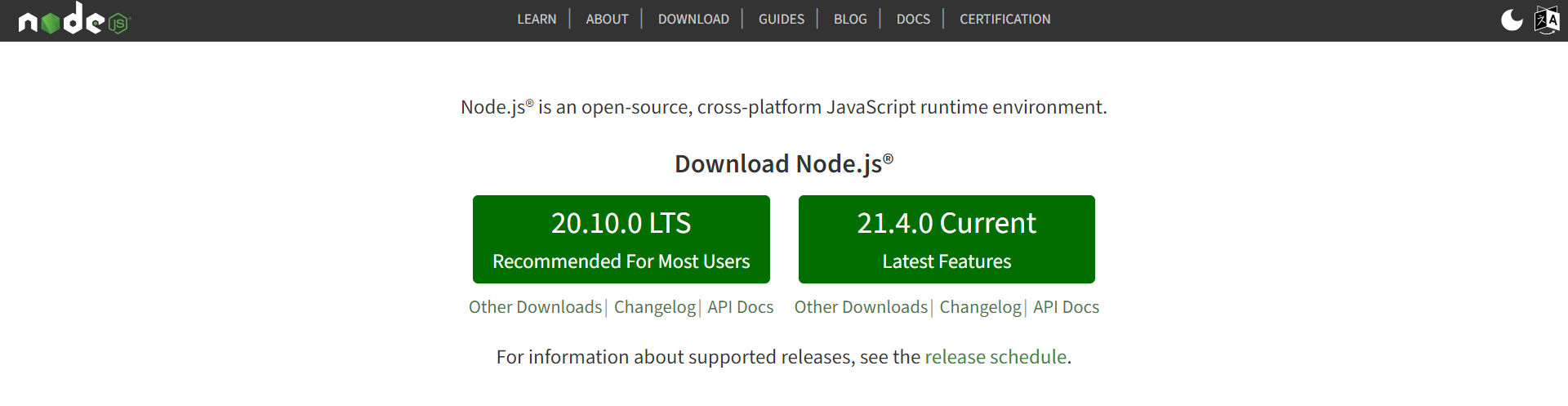
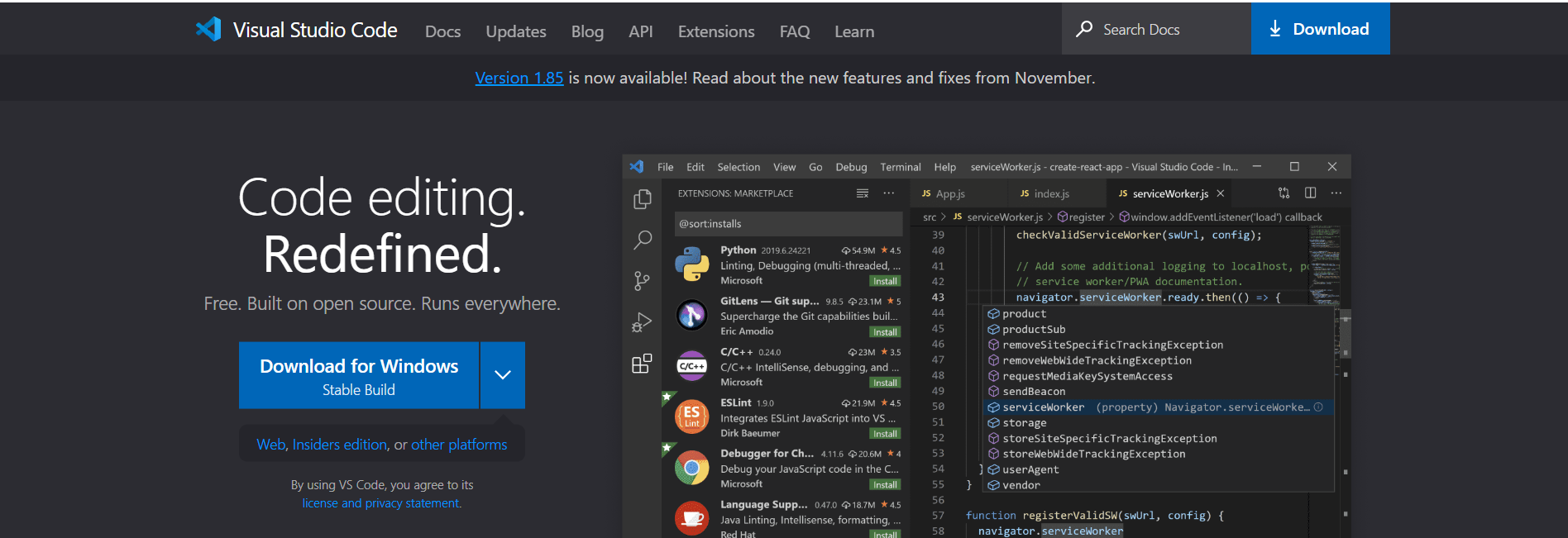
**Guide d’installation de l'environnement React Nativ + android studio avec VsCode**

# Prés requis:

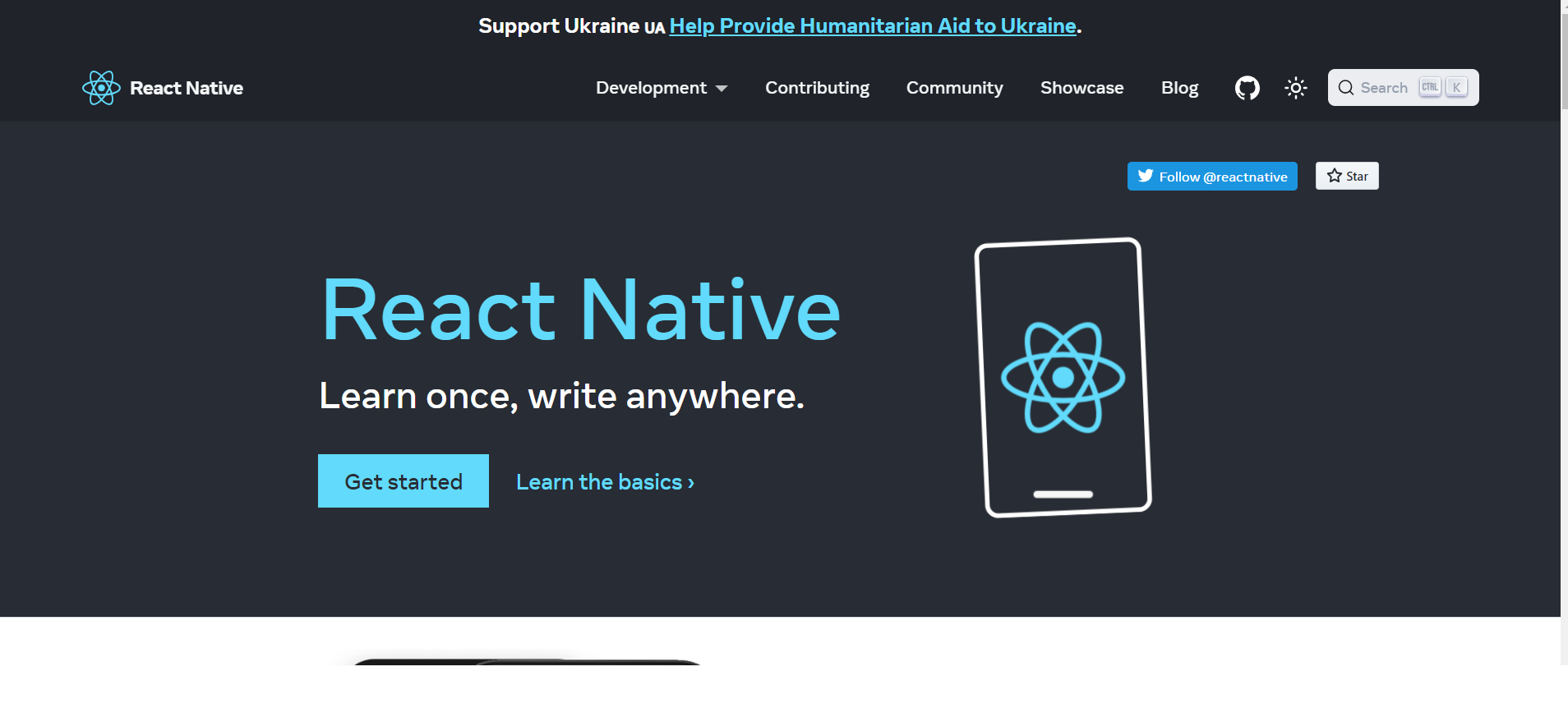
\*Installer Node.JS sur https://nodejs.org/en  


\*Installer VsCode sur https://code.visualstudio.com/

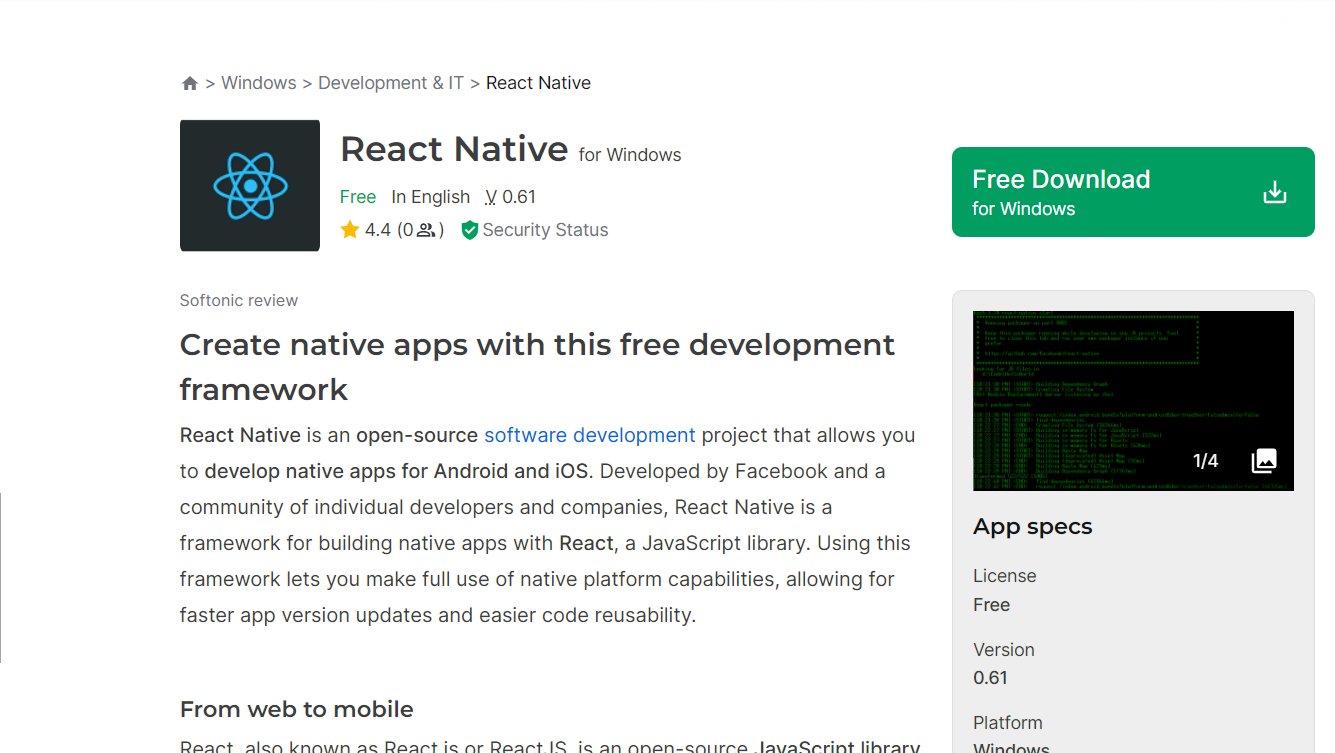


# Installation de React native

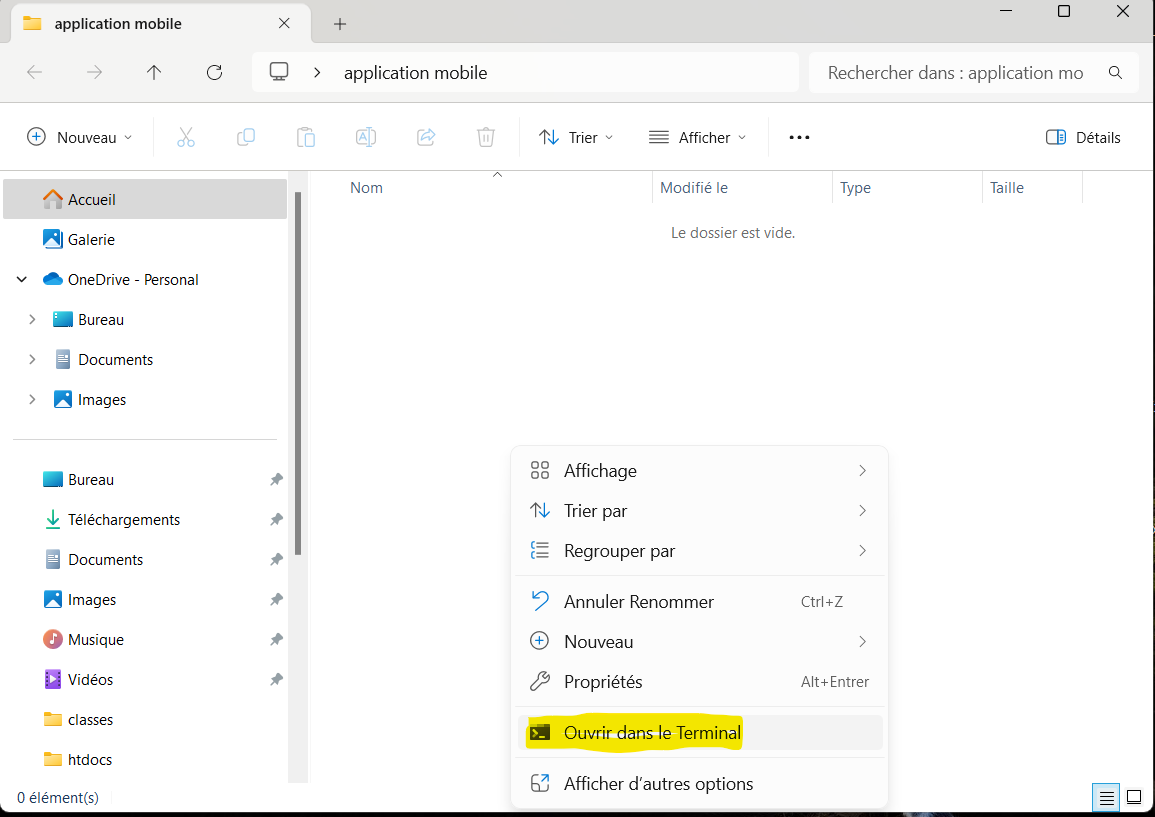
on retrouve toute la documentation nécessaire pour utiliser le framework react nativ sur https://reactnative.dev/



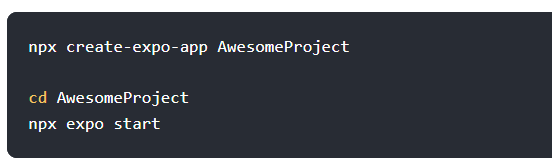
on le télécharge sur <https://react-native.en.softonic.com/>



une fois React Native installer avec NodeJS, on peut aller à l’endroit où l’on veut créer un nouveau projet pour une application mobile, ouvrir un terminal

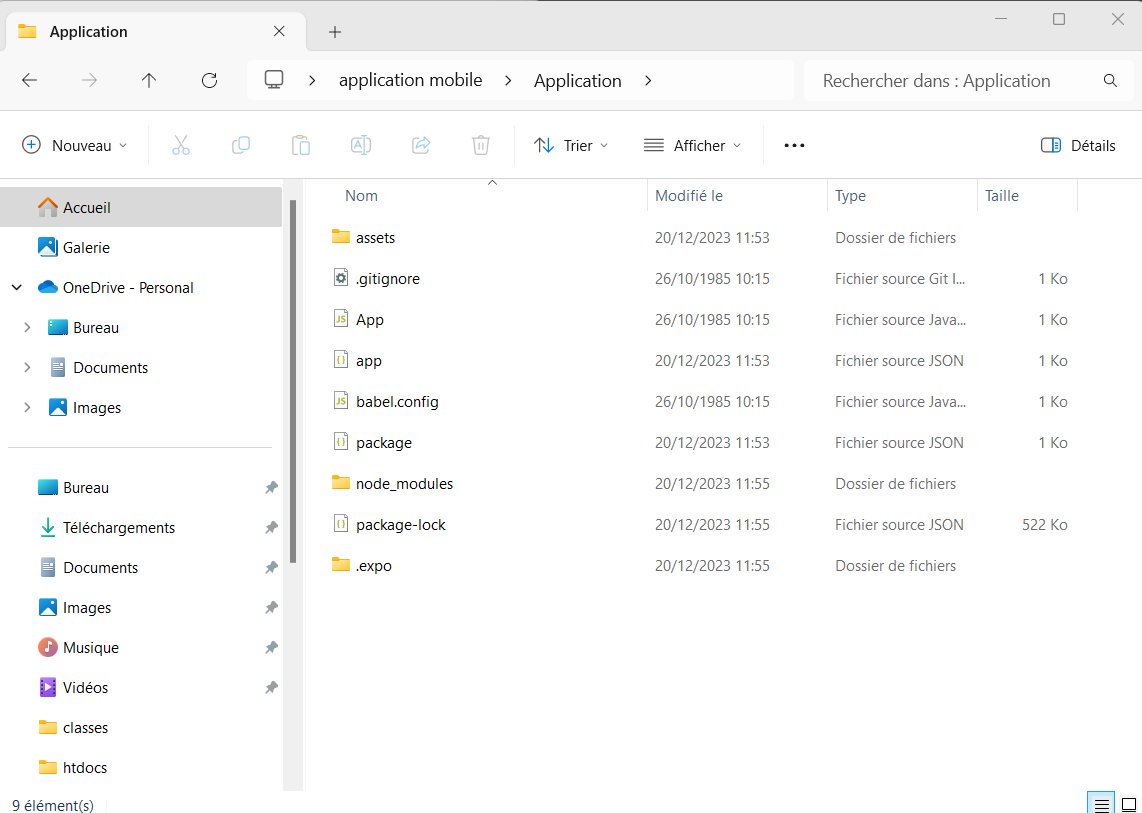


Puis on lance la commande suivante:

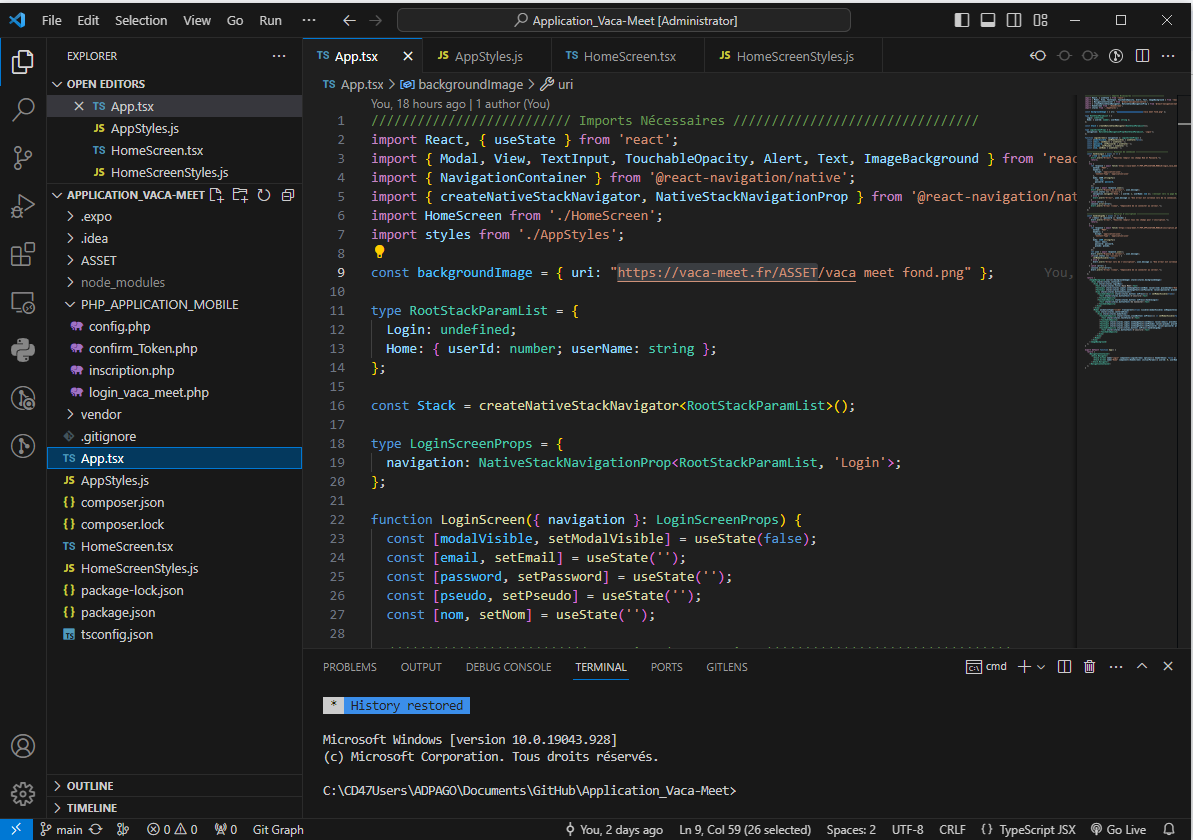


Sur cette commande ‘AwesomePojet’ est remplacé par le nom de notre projet.  
A la suite de cette commande ça va nous créer une structure de dossier avec un environnement pour développer notre application mobile

PHOto dossier



Ensuite j’ouvre le dossier de mon projet avec VsCode



Maintenant on va pouvoir coder notre application dans le fichier App.tsx.

**Et pour gérer les bug et voir en temps réel notre développement on va utiliser android studio ou bien un smartphone (partie 2)**

# Debug avec un smartphone ou bien android studio

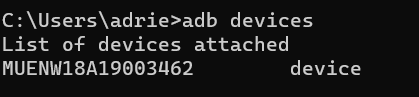
android studio va permettre de gérer des émulateurs mobiles pour avoir un aperçu en temps réel de l’application. On peut aussi utiliser notre téléphone portable. Voici les deux procédures.

## Configuration d’un téléphone comme simulateur de l’application

Voici la procédure pour utiliser son téléphone comme simulateur pour l’application:  
 \*Passer son téléphone en mode développeur (Accédez aux paramètres de l’appareil Android ;Allez dans « À propos du téléphone » ;Tout en bas, tapotez 7 fois « numéro de build » ; )  
 \*Activez le Débogage USB : Sur votre appareil Android,   
\*Connectez l'Appareil à votre Ordinateur : Utilisez un câble USB pour connecter votre appareil Android à l'ordinateur.

\*Télécharger l’application Expo sur le téléphone

\*Vérifiez la Connexion : Dans le terminal, exécutez adb devices pour vérifier si l'appareil est détecté. Si votre appareil est listé, il devrait être prêt à être utilisé avec Expo.



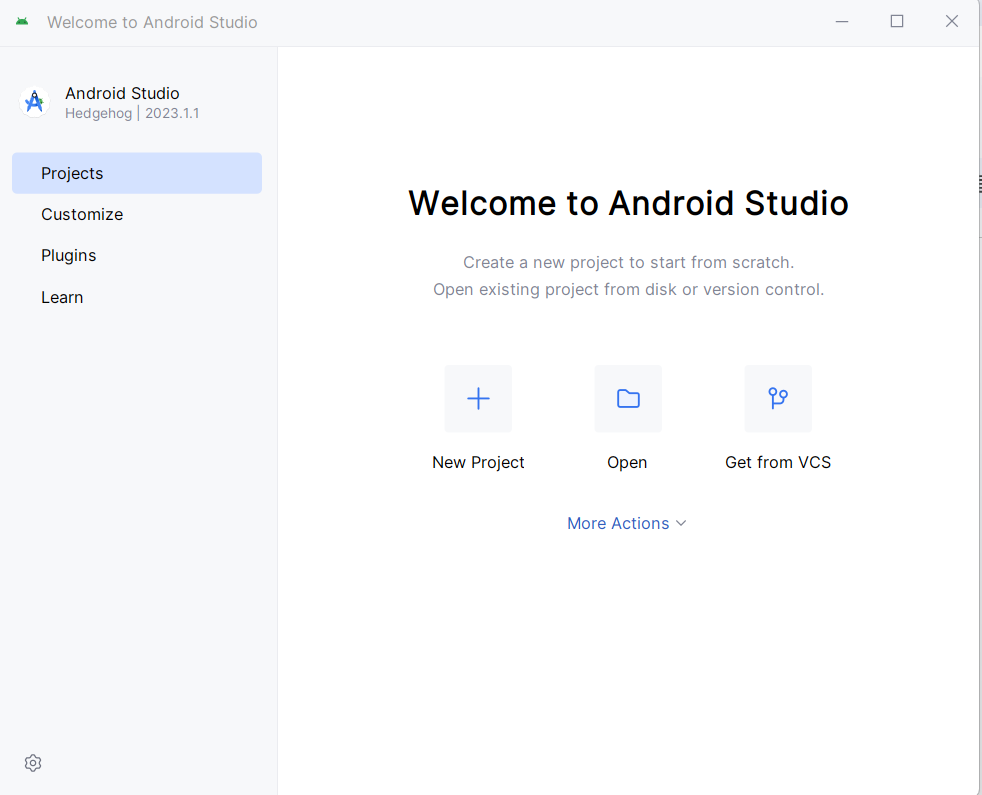
## Configuration d'un Émulateur Android

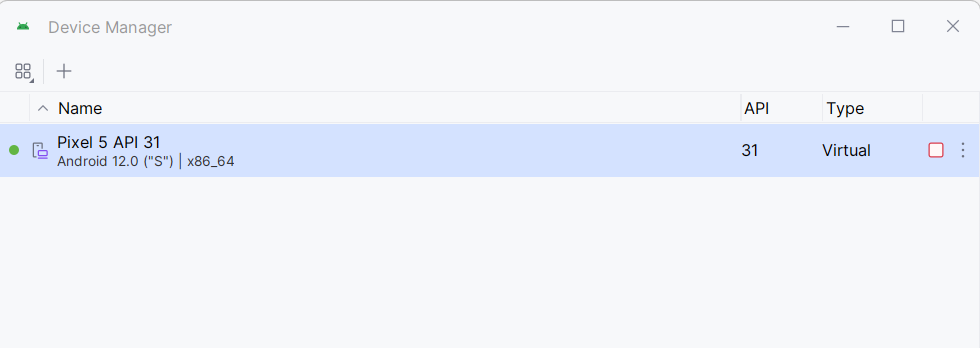
Installez Android Studio : Téléchargez et installez Android Studio, qui comprend l'émulateur Android.

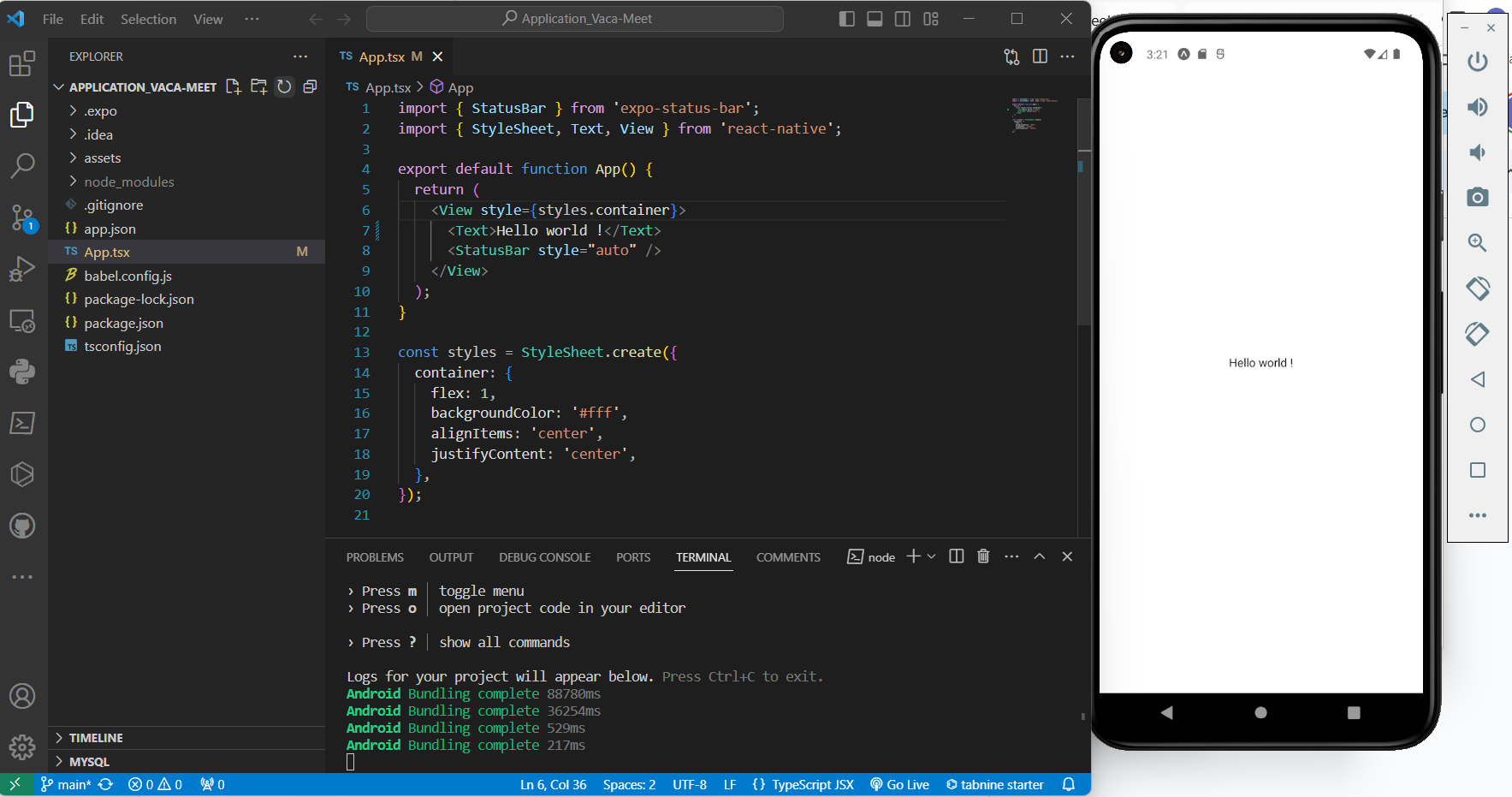
Configurez un Émulateur : Lancez Android Studio, allez dans le menu "AVD Manager" (Android Virtual Device Manager) et configurez un nouvel émulateur. Choisissez une image système et créez l'AVD.

Démarrez l'Émulateur : Depuis l'AVD Manager, démarrez l'émulateur que vous avez configuré.

Vérifiez avec ADB : Une fois l'émulateur en cours d'exécution, exécutez adb devices dans le terminal pour confirmer que l'émulateur est détecté.







Maintenant on peut utiliser le framework react Native avec un aperçu en direct de notre application

Voici en plus des commande et tips d'utilisation de React native importante

# 

# Tips React Native

Mettre à jours les dépendance d’un projet:  
$ npm install

Lancer le projet sur un émulateur android ou smartphone  
$ npm run android

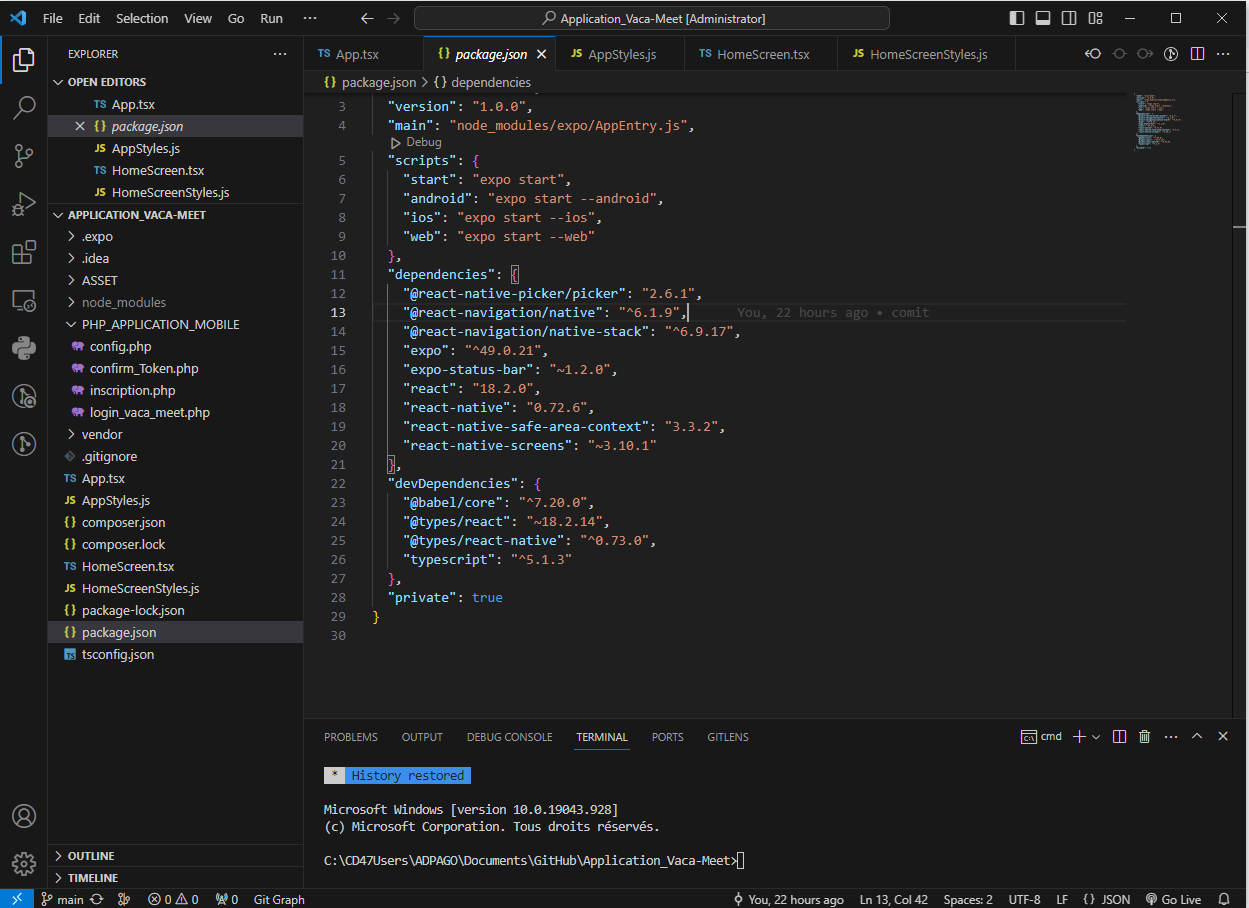
Pour diriger un utilisateur sur une nouvelle page de l’application il faut installer React Navigation:  
$ npm install @react-navigation/native  
$ npm install react-native-screens react-native-safe-area-context

Et si on utilise une pile de navigation (le plus souvent) il faut aussi:

$ npm install @react-navigation/native-stack

**Ensuite il faut envelopper l’application dans un ‘NavigationContainer’**

**Bien faire aussi attention au problème de compatibilité des version dans le fichier package.json**



En cas de modification sur le fichier package pour avoir des version compatible il faut faire:  
$ rm -rf node\_modules  
$ npm cache clean –force  
$ npm install  
  
Ces 3 commande vont vidé le cache des ancienne version et mettre à jours

Installation du package picker pour utiliser les combo:  
$ npm install @react-native-picker:picker

Installation du package autocomplete-input pour avoir un champ de saisie avec autocomplétion

$ npm install react-native-autocomplete-input –save

En cas de conflit avec les différent packages et dépendances:  
Utilisez l’option ‘--force” pour force npm à accepter les dépendances potentiellement incompatibles:  
 $ npm –force

Ou bien utiliser l’option ‘--legacy-peer-deps’ pour ignorer la vérification des dépendances de pair et revenir au comportement de npm version 4 à 6:  
$ npm install –legacy-peer-deps

nettoyage du cache npm:  
$ npm cache clean –force

Installer combo datetime:

$ npm install @react-native-community/datetimepicker

$ npm install react-native-datepicker –save

Naviguer de route:  
$ npm install @react-navigation/native

$ npm install react-native-screens react-native-safe-area-context

si on utilise un stack navigateur:  
$ npm install @react-navigation/native-stack

Installer selection date :  
$ npm install @react-native-community/datetimepicker

Lancer expo sur téléphone :  
$ npm install -g expo-cli

$ expo start

$ npx expo // lancer en local

$ expo upgrade

Réinstaller les dependance  
$ npm install --legacy-peer-deps

Forcer upgrade manuellement

$ npm install [expo@^50.0.0](mailto:expo@%5e50.0.0)

$ npm audit fix --force

$ npm install

Mettre les dépendance à jours pour faire passer le update

$ npm install expo@^50.0.0 --legacy-peer-deps

$ npm audit fix –force

$ npm cache clean –force

//Pour mon cas à mois

$ expo install @react-native-community/datetimepicker @react-native-picker/picker expo-status-bar react-native@0.73.6 react-native-safe-area-context@4.8.2 [react-native-screens@3.29.0](mailto:react-native-screens@3.29.0)

$ npm install --save-dev @types/react-native-picker\_\_picker

Suprimer tous les module puis toutes les réinstaller

$ Remove-Item -Recurse -Force node\_modules

$ npm install