DOCUMENT TECHNIQUE REACT NATIVE

Table des matières

[Table des Figures 4](#_Toc86316739)

[**CHAPITRE 1 : SET UP POUR L’ENVIRONEMMENT WINDOWS** 5](#_Toc86316740)

[**Configuration pour Windows :** 6](#_Toc86316741)

[**CHAPITRE 2 : BONJOUR LE MONDE** 7](#_Toc86316742)

[**Bonjour Tout le monde!** 8](#_Toc86316743)

[**Chapitre 3 : Coiffant (stylesheet)** 10](#_Toc86316744)

[**Introduction :** 10](#_Toc86316745)

[**Syntaxe :** 10](#_Toc86316746)

[**Remarques :** 10](#_Toc86316747)

[**Examples :** 10](#_Toc86316748)

[**Chapitre 6 : Composants** 12](#_Toc86316749)

[**Examples :** 12](#_Toc86316750)

[Composant de base : 12](#_Toc86316751)

[Composant avec état 12](#_Toc86316752)

[Composant sans état ( composant without state) 13](#_Toc86316753)

[**Chapitre 7 : Disposition** 14](#_Toc86316754)

[**Flexbox** : 14](#_Toc86316755)

[**Chaptire 8 : States ( les états)** 16](#_Toc86316756)

[**Syntaxe :** 16](#_Toc86316757)

[**Examples :** 16](#_Toc86316758)

[**SetState :** 16](#_Toc86316759)

[**Chapitre 10 : Exécuter une application sur l’appareil (version android ) & Expo** 18](#_Toc86316760)

[**Exécuter sur l’appareil :** 18](#_Toc86316761)

[**Chapitre 11 : Images** 20](#_Toc86316762)

[Examples : 20](#_Toc86316763)

[Module d’image 20](#_Toc86316764)

[ Image locale  et Image externe : 20](#_Toc86316765)

[**Source d’image conditionnelle :** 21](#_Toc86316766)

[**Screen :** 21](#_Toc86316767)

[21](#_Toc86316768)

[21](#_Toc86316769)

[**Chapitre 12 : Instructions en ligne de commande** 22](#_Toc86316770)

[**Examples :** 22](#_Toc86316771)

[**Vérifier la version** 22](#_Toc86316772)

[**Examples de sortie** 22](#_Toc86316773)

[**Chapitre 13 : Le débogage** 23](#_Toc86316774)

[Syntaxe : 23](#_Toc86316775)

[Le débogage avec Expo (Exponent ) et VSCode : 23](#_Toc86316776)

[ C’est quoi Exponent ? 23](#_Toc86316777)

[Etapes de débogage par Expo & VScode : 23](#_Toc86316778)

[**Chapitre 14 : Le Routage** 25](#_Toc86316779)

[Exemples : 25](#_Toc86316780)

[Stack Composant : 25](#_Toc86316781)

[Top Tabs : 27](#_Toc86316782)

[Bottom Tabs 28](#_Toc86316783)

[**Chapitre 15 : Liaison de l’API native** 29](#_Toc86316784)

[Introduction : 29](#_Toc86316785)

[Exemples : 29](#_Toc86316786)

[Liens sortants 29](#_Toc86316787)

[Liens entrants 29](#_Toc86316788)

[**Chapitre 16 : ListView** 31](#_Toc86316789)

[FlatList 31](#_Toc86316790)

[SectionList 32](#_Toc86316791)

[Screen : 32](#_Toc86316792)

[32](#_Toc86316793)

[Map Array 33](#_Toc86316794)

[**Chapitre 17 : Input Data** 34](#_Toc86316795)

[Exemples : 34](#_Toc86316796)

[TextInput : 34](#_Toc86316797)

[**Chapitre 18 : Modal** 35](#_Toc86316798)

[Introduction : 35](#_Toc86316799)

[Paramètres : 35](#_Toc86316800)

# Table des Figures

[Figure 1:react native logo 5](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316698)

[Figure 2: Hello World 7](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316699)

[Figure 3: Hello World Screen 8](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316700)

[Figure 4: basic component 11](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316701)

[Figure 5: component avec état 12](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316702)

[Figure 6: composant sans état 12](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316703)

[Figure 7: flexdirection : 'row' 14](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316704)

[Figure 8 : flexdirection row screen 15](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316705)

[Figure 9: counter add 16](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316706)

[Figure 10: counter add screen 17](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316707)

[Figure 11: adb devices 18](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316708)

[Figure 12: expo start server 19](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316709)

[Figure 13: expo run on android emulator 19](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316710)

[Figure 14: image locale et externe 20](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316711)

[Figure 15: image locale & externe screen 20](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316712)

[Figure 16: image conditionnelle 21](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316713)

[Figure 17: image conditionnelle screen 21](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316714)

[Figure 18: react native check version 22](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316715)

[Figure 19: expo logo 23](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316716)

[Figure 20: react native tools 23](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316717)

[Figure 21 : debug in exponent 24](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316718)

[Figure 22: debug 24](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316719)

[Figure 23 : Navigator 25](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316720)

[Figure 24 : MyStack 25](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316721)

[Figure 25: First Screen 26](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316722)

[Figure 26: Second Screen 26](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316723)

[Figure 27: Top Tabs 27](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316724)

[Figure 28: Tab Tops screen 27](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316725)

[Figure 29: Bottom tabs 28](#_Toc86316726)

[Figure 30: lien sortant 29](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316727)

[Figure 31: lien sortant screen 29](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316728)

[Figure 32 : lien entrant 29](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316729)

[Figure 33: Linking 29](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316730)

[Figure 34: lien entrant 30](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316731)

[Figure 35:flatList 31](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316732)

[Figure 36: flatlist users 31](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316733)

[Figure 37:SectionList 32](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316734)

[Figure 38: SectionList Screen 32](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316735)

[Figure 39:map array view 33](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316736)

[Figure 40: TextInput 34](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316737)

[Figure 41: add user screen 34](file:///C:\Users\ahafdi\Desktop\react%20native\ignite\PizzaApp\documentation_RN\documentation\DOCUMENT%20TECHNIQUE%20REACT%20NATIVE.docx#_Toc86316738)

# **Liste des Tableaux :**

[Tableau 1: Modal parameters 36](#_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 :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 : Hello World

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

****

Figure : 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 : 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 : component avec état

### Composant sans état ( composant without state)

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

Figure : 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 : flexdirection : 'row'

**screen :**

****

Figure : 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 : counter add

**Screen :**

****

Figure : 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 : 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 : expo start server

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

Figure : 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 : image locale et externe

* **Screen :**

Figure : image locale & externe screen

****

**Source d’image conditionnelle :**

Figure : image conditionnelle

**Screen :**

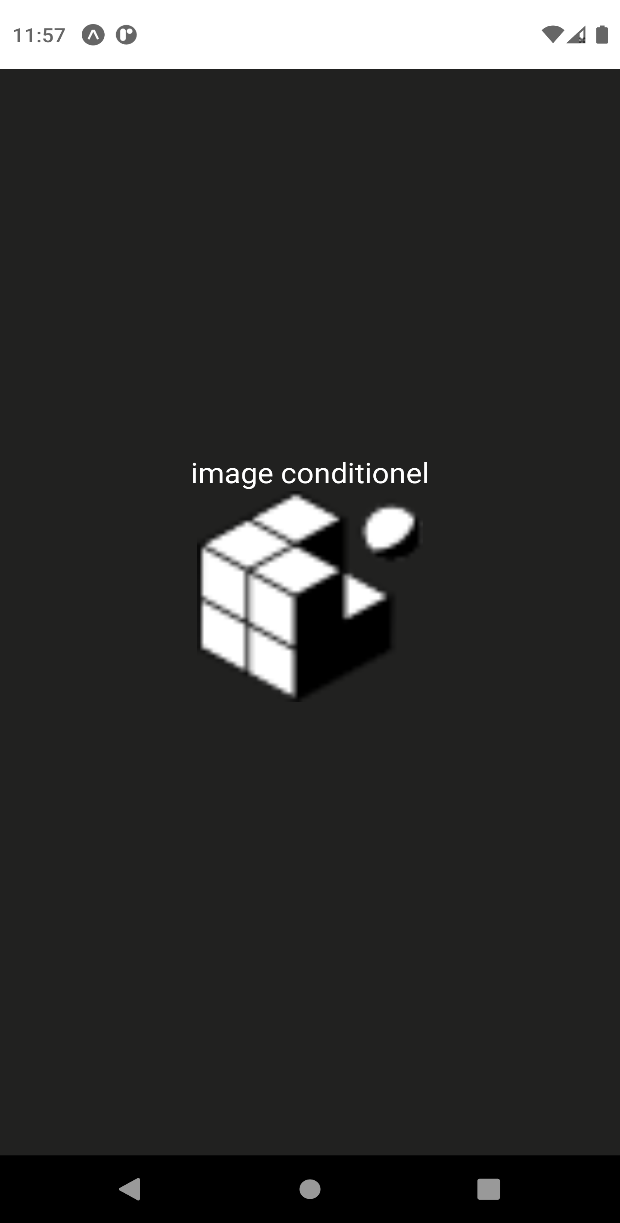
****

Figure : image conditionnelle screen

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

### **Examples :**

### **Vérifier la version**

* **$ react-native -v**

### **Examples de sortie**

Figure : 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 : expo logo

### Etapes de débogage par Expo & VScode :

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

Figure : react native tools



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



Figure : debug in exponent

1. On lance le debug de l’application :



Figure : 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 :

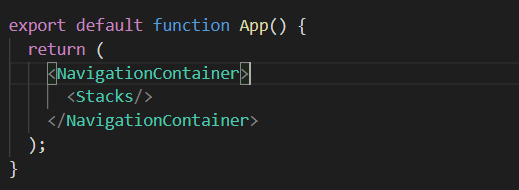
Exemple de navigation entre deux screens :

Figure : Navigator

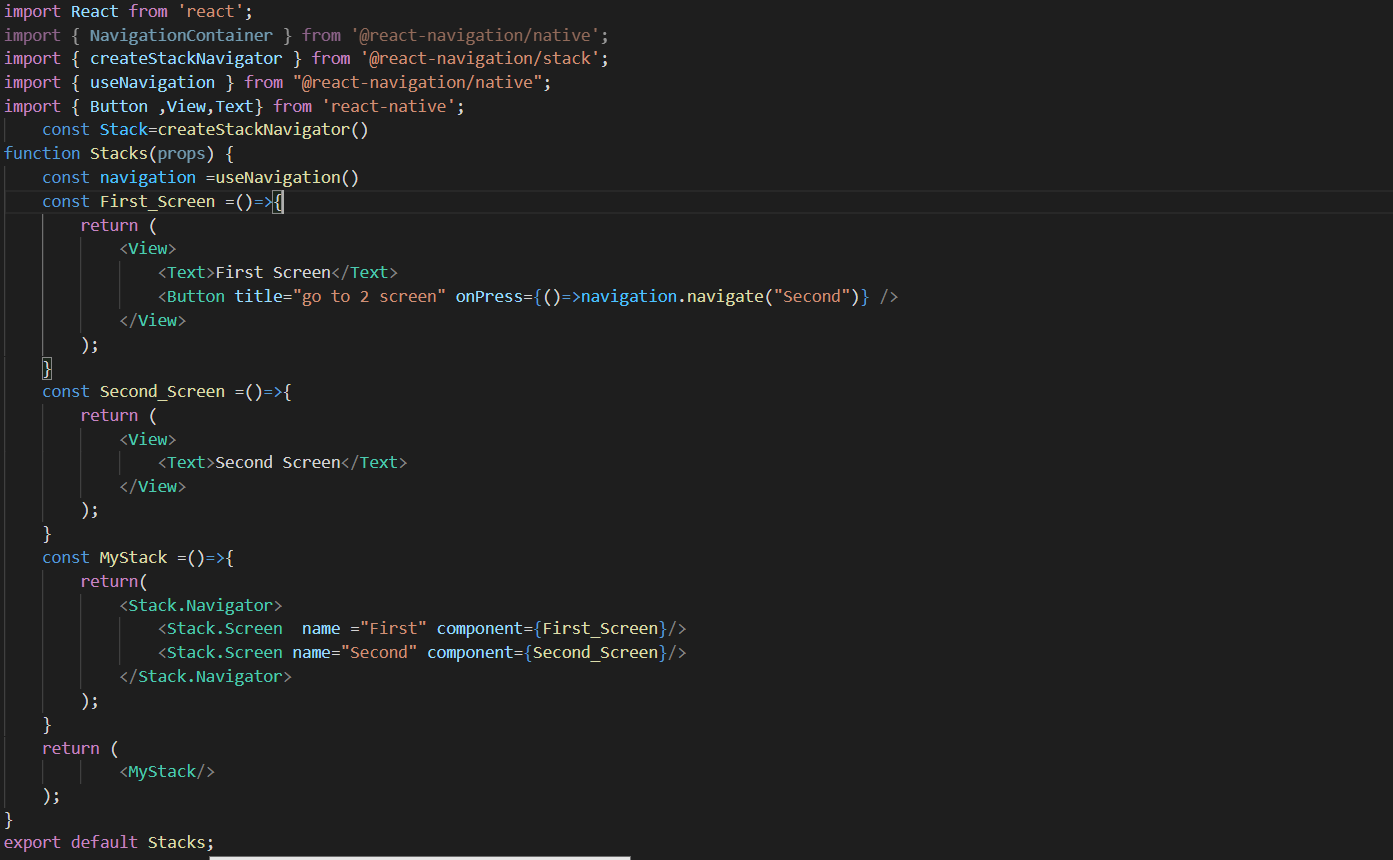


Figure : MyStack

**Screens :**

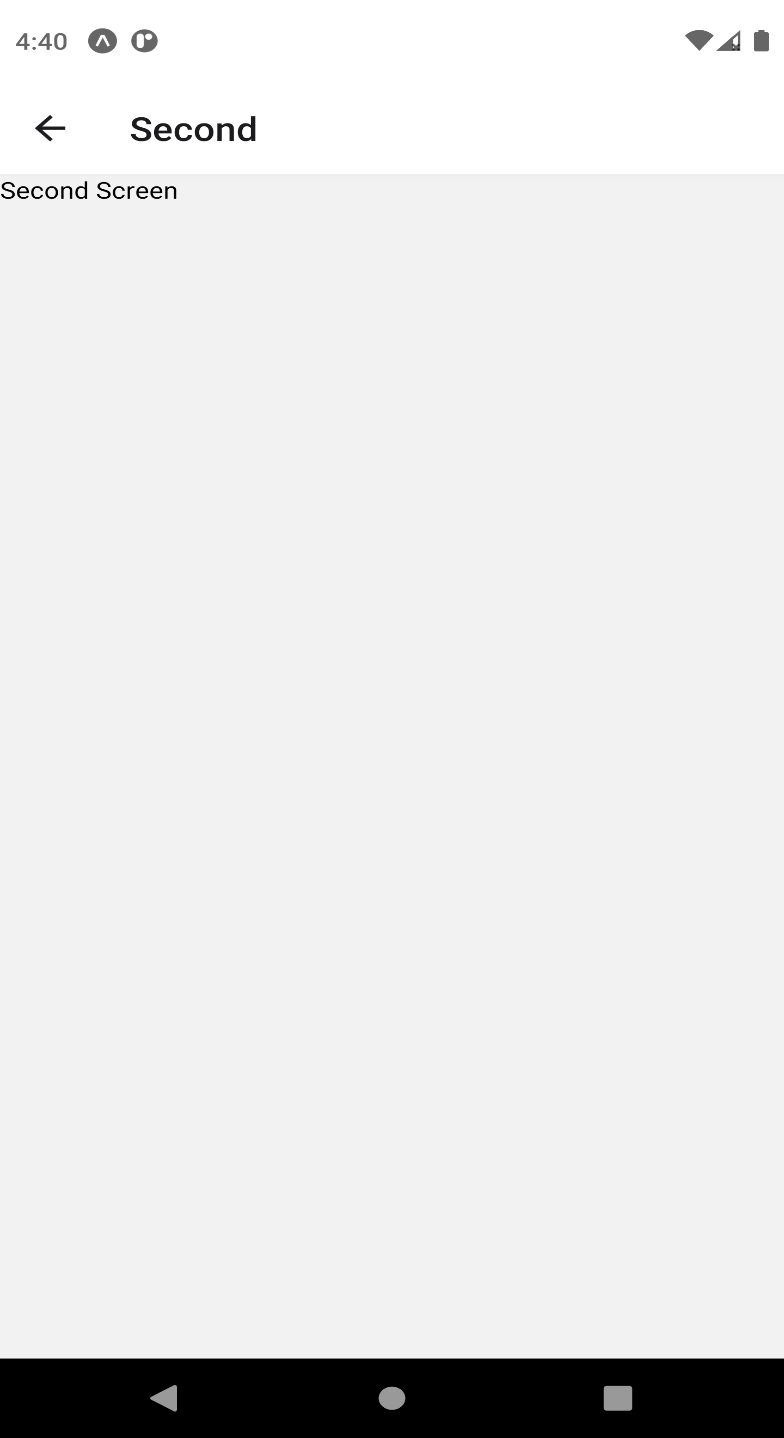
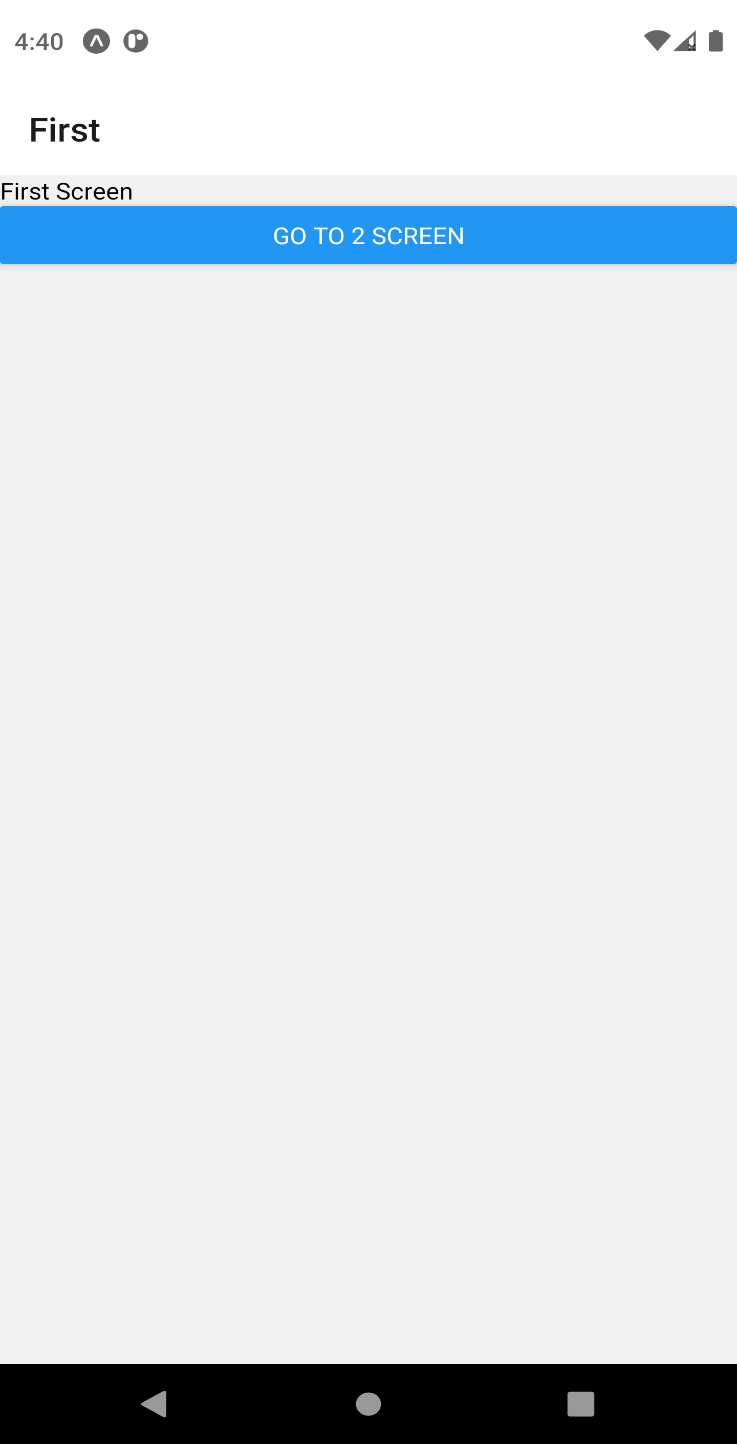
****

Figure : First Screen

Figure : Second Screen

### Top Tabs :

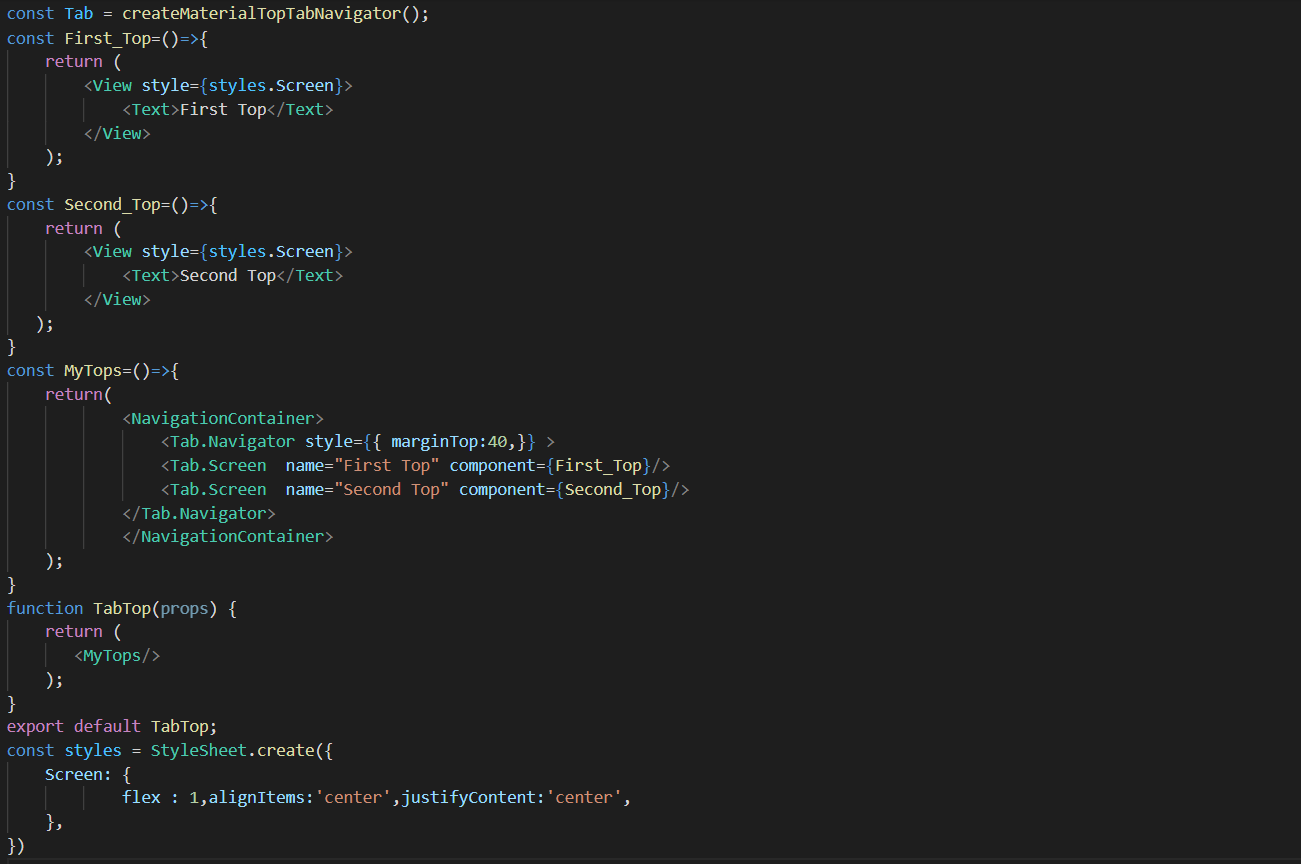


Figure : Top Tabs

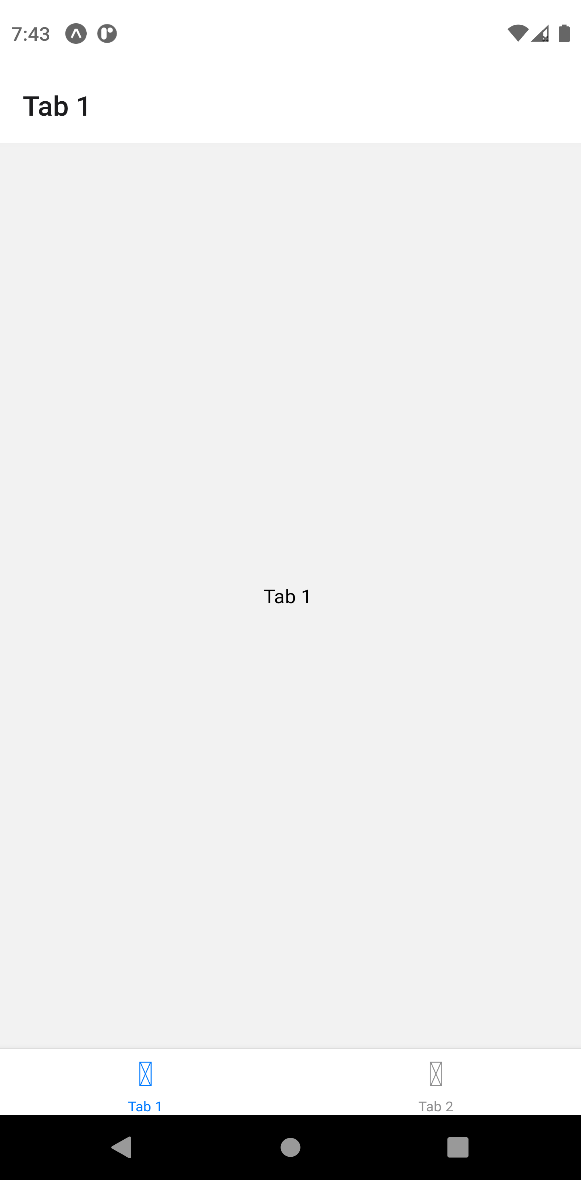
Screen :

Figure : Tab Tops screen

### Bottom Tabs



Figure : Bottom tabs

Screen :

# **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

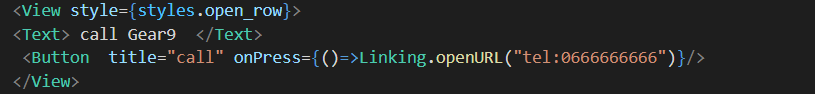


Figure : lien sortant

### Liens entrants

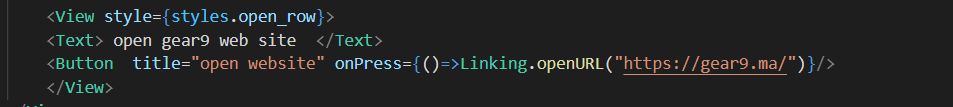
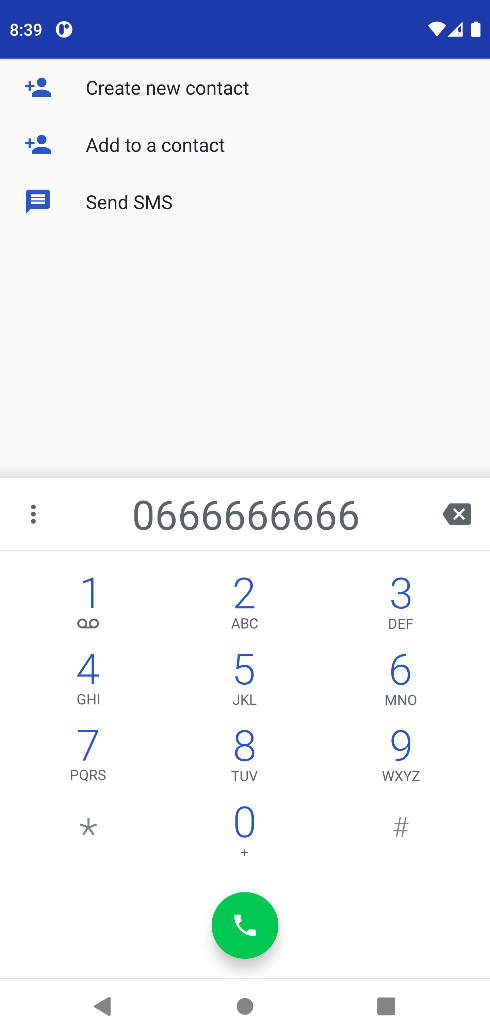


Figure : lien sortant screen

Figure : lien entrant

Screens :

Figure : Linking

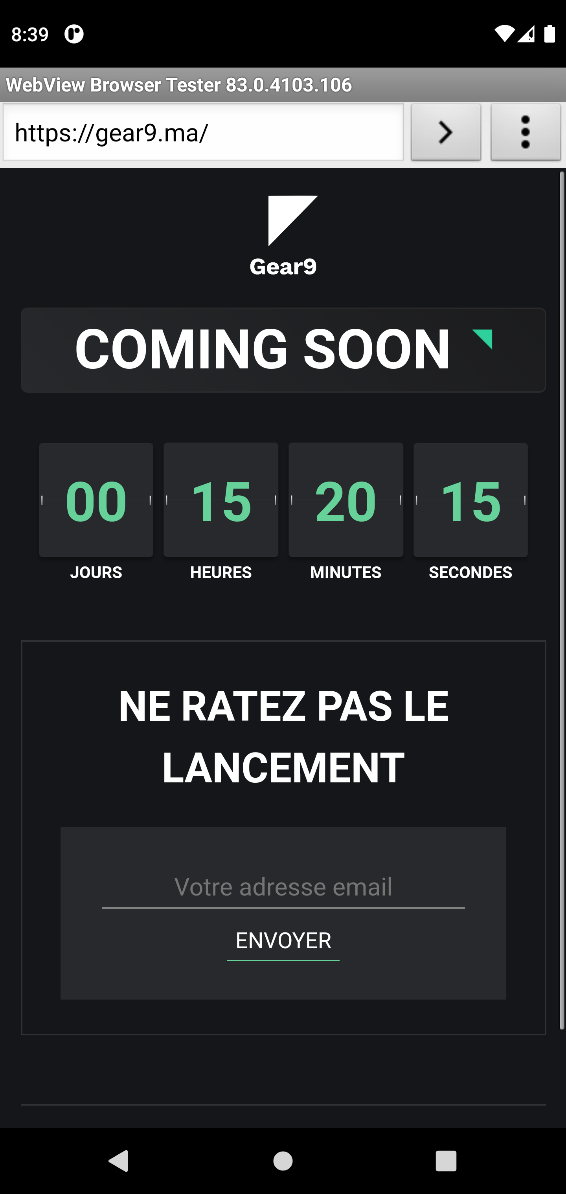


Figure : lien entrant

# **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 :flatList

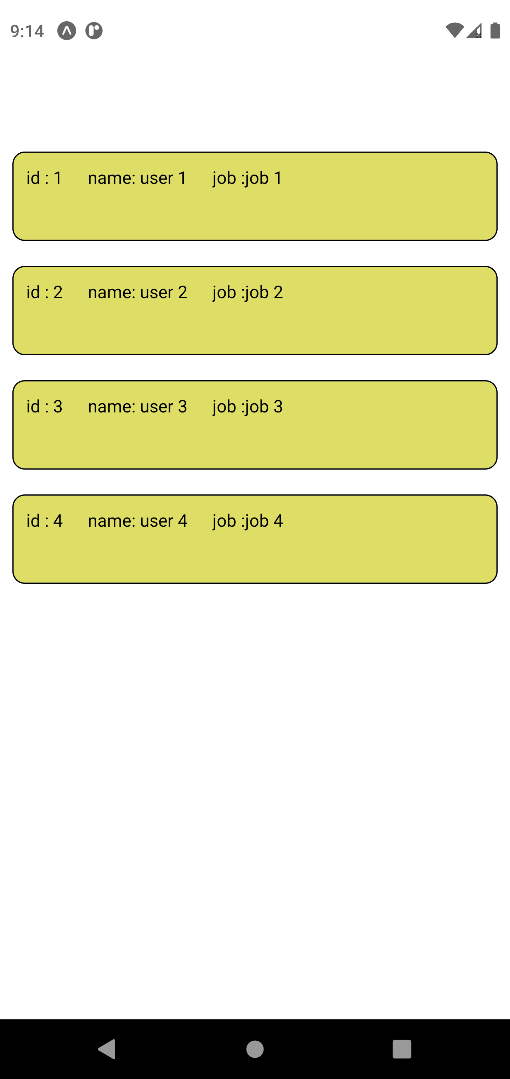
Screen :

Figure : flatlist users

### SectionList



Figure :SectionList

### Screen :

### 

Figure : SectionList Screen

### Map Array

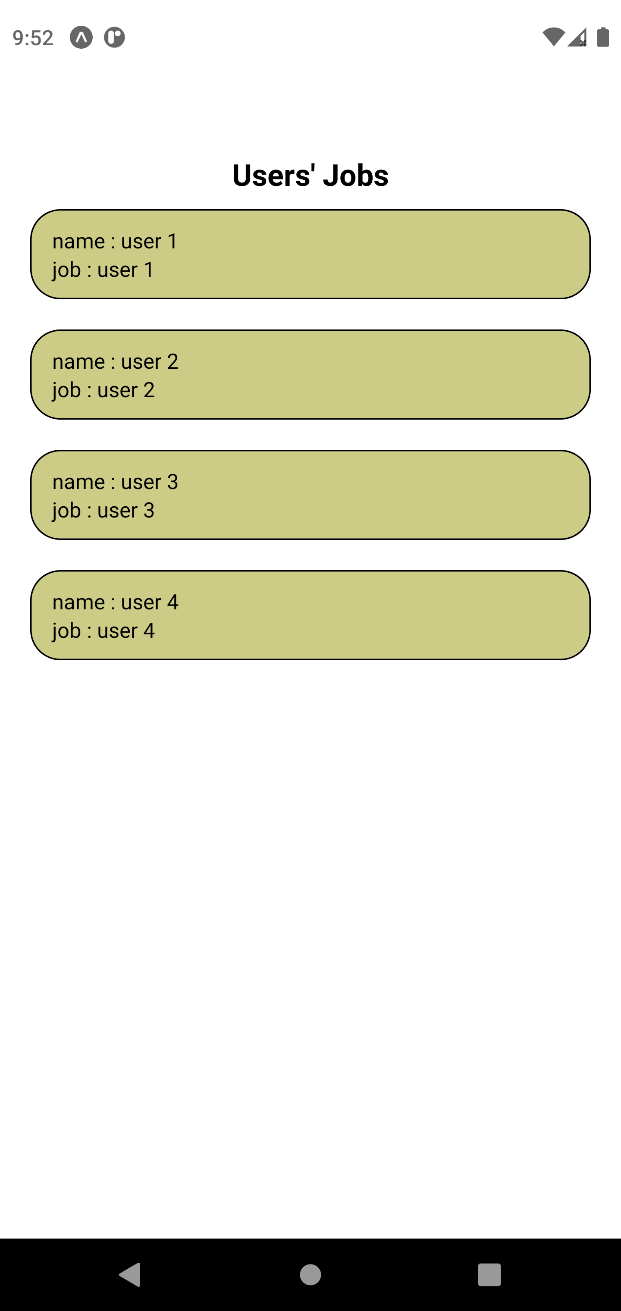
screen :

Figure :map array view

# **Chapitre 17 : Input Data**

### Exemples :

### TextInput :

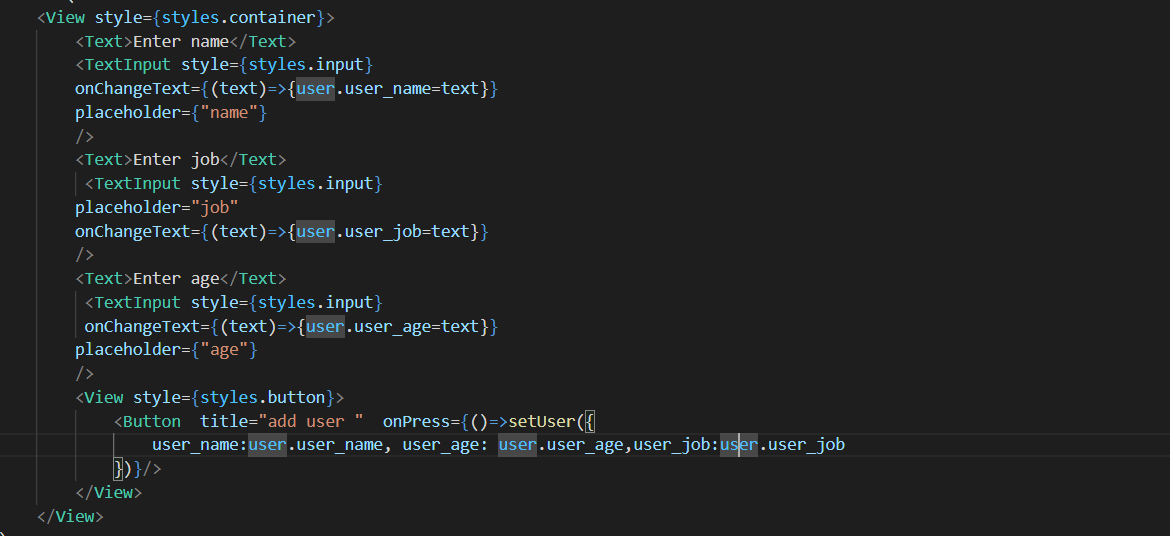
TextInput est un composant pour entrer les données via le clavier .

Figure : TextInput

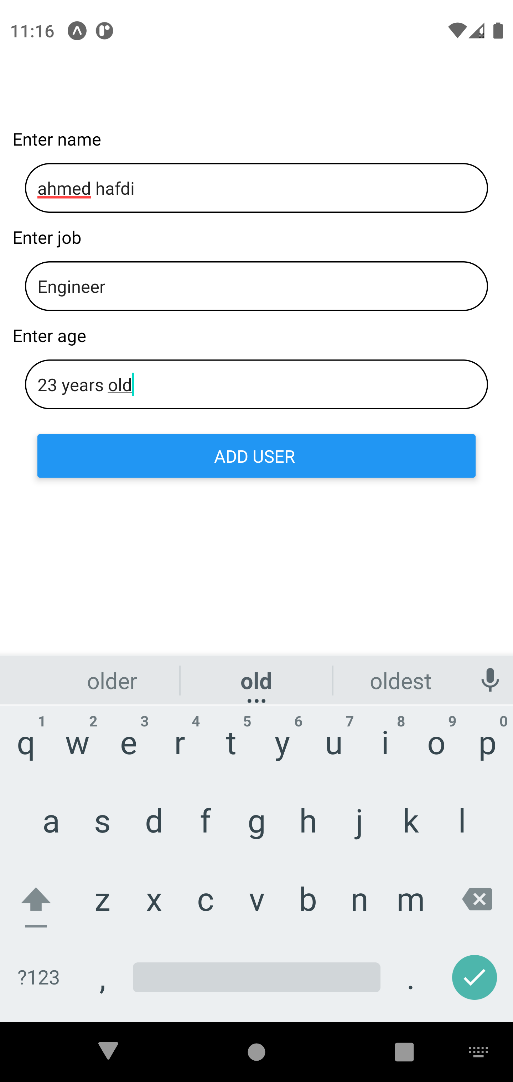


Figure : add user screen

Screen :

# **Chapitre 18 : Modal**

### 

### Introduction :

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

### Paramètres :

Tableau : 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») |