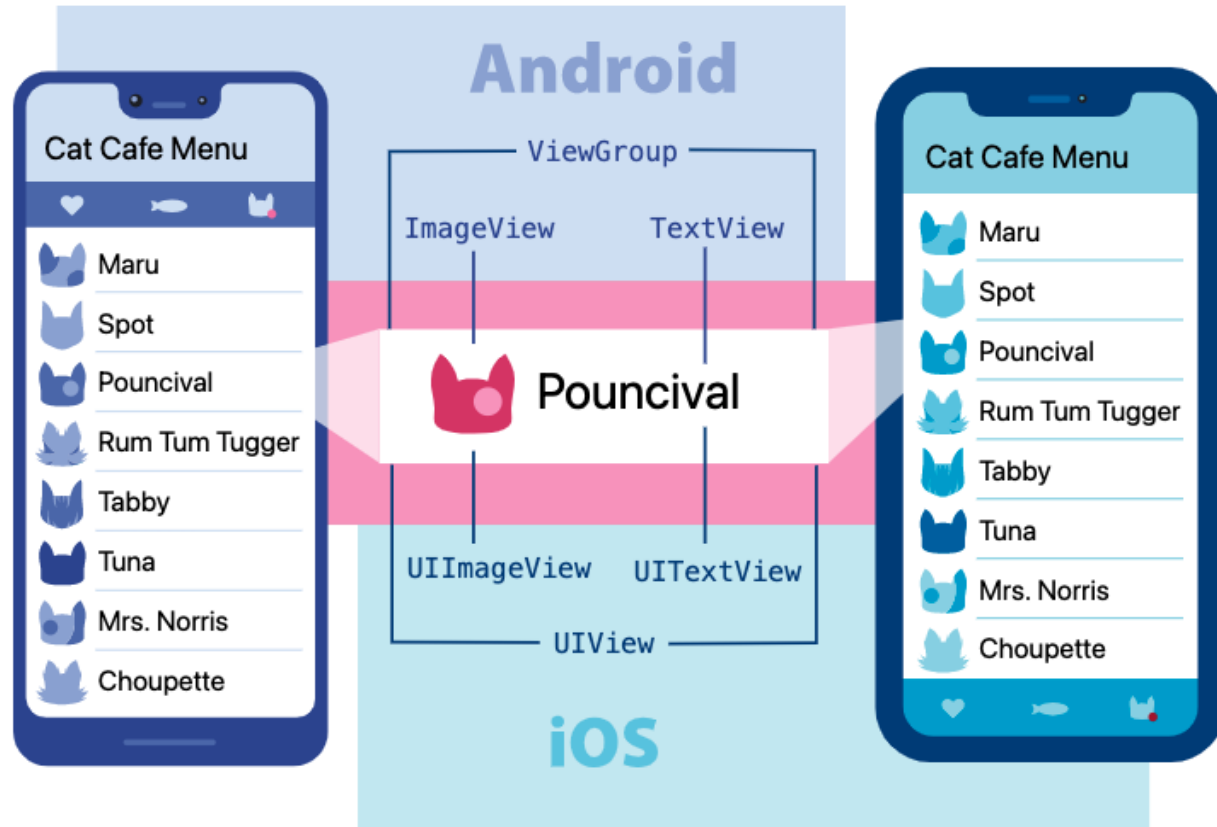


# Programación multimedia y dispositivos móviles

INTRODUCCIÓN A REACT-NATIVE



# Core components and Native Components



```
<View>  
  <Image></Image>  
  </Text></Text>  
</View>
```

# Core components and Native Components

REACT NATIVE UI COMPONENT	ANDROID VIEW	IOS VIEW	WEB ANALOG	DESCRIPTION
<code>&lt;View&gt;</code>	<code>&lt;ViewGroup&gt;</code>	<code>&lt;UIView&gt;</code>	A non-scrolling <code>&lt;div&gt;</code>	A container that supports layout with flexbox, style, some touch handling, and accessibility controls
<code>&lt;Text&gt;</code>	<code>&lt;TextView&gt;</code>	<code>&lt;UITextView&gt;</code>	<code>&lt;p&gt;</code>	Displays, styles, and nests strings of text and even handles touch events
<code>&lt;Image&gt;</code>	<code>&lt;ImageView&gt;</code>	<code>&lt;UIImageView&gt;</code>	<code>&lt;img&gt;</code>	Displays different types of images
<code>&lt;ScrollView&gt;</code>	<code>&lt;ScrollView&gt;</code>	<code>&lt;UIScrollView&gt;</code>	<code>&lt;div&gt;</code>	A generic scrolling container that can contain multiple components and views
<code>&lt;TextInput&gt;</code>	<code>&lt;EditText&gt;</code>	<code>&lt;UITextField&gt;</code>	<code>&lt;input type="text"&gt;</code>	Allows the user to enter text

# React Fundamentals

## Conceptos básicos de react

- components
- JSX
- props
- state

# Your first component

Import de los elementos necesarios

JS App.js

```
1 import React from 'react';  
2 import { Text } from 'react-native';
```

# Your first component

Crear la función principal

JS App.js

```
1  import React from 'react';
2  import { Text } from 'react-native';
3
4  function App() {
5    ..
6  }
7
8  export default App;
```

# Your first component

Renderizamos

```
JS App.js
1  import React from 'react';
2  import { Text } from 'react-native';
3
4  function App () {
5    return (
6      <Text>Hello, I am a cat!</Text>
7    );
8  }
9
10 export default App;
```

# Your first component

Otra forma de definir la función

```
JS App.js
1  import React from 'react';
2  import { Text } from 'react-native';
3
4  const App = () => {
5    return (
6      <Text>Hello, I am a cat!</Text>
7    );
8  }
9
10 export default App;
```



# Your first component

Exportar la función en su declaración

```
JS App.js
1  import React from 'react';
2  import { Text } from 'react-native';
3
4  export default function App () {
5    return (
6      <Text>Hello, I am a cat!</Text>
7    );
8  }
```

# JSX

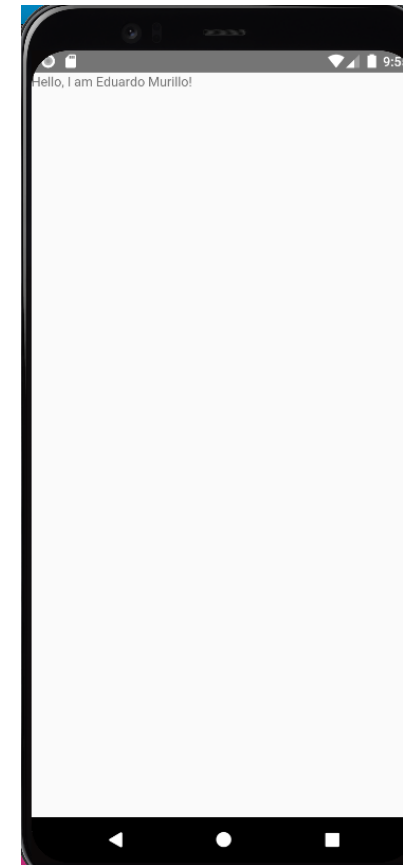
Permite escribir elementos y usar variables dentro de JavaScript

```
JS App.js
1  import React from 'react';
2  import { Text } from 'react-native';
3
4
5  export default function App () {
6    const name = "Eduardo";
7    return (
8      <Text>Hello, I am a {name}</Text>
9    );
10 }
```

# JSX

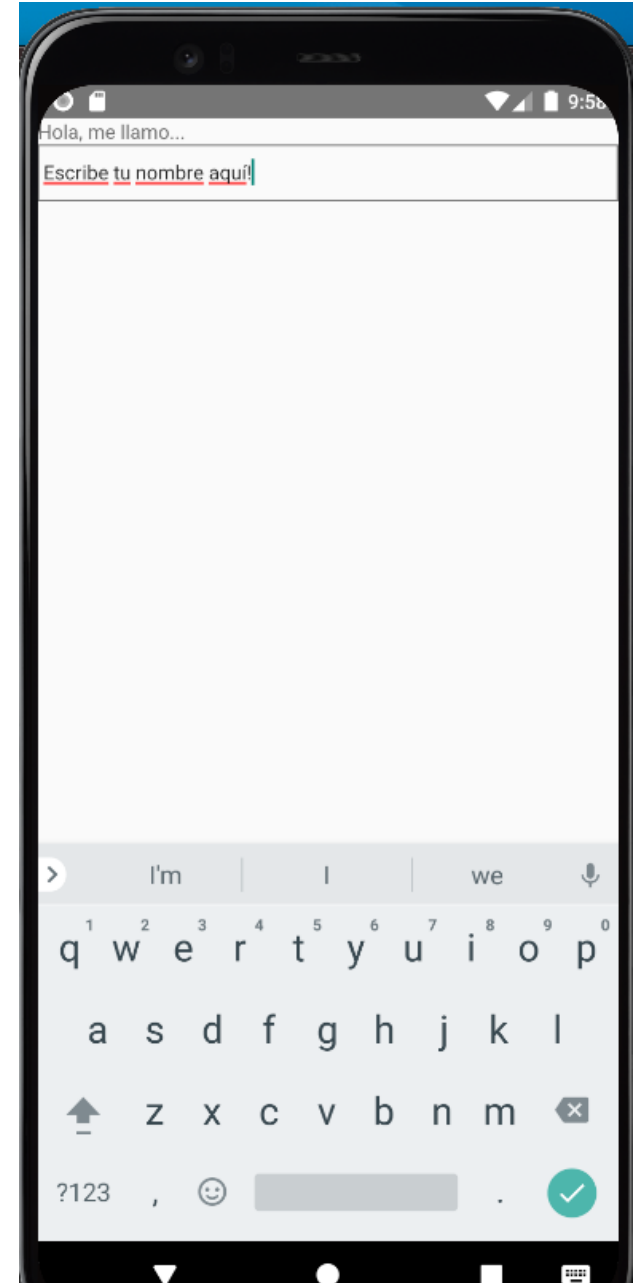
Permite escribir elementos y usar variables dentro de JavaScript

```
JS App.js ×  
1 import React from 'react';  
2 import {Text} from 'react-native';  
3  
4  
5 export default function App() {  
6   const getFullName = (firstName, secondName) => {  
7     return firstName + " " + secondName;  
8   }  
9  
10  return (  
11    <Text>Hello, I am {getFullName("Eduardo", "Murillo",)}!</Text>  
12  );  
13 }
```



# Custom Components

```
JS App.js  ×
1  import React from 'react';
2  import { Text, TextInput, View } from 'react-native';
3
4
5  export default function App () {
6
7    return (
8      <View>
9        <Text>Hola, me llamo...</Text>
10       <TextInput
11         style={{
12           height: 40,
13           borderColor: 'gray',
14           borderWidth: 1
15         }}
16         defaultValue="Escribe tu nombre aquí!"
17       />
18     </View>
19   );
20 }
```



# Custom Components

Está ensombrecido

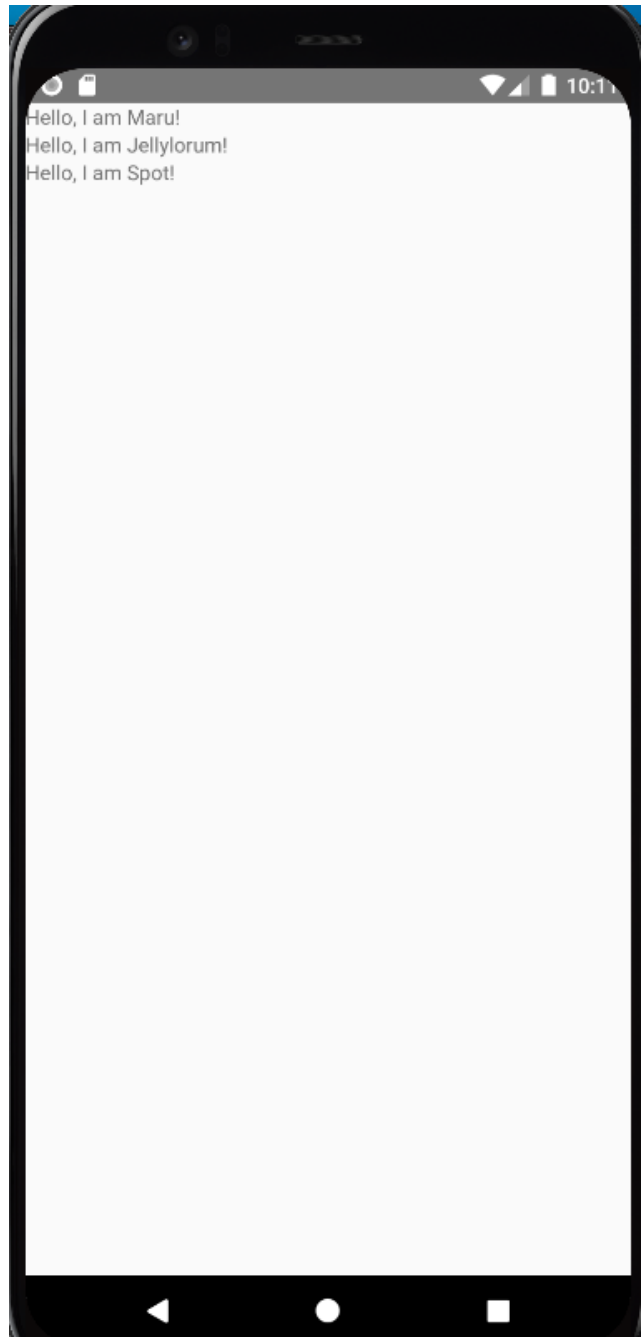
```
JS App.js  x
1  import React from 'react';
2  import { Text, TextInput, View } from 'react-native';
3
4  const Human = () => {
5    return (
6      <View>
7        <Text>I am a human</Text>
8      </View>
9    );
10 }
11
12
13
14 export default function App () {
15
16   return (
17     <View>
18       <Text>Welcome!</Text>
19       <Human />
20       <Human />
21       <Human />
22     </View>
23   );
24 }
```



# Props

JS App.js

```
1  import React from 'react';
2  import { Text, View } from 'react-native';
3
4  const Human = (props) => {
5    return (
6      <View>
7        <Text>Hello, I am {props.name}!</Text>
8      </View>
9    );
10 }
11
12 export default function App() {
13
14   return (
15     <View>
16       <Human name="Maru" />
17       <Human name="Jellylorum" />
18       <Human name="Spot" />
19     </View>
20   );
21 }
```



# State

Las variables que usamos en nuestra App. Se refrescan cada vez que haya un cambio en el renderizado.

Se definen:

```
const [isHungry, setIsHungry] = useState(true);
```

```
const [count, setCount] = useState(0);
```

```
const [name, setName] = useState(null);
```

# State

JS App.js ×

```
1 import React, { useState } from "react";
```

```
const [count, setCount] = useState(true);
```



# State

```
JS App.js ×
1  import React, { useState } from "react";
2  import { Button, Text, View } from "react-native";
3
4  export default function App() {
5    const [count, setCount] = useState(true);
6    return (
7      <View>
8        <Button
9          onPress={() => {
10            setCount(count+1);
11          }}
12          title="Cuenta"
13        />
14        <Text>Cada vez que pulso sumo 1 y llevo: {count}</Text>
15      </View>
16    );
17  }
```



# Design

```
JS App.js
1  import React from 'react';
2  import { StyleSheet, Text, View } from 'react-native';
3
4  const App = () => {
5    return (
6      <View style={styles.container}>
7        <Text style={styles.red}>just red</Text>
8        <Text style={styles.bigBlue}>just bigBlue</Text>
9        <Text style={[styles.bigBlue, styles.red]}>bigBlue, then red</Text>
10       <Text style={[styles.red, styles.bigBlue]}>red, then bigBlue</Text>
11     </View>
12   );
13 };
14
15 const styles = StyleSheet.create({
16   container: {
17     marginTop: 50,
18     backgroundColor: 'white'
19   },
20   bigBlue: {
21     color: 'blue',
22     fontWeight: 'bold',
23     fontSize: 30,
24   },
25   red: {
26     color: 'red',
27   },
28 });
29
30 export default App;
```



# Condicionales

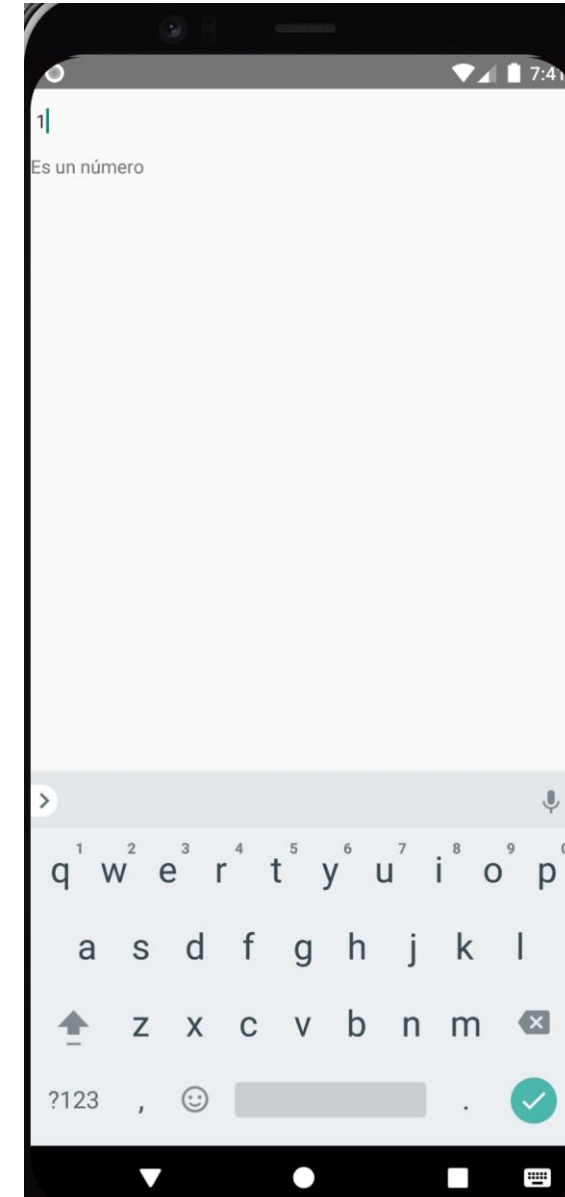
```
import React, { useState } from 'react';
import { Text, TextInput, View } from 'react-native';

export default function App() {

  const [texto, setTexto] = useState('');

  function checkInp() {
    var regex = /^[a-zA-Z]+$/;
    if (!texto.match(regex)) {
      return <Text>Es un número</Text>
    } else {
      return <Text>Es letra</Text>
    }
  }

  return (
    <View>
      <TextInput
        defaultValue=""
        placeholder="Texto"
        onChangeText={texto => setTexto(texto)}
      />
      {checkInp()}
    </View>
  )
}
```



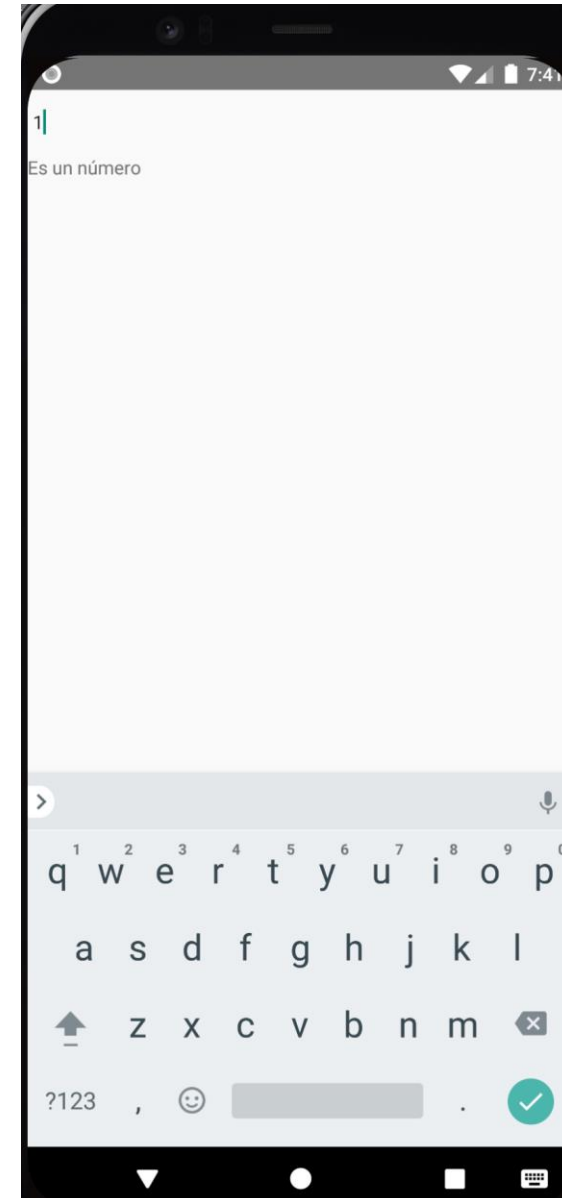
# Condicionales

```
import React, { useState } from 'react';
import { Text, TextInput, View } from 'react-native';

export default function App() {

  const [texto, setTexto] = useState('');
  var regex=/^[a-zA-Z]+$/;

  return (
    <View>
      <TextInput
        defaultValue=""
        placeholder="Texto"
        onChangeText={texto => setTexto(texto)}
      />
      {!texto.match(regex) ? <Text>Es un número</Text> : <Text>Es letra</Text>}
    </View>
  )
}
```



# Ejercicio

12:58 PM 🔔 🕒 📧 📅 ... 🔌 📶 🔋 28

Hola mi nombre es Eduardo Murillo

Escribe aquí tu edad

Edad

Si edad < 18 "¡Qué joven eres!"  
18 < edad < 19 "¡Qué buena edad!"  
Edad > 19 "¡Pedazo de edad"

Finalizar

Gracias por rellenar el formulario



# Problemas comunes

Cuerpo básico de una clase

```
JS App.js x
1 import React, { useState } from 'react';
2 import { Text, TextInput, View } from 'react-native';
3
4 export default function App() {
5
6   const [texto, setTexto] = useState('');
7
8   function miFuncion() {
9     return <Text>Hola</Text>
10  }
11
12  return (
13    <View>
14      <TextInput
15        defaultValue=""
16        placeholder="Texto"
17        onChangeText={x => setTexto(x)}
18      />
19      <Text>{texto}</Text>
20      {miFuncion()}
21    </View>
22  )
23 }
24
25
```

Definición de mi clase o función principal

Defino mis Hooks (variables que cambian en el tiempo)

Definición funciones auxiliares

Return o renderizado principal

# Problemas comunes

## Diferencia entre llamar a funciones y variables

Llamar a una variable

{miVariable}

Llamar a una función

{miFunción()}

# Problemas comunes

## Diferencias entre componentes

Button

```
<Button  
  title="Press me"  
  onPress={() => Alert.alert('Presiono')}  
>
```

TextInput

```
<TextInput  
  onChangeText={x => setText(x)}  
  value={text}  
>
```



# Especificaciones de desarrollo

1. El nombre de la clase usa el estilo UpperCamelCase, debe seguir el caso de camello y la primera letra debe estar en mayúscula;

**LoginPage/MiClase**

2. Los nombres de los métodos, los nombres de los parámetros, las variables y las variables locales usan el estilo lowerCamelCase de manera uniforme y deben seguir el caso camel, y la primera letra debe estar en minúsculas;

**MiFunción() / inputUserId**

3. Los nombres de las constantes todo en mayúscula y las palabras están separadas por guiones bajos, para que la expresión semántica sea completa y clara, y los nombres no sean demasiado largos

**MI\_CONSTANTE**

4. Ningún nombre en el código puede comenzar con un guión bajo o un signo de dólar, ni puede terminar con un guión bajo o un signo de dólar

**\_MiClase**

# Core components

## Image

```
<Image
  style={styles.myImageStyle}
  source={{
    uri: 'https://reactnative.dev/img/tiny_logo.png',
  }}
/>
```

# Core components

## Image

```
<Image
  style={{width: 50, height: 50}
  source={{
    uri: 'https://reactnative.dev/img/tiny_logo.png',
  }}
/>
```

# Core components

## Switch

```
<Switch  
  trackColor={{ false: 'green', true: 'yellow' }}  
  thumbColor={isEnabled ? 'blue' : 'red'}  
  onValueChange={() => setIsEnabled(previousState => !previousState)}  
  value={isEnabled}  
>
```

Valor del Switch

Color de la línea sobre la que se esplaza

Color del botón que se desplaza

Función que se activa al pulsar el switch

# Ejercicio

Hacer una pantalla que al activar un Switch aparezca una imagen

```
<Switch
  trackColor={{ false: 'green', true: 'yellow' }}
  thumbColor={isEnabled ? 'blue' : 'red'}
  onChange={() => setIsEnabled(previousState => !previousState)}
  value={isEnabled}
/>

<Image
  style={{width: 50, height: 50}}
  source={{
    uri: 'https://reactnative.dev/img/tiny\_logo.png',
  }}
/>
```

# Core components

## FlatList

Convierte un grupo de elementos en una lista

```
<FlatList  
  data={DATA}  
  renderItem={renderItem}  
  keyExtractor={item => item.id}  
>
```

Datos a mostrar

Función para mostrar los datos, recibe como parámetro un 'item', que es cada elemento de la lista

Se usa para cachear e indexar los elementos internamente. Suele ser el id de los elementos de entrada

# Core components

## FlatList

```
import React from 'react';
import { Text, View, FlatList } from 'react-native';

export default function App() {

  const DATA = [
    { id: 'bd7acbea-c1b1-46c2-aed5-3ad53abb28ba',
      title: 'Primer elemento', },
    { id: '3ac68afc-c605-48d3-a4f8-fbd91aa97f63',
      title: 'Segundo elemento', },
    { id: '58694a0f-3da1-471f-bd96-145571e29d72',
      title: 'Tercer elemento', },
  ];

  const renderItem = ({ item }) => (
    <View >
      <Text >{item.title}</Text>
    </View>
  );

  return (
    <View>
      <FlatList
        data={DATA}
        renderItem={renderItem}
        keyExtractor={item => item.id}
      />
    </View>
  )
}
```

# Ejercicio

Crear una lista de 20 elementos



# Core components

ScrollView

Permite hacer scroll en una pantalla

```
<ScrollView style={{backgroundColor: 'red'}}>  
  <Text style={{fontSize: 42,}}>  
    Lorem ipsum dolor sit amet, consectetur  
    adipiscing elit, sed do eiusmod tempor incididunt ut  
    labore et dolore magna aliqua. Ut enim ad minim  
    veniam, quis nostrud exercitation ullamco laboris nisi  
    ut aliquip ex ea commodo consequat. Duis aute  
    irure dolor in reprehenderit in voluptate velit esse  
    cillum dolore eu fugiat nulla pariatur. Excepteur sint  
    occaecat cupidatat non proident, sunt in culpa qui  
    officia deserunt mollit anim id est laborum.  
  </Text>  
</ScrollView>
```

# Ejercicio

Crear pantalla sin scroll para luego meter el scrollview

# Core components

TouchableHighlight

TouchableOpacity

```
return (  
  <View>  
    <TouchableOpacity  
      style={{ alignItems: "center", backgroundColor: "red", padding: 10 }}  
      onPress={() => setCount(count + 1)}  
    >  
      <Text>Press Here</Text>  
    </TouchableOpacity>  
    <TouchableHighlight  
      style={{ alignItems: "center", backgroundColor: "blue", padding: 10 }}  
      onPress={() => setCount(count + 1)}  
    >  
      <Text>Press Here</Text>  
    </TouchableHighlight>  
  </View>  
)
```

# Core components

TouchableHighlight

TouchableOpacity

```
return (  
  <View>  
    <TouchableOpacity  
      style={{ alignItems: "center", backgroundColor: "red", padding: 10 }}  
      onPress={() => setCount(count + 1)}  
    >  
      <Text>Press Here</Text>  
    </TouchableOpacity>  
    <TouchableHighlight  
      style={{ alignItems: "center", backgroundColor: "blue", padding: 10 }}  
      onPress={() => setCount(count + 1)}  
    >  
      <Text>Press Here</Text>  
    </TouchableHighlight>  
  </View>  
)
```

# Styles

Flex

Define la distribución de los elementos en la pantalla

```
return (  
  <View style={[styles.container, {  
    flexDirection: "column"  
  }]}>  
    <View style={{ flex: 1, backgroundColor: "red" }} />  
    <View style={{ flex: 1, backgroundColor: "blue" }} />  
    <View style={{ flex: 1, backgroundColor: "green" }} />  
  </View>  
);  
  
const styles = StyleSheet.create({  
  container: {  
    flex: 1,  
    padding: 20,  
  },  
});
```

# Navigating Between Screens

React Navigation

**Documentación:** <https://reactnavigation.org/docs/getting-started/>

# Navigating Between Screens

## Instalar React Navigation

```
npm install @react-navigation/native
```

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

```
npm install @react-navigation/native-stack
```

# Navigating Between Screens

## Uso

```
import React from 'react';
import { Text, View } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';

function HomeScreen() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
    </View>
  );
}

const Stack = createNativeStackNavigator();

export default function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator initialRouteName="Home">
        <Stack.Screen name="Home" component={HomeScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

Dentro irán todas las pantallas de la app. Suele ir en el index o App.js

Los Stack devuelven dos propiedades:  
Screen  
Navigator



# Navigating Between Screens

## Uso

```
import React from 'react';
import { Text, View } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';

function HomeScreen() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
    </View>
  );
}

const Stack = createNativeStackNavigator();

export default function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator initialRouteName="Home">
        <Stack.Screen name="Home" component={HomeScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

Nombre para identificar la navegación

Renderizado de la pantalla

# Navigating Between Screens

## Uso

```
import React from 'react';
import { Text, View } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';

function HomeScreen() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
    </View>
  );
}

const Stack = createNativeStackNavigator();

export default function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator initialRouteName="Home">
        <Stack.Screen name="Home" component={HomeScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

Dentro irán todas las pantallas de la app. Suele ir en el index o App.js

Los Stack devuelven dos propiedades:  
Screen  
Navigator

# Navigating Between Screens

## React Navigation: initialRouteName

```
import React from 'react';
import { Text, View } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';

function HomeScreen() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
    </View>
  );
}

function DetailsScreen() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Details Screen1</Text>
    </View>
  );
}

const Stack = createNativeStackNavigator();

export default function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator initialRouteName="Details">
        <Stack.Screen name="Home" component={HomeScreen} />
        <Stack.Screen name="Details" component={DetailsScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

Indica la ruta que se inicia dentro del Stack

# Navigating Between Screens

## React Navigation: Options

<https://reactnavigation.org/docs/native-stack-navigator/>

```
import React from 'react';
import { Text, View } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';

function HomeScreen() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
    </View>
  );
}

function DetailsScreen() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Details Screen</Text>
    </View>
  );
}

const Stack = createNativeStackNavigator();

export default function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator initialRouteName="Details">
        <Stack.Screen name="Home" component={HomeScreen} />
        <Stack.Screen name="Details" component={DetailsScreen} options={{title: 'Mis detalles'}} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

Opciones de la  
pantalla



# Navigating Between Screens

## React Navigation

```
function DetailsScreen({navigation}) {  
  return (  
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>  
      <Text>Details Screen1</Text>  
      <Button  
        title="Go to Details... again"  
        onPress={() => navigation.push('Details')}  
      />  
      <Button  
        title="Go to Home"  
        onPress={() => navigation.navigate('Home')}  
      />  
      <Button title="Go back" onPress={() => navigation.goBack()} />  
      <Button  
        title="Go back to first screen in stack"  
        onPress={() => navigation.popToTop()}  
      />  
    </View>  
  );  
}
```

Inserta al principio de la cola  
la pantalla Details

Navega al Home

Va a la pantalla anterior

Va a la primera pantalla del  
Stack

# Navigating Between Screens

## React Navigation: Parameters

```
navigation.navigate('RouteName', { /* params go here */ })
```

```
navigation.navigate(Details, { userName: 'Antonio', edad: 23 })
```

Los parámetros se guardan como si fueran variables:

```
const { userName } = route.params;
```

# Navigating Between Screens

## React Navigation: Initial parameters

```
<Stack.Screen name="Home" component={HomeScreen} initialParams={{userName:"Antonio"}}  
/>
```

# Navigating Between Screens

## React Navigation: Anti-pattern

```
navigation.navigate('Profile',{
  user: {
    id: 'jane',
    firstName: 'Jane',
    lastName: 'Done',
    age: 25,
  },
});
```



# Navigating Between Screens

## React Navigation: Parameters

```
import React from 'react';
import { Text, View, Button } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';

function HomeScreen({navigation, route}) {
  console.log("-----route: ", route);
  const { userName } = route.params;
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Home Screen</Text>
      <Text>Hola {userName}</Text>
      <Button
        title="Go to Details"
        onPress={() => navigation.navigate('Details', {userName: 'Manolillo'})}
      />
    </View>
  );
}

function DetailsScreen({navigation, route}) {
  console.log("-----route: ", route);
  const { userName } = route.params;
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <Text>Details Screen</Text>
      <Text>Hola {userName}</Text>
      <Button
        title="Go to Home"
        onPress={() => navigation.navigate('Home', {userName: 'Jorge'})}
      />
    </View>
  );
}

const Stack = createNativeStackNavigator();

export default function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator initialRouteName="Home">
        <Stack.Screen name="Home" component={HomeScreen} initialParams={{userName: "Antonio"}} />
        <Stack.Screen name="Details" component={DetailsScreen} options={{title: 'Mis detalles'}} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

# Navigating Between Screens

## Ejercicio

La App tiene las siguientes características:

- Crear una navegación entre dos pantallas con el título de la pantalla centrado
- La primera pantalla recibe el valor de la variable username como parámetro inicial del stack
- La primera pantalla envía a la segunda la variable 'userName' como parámetro y la segunda la pinta por pantalla

# Navigating Between Screens

## Ejercicio

La App tiene las siguientes características:

- Crear una navegación entre dos pantallas con el título de la pantalla centrado
- La primera pantalla tiene dos `textInput` donde le pregunta al usuario su nombre y edad para enviarlo a la segunda pantalla
- La segunda pantalla muestra el nombre y la edad del usuario

# Navigating Between Screens

## React Navigation: Header Styles

```
export default function App() {  
  return (  
    <NavigationContainer>  
      <Stack.Navigator initialRouteName="Home">  
        <Stack.Screen name="Home" component={HomeScreen} options={{  
          title: 'My home',  
          headerStyle: {  
            backgroundColor: '#f4511e',  
          },  
          headerTintColor: 'white',  
          headerTitleStyle: {  
            fontWeight: 'bold',  
          },  
        }} />  
      </Stack.Navigator>  
    </NavigationContainer>  
  );  
}
```

# Navigating Between Screens

## React Navigation: Header Styles

```
export default function App() {  
  return (  
    <NavigationContainer>  
      <Stack.Navigator initialRouteName="Home">  
        <Stack.Screen name="Home" component={HomeScreen} options={{  
          title: 'My home',  
          headerStyle: {  
            backgroundColor: '#f4511e',  
          },  
          headerTintColor: 'white',  
          headerTitleStyle: {  
            fontWeight: 'bold'  
          },  
          headerTitleAlign: 'center',  
          headerRight: () => (  
            <Button  
              onPress={() => alert('This is a button!')}  
              title="Info"  
              color="#00cc00"  
            />  
          ),  
          // headerLeft: () => (<View />)  
        }} />  
      </Stack.Navigator>  
    </NavigationContainer>  
  );  
}
```

# Navigating Between Screens

## React Navigation: Tab Screen

```
npm install --save react-native-vector-icons
```

Editar android/app/build.gradle ( NO android/build.gradle ) y añadir  
apply from: "../node\_modules/react-native-vector-icons/fonts.gradle"  
(línea 85)

# Navigating Between Screens

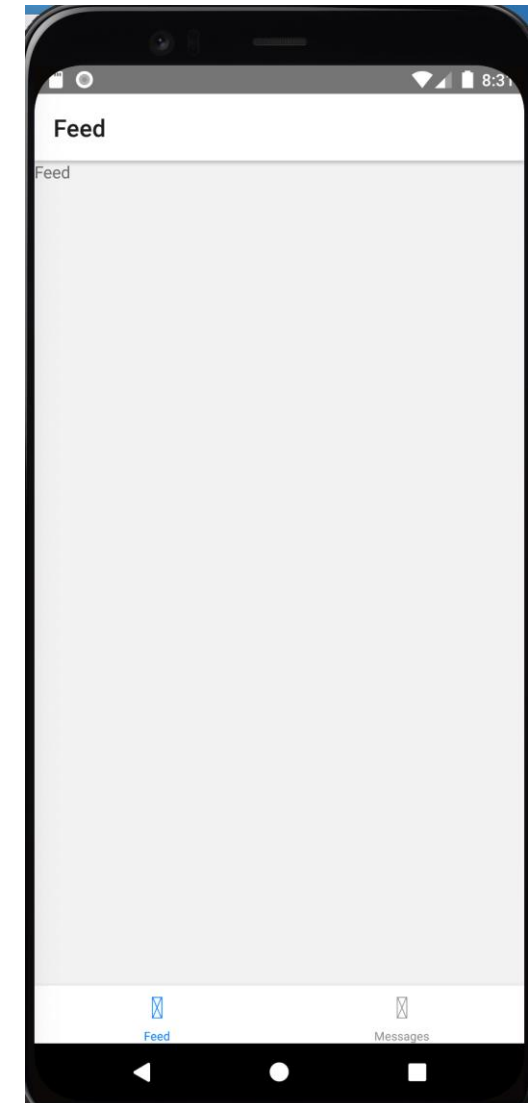
**React Navigation: Tab Screen**

```
npm install @react-navigation/bottom-tabs
```

# Navigating Between Screens

## React Navigation: Tab screen

```
function Home() {  
  return (  
    <Tab.Navigator>  
      <Tab.Screen name="Feed" component={FeedScreen} />  
      <Tab.Screen name="Messages" component={MessagesScreen} />  
    </Tab.Navigator>  
  );  
}  
  
export default function App() {  
  return (  
    <NavigationContainer>  
      <Stack.Navigator>  
        <Stack.Screen  
          name="Home"  
          component={Home}  
          options={{ headerShown: false }}  
        />  
        <Stack.Screen name="Profile" component={ProfileScreen} />  
        <Stack.Screen name="Settings" component={SettingsScreen} />  
      </Stack.Navigator>  
    </NavigationContainer>  
  );  
}
```





# Navigating Between Screens

## React Navigation: Tab screen

```
function Home() {  
  return (  
    <Tab.Navigator  
      screenOptions={({ route }) => ({  
        tabBarIcon: ({ focused, color, size }) => {  
          let iconName;  
          if (route.name === 'Feed') {  
            iconName = focused  
              ? 'ios-information-circle'  
              : 'ios-information-circle-outline';  
          } else if (route.name === 'Messages') {  
            iconName = focused ? 'paw' : 'paw-outline';  
          }  
          return <Ionicons name={iconName} size={size} color={color} />;  
        },  
        tabBarActiveTintColor: 'tomato',  
        tabBarInactiveTintColor: 'grey',  
      })  
    <Tab.Screen name="Feed" component={FeedScreen} />  
    <Tab.Screen name="Messages" component={MessagesScreen} />  
  </Tab.Navigator>  
  );  
}
```



# Navigating Between Screens

## React Navigation: Tab screen

```
export default function App() {  
  return (  
    <NavigationContainer>  
      <Tab.Navigator  
        screenOptions={({ route }) => ({  
          tabBarIcon: ({ focused, color, size }) => {  
            let iconName;  
            if (route.name === 'Home') {  
              iconName = focused  
                ? 'ios-information-circle'  
                : 'ios-information-circle-outline';  
            } else if (route.name === 'Settings') {  
              iconName = focused ? 'paw' : 'paw-outline';  
            }  
            return <Ionicons name={iconName} size={size} color={color} />;  
          },  
          tabBarActiveTintColor: 'tomato',  
          tabBarInactiveTintColor: 'grey',  
        ))}  
      <Tab.Screen options={{ headerShown: false }} name="Home" component={HomeStackScreen} />  
      <Tab.Screen options={{ headerShown: false }} name="Settings" component={SettingsStackScreen} />  
    </Tab.Navigator>  
  </NavigationContainer>  
);  
}
```



# Navigating Between Screens

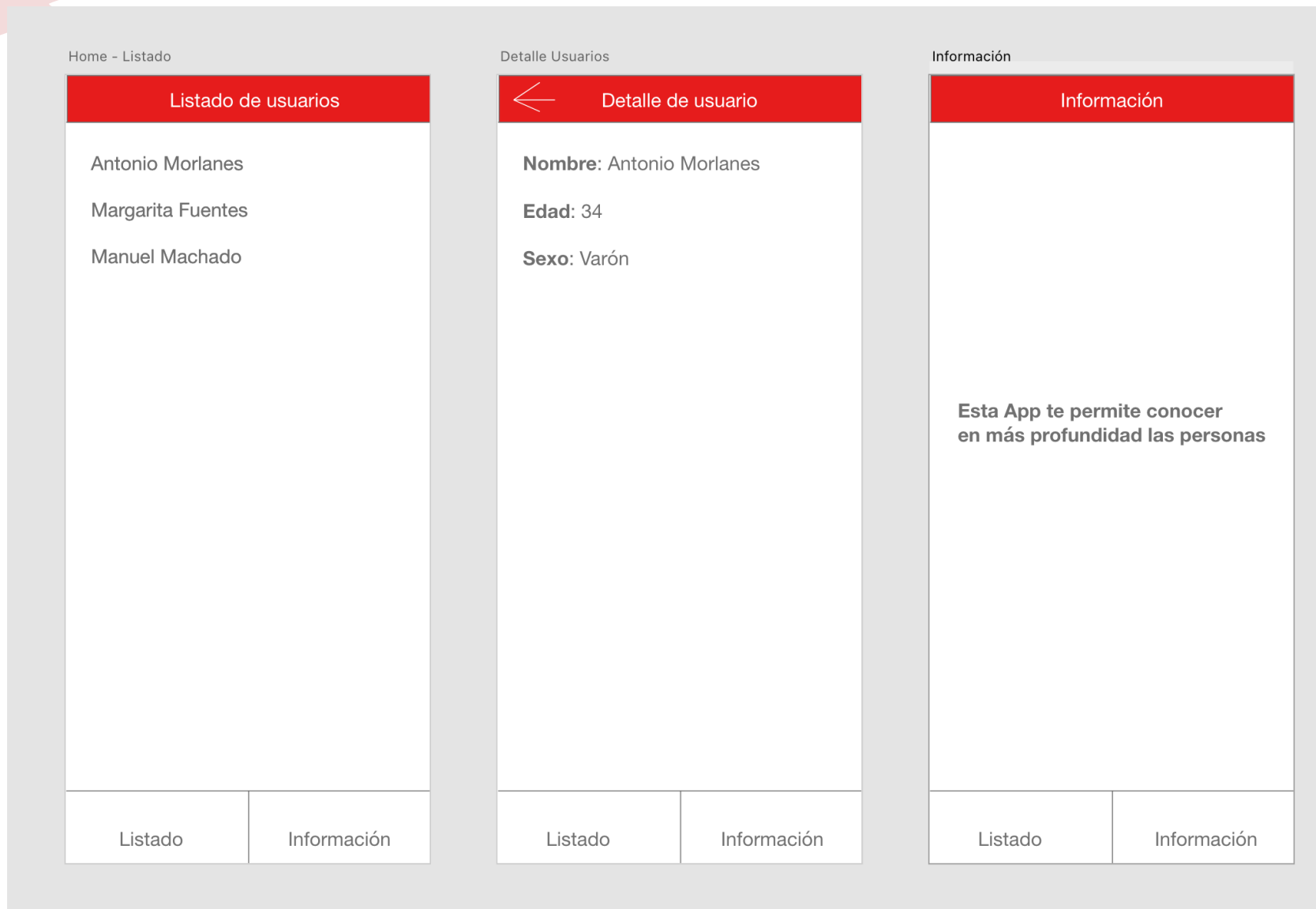
## Ejercicio

La App tiene las siguientes características:

- La página principal tiene 2 tabs (Listado e Info)
- Pantalla Listado: Contiene un listado de nombre de usuarios
- Al pinchar en el usuario te abre una pantalla de detalle con el nombre, edad y sexo
- Info: Contiene información sobre el uso de la app

# Navigating Between Screens

## Ejercicio



# Structure folders

src

**api** (llamadas a backed)

**assets** (imágenes)

**components** (componentes creados por nosotros)

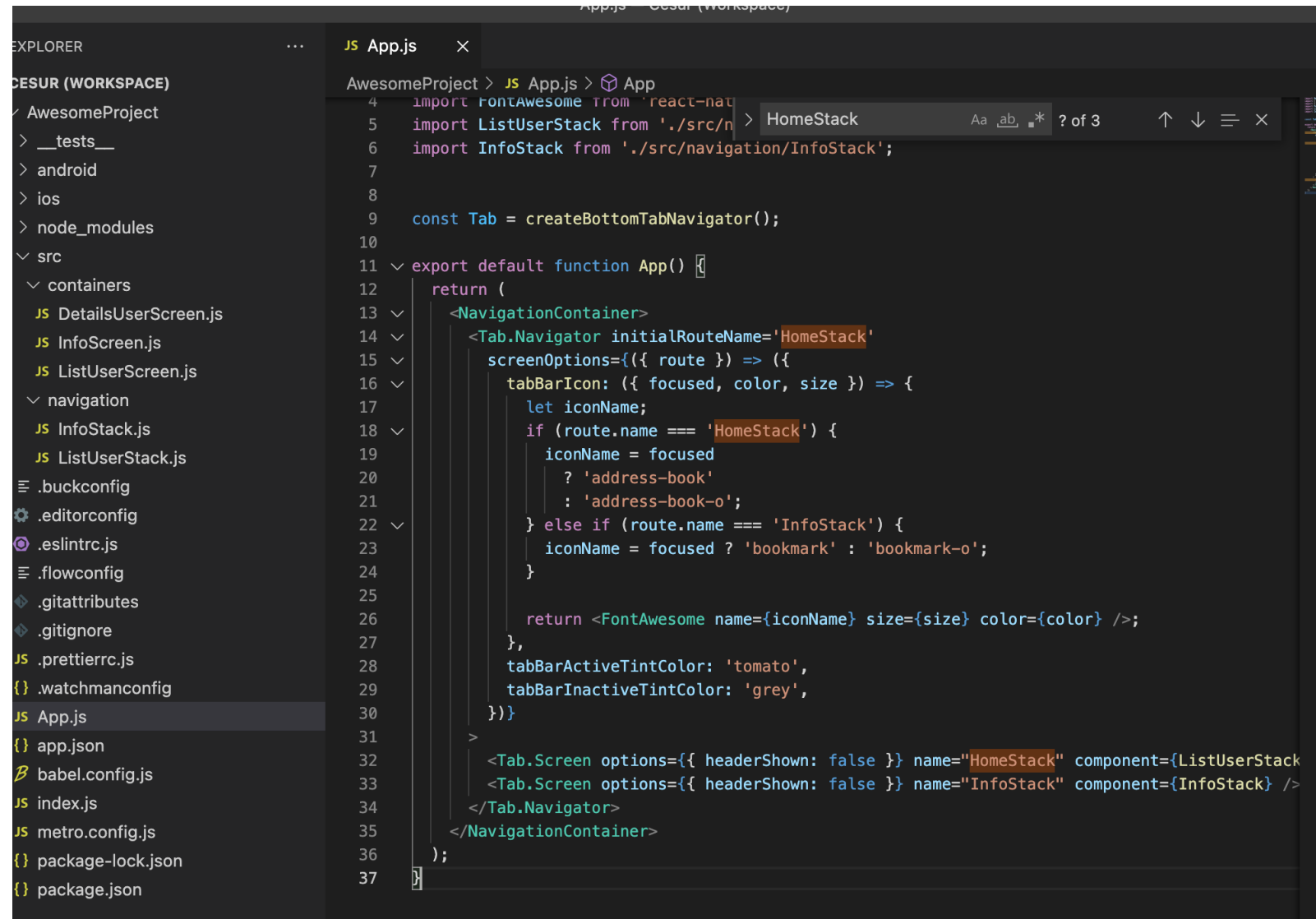
**containers** (pantallas)

**navigation** (Stack o tab o cualquier otro componente de navegación)

**styles**

**Utilities** (ficheros que contengan reglas o elementos comunes como un regex)

# Structure folders



The image shows a screenshot of the Visual Studio Code editor. On the left, the 'EXPLORER' sidebar displays the project structure for 'CESUR (WORKSPACE)'. The structure includes a root folder 'AwesomeProject' with subfolders '.\_\_tests\_\_', 'android', 'ios', 'node\_modules', and 'src'. The 'src' folder is expanded, showing 'containers' (with 'DetailsUserScreen.js', 'InfoScreen.js', and 'ListUserScreen.js') and 'navigation' (with 'InfoStack.js' and 'ListUserStack.js'). Other files in the workspace include '.buckconfig', '.editorconfig', '.eslintrc.js', '.flowconfig', '.gitattributes', '.gitignore', '.prettierrc.js', '.watchmanconfig', 'App.js' (selected), 'app.json', 'babel.config.js', 'index.js', 'metro.config.js', 'package-lock.json', and 'package.json'.

The main editor area shows the 'App.js' file. The code is as follows:

```
4 import { FontAwesomeIcon } from 'react-native';
5 import ListUserStack from './src/navigation/ListUserStack';
6 import InfoStack from './src/navigation/InfoStack';
7
8
9 const Tab = createBottomTabNavigator();
10
11 export default function App() {
12   return (
13     <NavigationContainer>
14       <Tab.Navigator initialRouteName='HomeStack'
15         screenOptions={({ route }) => ({
16           tabBarIcon: ({ focused, color, size }) => {
17             let iconName;
18             if (route.name === 'HomeStack') {
19               iconName = focused
20                 ? 'address-book'
21                 : 'address-book-o';
22             } else if (route.name === 'InfoStack') {
23               iconName = focused ? 'bookmark' : 'bookmark-o';
24             }
25
26             return <FontAwesomeIcon name={iconName} size={size} color={color} />;
27           },
28           tabBarActiveTintColor: 'tomato',
29           tabBarInactiveTintColor: 'grey',
30         })
31       >
32         <Tab.Screen options={{ headerShown: false }} name="HomeStack" component={ListUserStack} />
33         <Tab.Screen options={{ headerShown: false }} name="InfoStack" component={InfoStack} />
34       </Tab.Navigator>
35     </NavigationContainer>
36   );
37 }
```

# Examen

1. La aplicación tendrá dos tabs, Historia y Usuarios
2. El menú Historia tendrá una recopilación de imágenes y texto de la historia de los dispositivos móviles
3. El menú Usuarios tendrá una primera pantalla para filtrar por edad un conjunto de usuarios previamente cargados
4. Al pulsar el botón 'Buscar', cambiará de pantalla para mostrar los usuarios que tengan una edad menor a la insertada

# Examen

