

2)Configuration de l’option Message

3)Installation du service GoogleService dans le projet et ajout de cette option dans App.json

4)Correspondance entre Projet React et FireBase via la Key

Pour fixer les warning des dependences

npx expo install --fix

**Build APK**

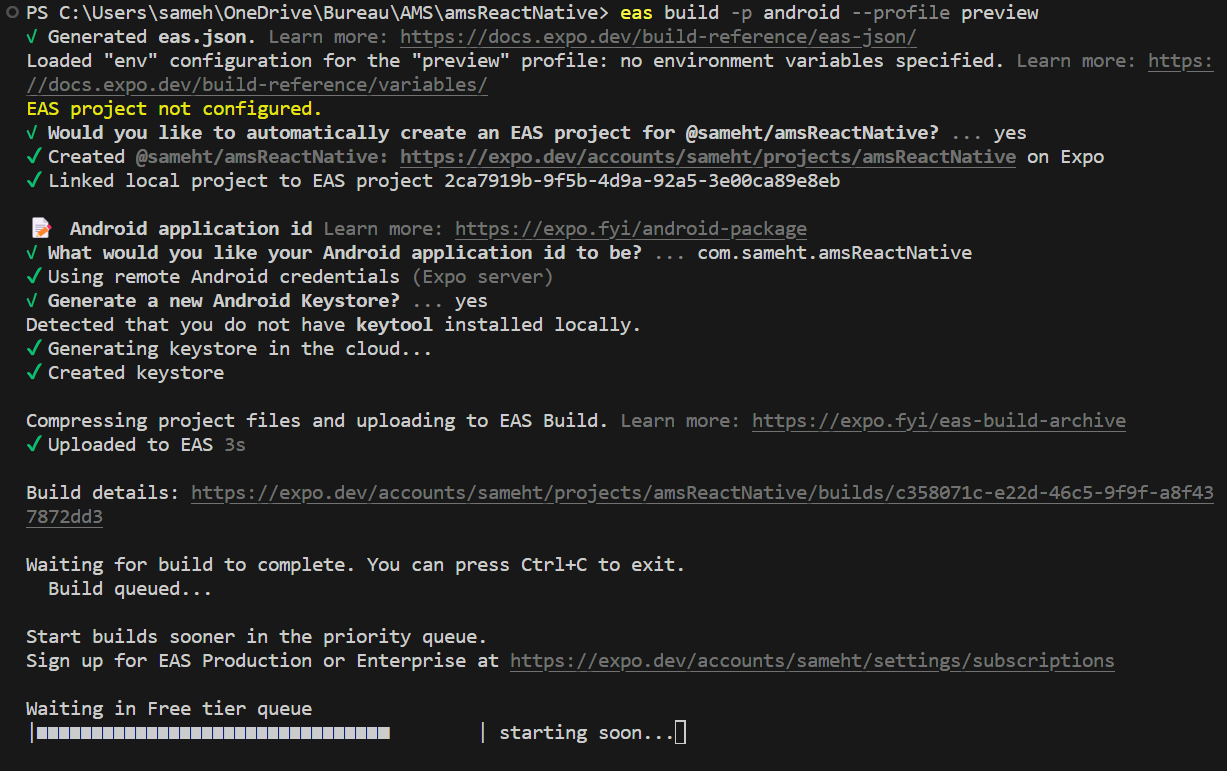
Il faut avoir un compte sur<https://expo.dev/>

npm install -g eas-cli

**eas-cli** est un outil essentiel pour les développeurs utilisant Expo pour gérer le processus de construction et de distribution de leurs applications React Native.

eas build -p android --profile preview

Cette commande lance le processus de construction d'une application pour la plateforme Android en utilisant Expo Application Services.



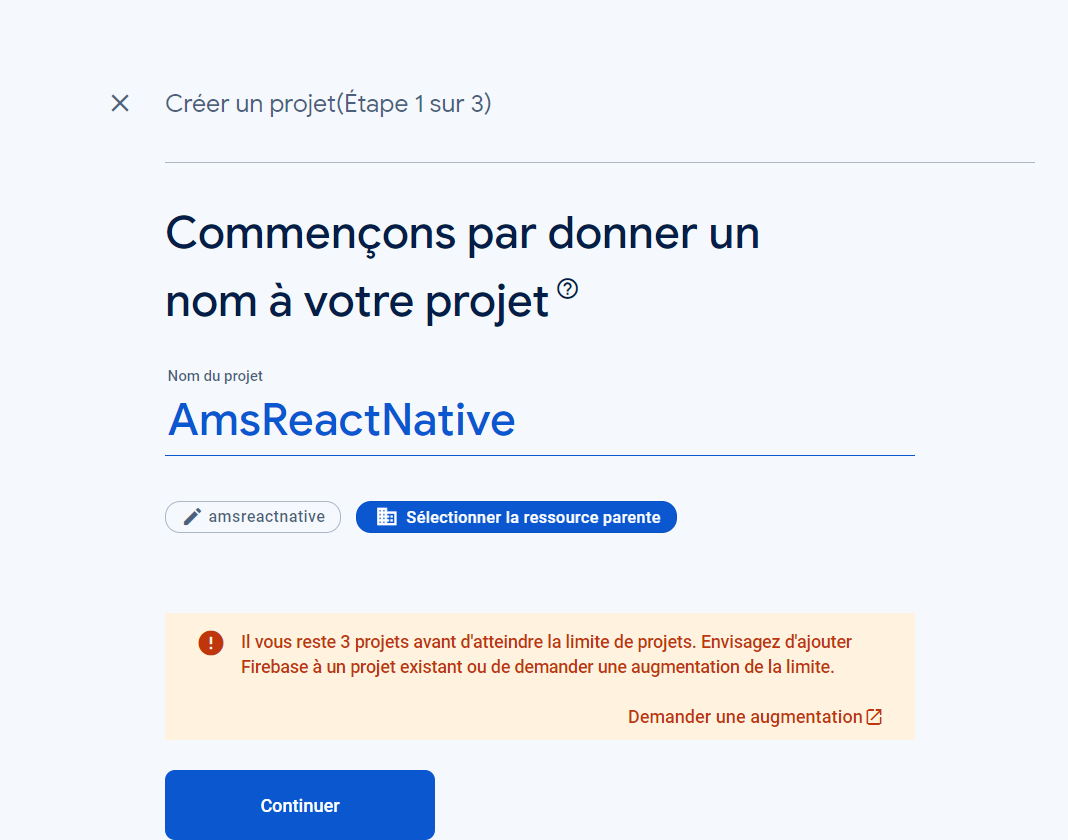
**Push notifications**

<https://firebase.google.com/>

On doit utiliser **Firebase** est une plateforme de développement d'applications mobiles et web développée par Google.Elle offre une variété de services et d'outils pour aider les développeurs à créer, améliorer et faire évoluer leurs applications plus rapidement et plus efficacement.

On doit utiliser **Firebase Cloud Messaging (FCM)** : Un service de messagerie qui permet d'envoyer des notifications et des messages à des utilisateurs sur Android, iOS et les applications web.

1-Création d’un nouveau projet

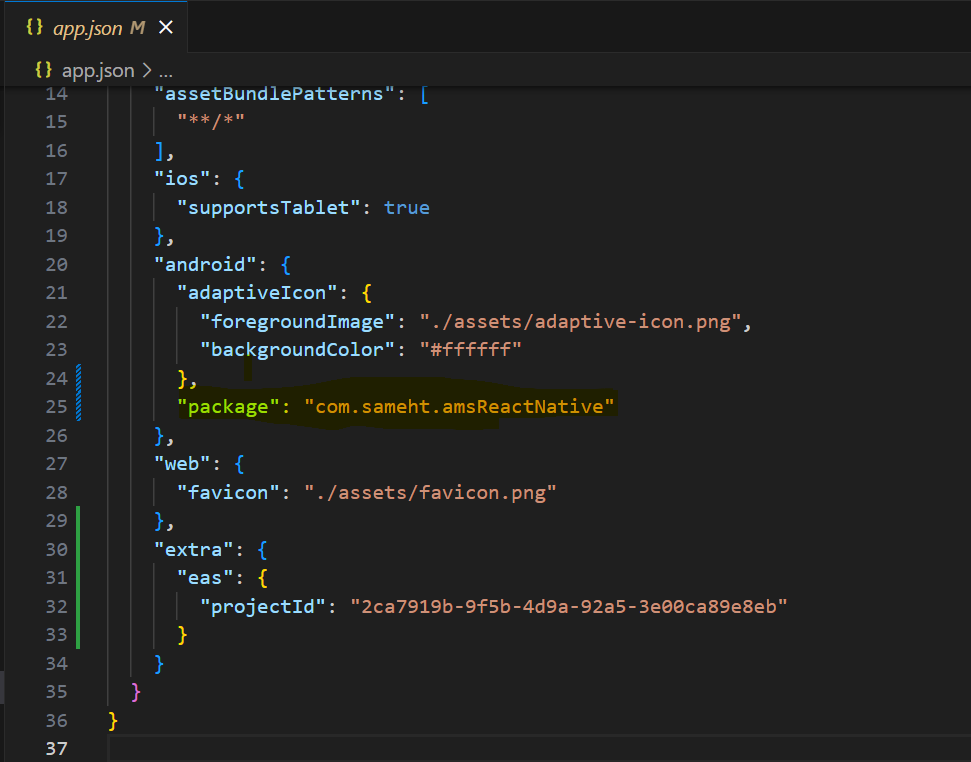


On doit ajouter une application Android





Le nom du package doit être le même dans le fichier **app.json**.

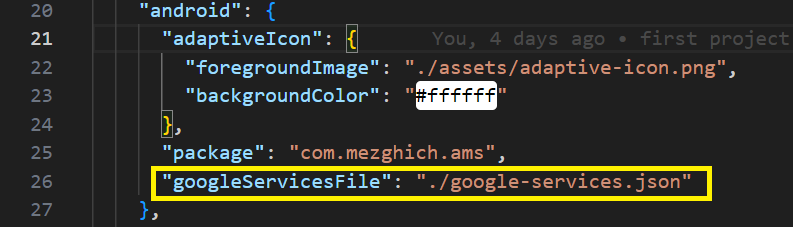


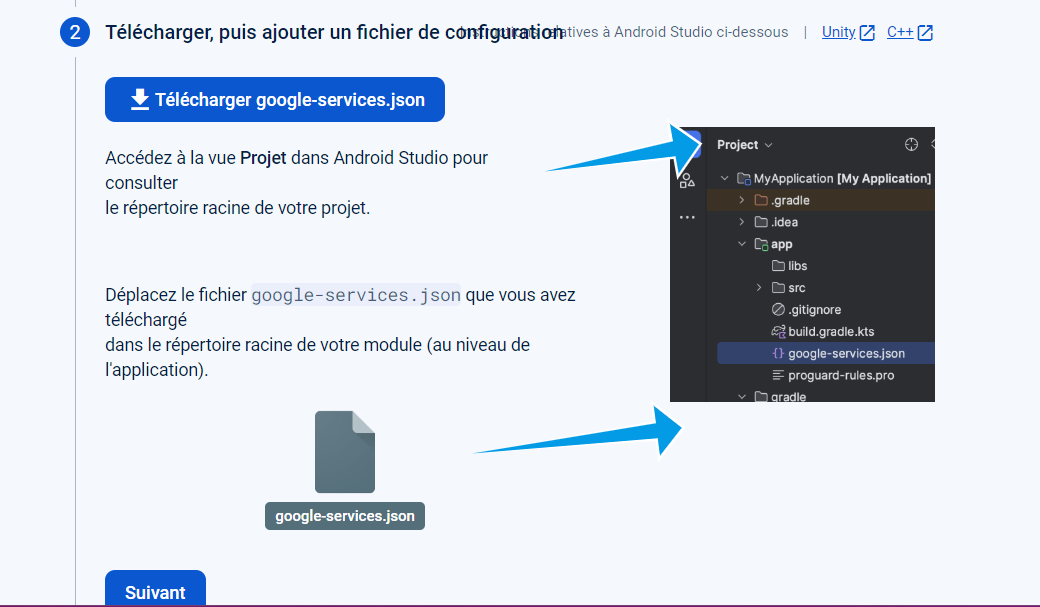
Ensuite, il faut télécharger le fichier **google-services.json** et l'ajouter à votre projet.

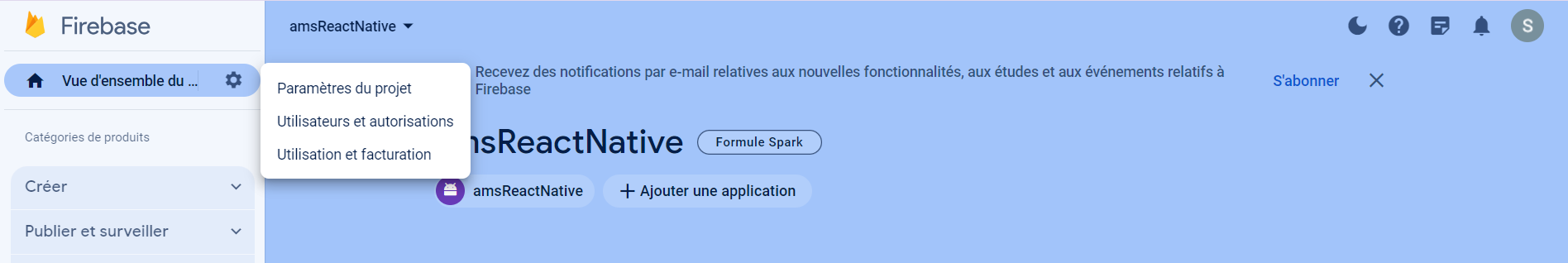
Et ajouter l’instruction suivante dans app.json

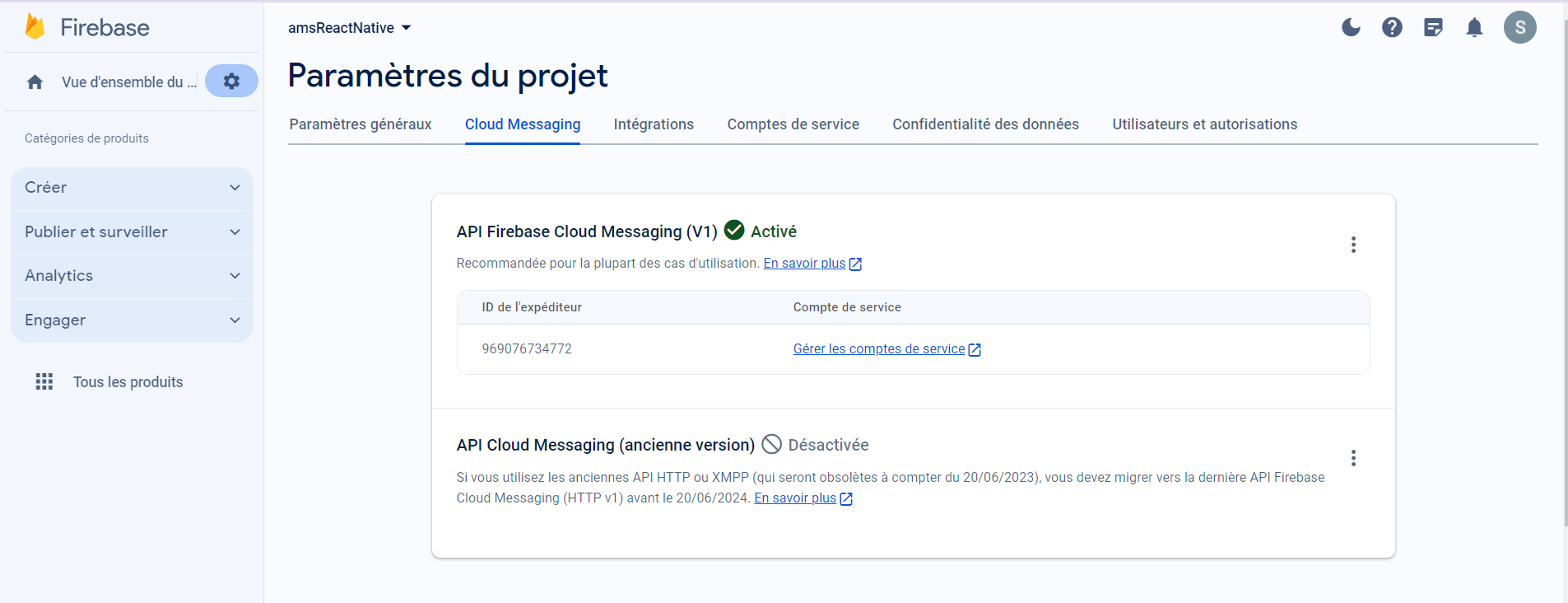
    "package": "com.mezghich.ams",

      "googleServicesFile": "./google-services.json"

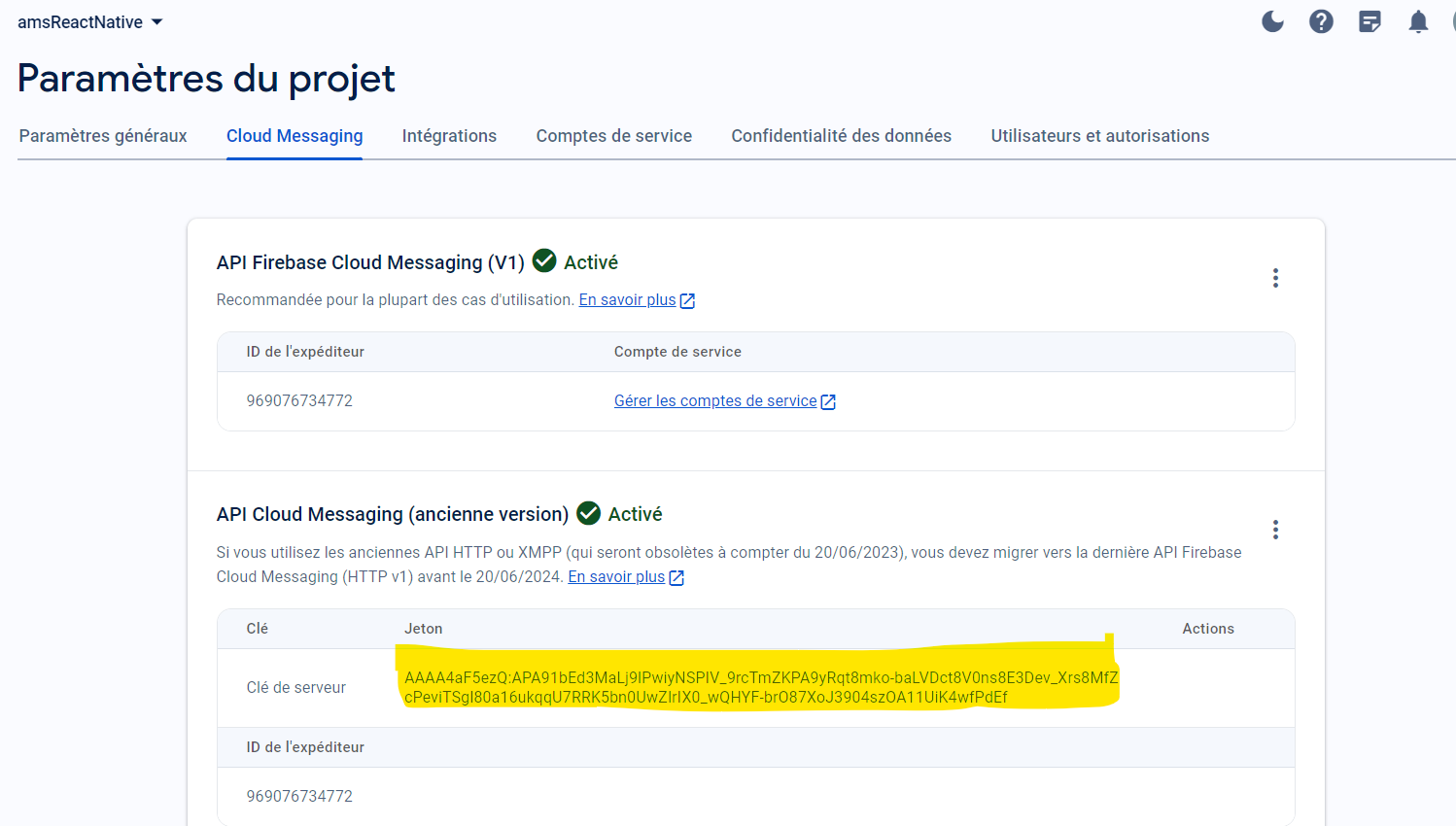
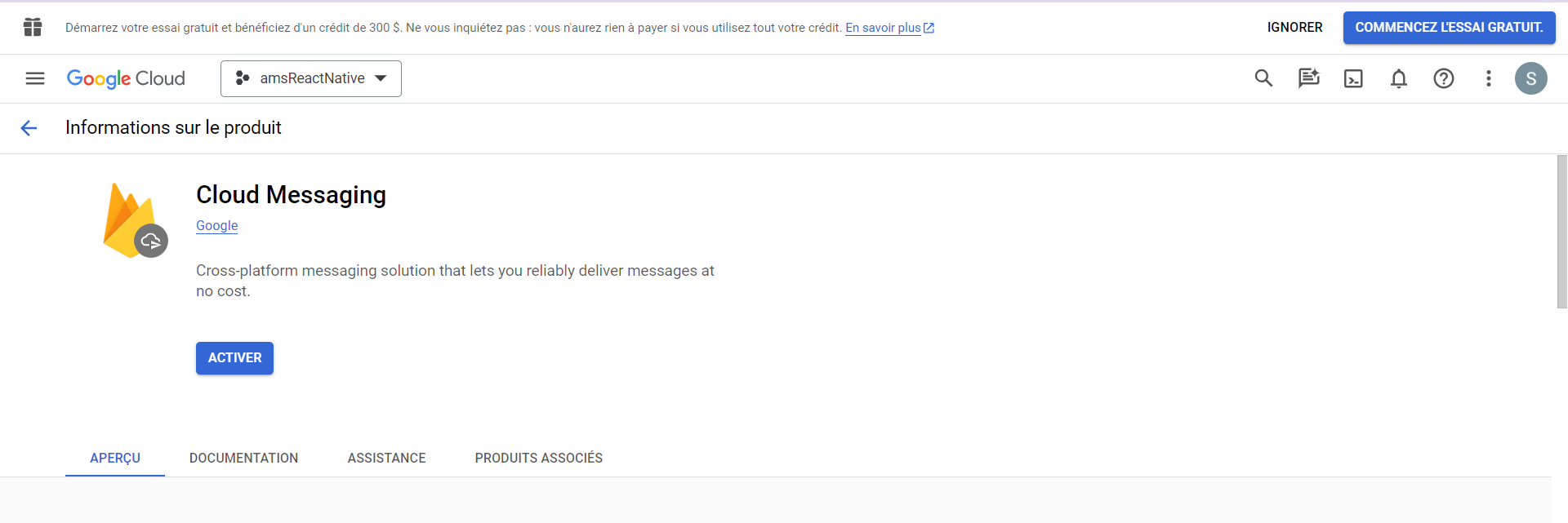
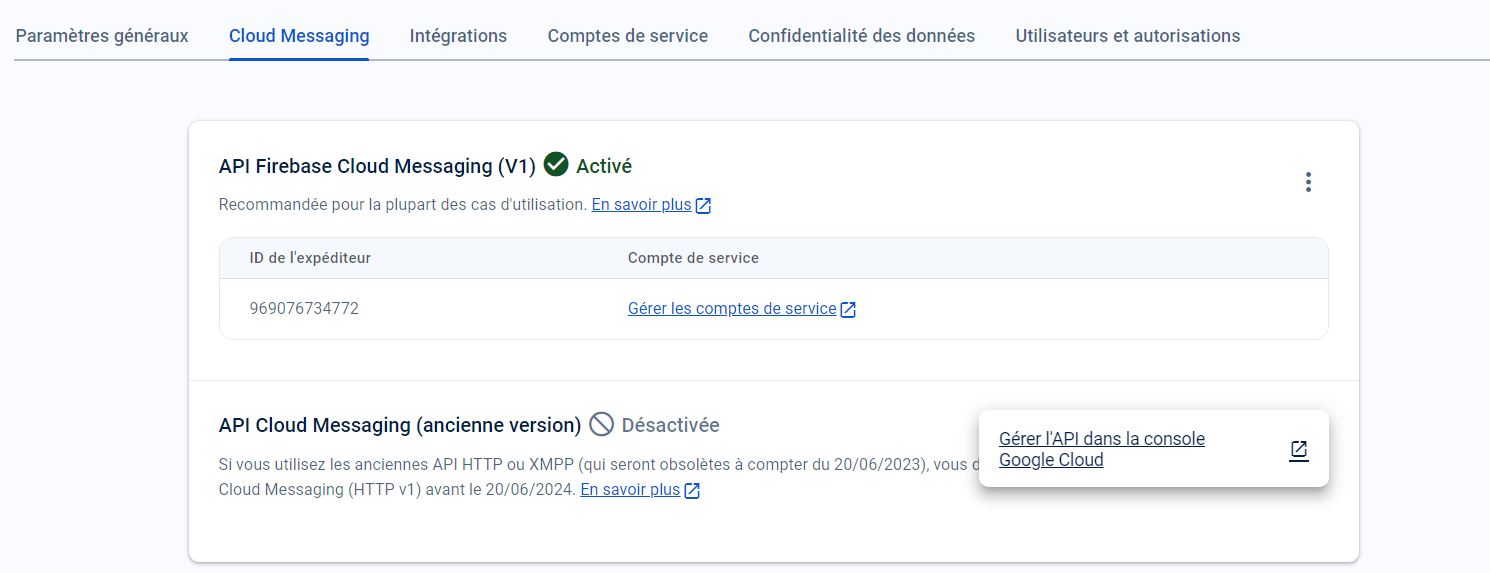






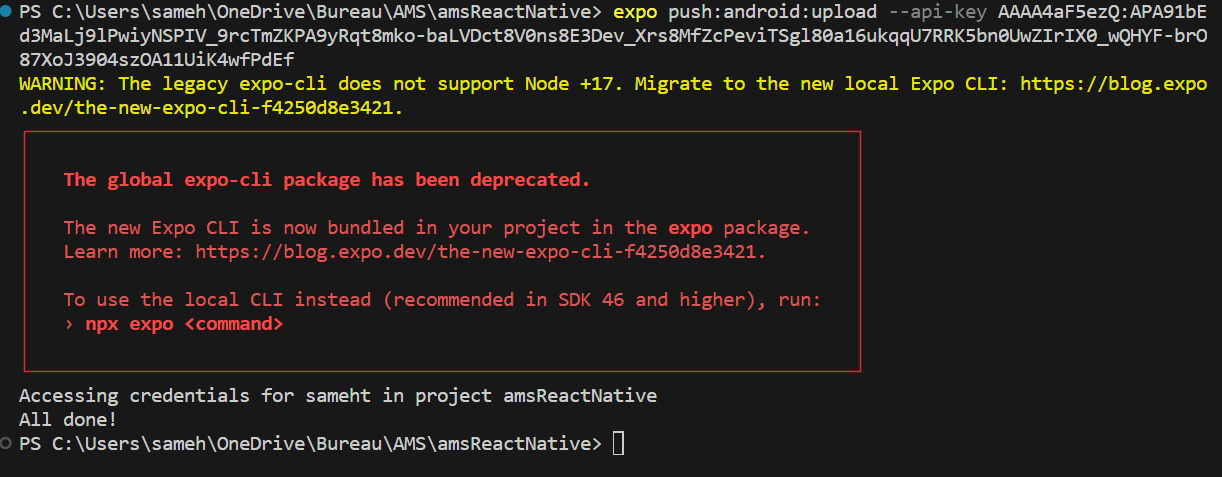


Maintenant, il est nécessaire d'activer l'API Cloud Messaging pour générer une clé de serveur.



Enfin on doit exécuter cette commande

expo push:android:upload --api-key AAAAEyH3cYk:APA91bGjDxvwBpreg7kyD2-JLYqKVa-BhrOxP246USbfM5NFJ3eU2ECUuOlcBdGQm3oqrsyy9LB6yHjTuMP73aVWxczBT6sW9ONaURBPI0C8bSgk52rMk1eMhTH-eeINGaOmpEU2gcwb



=>On doit exécuter 3 commandes

npm install expo-device

fournit des informations sur l'appareil sur lequel votre application s'exécute

npm install expo-notifications

offre une API complète pour gérer les notifications push dans les applications Expo. Il permet de programmer l'envoi de notifications, de les recevoir, et de gérer les permissions liées aux notifications.

npm install expo-constants

La commande **npm install expo-constants** est utilisée pour installer le module **expo-constants** dans votre projet Expo ou React Native (si vous utilisez Expo Managed Workflow). Ce module fournit un ensemble d'API qui vous donnent accès à des constantes systèmes qui sont utiles pour votre application.

Voici ce que fait spécifiquement cette commande :

* **Installation du paquet expo-constants**: Elle télécharge le paquet **expo-constants** depuis le registre npm et l'ajoute à votre projet.
* **Mise à jour du fichier package.json**: Elle ajoute une entrée pour **expo-constants** dans la section **dependencies** de votre fichier **package.json**, ce qui assure que le même paquet peut être installé par d'autres personnes travaillant sur le projet ou lors du déploiement.
* **Installation des dépendances**: Elle installe également toutes les dépendances nécessaires à **expo-constants**.

Après l'installation, vous pouvez utiliser **expo-constants** dans votre projet pour accéder à des informations telles que :

* **Informations sur l'appareil**: Comme le modèle de l'appareil, la version du système d'exploitation.
* **Informations sur l'application**: Comme la version de l'application, le nom du bundle, les paramètres de l'environnement d'exécution.
* **Constantes de configuration**: Comme celles définies dans **app.json** ou **app.config.js**.

Ajouter les imports suivants

import \* as Device from "expo-device";

import \* as Notifications from "expo-notifications";

import Constants from "expo-constants";

On doit ajouter cette fonction dans **HomeScreen**

async function registerForPushNotificationsAsync() {  
 let token;  
 if (Device.***isDevice***) {  
 const { status: existingStatus } = await Notifications.getPermissionsAsync();  
 let finalStatus = existingStatus;  
 if (existingStatus !== 'granted') {  
 const { status } = await Notifications.requestPermissionsAsync();  
 finalStatus = status;  
 }  
 if (finalStatus !== 'granted') {  
 alert('Failed to get push token for push notification!');  
 return;  
 }  
 const response = await Notifications.getExpoPushTokenAsync({  
 projectId: ***Constants***.expoConfig.extra.eas.projectId,  
 });  
 token = response.data;  
 ***console***.log(token);  
  
  
 const id = await ***asyncStorage***.getItem("id");  
 await axios.post(  
 "https://ams.smart-it-partner.com/api/push\_notives",  
 { token: token, user: id },  
 { headers: { 'Content-Type': 'application/ld+json' } }  
 );  
  
  
  
  
 } else {  
 alert('Must use physical device for Push Notifications');  
 }  
  
 if (***Platform***.*OS* === 'android') {  
 Notifications.setNotificationChannelAsync('default', {  
 name: 'default',  
 importance: Notifications.AndroidImportance.*MAX*,  
 vibrationPattern: [0, 250, 250, 250],  
 lightColor: '#FF231F7C',  
 });  
 }  
  
 return token;  
}

**useEffect(() => {**

**registerForPushNotificationsAsync();**

**Notifications.setNotificationHandler({**

**handleNotification: async () => ({**

**shouldShowAlert: true,**

**shouldSetBadge: true,**

**}),**

**});**

**}, []);**

**Push notifications vers lien spécifique**

Ensuite, on doit modifier **registerForPushNotificationsAsync** dans **HomeScreen**

async function registerForPushNotificationsAsync() {  
 let token;  
 if (Device.***isDevice***) {  
 const {status: existingStatus} = await Notifications.getPermissionsAsync();  
 let finalStatus = existingStatus;  
 if (existingStatus !== 'granted') {  
 const {status} = await Notifications.requestPermissionsAsync();  
 finalStatus = status;  
 }  
 if (finalStatus !== 'granted') {  
 alert('Failed to get push token for push notification!');  
 return;  
 }  
 const response = await Notifications.getExpoPushTokenAsync({  
 projectId: ***Constants***.expoConfig.extra.eas.projectId,  
 });  
 token = response.data;  
 ***console***.log(token);  
  
  
 const id = await ***asyncStorage***.getItem("id");  
 await axios.post(  
 "https://ams.smart-it-partner.com/api/push\_notives",  
 {token: token, user: id},  
 {headers: {'Content-Type': 'application/ld+json'}}  
 );  
  
  
 } else {  
 alert('Must use physical device for Push Notifications');  
 }  
  
 if (***Platform***.*OS* === 'android') {  
 Notifications.setNotificationChannelAsync('default', {  
 name: 'default',  
 importance: Notifications.AndroidImportance.*MAX*,  
 vibrationPattern: [0, 250, 250, 250],  
 lightColor: '#FF231F7C',  
 });  
 }  
  
 return token;  
}  
  
useEffect(() => {  
 registerForPushNotificationsAsync();  
  
 // Configure le comportement de la notification lorsque l'application est ouverte  
 Notifications.setNotificationHandler({  
 handleNotification: async () => ({  
 shouldShowAlert: true, // Affiche une alerte même si l'app est ouverte  
 shouldPlaySound: true,  
 shouldSetBadge: false,  
 }),  
 });  
  
 // Gestionnaire pour les interactions avec les notifications  
 const responseSubscription = Notifications.addNotificationResponseReceivedListener(response => {  
 const data = response.notification.request.content.data;  
 ***console***.log(data);  
 // Vérifie si l'application était en premier plan au moment où l'utilisateur a interagi avec la notification  
 if (response.notification.request.trigger.type === 'push' && data.screen && navigation) {  
 // Naviguez vers l'écran spécifié  
 // navigation.navigate(data.screen, data.params);  
  
 navigation.navigate(data.screen, {  
 title: data.title,  
 id: data.id  
  
 })  
 }  
 });  
  
 return () => {  
 responseSubscription.remove();  
 };  
}, [navigation]);

****

***Bon travail !***