APLICACIONES MÓVILES MULTIPLATAFORMA

LABORATORIO N° 03

React Native

Alumno(s):		***Nota***	
Grupo:		***Ciclo:V***	

Criterio de Evaluación	*Excelente (4pts)*	*Bueno (3pts)*	*Requiere mejora (2pts)*	*No acept. (0pts)*	*Puntaje Logrado*
Instala con éxito React Native					
Utiliza Next-Gen Javascript					
Desarrolla aplicaciones con React Native					
Realiza con éxito lo propuesto en el laboratorio					
Es puntual y redacta el informe adecuadamente					

Laboratorio 03:
React Native

Objetivos:

Al finalizar el laboratorio el estudiante será capaz de:

- Entender el funcionamiento de React Native
- Desarrollar aplicaciones web enfocadas a componentes
- Implementación correcta de Next-Gen Javascript

Seguridad:

- Ubicar maletines y/o mochilas en el gabinete del aula de Laboratorio.
- No ingresar con líquidos, ni comida al aula de Laboratorio.
- Al culminar la sesión de laboratorio apagar correctamente la computadora y la pantalla, y ordenar las sillas utilizadas.

Equipos y Materiales:

- Una computadora con:
  - Windows 7 o superior
  - VMware Workstation 10+ o VMware Player 7+
  - Conexión a la red del laboratorio
- Máquinas virtuales:
  - Windows 7 Pro 64bits Español - Plantilla
- Instalador de node.js

Procedimiento:

Lab Setup

1. **Instalación y configuración de ReactJS**
   1.1. Ingrese a la siguiente web y descargue el paquete Chocolatey
   https://chocolatey.org/

   1.2. Después de instalar Chocolatey (que es un asistente para la configuración amigable en Windows), abrir un **Símbolo de Sistema como Administrador** y ejecutar el siguiente comando:

```
$ choco install -y nodejs.install python2 jdk8
```

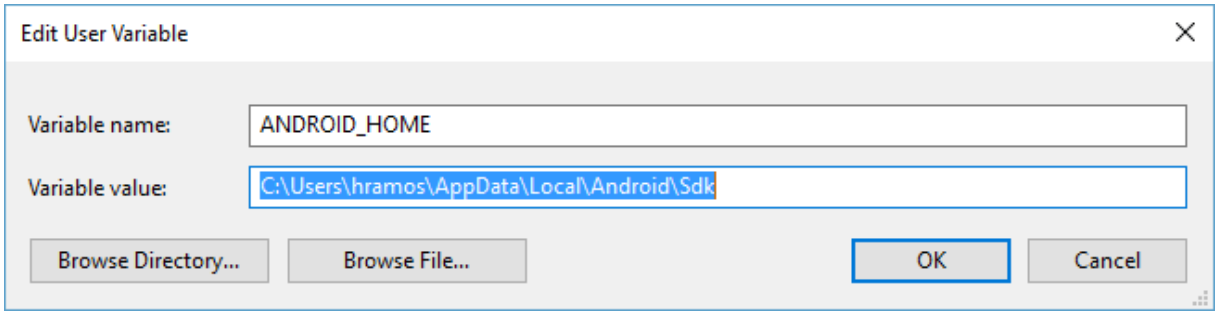
   1.3. Ahora procedemos con la instalación global de React Native. Ejecutaremos la siguiente instrucción.

```
$ npm install -g react-native-cli
```

   1.4. Debemos tener configurados todos los SDKs de Android para poder continuar. En caso ya tengas configurado Android Studio y sus variables de entorno, puedes continuar con la parte 2 del laboratorio. En caso contrario, descarga e instala Android Studio (solamente lo usaremos por los drivers y las sdk, no lo ejecutaremos durante el laboratorio). Después de instalado, en la pantalla de bienvenida (la que aprecias a continuación) nos mostrará la opción Configurar, en la que seleccionaremos la opción SDK manager.

1.5. Haz click en **Show Package Details**, luego expande la opción de **Android 9 (Pie)** y asegúrate de que los siguientes ítems estén seleccionados:
- **Android SDK Platform 28**
- **Intel x86 Atom_64 System Image** o **Google APIs Intel x86 Atom System Image**

1.6. Agrega ahora en tus variables de entorno la variable ANDROID_HOME.
Si tienes problemas para agregar las variables de entorno, hice un vídeo para ti:
https://www.youtube.com/watch?v=hxQGKOsQwH4



2. Creación de proyecto

2.1. Ejecuta el siguiente comando para crear nuestro proyecto tecsupApp

```
$ react-native init tecsupApp
```

2.2. Una vez terminado, estamos listos para ejecutar nuestra aplicación. Debemos alistar entonces nuestro dispositivo Android.
2.3. En caso decidas usar un emulador: No debes hacer nada especial, solamente iniciar dicho emulador antes de iniciar el proyecto.
2.4. En caso decidas usar tu propio equipo físico: Debes conectar tu equipo con un cable USB a tu equipo y activar el modo desarrollador. Puedes encontrar una guía aquí:
https://facebook.github.io/react-native/docs/running-on-device

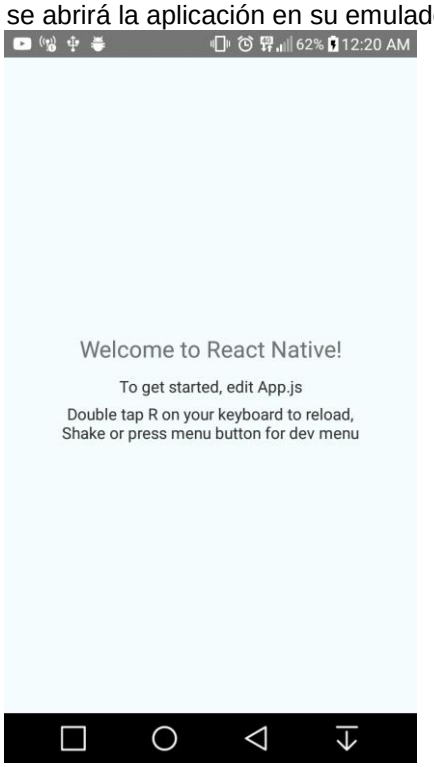
2.5. Finalmente, ubiquémonos en la carpeta del proyecto y corramos el comando react-native run-android (si fueras a ejecutar en iOS, será run-ios)

```
Run instructions for iOS:
  • cd C:\Users\favio\code\tecsup\tecsupApp && react-native run-ios
  - or -
  • Open ios\tecsupApp.xcodeproj in Xcode
  • Hit the Run button

Run instructions for Android:
  • Have an Android emulator running (quickest way to get started), or a device connected.
  • cd C:\Users\favio\code\tecsup\tecsupApp && react-native run-android
```

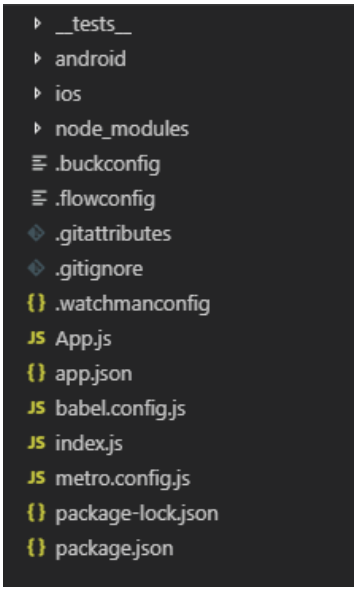
```
>cd tecsupApp

\tecsupApp>react-native run-android
```

3. Estructura del proyecto

3.1. Una vez iniciado el proyecto, se abrirá la aplicación en su emulador o dispositivo físico.



3.2. Veamos la estructura del proyecto. Vemos que React Native ha creado dos carpetas, iOS y android, ambas conteniendo el código nativo. Sin embargo, nuestro punto de entrada está en el archivo App.js



3.3. La primera parte de App.js nos muestra nuestro componente básico, justamente el que vemos reflejado en el celular.

```
import React, {Component} from 'react';
import {Platform, StyleSheet, Text, View} from 'react-native';

const instructions = Platform.select({
  ios: 'Press Cmd+R to reload,\n' + 'Cmd+D or shake for dev menu',
  android:
    'Double tap R on your keyboard to reload,\n' +
    'Shake or press menu button for dev menu',
});

type Props = {};
export default class App extends Component<Props> {
  render() {
    return (
      <View style={styles.container}>
        <Text style={styles.welcome}>Welcome to React Native!</Text>
        <Text style={styles.instructions}>To get started, edit App.js</Text>
        <Text style={styles.instructions}>{instructions}</Text>
      </View>
    );
  }
}
```

3.4. La segunda parte de App.js nos muestra los estilos aplicados.

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: '#F5FCFF',
  },
  welcome: {
    fontSize: 20,
    textAlign: 'center',
    margin: 10,
  },
  instructions: {
    textAlign: 'center',
    color: '#333333',
    marginBottom: 5,
  },
});
```

3.5. Hagamos una modificación en el componente. Modifiquemos el texto de bienvenida para saludar a los alumnos de Tecsup.

```
render() {
  return (
    <View style={styles.container}>
      <Text style={styles.welcome}>Bienvenido alumno de Tecsup</Text>
      <Text style={styles.instructions}>To get started, edit App.js</Text>
      <Text style={styles.instructions}>{instructions}</Text>
    </View>
  );
}
```

3.6. Para refrescar el dispositivo, siga las instrucciones del siguiente vídeo: https://youtu.be/bUXtWKCEKKU

3.7. En React Native, usamos el mismo principio de React JS pero sin elementos HTML. Para poder utilizar elementos nativos, debemos importarlos de la misma librería de React Native. Entre los principales tenemos:

- **View**: Equivalente al DIV en HTML. Es el elemento básico como contenedor.
- **Text**: Equivalente al SPAN en HTML. Nos permite mostrar textos y adornarlos con la decoración necesaria.
- **Image**: Equivalente al IMG en HTML. Permite mostrar imágenes de la web o locales.
- **TextInput**: Equivalente al INPUT en HTML. Elemento básico para los formularios.
- **ScrollView**: Contenedor para poder hacer "scroll" y mostrar varios componentes.
- **StyleSheet**: Elemento necesario para la creación de hojas de estilos.

3.8. Procedamos a agregar una imagen. Modifique el código para que ahora se importe el componente Image y también se agregue en el render principal.

```
import React, {Component} from 'react';
import {Image, StyleSheet, Text, View} from 'react-native';

type Props = {};
export default class App extends Component<Props> {
  render() {
    return (
      <View style={styles.container}>
      <Image
        source={require('./logo.png')}
      />
        <Text style={styles.welcome}>Bienvenido alumno de Tecsup</Text>
      </View>
    );
  }
}
```

3.9. Copie el archivo logo.png proveído en el laboratorio en la raíz de su proyecto para verificar el funcionamiento correcto de lo modificado.

3.10. Adjunte una captura de la aplicación hasta el momento.

4. **Finalizar la sesión**

4.1. Apagar el equipo virtual

4.2. Apagar el equipo

Conclusiones:

Indicar las conclusiones que llegó después de los temas tratados de manera práctica en este laboratorio.