

**DOCUMENT TECHNIQUE DE LA FORMATION DE REACT NATIVE**

**Réalisé par : HAFDI AHMED Encadré par : Mr SERRAJ Mohammed Amine**

Table des matières

[Table des Figures 6](#_Toc86646453)

[**Liste des Tableaux :** 7](#_Toc86646454)

[**CHAPITRE 1 : SET UP POUR L’ENVIRONEMMENT WINDOWS** 9](#_Toc86646455)

[**Configuration pour Windows :** 9](#_Toc86646456)

[**CHAPITRE 2 : BONJOUR LE MONDE** 11](#_Toc86646457)

[**Bonjour Tout le monde!** 11](#_Toc86646458)

[**Chapitre 3 : Coiffant (stylesheet)** 13](#_Toc86646459)

[**Introduction :** 13](#_Toc86646460)

[**Syntaxe :** 13](#_Toc86646461)

[**Remarques :** 13](#_Toc86646462)

[**Examples :** 13](#_Toc86646463)

[**Chapitre 6 : Composants** 15](#_Toc86646464)

[**Examples :** 15](#_Toc86646465)

[Composant de base : 15](#_Toc86646466)

[Composant avec état 15](#_Toc86646467)

[Composant sans état ( composant without state) 16](#_Toc86646468)

[**Chapitre 7 : Disposition** 17](#_Toc86646469)

[**Flexbox** : 17](#_Toc86646470)

[**Chaptire 8 : States ( les états)** 19](#_Toc86646471)

[**Syntaxe :** 19](#_Toc86646472)

[**Examples :** 19](#_Toc86646473)

[**SetState :** 19](#_Toc86646474)

[**Chapitre 10 : Exécuter une application sur l’appareil (version android ) & Expo** 21](#_Toc86646475)

[**Exécuter sur l’appareil :** 21](#_Toc86646476)

[**Chapitre 11 : Images** 23](#_Toc86646477)

[Examples : 23](#_Toc86646478)

[Module d’image 23](#_Toc86646479)

[ Image locale  et Image externe : 23](#_Toc86646480)

[**Source d’image conditionnelle :** 24](#_Toc86646481)

[**Screen :** 24](#_Toc86646482)

[24](#_Toc86646483)

[25](#_Toc86646484)

[**Chapitre 12 : Instructions en ligne de commande** 25](#_Toc86646485)

[**Examples :** 25](#_Toc86646486)

[**Vérifier la version** 25](#_Toc86646487)

[**Examples de sortie** 25](#_Toc86646488)

[**Chapitre 13 : Le débogage** 26](#_Toc86646489)

[Syntaxe : 26](#_Toc86646490)

[Le débogage avec Expo (Exponent ) et VSCode : 26](#_Toc86646491)

[ C’est quoi Exponent ? 26](#_Toc86646492)

[Etapes de débogage par Expo & VScode : 26](#_Toc86646493)

[**Chapitre 14 : Le Routage** 28](#_Toc86646494)

[Exemples : 28](#_Toc86646495)

[Stack Composant : 28](#_Toc86646496)

[Top Tabs : 30](#_Toc86646497)

[Bottom Tabs 31](#_Toc86646498)

[**Chapitre 15 : Liaison de l’API native** 32](#_Toc86646499)

[Introduction : 32](#_Toc86646500)

[Exemples : 32](#_Toc86646501)

[Liens sortants 32](#_Toc86646502)

[Liens entrants 32](#_Toc86646503)

[**Chapitre 16 : ListView** 34](#_Toc86646504)

[FlatList 35](#_Toc86646505)

[SectionList 36](#_Toc86646506)

[Screen : 36](#_Toc86646507)

[36](#_Toc86646508)

[Map Array 38](#_Toc86646509)

[**Chapitre 17 : Input Data** 39](#_Toc86646510)

[Exemples : 39](#_Toc86646511)

[TextInput : 39](#_Toc86646512)

[**Chapitre 18 : Modal** 40](#_Toc86646513)

[Introduction : 40](#_Toc86646514)

[Paramètres : 40](#_Toc86646515)

[**Chapitre 19 : Module Plateforme** 42](#_Toc86646516)

[Introduction : 42](#_Toc86646517)

[Exemple : 42](#_Toc86646518)

[**Chapitre 20 : Notification push** 43](#_Toc86646519)

[Expo Push Notifications Tool : 43](#_Toc86646520)

[**Chapitre 21 : Polices Personnalisés** 45](#_Toc86646521)

[Exemples : 45](#_Toc86646522)

[**Chapitre 22 : RefreshControl** 46](#_Toc86646523)

[**Chapitre 23 : Requêtes http** 47](#_Toc86646524)

[Syntaxe : 47](#_Toc86646525)

[Exemple : Posting application avec json server 47](#_Toc86646526)

[**Chapitre 24 : Test d’unité** 49](#_Toc86646527)

[Introduction : 49](#_Toc86646528)

[Exemples : 49](#_Toc86646529)

[**Chapitre 25 : Mobx** 50](#_Toc86646530)

[Introduction : 50](#_Toc86646531)

[C’est quoi Mobx : 50](#_Toc86646532)

[Les Principes de Mobx : 51](#_Toc86646533)

[Exemple 1 : une application simple des taches avec state management useState () 51](#_Toc86646534)

[Exemple 2 :une application simple des taches avec state management Mobx 53](#_Toc86646535)

[Exemple 3 : Exemple 2 :une application simple des taches avec state management MST (Mobx State Tree) 54](#_Toc86646536)

[54](#_Toc86646537)

[C’est quoi MST : 54](#_Toc86646538)

# Table des Figures

[Figure 1:react native logo 9](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646385)

[Figure 2: Hello World 11](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646386)

[Figure 3: Hello World Screen 12](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646387)

[Figure 4: basic component 15](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646388)

[Figure 5: component avec état 15](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646389)

[Figure 6: composant sans état 16](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646390)

[Figure 7: flexdirection : 'row' 17](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646391)

[Figure 8 : flexdirection row screen 18](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646392)

[Figure 9: counter add 19](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646393)

[Figure 10: counter add screen 20](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646394)

[Figure 11: adb devices 21](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646395)

[Figure 12: expo start server 22](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646396)

[Figure 13: expo run on android emulator 22](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646397)

[Figure 14: image locale et externe 23](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646398)

[Figure 15: image locale & externe screen 23](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646399)

[Figure 16: image conditionnelle 24](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646400)

[Figure 17: image conditionnelle screen 25](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646401)

[Figure 18: react native check version 25](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646402)

[Figure 19: expo logo 26](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646403)

[Figure 20: react native tools 26](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646404)

[Figure 21 : debug in exponent 27](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646405)

[Figure 22: debug 27](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646406)

[Figure 23 : Navigator 28](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646407)

[Figure 24 : MyStack 28](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646408)

[Figure 25: First Screen 29](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646409)

[Figure 26: Second Screen 29](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646410)

[Figure 27: Top Tabs 30](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646411)

[Figure 28: Tab Tops screen 30](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646412)

[Figure 29: Bottom tabs 31](#_Toc86646413)

[Figure 30: lien sortant 32](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646414)

[Figure 32 : lien entrant 32](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646415)

[Figure 31: lien sortant screen 32](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646416)

[Figure 33: Linking 32](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646417)

[Figure 34: lien entrant 33](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646418)

[Figure 35:flatList 35](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646419)

[Figure 36: flatlist users 35](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646420)

[Figure 37:SectionList 36](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646421)

[Figure 38: SectionList Screen 36](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646422)

[Figure 39:map array view 38](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646423)

[Figure 40: TextInput 39](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646424)

[Figure 41: add user screen 39](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646425)

[Figure 42: Modal 40](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646426)

[Figure 43: Modal 1 41](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646427)

[Figure 44: modal 2 41](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646428)

[Figure 45: Platforme 42](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646429)

[Figure 46: Platform Screen 42](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646430)

[Figure 47: Push Notification 43](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646431)

[Figure 48: expo push notification tool 43](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646432)

[Figure 49: notification screen 44](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646433)

[Figure 50: fonts 45](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646434)

[Figure 51: fonts exemples 45](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646435)

[Figure 52: refresh control 46](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646436)

[Figure 53:Refresh control screen 46](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646437)

[Figure 54: fetch1 47](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646438)

[Figure 55: fetch 2 47](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646439)

[Figure 56: Test for buttons component 49](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646440)

[Figure 57: buttonns.test.js 49](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646441)

[Figure 58: Test passed 49](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646442)

[Figure 59:Mobx + React Native 50](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646443)

[Figure 60: Mobx state architecture 50](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646444)

[Figure 61: Task with UseState 51](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646445)

[Figure 62: Tasks UseState screen 52](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646446)

[Figure 63: Tasks With Mobx 53](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646447)

[Figure 64: Mobx Tasks Screen 53](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646448)

[Figure 65: MST RN 54](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646449)

[Figure 66: Book Store 54](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646450)

[Figure 67: BookView 1 55](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646451)

[Figure 68: BookView 2 55](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86646452)

# **Liste des Tableaux :**

[Tableau 1: Modal parameters 40](#_Toc86316847)

# **CHAPITRE 1 : SET UP POUR L’ENVIRONEMMENT WINDOWS**

**React Native** vous permet de créer des applications mobiles en utilisant uniquement JavaScript. Il utilise la même conception que React, vous permettant de composer une interface utilisateur mobile riche à partir de composants déclaratifs. Avec React Native, vous ne créez pas une «application Web mobile», une «application HTML5» ou une «application hybride».

Figure 1:react native logo

Vous construisez une véritable application mobile qui ne se distingue pas d'une application créée avec Objective-C ou Java. React Native utilise les mêmes éléments fondamentaux que les applications iOS et Android classiques. Vous venez de mettre ces blocs de construction ensemble en utilisant JavaScript et React.

Il est open-source et maintenu par Facebook.

• Site Internet (https://reactnative.dev/)

• Documentation (https://reactnative.dev/docs/getting-started)

• GitHub Repository(<https://github.com/facebook/react-native>)

## **Configuration pour Windows :**

**Remarque:** vous ne pouvez pas développer d’applications réactives pour iOS sur Windows, mais uniquement des applications Android réactives.

**Outils / Environnement**

* Windows 10
* outil de ligne de commande (par exemple, ligne de commande Powershell ou Windows)
* Chocolaté ( étapes pour configurer via PowerShell via ce lien https://blog.fbalashov.com/2016/07/react-native-android-apps-on-windows.html#setup-choco)
* Le JDK (version 8)
* Studio Android Une machine Intel avec la technologie de virtualisation activée pour HAXM (facultatif, nécessaire uniquement si vous souhaitez utiliser un émulateur)

1. **Configurez votre machine pour réagir au développement natif :**

Démarrez la ligne de commande en tant qu'administrateur exécutez les commandes suivantes:

***choco install nodejs.install***

***choco install python2***

Redémarrez la ligne de commande en tant qu'administrateur pour pouvoir exécuter npm

***npm install -g react-native-cli***

1. **Définissez vos variables d'environnement :**

Ouvrez la fenêtre Variables d'environnement en naviguant vers: [Clic droit] Menu "Démarrer" -> Système -> Paramètres système avancés -> Variables d'environnement Dans la section inférieure, recherchez la variable système "Path" et ajoutez l'emplacement d'installation de react-native à l'étape 1. Si vous n'avez pas ajouté de variable d'environnement ANDROID\_HOME, vous devrez également le faire ici. Dans la fenêtre "Variables d'environnement", ajoutez une nouvelle variable système nommée "ANDROID\_HOME" et la valeur correspondant au chemin d'accès à votre SDK Android. Redémarrez ensuite la ligne de commande en tant qu'administrateur pour pouvoir y exécuter des commandes réactives.

1. **Créez votre projet**

Créez votre projet En ligne de commande, accédez au dossier dans lequel vous souhaitez placer votre projet et exécutez la commande suivante:

**react-native init ProjectName**

1. **Lancez votre projet**

Démarrez un émulateur depuis Android Studio Accédez au répertoire racine de votre projet en ligne de commande et exécutez-le:

***cd ProjectName***

***react-native run-android***

# **CHAPITRE 2 : BONJOUR LE MONDE**

On crée un projet avec la commande :

***expo init Hello World***

on lance le project avec la commande : (on lance la commande sur le path du projet )

***expo start***

on ouvrit le fichier App.js

Après on écrit *<Text>Bonjour Tout le Monde </Text> :*

On rafraichit le fichier avec la commande « control + S »

Et voila Félicitations! Vous avez écrit avec succès votre premier Hello World!

### **Bonjour Tout le monde!**

****

Figure 2: Hello World

Vous devriez voir Hello World! écrit à l'écran!

****

Figure 3: Hello World Screen

# **Chapitre 3 : Coiffant (stylesheet)**

### **Introduction :**

Les styles sont définis dans un objet JSON avec des noms d'attributs de style similaires, comme dans CSS. Un tel objet peut soit être mis en ligne dans le style prop d'un composant, soit être transmis à la fonction StyleSheet.create(StyleObject) et être stocké dans une variable pour un accès en ligne plus court en utilisant un nom de sélecteur similaire à une classe. en CSS.

### **Syntaxe :**

* <Component style={styleFormaStyleSheet }/>
* <Component style={styleObject }/>
* <Component style={style1,style2 }/>

### **Remarques :**

La plupart des styles React Native sont leurs formulaires CSS, mais dans un cas camel. Ainsi, la text-decoration devient textDecoration . Contrairement aux CSS, les styles ne sont pas hérités. Si vous souhaitez que les composants enfants héritent d'un certain style, vous devez le fournir explicitement à l'enfant. Cela signifie que vous ne pouvez pas définir une famille de polices pour une View entière. La seule exception à cette règle est le composant Text : les Text imbriqués héritent de leurs styles parents.

### **Examples :**

Il y a plusieurs manière pour définir un style :

* Style en ligne :

***<Text style={{color : ‘red}} >Red Text </Text>***

* Styling à l’aide d’une feuille de style :

***Const styles = StyleSheet.create {{***

***red : {***

***color : ‘red’***

***},***

***}}***

* Styling à l’aide de plusieurs styles :

const styles = StyleSheet.create({

red: { color: 'red' },

greenUnderline: { color: 'green',

textDecoration: 'underline' },

big: { fontSize: 30 } });

//code of the class

<Text style={{styles.red,styles.big > Big red </Text>

<Text style={{styles.red,styles.greenUnderline > Big red </Text>

# **Chapitre 6 : Composants**

**Examples :**

### Composant de base :



Figure 4: basic component

Composant avec état

Ces composants auront des états changeants.(componant with states)

On utiliser un constructor pour créer un état / state car on utilise ici une Classe comme un composant



Figure 5: component avec état

### Composant sans état ( composant without state)

(ce sont les fonctions , funtional componants are stateless , class componants are statefull )

Figure 6: composant sans état

# **Chapitre 7 : Disposition**

Examples :

### **Flexbox** :

Flexbox est une mode de mise en page permettant la disposition des éléments sur une page de manière à ce que les éléments se comportent de manière prévisible lorsque la mise en page doit

Prendre en charge différents tailles d’écran et différents périphériques d’affichage .par défaut ,

Flexbox représente les enfants (childreen ) sous forme d’une colonne (column), Mais on peut le changer en utilisant « flexDirection : ‘row » dans le styling .

**flexDirection**

Figure 7: flexdirection : 'row'

**screen :**

****

Figure 8 : flexdirection row screen

Ila ya plusieurs possibilité pour changer vers une colonne inversé ou ligne inversé (flexDirection : ‘row-reverse ‘ or ‘column-reverse ‘)

# **Chaptire 8 : States ( les états)**

### **Syntaxe :**

* Void **setState** (fonction |object nextState , [rappel de fonction])

### **Examples :**

Pour créer des applications qui contient par exemple des composants qui ont besoin d’étre changé instantanément et étre affiché sur l’écran , alors React Native offre les états ou states ,

Pour modifer la vue d’une application , on peut utiliser **setState** ,setState effectue une fusion superficielle entre l’état nouveau et précédent et déclaneche un re-render du composant .

Pour bien expliquer setState et comment ça fonctionne je vais prendre l’exemple d’un compteur avec button

### **SetState :**



Figure 9: counter add

**Screen :**

****

Figure 10: counter add screen

# **Chapitre 10 : Exécuter une application sur l’appareil (version android ) & Expo**

### **Exécuter sur l’appareil :**

1. Adb devices ( cette commande pour afficher tous les appareil /emulator connectés à l’ordinateur



Figure 11: adb devices

1. emulator -avd Pixel\_XL\_API\_30 ( lancer l’emulator /simulateur)
2. react-native run-android ( exécuter le projet sur le simulateur android)
3. pour Expo (j’utilise expo sur cette documentation )
4. expo start ( on lance cette commande sur la destination du projet pour lancer le serveur d’Expo)

Figure 12: expo start server

1. On clique sur ‘ run on Android device/emulator’ :

Figure 13: expo run on android emulator

# **Chapitre 11 : Images**

### Examples :

### Module d’image

On peut importer Image à partir du package « react-native »

* Image locale  et Image externe :

Figure 14: image locale et externe

* **Screen :**

Figure 15: image locale & externe screen

****

**Source d’image conditionnelle :**

Figure 16: image conditionnelle

**Screen :**

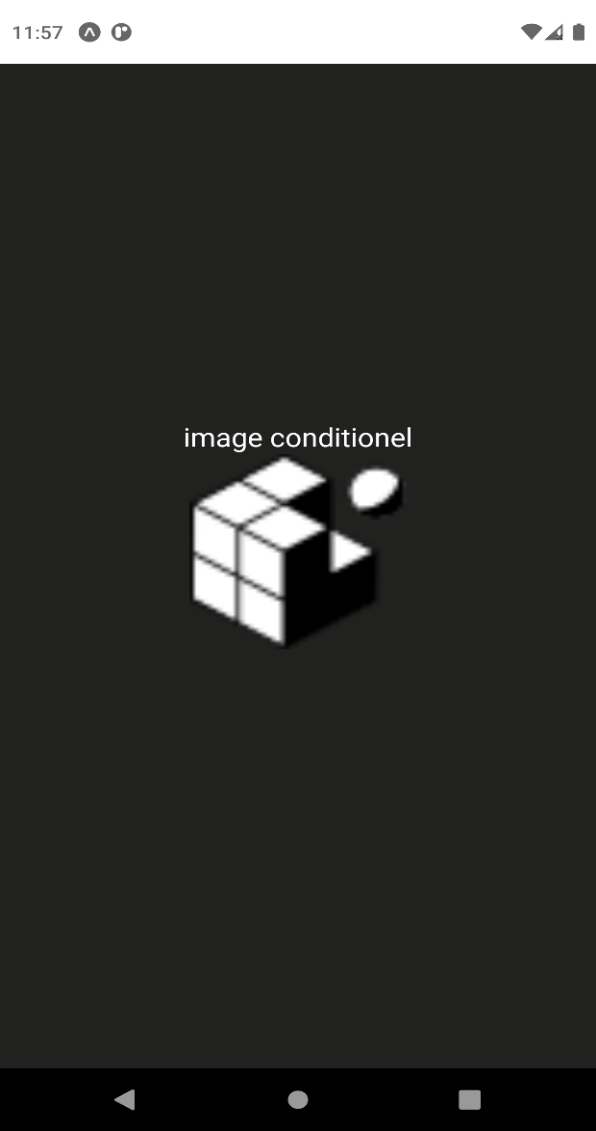
****

Figure 17: image conditionnelle screen

# **Chapitre 12 : Instructions en ligne de commande**

### **Examples :**

### **Vérifier la version**

* **$ react-native -v**

### **Examples de sortie**

Figure 18: react native check version

Pour Initialiser

* **$ react-native init MyFirstProjectRN**
* **$ expo init MyFirstProjectEXPO**

Pour courir pour Android

* **$ cd MyFirstProjectRN**
* **$ react-native run-android**

# **Chapitre 13 : Le débogage**

### Syntaxe :

* Débogueur

### Le débogage avec Expo (Exponent ) et VSCode :

### C’est quoi Exponent ?

**Exponent /Expo** est un framework et une plateforme pour les applications de React . c’est un ensemble d’outils et services construits autour de React Native et des plate-formes natives qui aident à développer , créer , déployer et itérer rapidement sur ios ,android et Web à partir de la même base de code JavaScript/TypeScript .

Figure 19: expo logo

### Etapes de débogage par Expo & VScode :

1. Installation de l’extension « React Native Tools » sur VSCode

Figure 20: react native tools



1. Après l’ajout de la configuration « Exponent » sur le fichier launche.json dand le dossier. Vscode



Figure 21 : debug in exponent

1. On lance le debug de l’application :



Figure 22: debug

# **Chapitre 14 : Le Routage**

Introduction :

Le routage ou la navigation permet de naviguer entre différents écrans . le routage est très important pour les applications mobile car il fournit un contexte à l’utilisateur sur l’endroit ou il se trouve , il se déplace entre les écrans et le fenêtres .

### Exemples :

StepUp Préparation :

Installation des modules :

* npm install @react-navigation/native @react-navigation/native-stack
* expo install react-native-screens react-native-safe-area-context

(c’est mieux d’utiliser **yarn** cli )

### Stack Composant :

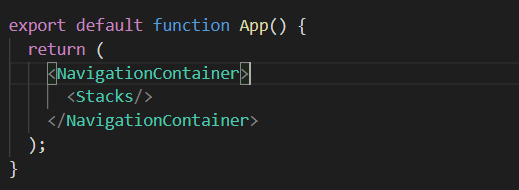
Exemple de navigation entre deux screens :

Figure 23 : Navigator

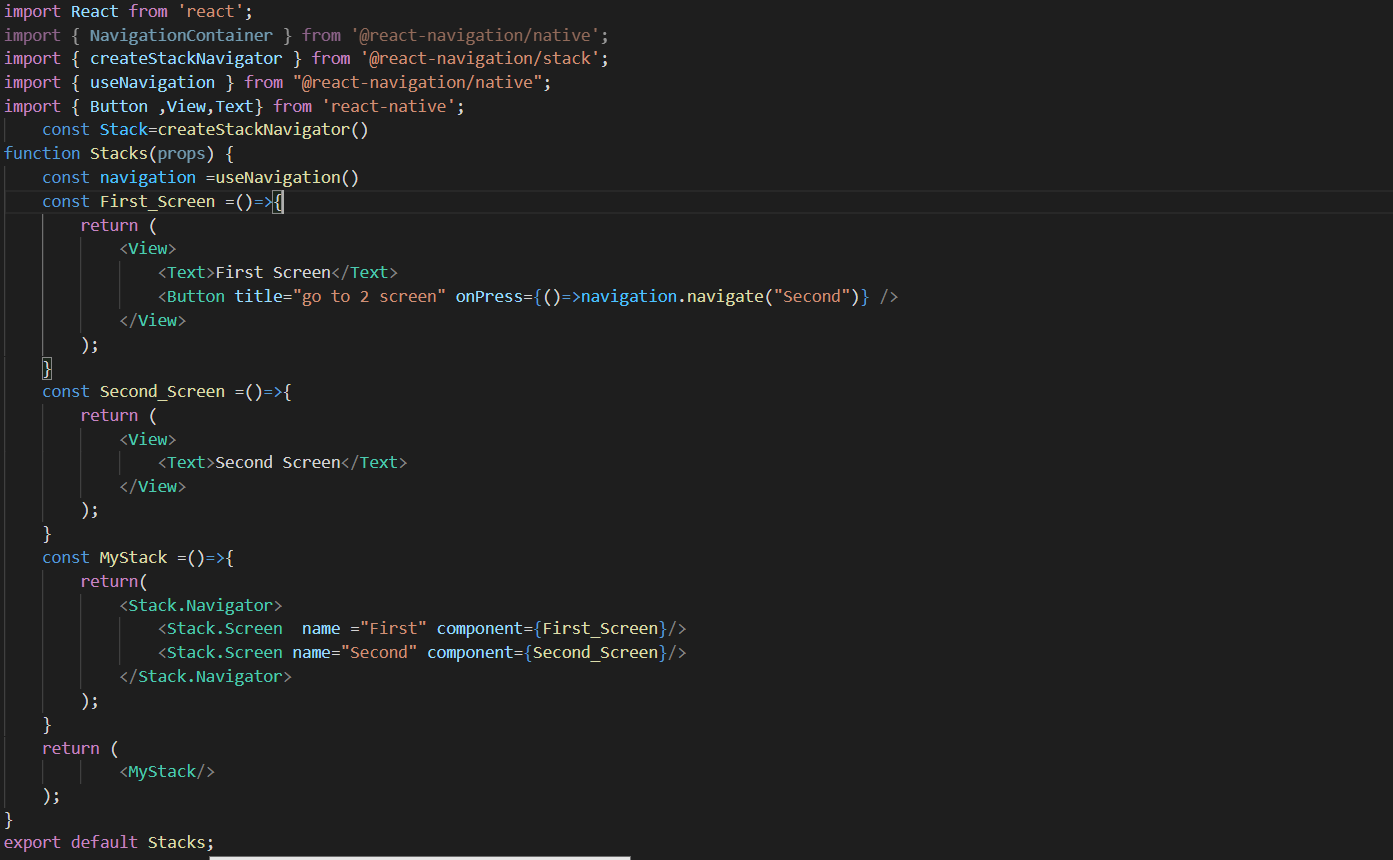


Figure 24 : MyStack

**Screens :**

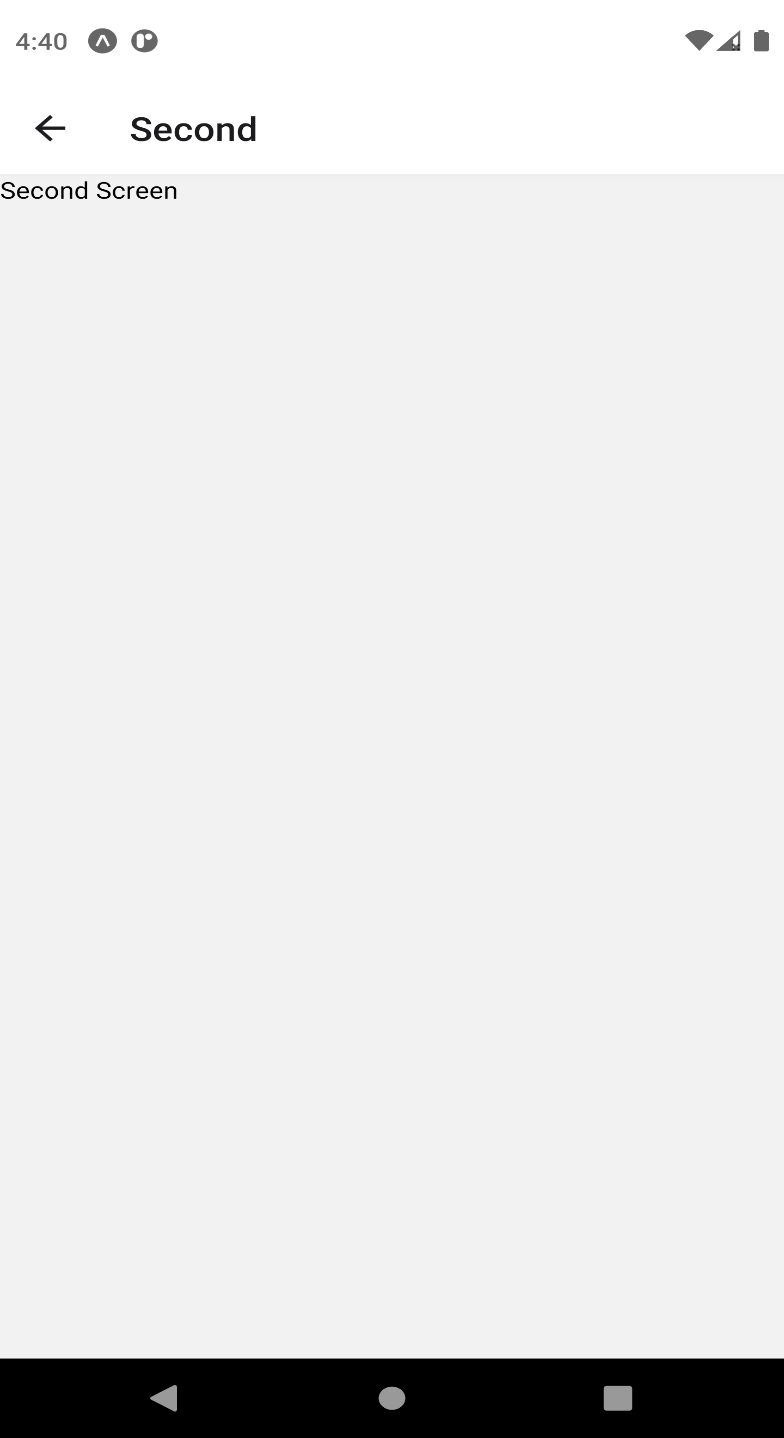
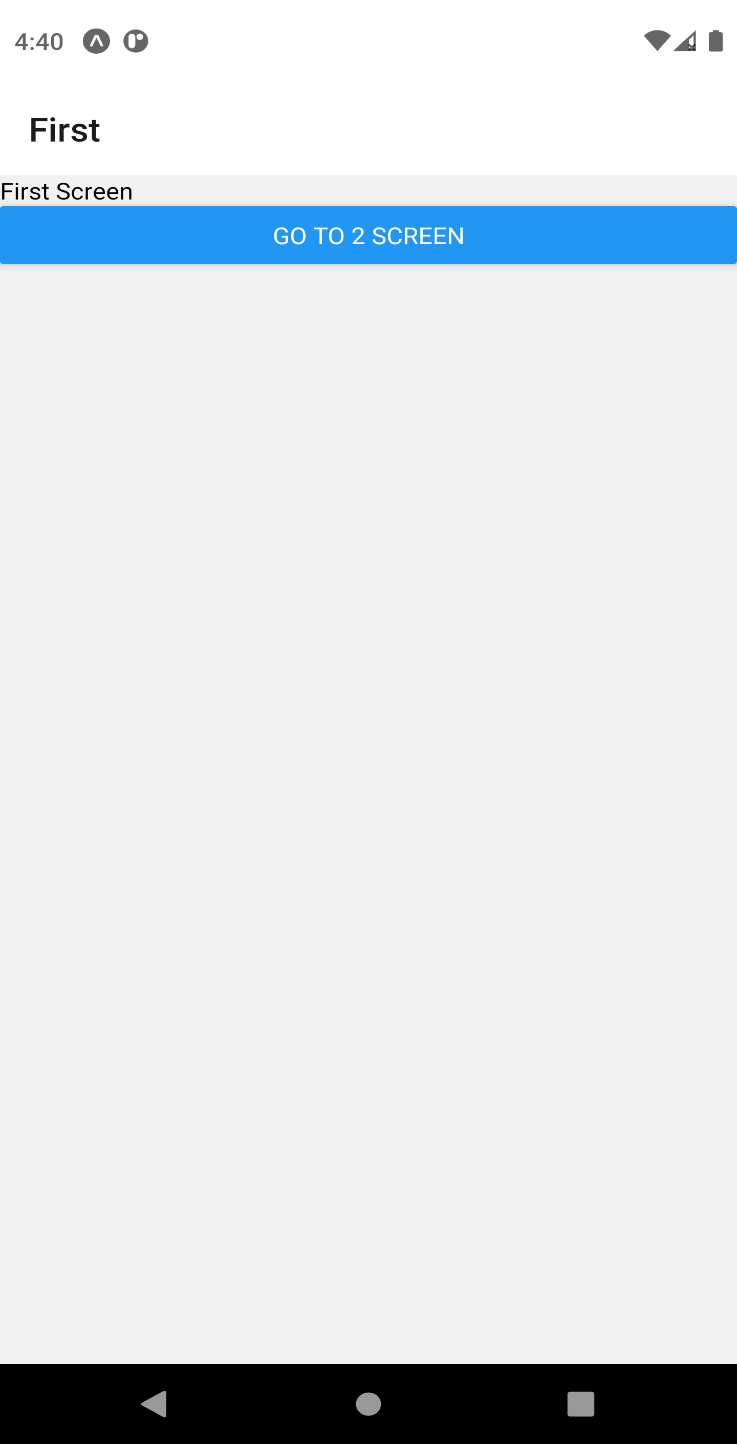
****

Figure 25: First Screen

Figure 26: Second Screen

### Top Tabs :

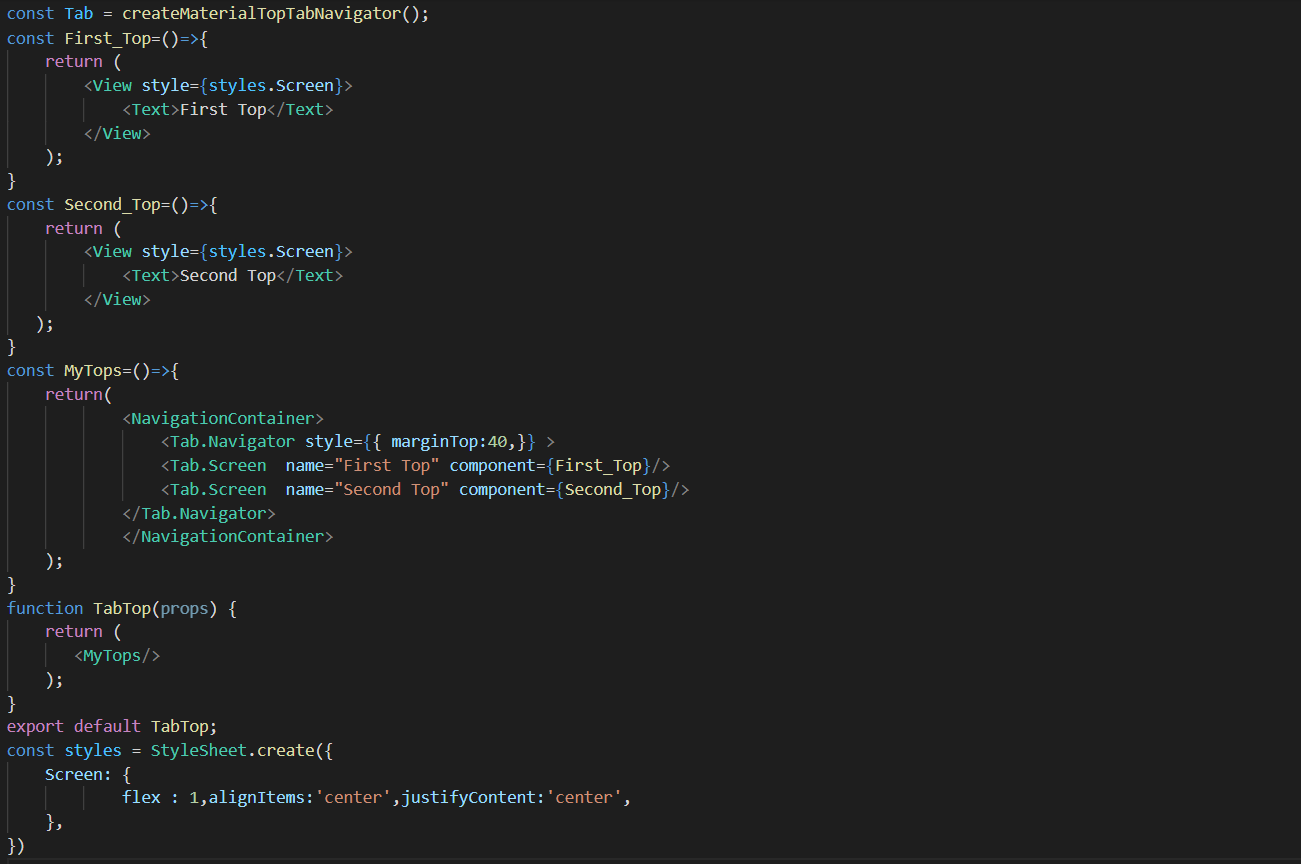


Figure 27: Top Tabs

Screen :

Figure 28: Tab Tops screen

### Bottom Tabs



Figure 29: Bottom tabs

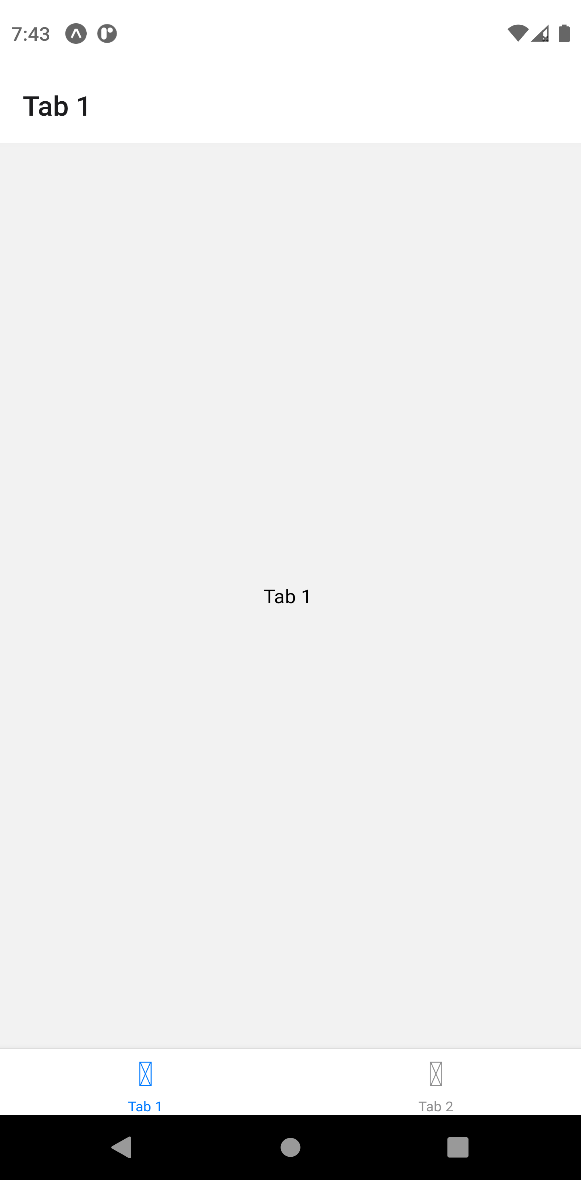
Screen :

Figure : Bottom tabs1

# **Chapitre 15 : Liaison de l’API native**

### Introduction :

L’api de liaison permet d’envoyer et de recevoir des liens entre les applications . par exemple ouvrir l’application téléphone avec le numéro composé ou Google Maps .on peut également utiliser **Linking**

Pour répondre aux liens externes depuis d’autres applications .

Pour utiliser ‘Linking ‘ on doit l’importer depuis react-native.

### Exemples :

### Liens sortants

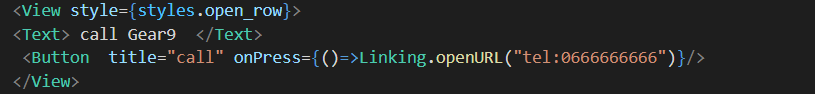


Figure 31: lien sortant

### Liens entrants

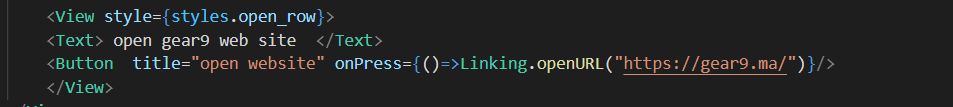
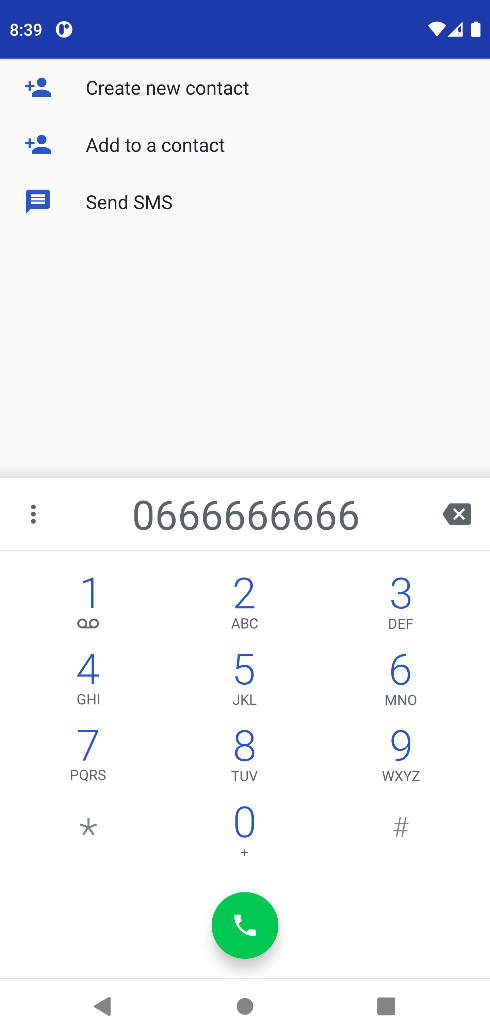


Figure 32 : lien entrant

Screens :

Figure 33: lien sortant screen

Figure 34: Linking

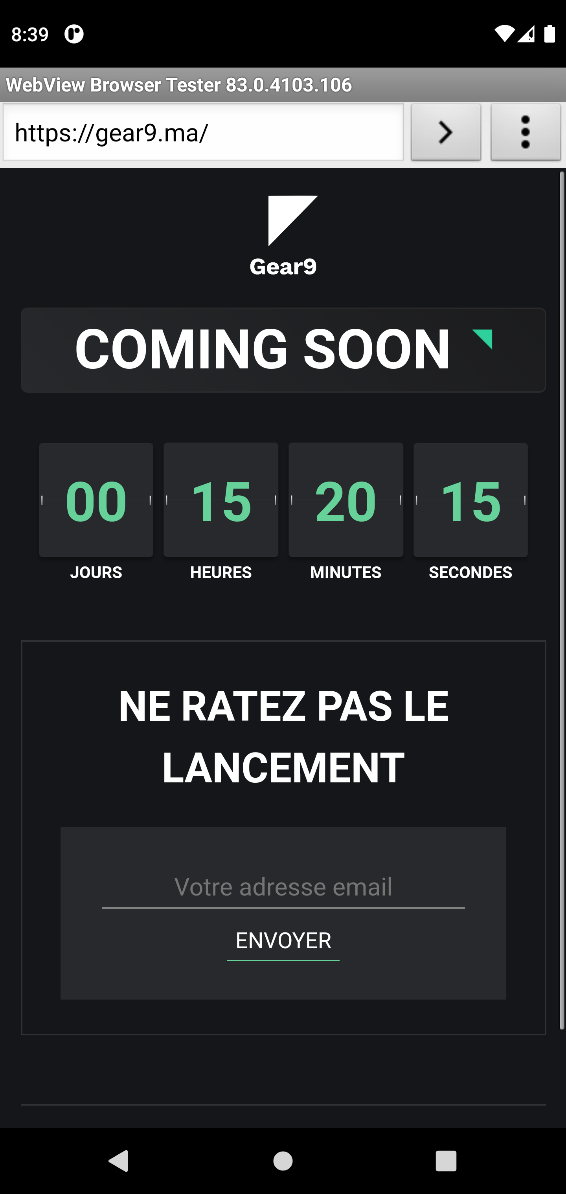


Figure 35: lien entrant

# **Chapitre 16 : ListView**

Exemples :

ListView est un composant cançu pour un affichage efficace des listes de données changeanyes défilant verticalement .il y a plusieurs de types pour l’utilisation de listview FlatList

Figure 36:flatList

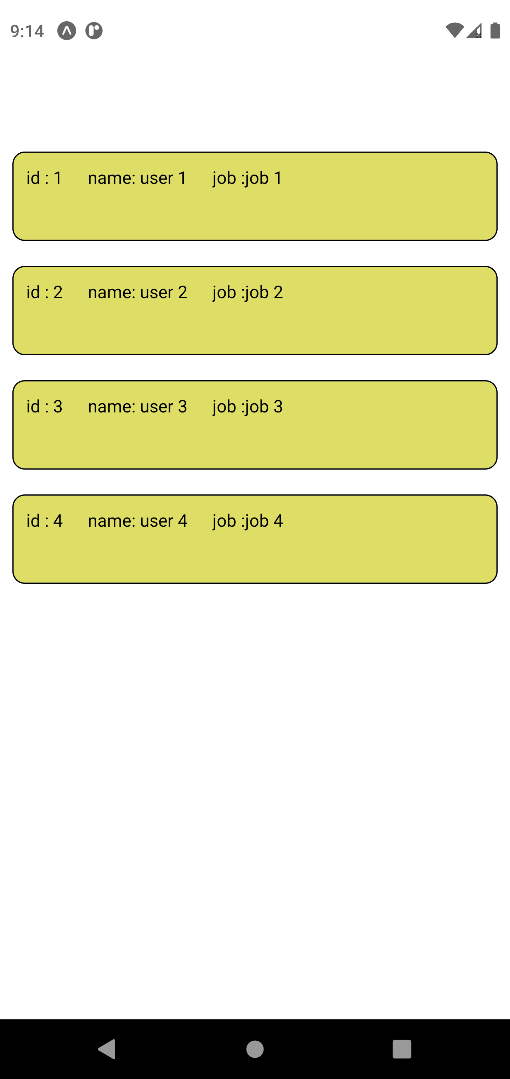
Screen :

Figure 37: flatlist users

### SectionList



Figure 38:SectionList

### Screen :

### 

Figure 39: SectionList Screen

### Map Array

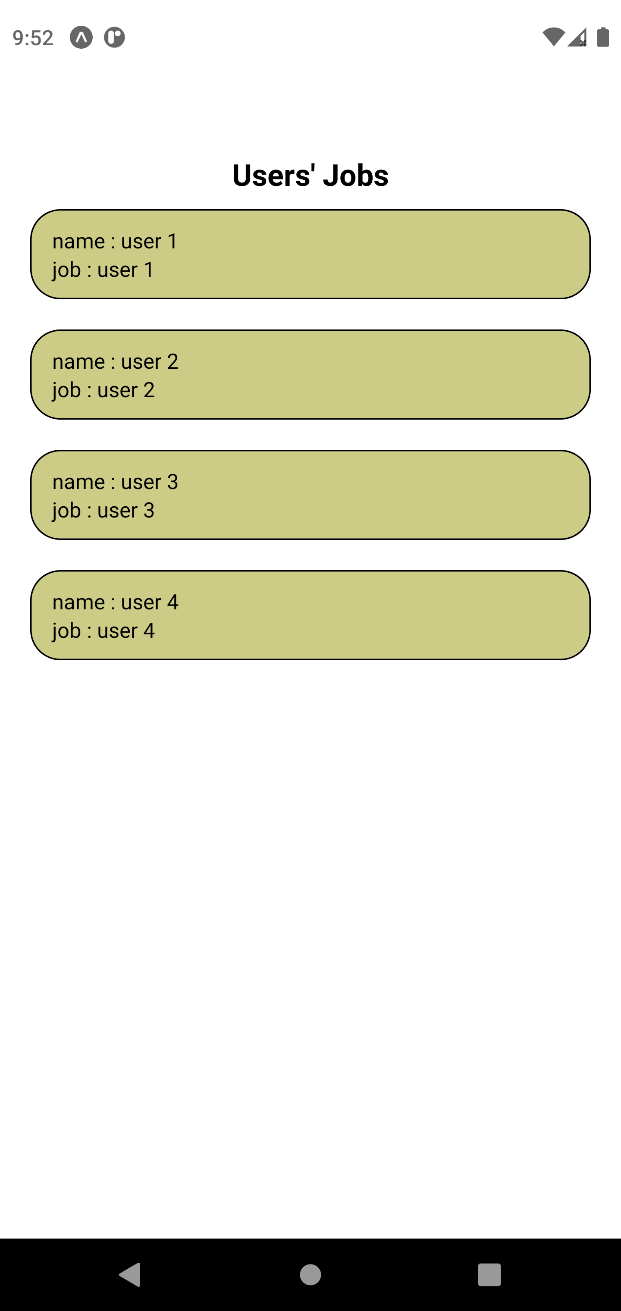
screen :

Figure 40:map array view

# **Chapitre 17 : Input Data**

### Exemples :

### TextInput :

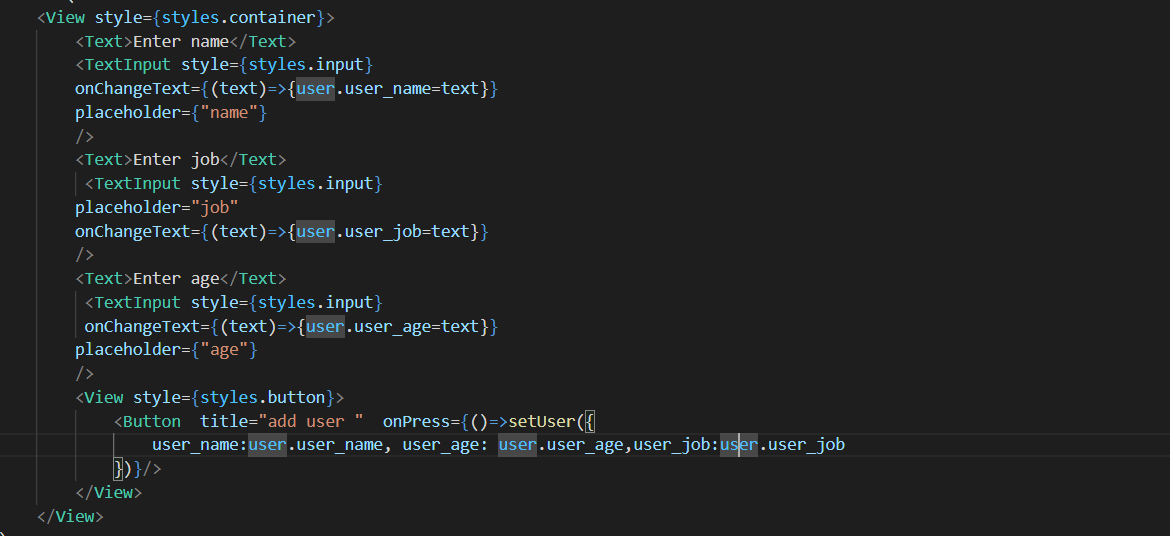
TextInput est un composant pour entrer les données via le clavier .

Figure 41: TextInput

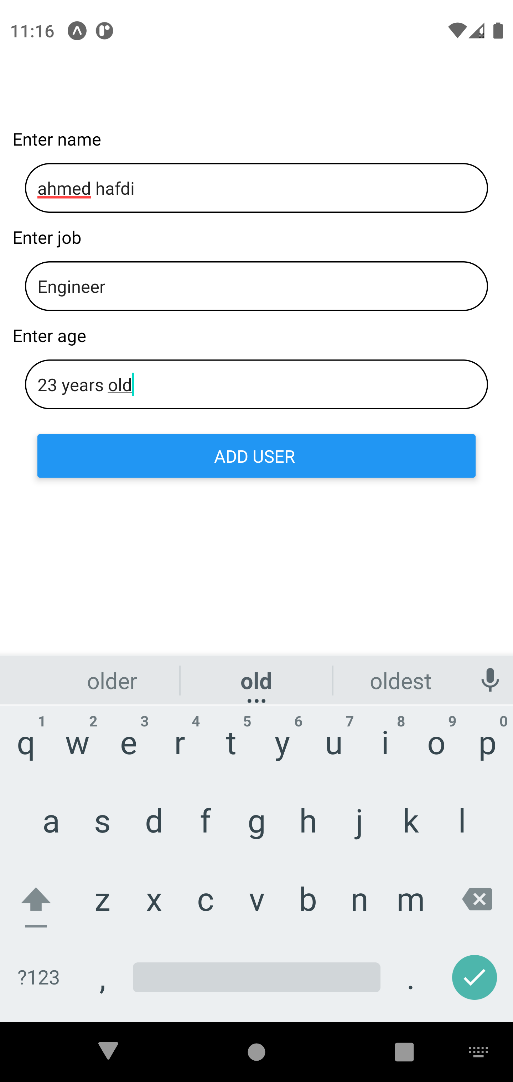


Figure 42: add user screen

Screen :

# **Chapitre 18 : Modal**

### Introduction :

Modal est composant ou un moyen simple de présenter du contenu au-dessus d’une vue englobante.

### Paramètres :

Tableau 1: Modal parameters

|  |  |
| --- | --- |
| **Fonction** | **Détails** |
| animationType | (‘none’,’slide’,’fade’) il contrôle l’animation modele. |
| Visible | Est un booléen qui contrôle la visibilité. |
| onShow | Il permet de passer une fonction qui sera une fois le modal affiché. |
| transparent | Bool pour la transparence. |
| onRequestClose (android) | Méthode pour le bouton retourner |
| onOrientationChange(ios) | Méthode lorsque l’orientation change |
| SupportedOrientations(IOS) | enum («portrait», «portrait à l'envers», «paysage», «paysage à gauche», «paysage à droite») |

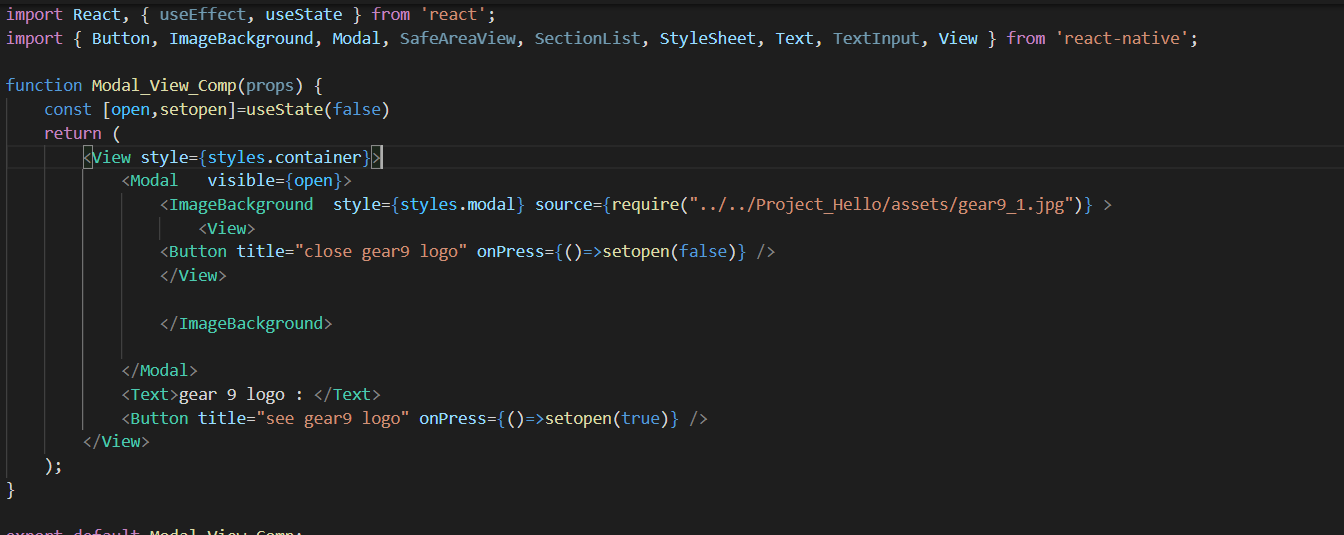


Figure 43: Modal

Screens :



Figure 44: Modal 1

Figure 45: modal 2

# **Chapitre 19 : Module Plateforme**

### Introduction :

Module plateforme permet de détecter le type de système d’exploitation.

### Exemple :

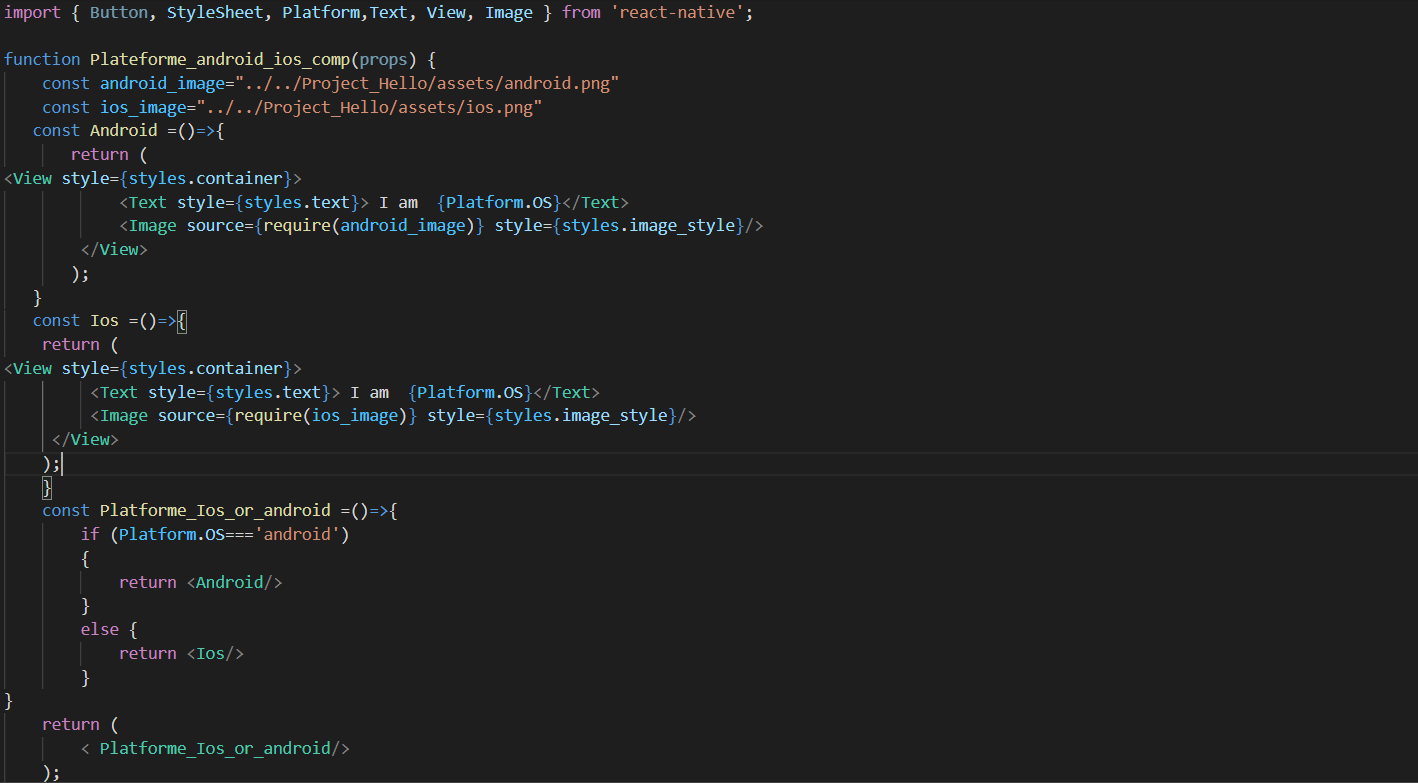


Figure 46: Platforme

Scren :

Figure 47: Platform Screen

# **Chapitre 20 : Notification push**

On utilise Push Notification pour réagir avec une application native en utilisant le module react-native-push-notification pour cette documentation j’ai utilisé un autre module pour Expo

C’est expo-permissions et expo-notifications.

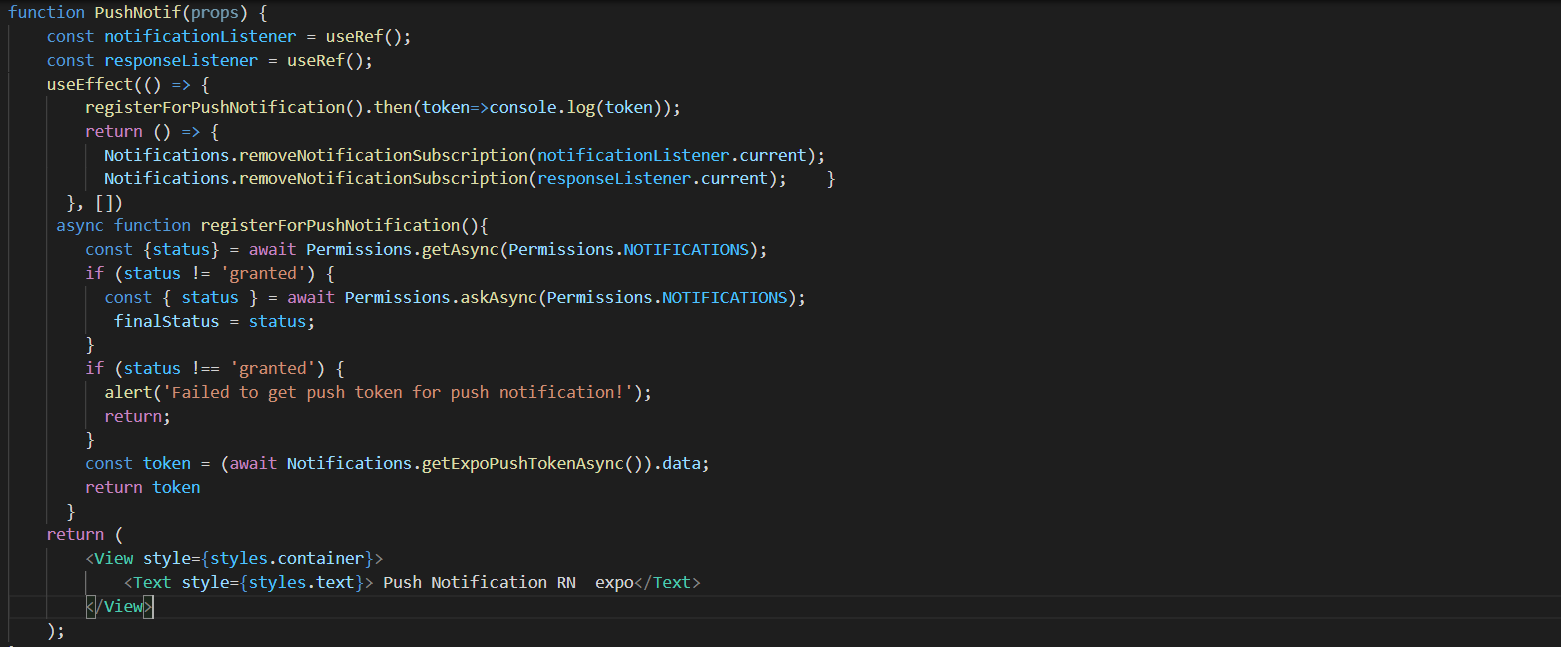
Exemple :

Figure 48: Push Notification

### Expo Push Notifications Tool :

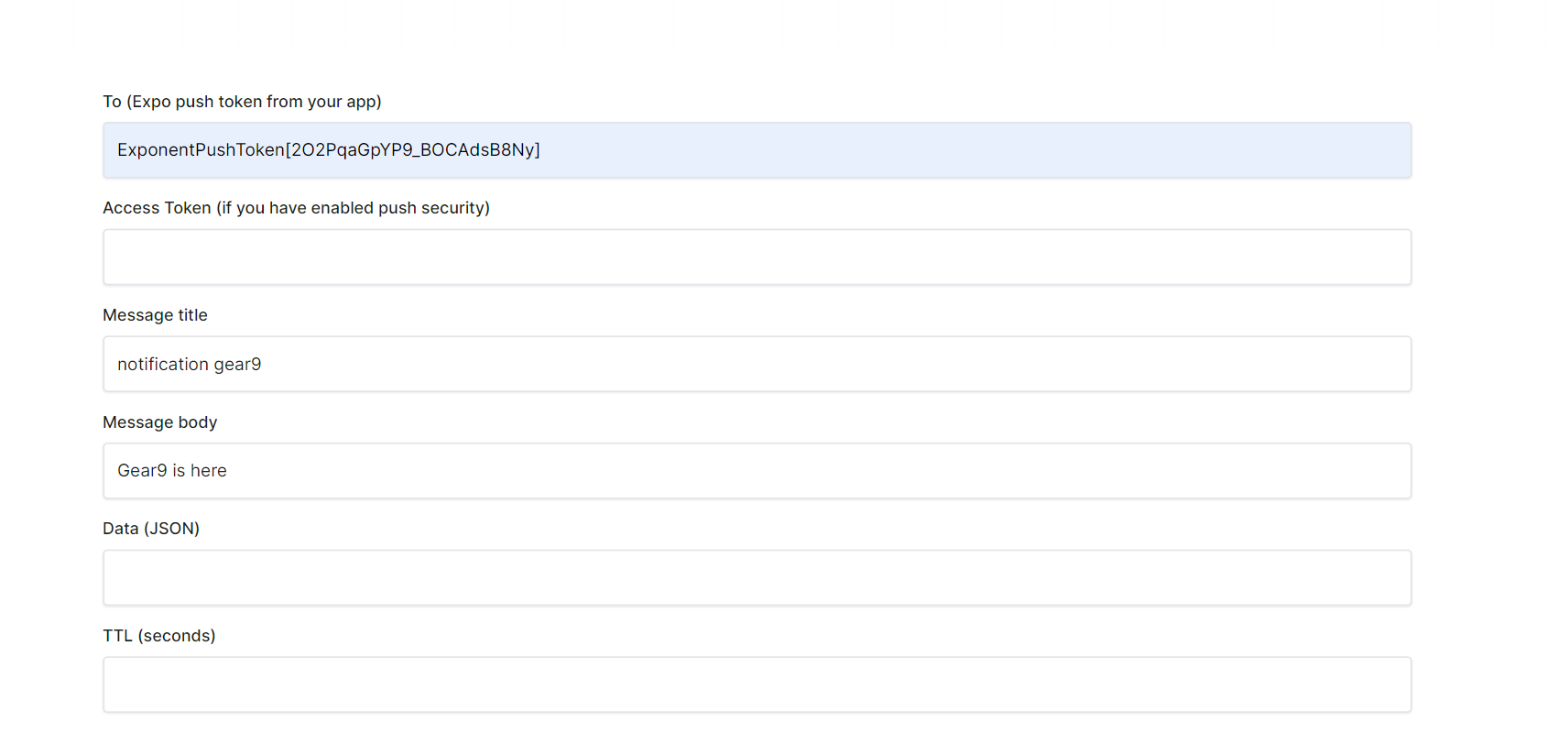


Figure 49: expo push notification tool

Screen :

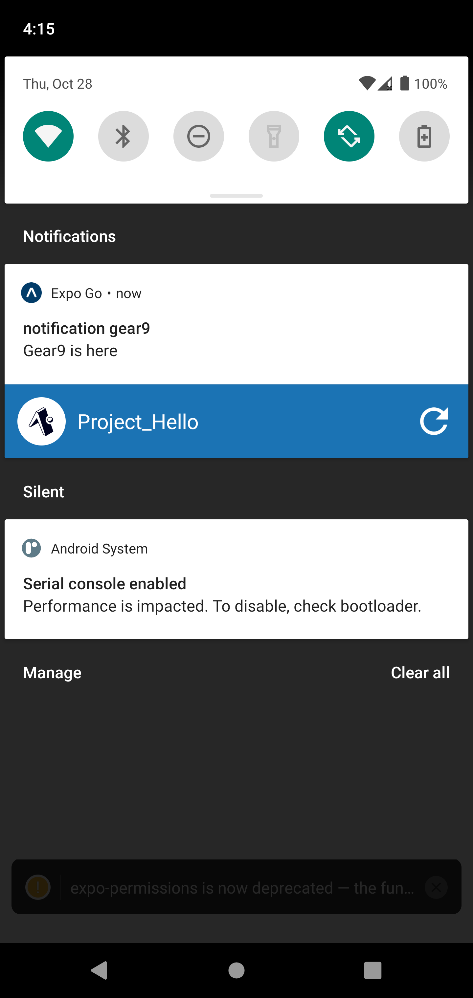


Figure 50: notification screen

# **Chapitre 21 : Polices Personnalisés**

### Exemples :

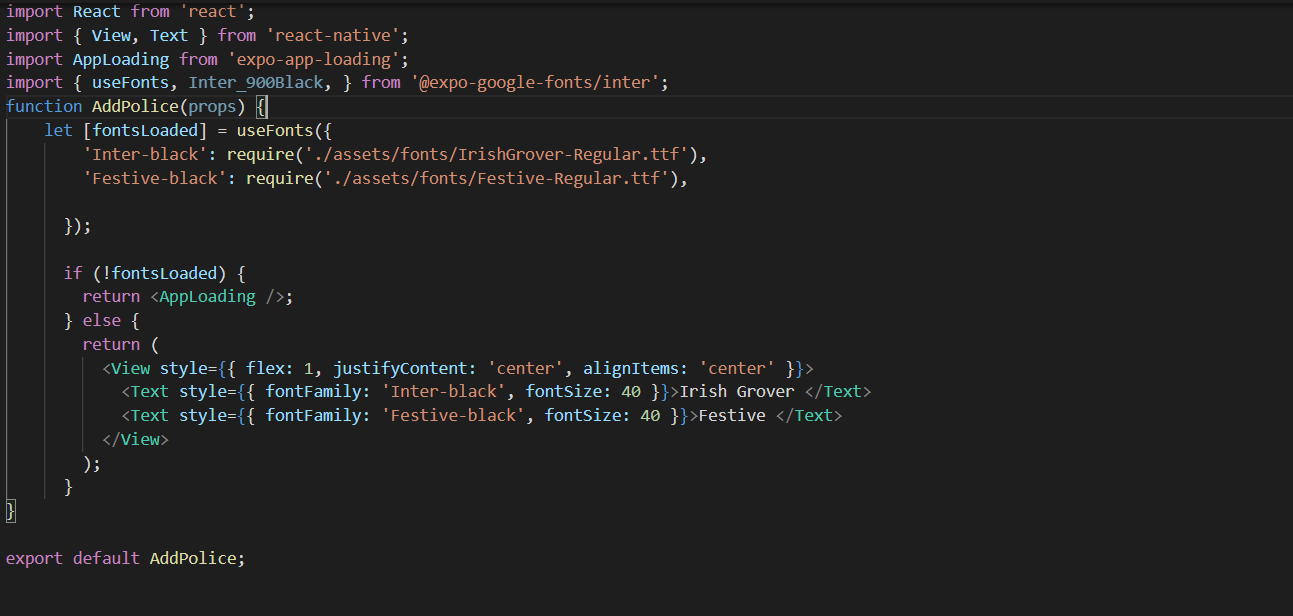
On Peut télécharger n’importe quelle type de police ( par exemple à partir de google fonts)

Figure 51: fonts



Figure 52: fonts exemples

Screen :

# **Chapitre 22 : RefreshControl**

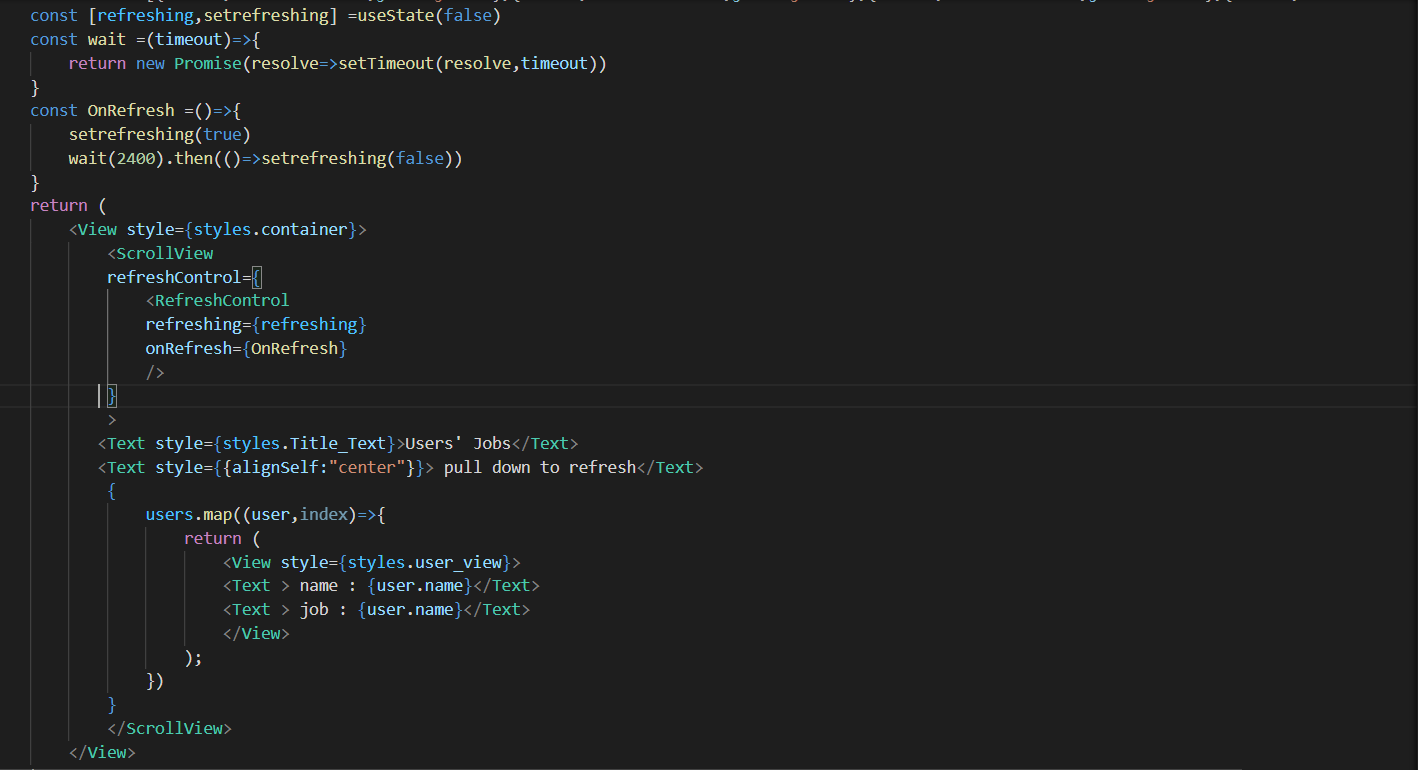


Figure 53: refresh control

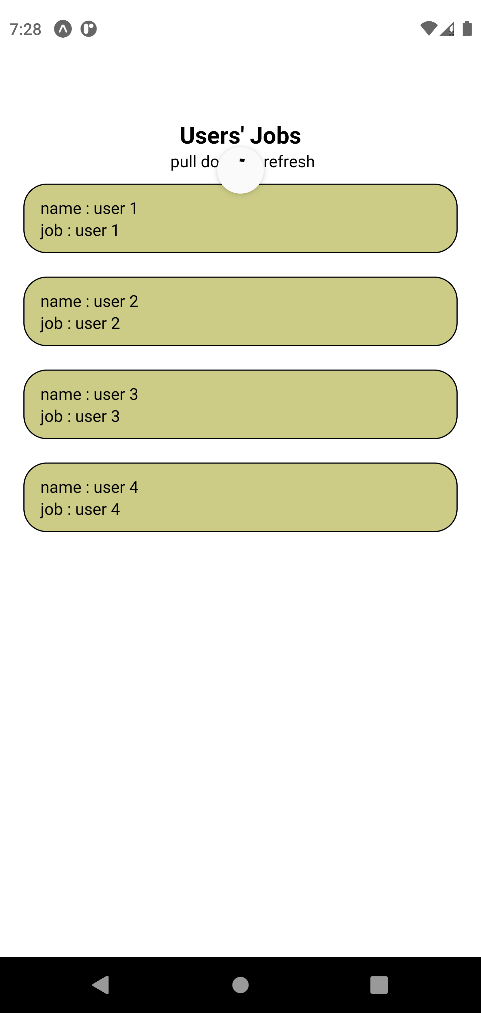
Screens :

Figure 54:Refresh control screen

# **Chapitre 23 : Requêtes http**

### Syntaxe :

* Fetch(url , options) [.alors(…)[.catch(…)]]

### Exemple : Posting application avec json server

Figure 55: fetch1

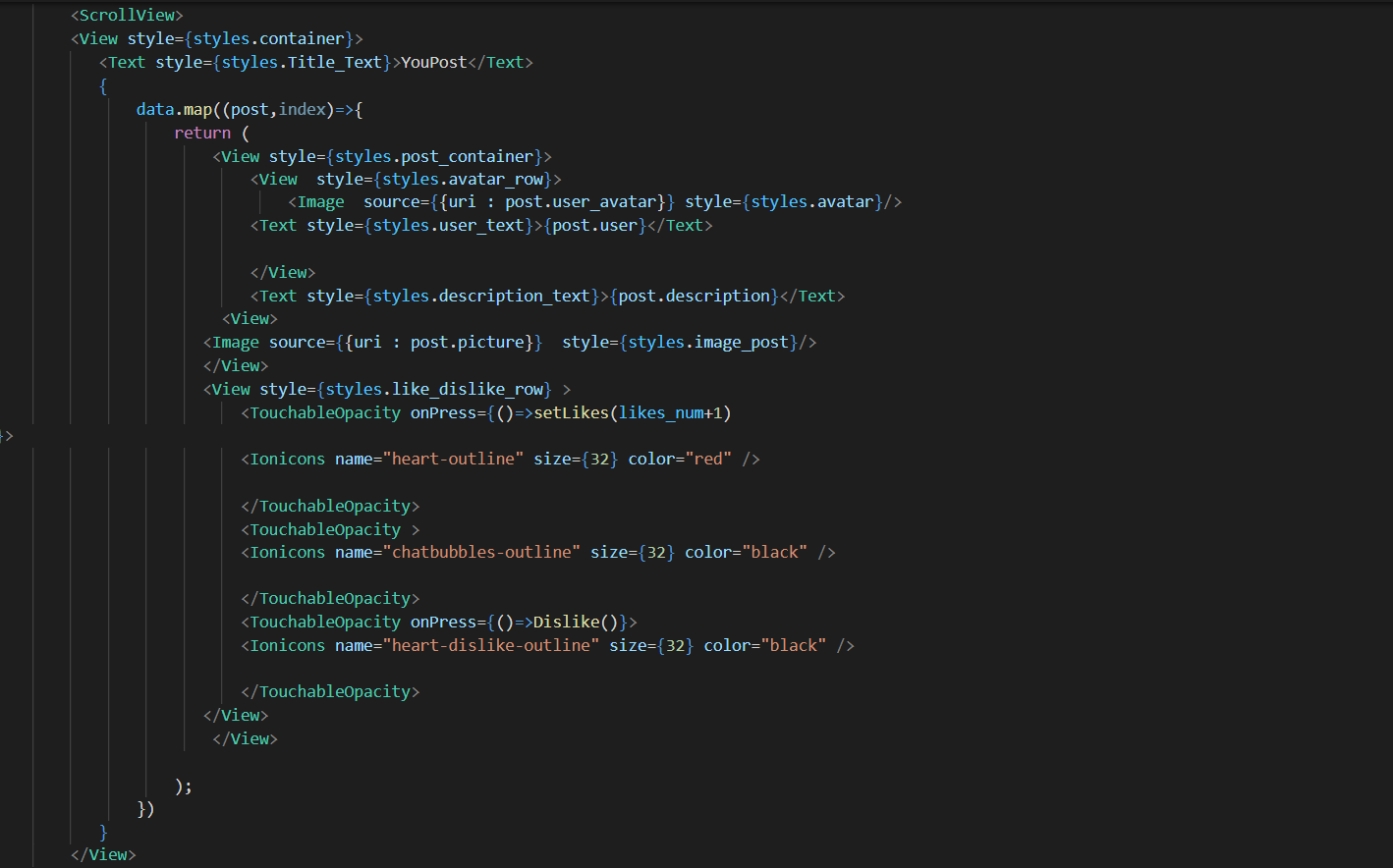
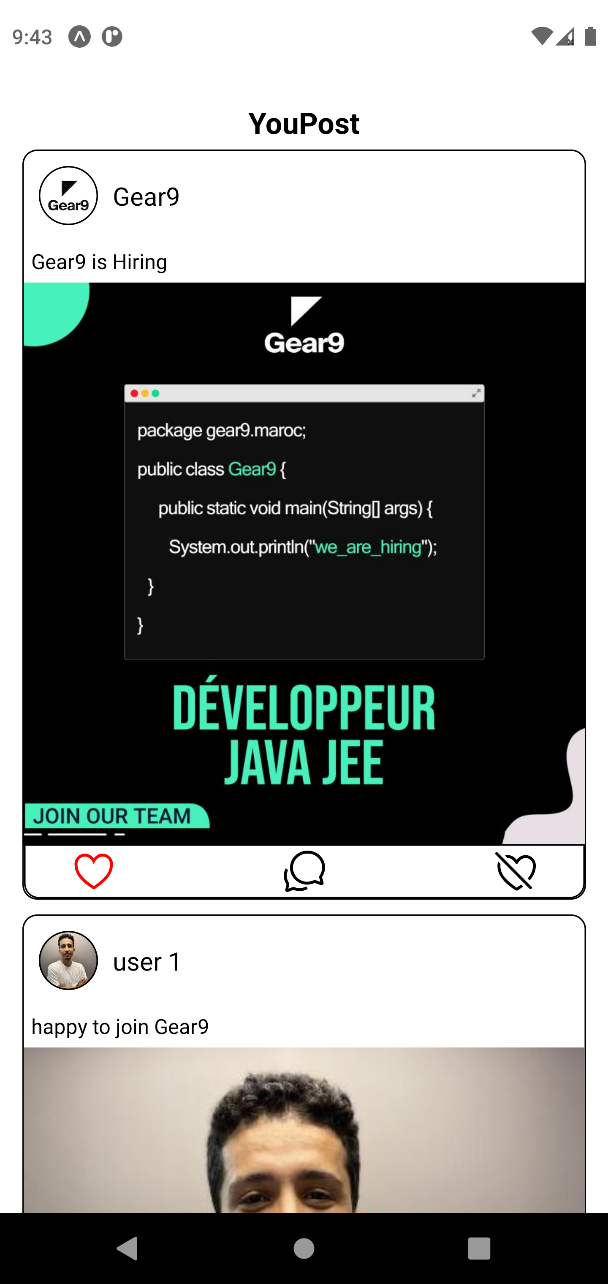


Figure 56: fetch 2

Screen :



# **Chapitre 24 : Test d’unité**

### Introduction :

Le tes unitaire est une pratique de test de bas niveau ou les plus petites unités ou composants du code son testés.

### Exemples :



Figure 57: Test for buttons component

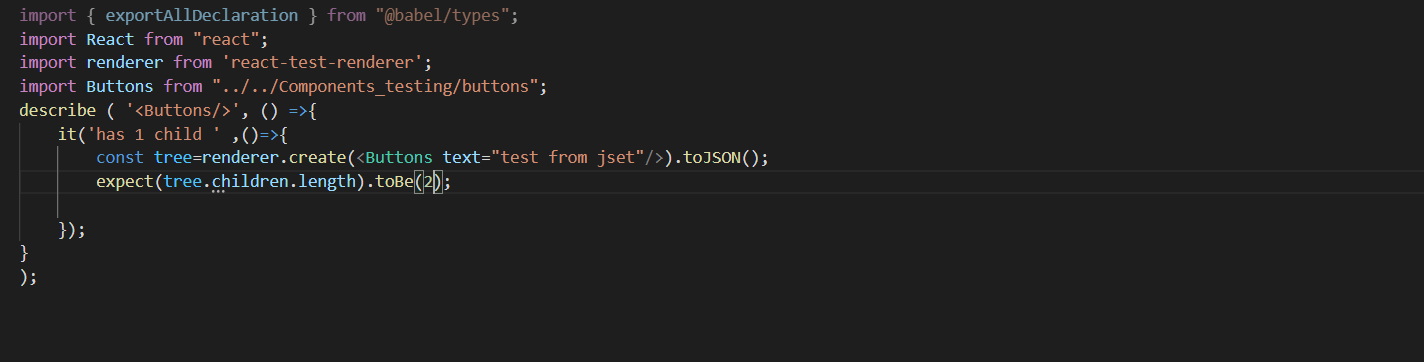


Figure 58: buttonns.test.js

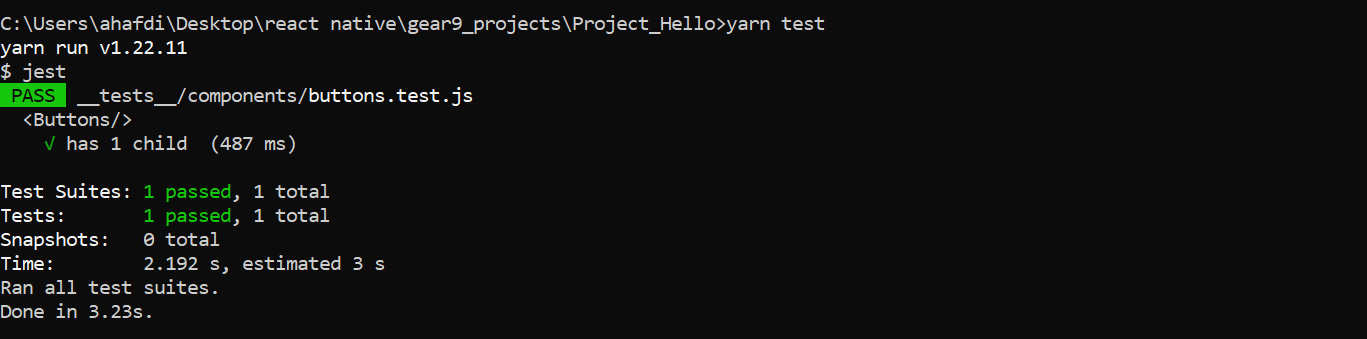


Figure 59: Test passed

# **Chapitre 25 : Mobx**

### Introduction :

Figure 60:Mobx + React Native

### 

State Management ou la gestion de états fait partie de développement des applications avec JavaScript. En particulier , les applications de React et React Native . dans cette documentation on va apprendre à utiliser la bibliothéque Mobx pour la gestion des états , comprendre les concepts de base .

### C’est quoi Mobx :

Mobx est une bibliothèque éprouvée qui simplifier la gestion des états .

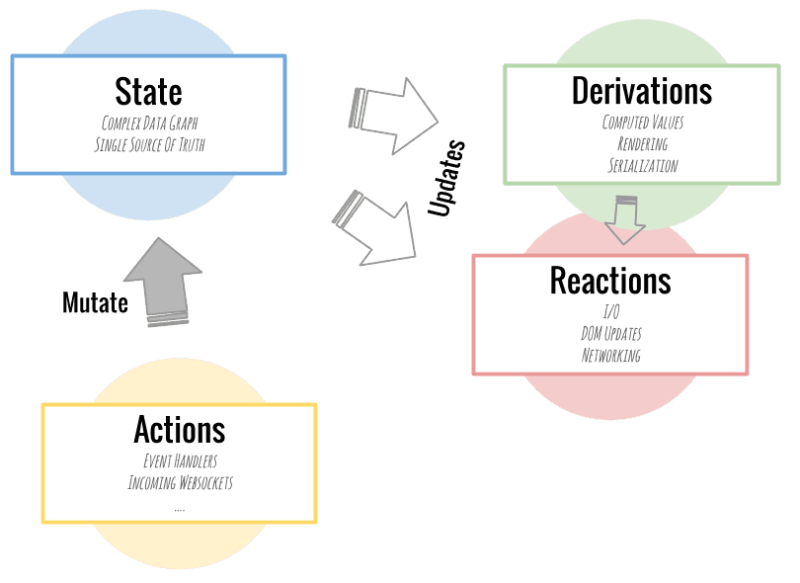


Figure : Mobx state architecture

### Les Principes de Mobx :

* **STATE**

STATE correspond aux données que contient l’application , il s’agit aussi le contenu de sa mémoire . ceci s’applique également à les composants .

* **DERIVATIONS**

Dans Mobx ; tout ce qui peut être dérivé de l’état sans interactions est une dérivation .

* **ACTIONS**

Contrairement , aux dérivations les actions sont tous les components et les fonctions qui mettent un changement sur les données.

Exemple 1 : une application simple des taches avec state management useState ()



Figure 62: Task with UseState

Screen :

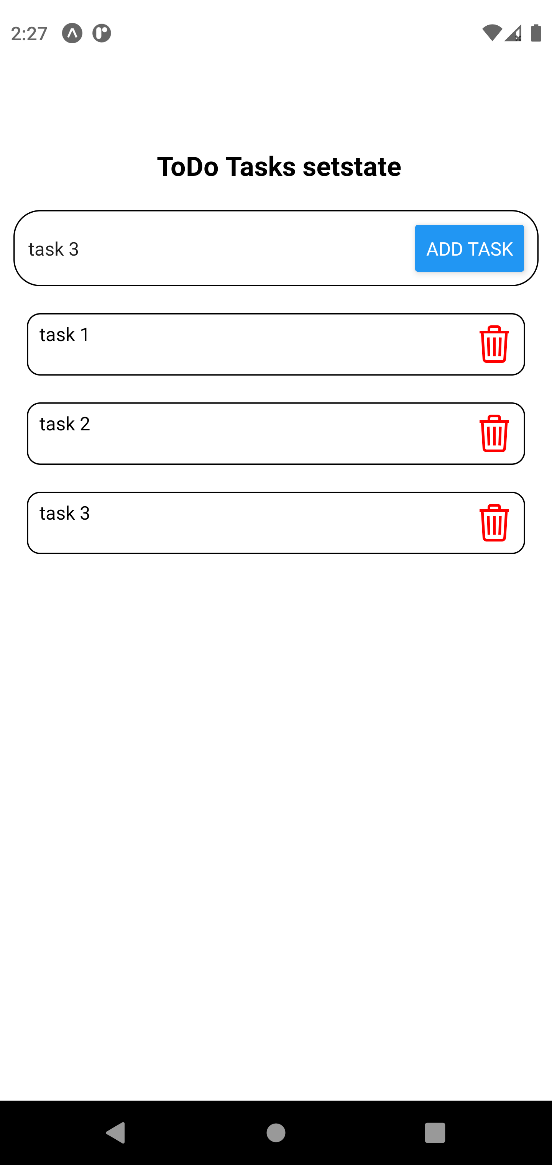


Figure 63: Tasks UseState screen

### Exemple 2 :une application simple des taches avec state management Mobx

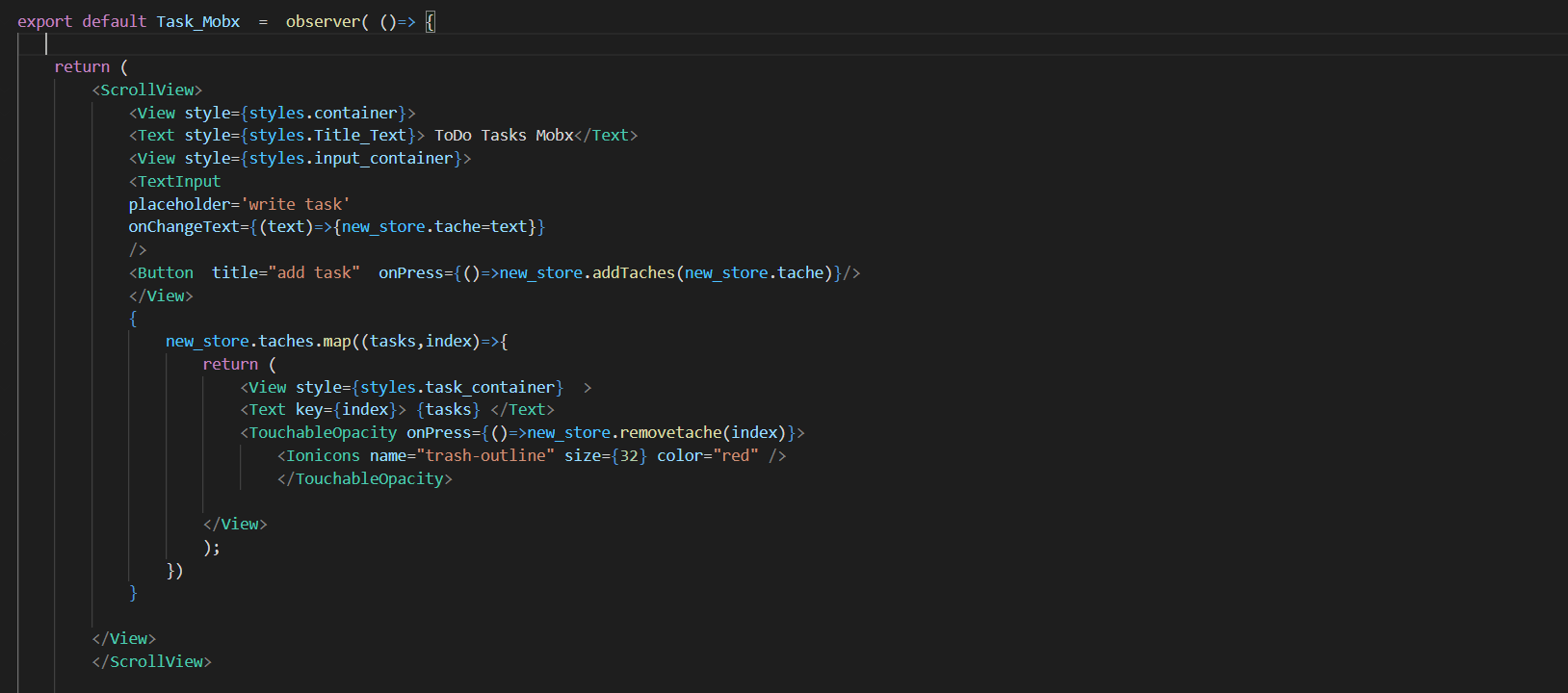


Figure 64: Tasks With Mobx

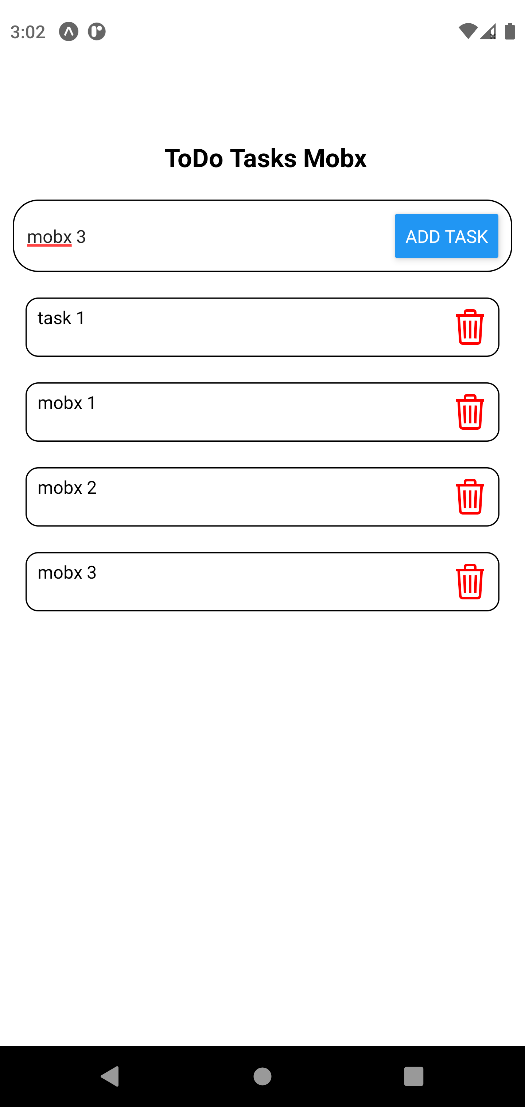
Screen :

Figure 65: Mobx Tasks Screen

### Exemple 3 : Exemple 2 :une application simple des taches avec state management MST (Mobx State Tree)

### 

Figure 66: MST RN

### C’est quoi MST :

Mobx State Tree est un bibliothèque de react et react native il permet de gérer les états mais mieux que Mobx car il permet de gérer les types de données et créer des modeles .



Figure 67: Book Store

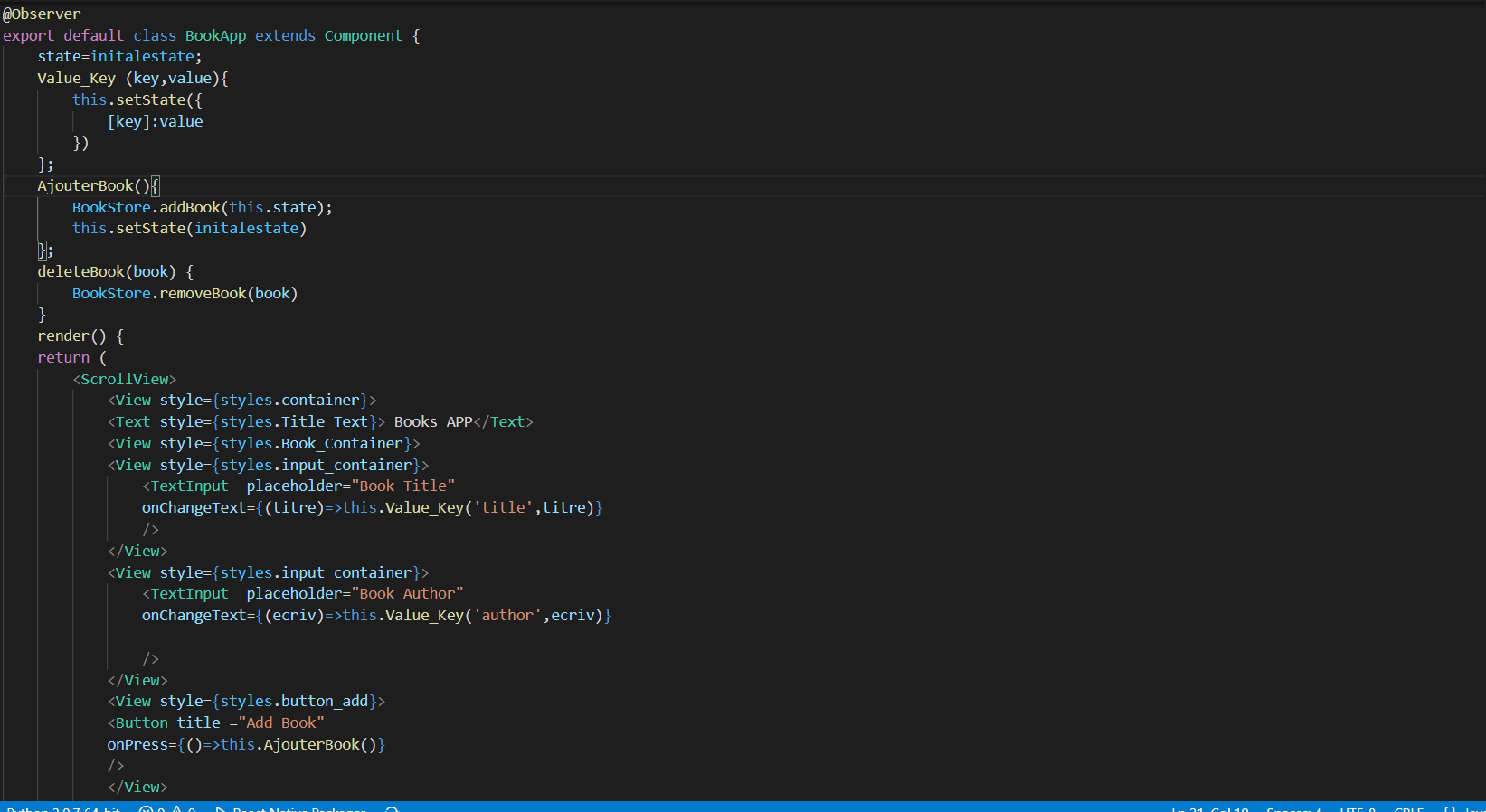
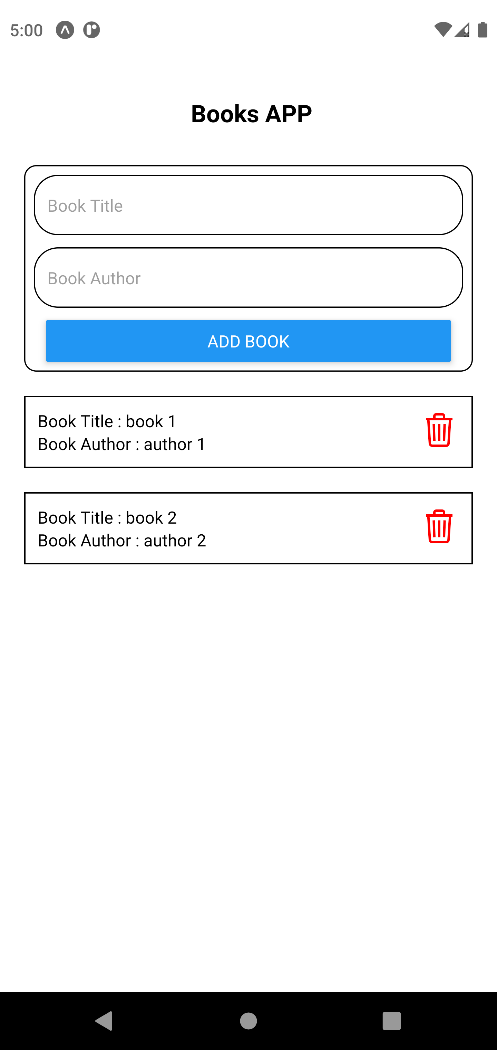


Figure 68: BookView 1



Figure 69: BookView 2

Screen :



# **Chapitre 26: Mini  Projet 1 ( Firebase users app )**

# 

Firebase users app : est une application simple d’ajouter /créer , supprimer et mettre à jour des comptes des utilisateurs c’est application pour admin pour gérer des comptes tout ça en utilisant la base de données firebase.

### Code source :

### Database/firebase.js :

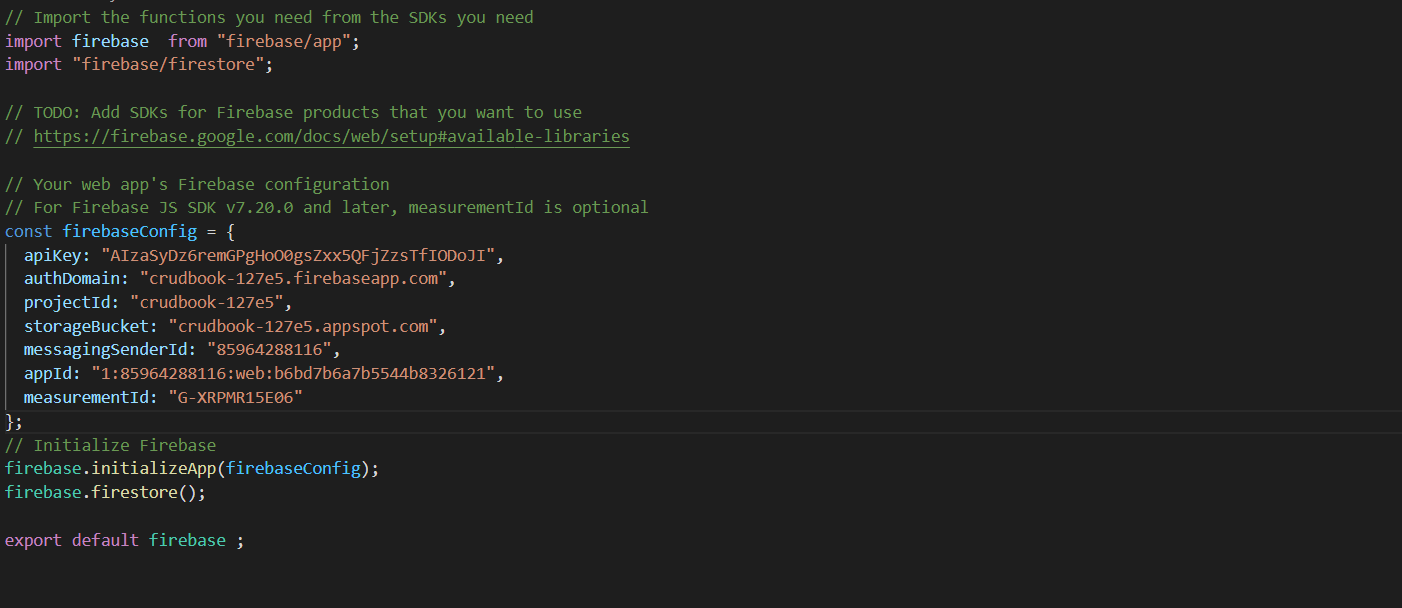


Figure : firebase

### Add User Screen :

Figure : adduserscreen1



Figure : adduserscreen2

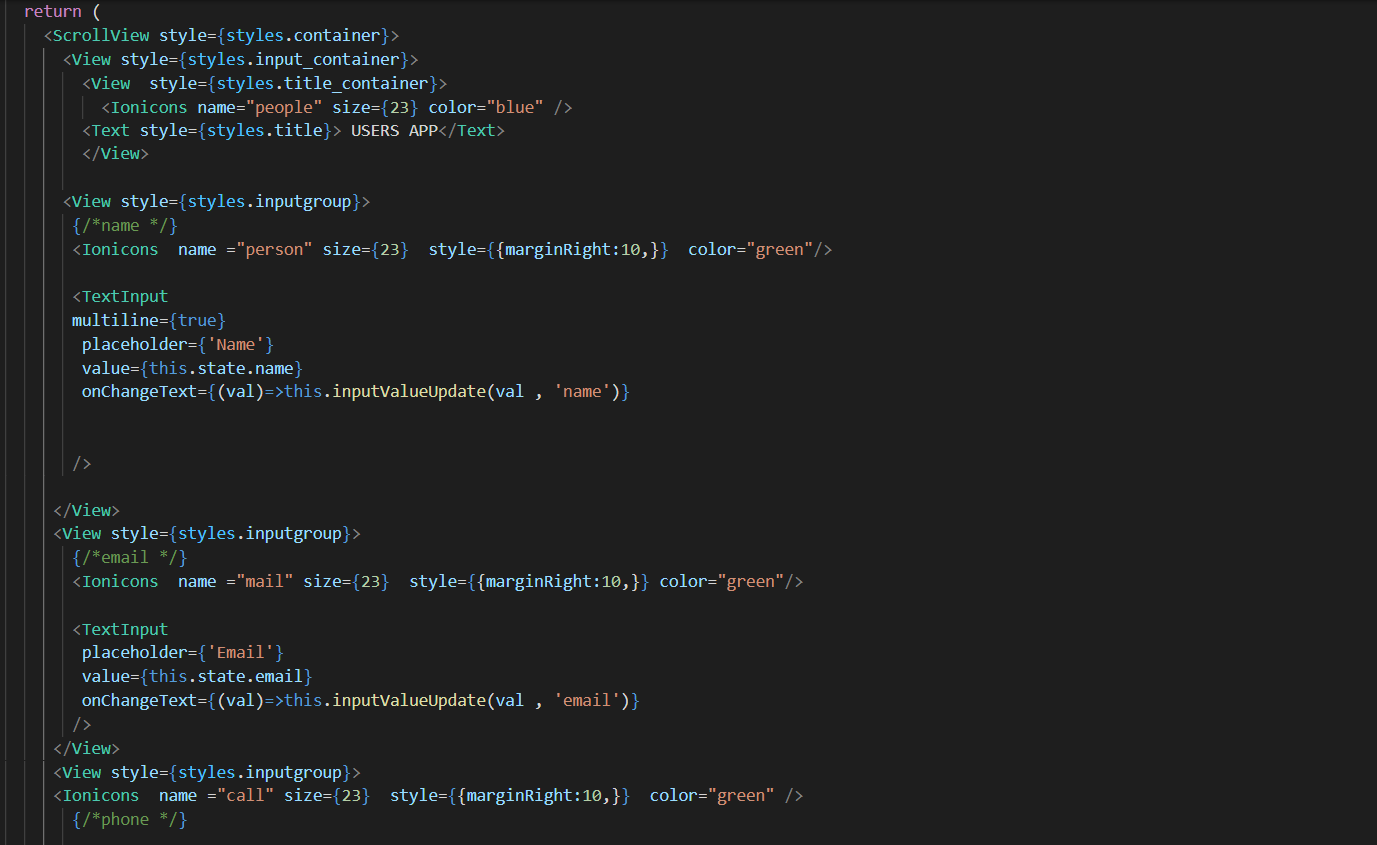


Figure : adduserscreen3



Figure : adduserscreen4

### List Users :

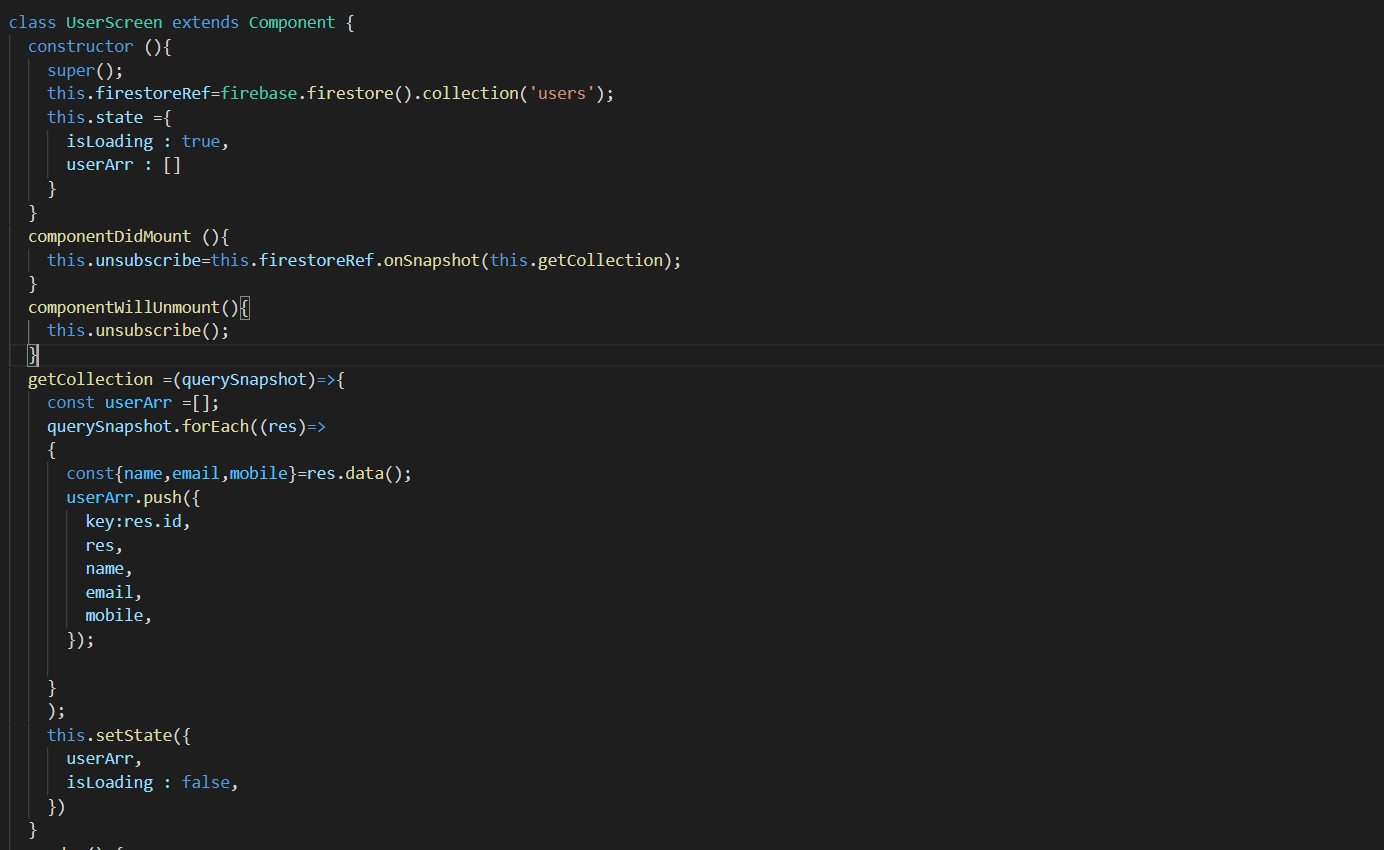


Figure : list users 1

### 

Figure : list users 2

### User Details Screen :

Figure : user screen details 1



Figure : user details screen 2



Figure : user screen details 3

Screens :

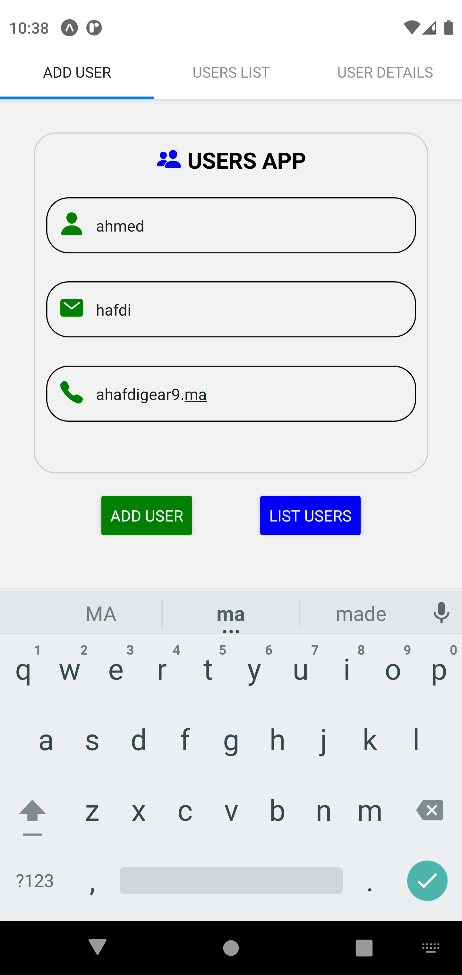
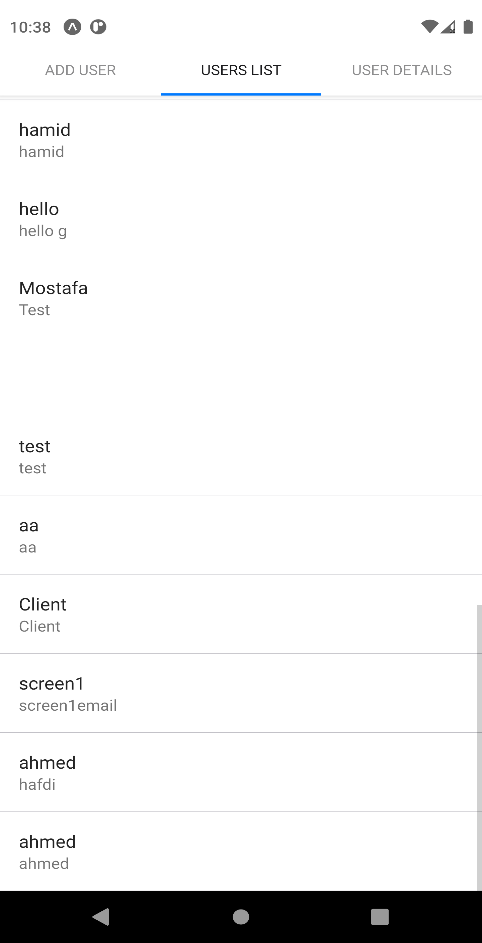
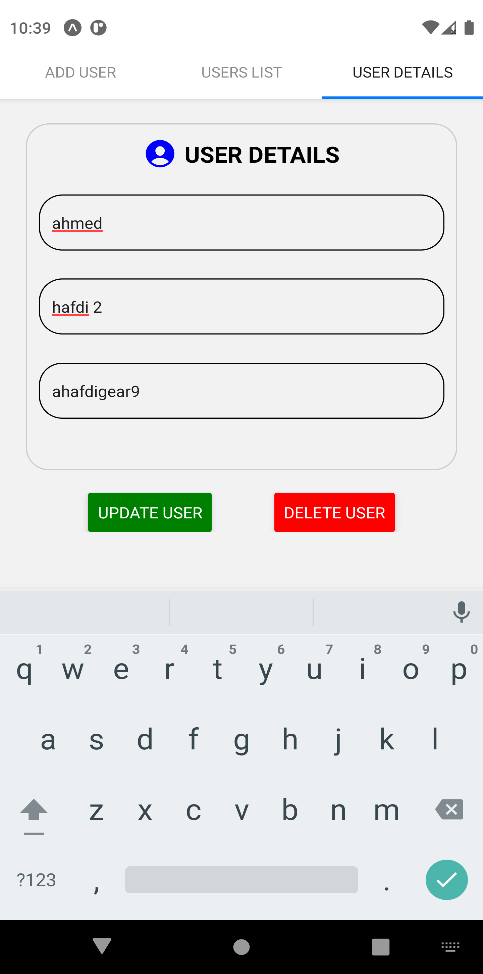


Figure : user details

Figure : add user

Figure : list users

# **Chapitre 27: Mini  Projet 2 ( Tasks app )**

Voir Chapitre 25