	<p style="text-align: center;">UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA, ESCUELA DE COMPUTACIÓN</p>
<p style="text-align: center;">Ciclo I</p>	<p style="text-align: center;">DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA</p> <p style="text-align: center;">Guía de Laboratorio No. 5 Configuración de entorno React Native</p>

I. RESULTADO DE APRENDIZAJE

Que el estudiante aprenda a:

- Configure el entorno de trabajo para realizar aplicaciones móviles.
- Utilice Visual Studio Code para la crear proyectos móviles.

II. CONFIGURANDO ENTORNO DE TRABAJO

Antes de comenzar con los primeros en el uso de React Native, debe verificar que posea el Software necesario para desarrollar y ejecutar esta tecnología. A continuación se le muestra las herramientas mínimas que debe poseer en su computadora:

1. **Software Developer Kit de Java** (SDK - Versión 12.0.2 o superior). El SDK reúne un grupo de herramientas que permiten la programación de aplicaciones móviles. Puede descargarlo en el siguiente enlace. Seleccione el tipo de instalador, según su la arquitectura y Sistema Operativo. Luego proceda a instalar el software.

<https://www.oracle.com/java/technologies/javase-downloads.html>



Java SE Development Kit 16.0.2		
This software is licensed under the Oracle Technology Network License Agreement for Oracle Java SE		
Product / File Description	File Size	Download
Linux ARM 64 RPM Package	144.87 MB	jdk-16.0.2_linux-aarch64_bin.rpm
Linux ARM 64 Compressed Archive	160.75 MB	jdk-16.0.2_linux-aarch64_bin.tar.gz
Linux x64 Debian Package	146.17 MB	jdk-16.0.2_linux-x64_bin.deb
Linux x64 RPM Package	155.01 MB	jdk-16.0.2_linux-x64_bin.rpm
Linux x64 Compressed Archive	170.04 MB	jdk-16.0.2_linux-x64_bin.tar.gz
macOS Installer	166.6 MB	jdk-16.0.2_osx-x64_bin.dmg
macOS Compressed Archive	167.21 MB	jdk-16.0.2_osx-x64_bin.tar.gz
Windows x64 Installer	150.88 MB	jdk-16.0.2_windows-x64_bin.exe
Windows x64 Compressed Archive	168.8 MB	jdk-16.0.2_windows-x64_bin.zip

2. **Node.js**, ideado como entorno de ejecución de JavaScript orientado a eventos asíncronos, Node.js está diseñado para crear aplicaciones network escalables. Puede descargarlo e instalarlo en su computadora. <https://nodejs.org/es/>



Node.js® es un entorno de ejecución para JavaScript construido con V8, motor de JavaScript de Chrome.

Descargar para Windows (x64)

18.17.1 LTS
Recomendado para la mayoría

20.5.1 Actual
Últimas características

Otras Descargas | Cambios | Documentación de la API

O eche un vistazo al Programa de soporte a largo plazo (LTS).

3. **npm**, es un administrador de paquetes de JavaScript y gestor de dependencias. También se desempeña como administrador de proyectos.



Comando de instalación
npm install [<package-spec> ...]

Comando para verificar la instalación
npm -v

4. **Expo**, es una plataforma de código abierto para crear aplicaciones nativas para Android, iOS y web con JavaScript y React. Puede verificar los pasos de instalación en la siguiente página:
<https://docs.expo.dev/workflow/expo-cli/>



Comando de instalación

`yarn global add expo-cli`



Comando para verificar la instalación

5. **Visual Studio Code**, es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. Puede descargarlo e instalarlo en su computadora: <https://code.visualstudio.com/download>



6. **Instalación de extensiones para Visual Studio Code**, puede ingresar a la siguiente página web para buscar las extensiones que se muestran en el cuadro de abajo <https://marketplace.visualstudio.com/VSCode> o puede utilizar su Visual Studio Code -> Menú "Ver" -> Extensiones (Ctrl + Mayús + X)

#	Extensión	Descripción
6.1	 Auto Close Tag <small>formulahendry.auto-close-tag</small> Jun Han 2,329,612 ★★★★★ Repository Automatically add HTML/XML close tag, same as Visual Studio IDE or Sublime Text Install	Permite agregar automáticamente la etiqueta de cierre HTML/XML.
6.2	 Bracket Pair Colorizer <small>coenraads.bracket-pair-colorizer</small> CoenraadS 2,304,900 ★★★★★ Repository License A customizable extension for colorizing matching brackets Install	Esta extensión permite identificar los corchetes con los colores. El usuario puede definir qué caracteres coincidir y qué colores usar.
6.3	 DotENV <small>mikestead.dotenv</small> mikestead 738,892 ★★★★★ Repository License Support for dotenv file syntax Install	Soporte para la sintaxis de archivos dotenv (variables de entorno)
6.4	 Firebase <small>toba.vsfire</small> toba 47,747 ★★★★★ Repository License Firestore Security Rules syntax highlighting Install	Resaltado de sintaxis de reglas de seguridad de Firebase.
6.5	 Prettier - Code formatter <small>esbenp.prettier-vscode</small> Esben Petersen 4,890,186 ★★★★★ Repository License Code formatter using prettier Install	Extensión que permite formatear (organizar) el código.

6.6	 Project Manager <small>alefragnani.project-manager</small> Alessandro Fragnani 830,749 ★★★★★ Repository Licer Easily switch between projects Install	Extensión que permite la organización y manejo fácil de proyectos.
6.7	 React Native Tools <small>msjsdiag.vscode-react-native</small> Microsoft 1,124,401 ★★★★★ Repository License Debugging and integrated commands for React Native Install	Extensión que permite la depuración de los comandos utilizados en React Native.

7. **Android Studio**, proporciona las herramientas más rápidas para crear aplicaciones en todo dispositivo Android. Puede descargarlo e instalarlo en el siguiente enlace: <https://developer.android.com/studio>
8. **AVD Manager**, configurar un dispositivo virtual en Android Studio.

IV. DESARROLLO DE PRÁCTICA

PARTE 1

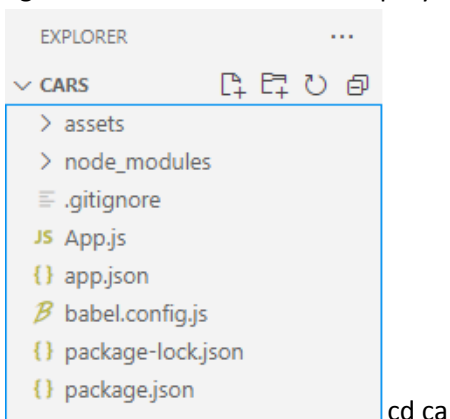
1. Abra la consola de comandos y ejecutar como administrador.
2. Ubicarse en consola en la dirección donde desea guardar su proyecto
3. En la consola de comandos ejecuta: **npx create-expo-app@latest cars --template blank**

```
C:\Users\Karens_Medrano\Desktop>npx create-expo-app@latest cars --template blank
```

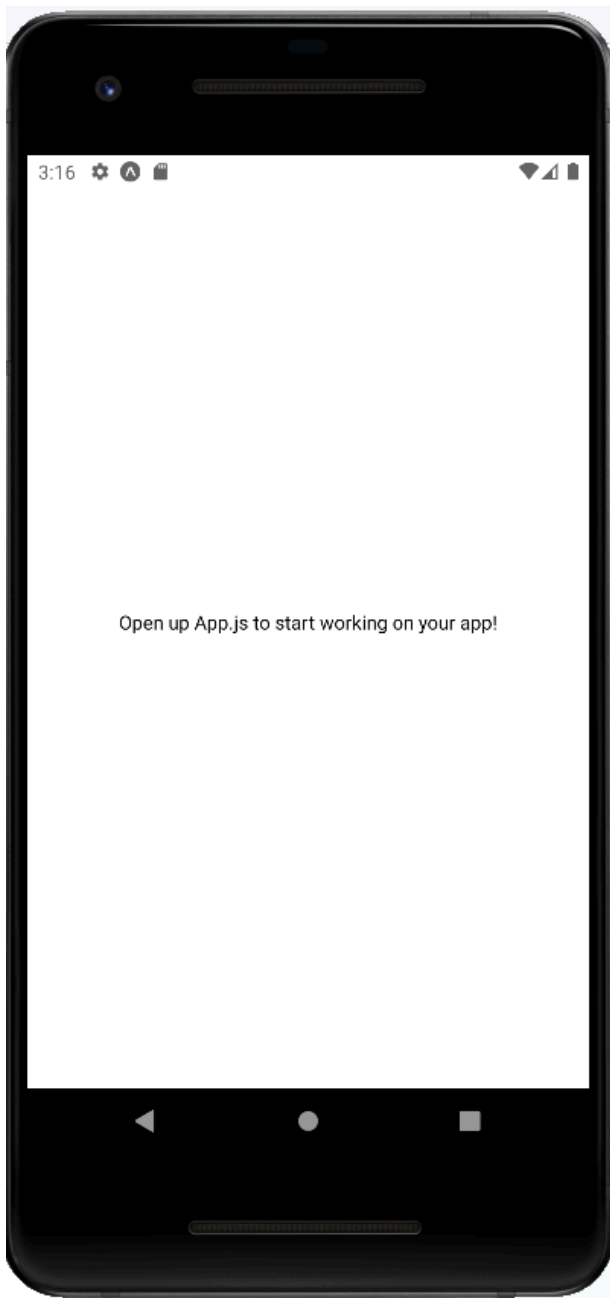
4. seleccionar la opción default new app:

```
C:\Users\Karens_Medrano\Desktop>npx create-expo-app@latest cars --template blank
✓ Downloaded and extracted project files.
> npm install
npm WARN deprecated osenv@0.1.5: This package is no longer supported.
```

5. Después de generar la aplicación deberemos abrir este proyecto en nuestro editor y deberemos observar la siguiente estructura en nuestro proyecto.



6. Ejecutamos nuestra aplicación, escribiendo en consola el siguiente comando: **npm run android/npm run ios/npm run web**
7. Deberíamos de visualizar lo siguiente en el emulador



8. Ahora cambiamos el archivo app.js de la aplicación para crear variables de objetos y mostrar en ella una lista de autos :

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

```

const DATA = [
  {
    id: '1',
    title: 'Toyota',
  },
  {
    id: '2',
    title: 'Mazda',
  },
  {
    id: '3',
    title: 'Mitsubishi',
  },
];

const Item = ({ title }) => (
  <View style={styles.item}>
    <Text style={styles.title}>{title}</Text>
  </View>
);

export default function App() {

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

  return (
    <SafeAreaView style={styles.container}>
      <FlatList
        data={DATA}
        renderItem={renderItem}
        keyExtractor={item => item.id}
      />
    </SafeAreaView>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    marginTop: StatusBar.currentHeight || 0,

```

```

    },
    item: {
      backgroundColor: '#f9c2ff',
      padding: 20,
      marginVertical: 8,
      marginHorizontal: 16,
    },
    title: {
      fontSize: 32,
    },
  });

```

9. Debería de visualizar lo siguiente en el emulador:



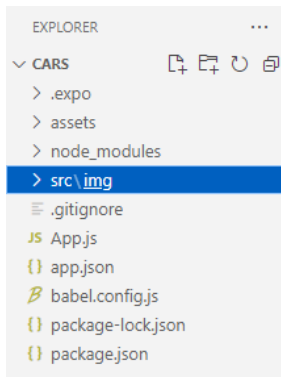
10. Ahora procedemos a agregar imagen a esta lista de automóviles, primero importamos la librería siguiente:

```

import {AppRegistry} from 'react-native';
import React from "react";
import { SafeAreaView, View, FlatList, StyleSheet, Text, StatusBar, Image } from 'react-native';

```

11. Dentro de nuestro proyecto deberemos crear la carpeta src, y dentro de la carpeta src una carpeta imgs. Se deberá de ver la estructura de nuestro proyecto de la siguiente forma:



12. Dentro de la carpeta imgs, pondremos las imágenes proporcionadas en recursos de esta guía.
13. Ahora vamos a modificar nuestro código para agregar las imágenes.
 - a. Primero en nuestro DATA vamos ir agregando la propiedad src, para cada uno de los elementos:

```
const DATA = [  
  {  
    id: '1',  
    title: 'Toyota',  
    src: require('./src/imgs/toyota.jpg'),  
  },  
  {  
    id: '2',  
    title: 'Mazda',  
    src: require('./src/imgs/mazda.jpg'),  
  },  
  {  
    id: '3',  
    title: 'Mitsubishi',  
    src: require('./src/imgs/mitsubishi.jpeg'),  
  },  
];
```

- b. vamos a modificar nuestra constante Item, y deberá quedarnos de la siguiente manera:

```
const Item = ({ title, img }) => (  
  <View style={styles.item}>  
    <Text style={styles.title}>{title}</Text>  
    <Image style={styles.img} source={img} />  
  </View>  
);
```

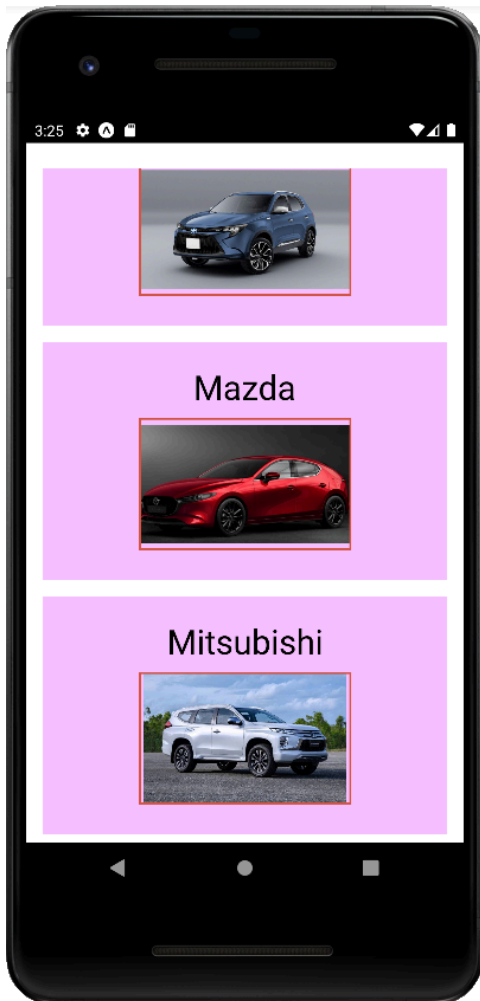
- c. para poder apreciar los cambios debemos realizar un cambio dentro de nuestro render Item:

```
const App = () => {  
  const renderItem = ({ item }) => (  
    <Item title={item.title} img={item.src} />  
  );  
};
```

- d. Ahora solo nos queda modificar los estilos:

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    marginTop: StatusBar.currentHeight || 0,
  },
  item: {
    backgroundColor: '#f9c2ff',
    padding: 20,
    marginVertical: 8,
    marginHorizontal: 16,
    alignItems: 'center'
  },
  title: {
    fontSize: 32,
  },
  img: {
    width: 200,
    height: 125,
    borderWidth: 2,
    borderColor: '#d35647',
    resizeMode: 'contain',
    margin: 8
  }
});
```

14. Procedemos a verificar los cambios dentro de nuestro emulador el cual debería verse de la siguiente manera:



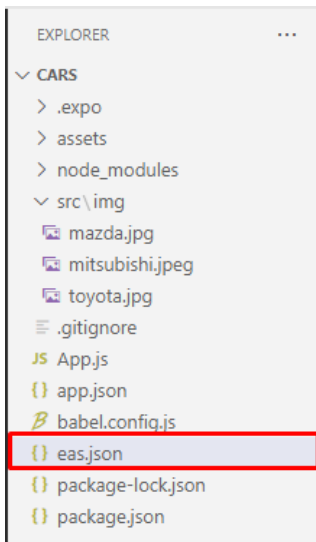
PARTE 2

1. En primer lugar, asegúrese de que su proyecto esté libre de errores. Eso significa que se está compilando y ejecutándose correctamente en el emulador o en un dispositivo Android.
2. Vamos a desarrollar nuestra propia versión de la aplicación en formato APK. Para lograrlo, es necesario sincronizar nuestro proyecto con un repositorio de Git.
3. Desde consola procederemos a instalar eas: **npm install -g eas-cli**
4. Procedemos a crear acceder a nuestra cuenta expo desde consola con el siguiente comando: **eas login**
5. Verificamos si estamos logueados con el siguiente comando: **eas whoami**
6. Desde consola debemos configurar nuestro proyecto para que sea compatible con las plataformas en las que planeamos utilizar esta aplicación. Utilizaremos el comando **eas build:configure** para llevar a cabo esta configuración.

```
C:\Users\Karens_Medrano\Desktop\cars>eas build:configure
⚠ The following process will configure your iOS and/or Android project to be compatible with EAS Build. These changes only apply to your local project files and you can safely revert them at any time.

? Which platforms would you like to configure for EAS Build? » - Use arrow-keys. Return to submit.
  All
  iOS
> Android
```

7. En nuestro proyecto se agrega el archivo eas.json



8. Ahora configuramos en archivo eas.json:

```
{
  "cli": {
    "version": ">= 7.2.0"
  },
  "build": {
    "development": {
      "developmentClient": true,
      "distribution": "internal"
    },
    "preview": {
      "android": {
        "buildType": "apk"
      }
    },
    "production": {}
  },
  "submit": {
    "production": {}
  }
}
```

9. Vamos a proceder a crear nuestro archivo de instalación para la aplicación y ejecutarlo desde la consola utilizando el comando `eas build -p android --profile preview`.
10. Como resultado, Expo nos proporcionará la dirección URL desde la cual podremos descargar nuestra APK.

Felicitaciones, acaba de generar un APK de compilación de lanzamiento nativo de React para Android.

V. EJERCICIO COMPLEMENTARIO

Realizar una app de comidas típicas salvadoreñas donde se muestre, una fotografía de la comida típica, el nombre de la comida y la cantidad calórica de cada platillo típico, para realizar la aplicación deberá utilizar los elementos card que se encuentra en [react-native-elements](https://reactnativeelements.com/):

