	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

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 ARM64 RPM Package	144.87 MB	jdk-16.0.2_linux-aarch64_bin.rpm
Linux ARM64 Compressed Archive	160.73 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	150.58 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/>



- 3. Yarn**, es un administrador de paquetes de JavaScript y gestor de dependencias. También se desempeña como administrador de proyectos. Puede verificar los pasos de instalación en la siguiente página: <https://classic.yarnpkg.com/lang/en/docs/install/#windows-stable>



	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

Comando de instalación

```
npm install --global yarn
```

Comando para verificar la instalación

```
yarn --version
```

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.

	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

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.

III. VERIFICANDO CONFIGURANDO ENTORNO DE TRABAJO

Ahora vamos a verificar que nuestro Visual Studio Code funcione correctamente.

1. Cree un folder nuevo con su número de Carnet
2. Ejecute Visual Studio Code y abra la carpeta creada como espacio de trabajo (Archivo -> Agregar carpeta al área de trabajo)
3. Cree un archivo index.js y digite el siguiente código


```
JS index.js X
verificacion > JS index.js > ...
1 function tablaMultiplicacion(numero){
2   for(let i=1;i<=10;i++){
3     console.log(numero+" x "+i+" = "+(numero*i));
4   }
5 }
6
7 tablaMultiplicacion(9);
```

4. Proceda a ejecutarlo haciendo clic en el menú "Terminal", luego ejecute el comando node + "nombre del archivo". Así como se muestra en la imagen.

```
TERMINAL COMMENTS PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN powershell + v
Windows PowerShell
Copyright (C) 2014 Microsoft Corporation. Todos los derechos reservados.

PS E: \PRACTICA #2\verificacion> node index.js
9 x 1 = 9
9 x 2 = 18
9 x 3 = 27
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
9 x 10 = 90
```

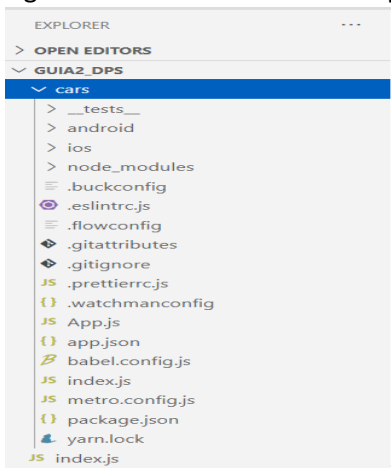
5. Confirmamos que nuestro entorno de trabajo está funcionando perfectamente.

	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO Nº 3

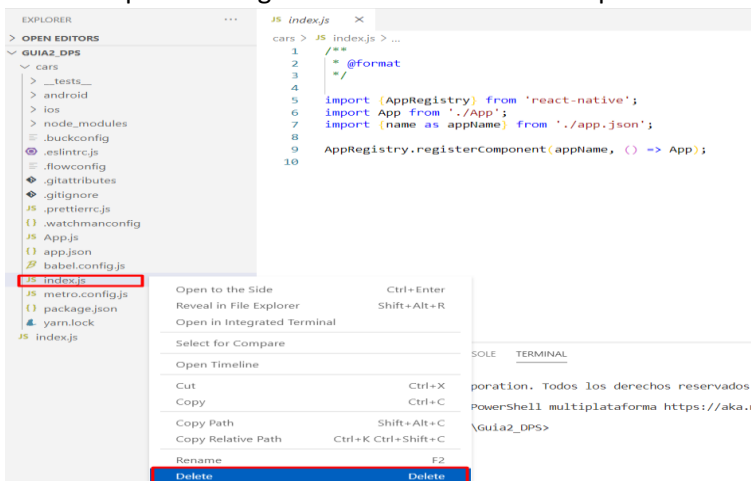
IV. DESARROLLO DE PRÁCTICA

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: **npm install -g react-native-cli**
4. Ve a la carpeta donde deseas colocar tu proyecto desde tu consola de comandos y en esa carpeta ejecutar:
expo init cars
5. Después de generar la aplicación deberemos abrir este proyecto en nuestro editor y deberemos observar la siguiente estructura en nuestro proyecto.




6. A continuación procedemos a eliminar el archivo index.js para poder crear nosotros, nuestro propio archivo debido a que este es generado automáticamente por react.



7. Ahora creamos nuestro propio archivo index.js
8. Empezamos importando las librerías

```
import React, {Component} from 'react';
import {AppRegistry} from 'react-native'
```

	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

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

9. Luego podemos empezar a crear nuestros componentes

```
const textosaludo =()=>{
  return (
    <View style={{flex:1,justifyContent:"center",alignItems:"center"}}>
      <Text>"Hola Mundo"</Text>
    </View>
  );
}
```

10. Procedemos a renderizar los componentes en nuestra pantalla

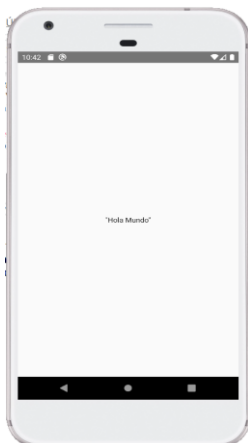
```
AppRegistry.registerComponent("cars", () => textosaludo);
```


11. El código completo debe verse de la siguiente manera

```
JS index.js
cars > JS index.js > ...
1  import React, {Component} from 'react';
2  import {AppRegistry} from 'react-native'
3  import {Text,View} from 'react-native';
4  const textosaludo =()=>{
5    return (
6      <View style={{flex:1,justifyContent:"center",alignItems:"center"}}>
7        <Text>"Hola Mundo"</Text>
8      </View>
9    );
10  };
11  }
12  AppRegistry.registerComponent("cars", () => textosaludo);
```

12. Ejecutamos nuestra aplicación, escribiendo en consola el siguiente comando: **expo start**

13. Deberíamos de visualizar lo siguiente en el emulador



	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

14. Ahora cambiemos la aplicación para crear variables de objetos y mostrar en ella una lista de autos:

```
import {AppRegistry} from 'react-native';
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>
);

const 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,
  },
});
```

	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

```

    title: {
      fontSize: 32,
    },
  });

```

```
AppRegistry.registerComponent("cars", () => App);
```

15. Debería de visualizar lo siguiente en el emulador:




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

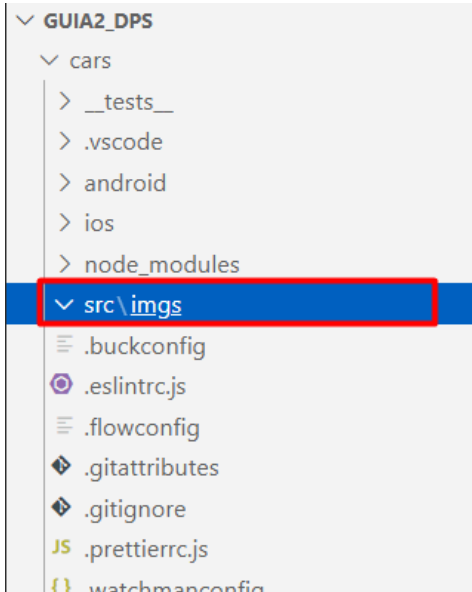
```

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

```

17. 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:

	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3



18. Dentro de la carpeta imgs, pondremos las imágenes proporcionadas en recursos de esta guía.

19. 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 debera 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:


	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

```
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
  }
});
```

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

	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3




PARTE 2

1. Procederemos a crear nuestro archivo aab, para Android.
2. En primer lugar, asegúrese de que su proyecto de Android esté libre de errores. Eso significa que se está compilando y ejecutándose correctamente en el emulador o en un dispositivo Android.
3. Necesitará una clave de firma generada por Java, que es un archivo de almacén de claves que se utiliza para generar un binario ejecutable React Native para Android.

Puede crear uno usando la herramienta de teclas en la terminal con el siguiente comando

```
sudo keytool -genkey -v -keystore my-upload-key.keystore -alias my-key-alias -keyalg RSA -keysize 2048 -validity 10000
```

Puede cambiar **my-upload-key** con el nombre que desee, así como my-key-alias . Esta clave usa el tamaño de clave 2048, en lugar del predeterminado 1024 por razones de seguridad.

	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

Por lo tanto, este comando le solicita la contraseña del almacén de claves, la clave real y los campos de nombre distinguido para su clave. Por lo tanto, todo debe ingresarse manualmente y con cuidado.

Ingrese su contraseña del almacén de claves: contraseña123

Vuelva a ingresar la nueva contraseña: contraseña123

¿Cuál es su nombre y apellido? [desconocido]: Karens Medrano

¿Cuál es el nombre de tu unidad organizacional? [desconocido]: udb

¿Cuál es el nombre de su organización? [desconocido]: udb

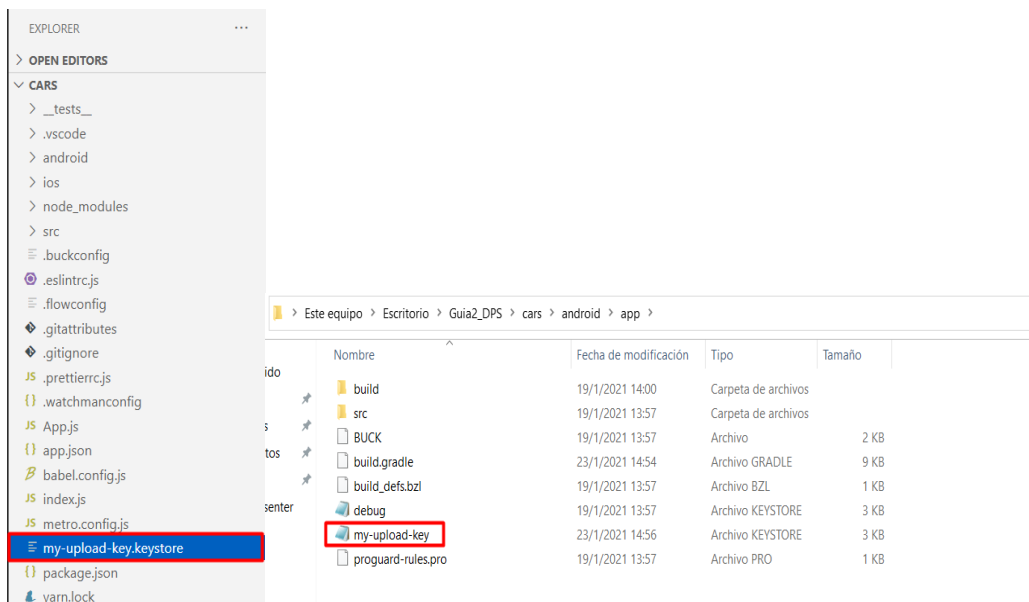
¿Cuál es el nombre de su ciudad o localidad? [desconocido]: San Salvador


¿Cuál es el nombre de su estado o provincia? [desconocido]: San Salvador

¿Cuál es el código de país de dos letras para esta unidad? [desconocido]: sv

NOTA: Como resultado, genera un archivo de almacén de claves en el directorio de su proyecto llamado my-release-key.keystore válido por 10000 días. Lo más importante es hacer una copia de seguridad de este archivo de almacén de claves y sus credenciales (contraseña de almacenamiento, alias y contraseña de alias) que se solicitarán más adelante.

- Ahora debe copiar el archivo your_key_name.keystore y pegarlo en el directorio android / app en la carpeta de su proyecto React Native.



	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

5. Edite el archivo `~/gradle/gradle.properties` o `android/gradle.properties`, y agregue lo siguiente (reemplácelo ***** con la contraseña correcta del almacén de claves, el alias y la contraseña de la clave),

`MYAPP_UPLOAD_STORE_FILE=my-upload-key.keystore`

`MYAPP_UPLOAD_KEY_ALIAS=my-key-alias`

`MYAPP_RELEASE_STORE_PASSWORD=*****`

`MYAPP_RELEASE_KEY_PASSWORD=*****`

6. El último paso de configuración que debe realizarse es configurar las versiones de versión para que se firmen mediante la clave de carga. Edite el archivo `android/app/build.gradle` en la carpeta de su proyecto y agregue la configuración de firma:


```
...
android {
    ...
    defaultConfig { ... }
    signingConfigs {
        release {
            if (project.hasProperty('MYAPP_UPLOAD_STORE_FILE')) {
                storeFile file(MYAPP_UPLOAD_STORE_FILE)
                storePassword MYAPP_UPLOAD_STORE_PASSWORD
                keyAlias MYAPP_UPLOAD_KEY_ALIAS
                keyPassword MYAPP_UPLOAD_KEY_PASSWORD
            }
        }
    }
    buildTypes {
        release {
            ...
            signingConfig signingConfigs.release
        }
    }
}
...
```


7. Ahora ejecutamos en la terminal

```
cd android
```

```
gradlew bundleRelease
```

8. La AAb ya esta generada se puede encontrar debajo `Android/app/build/outputs/bundle/release/app.aaby` está listo para cargarse en Google Play. la Apk se genera automáticamente en la carpeta `android\app\build\outputs\apk\release`.

	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

	UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN	CICLO II
	DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE	GUIA DE LABORATORIO N° 3

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 una pequeña descripción de sus ingredientes, para realizar la aplicación deberá utilizar los elementos card que se encuentra en **react-native-elements**:

