APLICACIONES MÓVILES MULTIPLATAFORMA

LABORATORIO N° 06

React-navigation

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

Criterio de Evaluación	*Excelente (4pts)*	*Bueno (3pts)*	*Requiere mejora (2pts)*	*No acept. (0pts)*	*Puntaje Logrado*
Agregar componentes de listados dinámicos					
Instalar y utilizar axios					
Consumir APIs de terceros					
Realiza con éxito lo propuesto en el laboratorio					
Es puntual y redacta el informe adecuadamente					

Laboratorio 06:
React-navigation

Objetivos:

Al finalizar el laboratorio el estudiante será capaz de:

- Entender el funcionamiento del componente Flatlist
- Desarrollar aplicaciones web enfocadas a componentes
- Manejar de manera básica axios para consumir APIs

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. **Configuración de proyecto**
   1.1. En su carpeta de trabajo, inicie un nuevo proyecto de react-native. En caso de usar una instalación nativa, utilice:

```
>react-native init lab06
```

   1.2. Iniciemos nuestro proyecto. En caso de usar la instalación nativa, utilizar:

```
>cd lab06

\lab06>react-native run-android
```

   1.3. Procederemos a instalar las dependencias para nuestra librería de navegación. Utilizaremos los siguientes tres comandos.

```
>npm install --save react-navigation
```

```
>yarn add react-native-gesture-handler
```

```
>react-native link react-native-gesture-handler
```

   1.4. Ahora configuraremos la parte gestual de nuestra aplicación (para habilitar los menús). Esto se hará solamente para Android. Modificamos el archivo **MainActivity.java** ubicado en **android/app/src/main/java/com/lab06**

```java
import com.facebook.react.ReactActivity;
import com.facebook.react.ReactActivityDelegate;
import com.facebook.react.ReactRootView;
import com.swmansion.gesturehandler.react.RNGestureHandlerEnabledRootView;

public class MainActivity extends ReactActivity {

    /**
     * Returns the name of the main component registered from JavaScript.
     * This is used to schedule rendering of the component.
     */
    @Override
    protected String getMainComponentName() {
        return "lab06";
    }

    @Override
    protected ReactActivityDelegate createReactActivityDelegate() {
        return new ReactActivityDelegate(this, getMainComponentName()) {
            @Override
            protected ReactRootView createRootView() {
                return new RNGestureHandlerEnabledRootView(MainActivity.this);
            }
        };
    }
```

1.5. Ahora que está todo configurado, reemplazaremos el contenido de nuestro archivo App.js

```javascript
import React from "react";
import { View, Text } from "react-native";
import { createStackNavigator, createAppContainer } from "react-navigation";

class HomeScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
        <Text>Hola mundo!</Text>
      </View>
    );
  }
}

const AppNavigator = createStackNavigator({
  Home: {
    screen: HomeScreen
  }
});

export default createAppContainer(AppNavigator);
```

1.6. Nuestra aplicación debe correr sin problemas y mostrar un Hola Mundo. Procederemos a crear una vista adicional, por lo que agregaremos lo siguiente en **App.js**

```
class DetailsScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
        <Text>Details Screen</Text>
      </View>
    );
  }
}

const AppNavigator = createStackNavigator({
  Home: {
    screen: HomeScreen
  },
  Details: DetailsScreen
},
{
  initialRouteName: "Home"
});
```

1.7. Note que, a diferencia de Home, que fue declarado como un objeto, **Details** llama directamente al componente. Ambos funcionarán sin ningún problema ya que ambas sintaxis son válidas.

Agregaremos un botón en la vista de Hola Mundo para navegar entre vistas. Probemos dicha funcionalidad y agregue un gif de dicho comportamiento.

```
import { View, Text, Button } from "react-native";
import { createStackNavigator, createAppContainer } from "react-navigation";

class HomeScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
        <Text>Hola mundo!</Text>
        <Button
          title="Go to Details"
          onPress={() => this.props.navigation.navigate('Details')}
        />
      </View>
    );
```

1.8. Agreguemos un botón para navegar también en Details para poder volver a la vista de inicio. Agregue un gif con dicho comportamiento.

```
class DetailsScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
        <Text>Details Screen</Text>
        <Button
          title="Go to Home"
          onPress={() => this.props.navigation.navigate('Home')}
        />
      </View>
    );
```

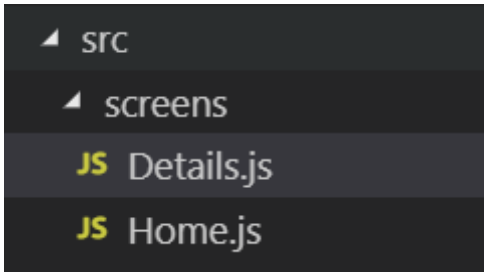
1.9. ¿Recuerdas que Home lo configuramos como un objeto? Ahora veremos qué ventajas tiene. Modificaremos su invocación, para poder pasarle opciones adicionales, como por ejemplo, el título de la vista.

```
const AppNavigator = createStackNav
  Home: {
    screen: Home,
    path: 'home/',
    navigationOptions: {
      title: 'Esta es la Home'
    }
  },
```

1.10. Al ejecutar el código debe usted obtener un error. Esto ha sido agregado a propósito, para que indique cuál es el problema (No se preocupe, es algo básico, no de esta lección).

2. Refactorizar nuestro código

2.1. Ahora que vemos el funcionamiento correcto de nuestro navegador, procederemos a refactorizar (ordenar) nuestro código. Crearemos la carpeta **src**, donde crearemos la carpeta **screens** con los archivos **Home.js** y **Details.js**



2.2. El contenido de **Home.js** será el siguiente:

```
import React from "react";
import { View, Text, Button } from "react-native";

class HomeScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
        <Text>Hola mundo!</Text>
        <Button
          title="Go to Details"
          onPress={() => this.props.navigation.navigate('Details')}
        />
      </View>
    );
  }
}

export default HomeScreen;
```

2.3. El contenido de **Details.js** será el siguiente:

```
import React from "react";
import { View, Text, Button } from "react-native";

class DetailsScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
        <Text>Details Screen</Text>
        <Button
          title="Go to Home"
          onPress={() => this.props.navigation.navigate('Home')}
        />
      </View>
    );
  }
}

export default DetailsScreen;
```

2.4. Modificaremos a **App.js** para invocar correctamente a los componentes escritos.

```
import { createStackNavigator, createAppContainer } from "react-navigation";

import HomeScreen from './src/screens/Home';
import DetailsScreen from './src/screens/Details';

const AppNavigator = createStackNavigator({
  Home: {
    screen: HomeScreen,
    path: 'home/',
    navigationOptions: {
      title: 'Esta es la Home'
    }
  },
  Details: DetailsScreen
},
{
  initialRouteName: "Home"
});

export default createAppContainer(AppNavigator);
```

3. Configuración por defecto

3.1. Nosotros podemos definir opcionales adicionales en el segundo objeto pasado a nuestro navegador. Procederemos a configurar algunas opciones, adjunte un gif mostrando dicho funcionamiento y comente que hace la propiedad title.

```
    Details: DetailsScreen
  },
  {
    initialRouteName: "Home",
    defaultNavigationOptions: {
      title: 'Un titulo genérico',
      headerStyle: {
        backgroundColor: '#f4511e',
      },
      headerTintColor: '#fff',
      headerTitleStyle: {
        fontWeight: 'bold',
      },
    },
  });
```

3.2. Agregaremos un archivo **Profile.js** en **screens** con el siguiente contenido:

```
import React from "react";
import { View, Text, Button } from "react-native";

class ProfileScreen extends React.Component {
  render() {
    let userName = 'Usuario';
    return (
      <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
        <Text>Bienvenido {userName}</Text>
        <Button
          title="Volver al Home"
          onPress={() => this.props.navigation.navigate('Home')}
        />
      </View>
    );
  }
}

export default ProfileScreen;
```

3.3. Modificaremos la vista **Details.js** para agregar un botón que nos lleve al perfil.

```
class DetailsScreen extends React.Component {
    goProfileHandler = () => {
        this.props.navigation.navigate('Profile')
    }
    render() {
        return (
        <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
            <Text>Details Screen</Text>
            <Button
                title="Go to Home"
                onPress={() => this.props.navigation.navigate('Home')}
            />
            <Button
                title="Ver Perfil"
                onPress={this.goProfileHandler}
            />
        </View>
```

3.4. Finalmente, agregaremos la ruta en nuestro **App.js**, adjunte un gif del comportamiento final.

```
import HomeScreen from './src/screens/Home';
import DetailsScreen from './src/screens/Details';
import ProfileScreen from './src/screens/Profile';

const AppNavigator = createStackNavigator({
  Home: {
    screen: HomeScreen,
    path: 'home/',
    navigationOptions: {
      title: 'Esta es la Home'
    }
  },
  Details: DetailsScreen,
  Profile: ProfileScreen,
},
```

4. Parámetros en ruta

4.1. Como durante el desarrollo de nuestras aplicaciones, tendremos vistas dinámicas, sería muy útil pasar parámetros a las vistas para indicarles cómo comportarse. Modificaremos **Details.js** para que le pase un parámetro de nombre a **Profile.js**

```
class DetailsScreen extends React.Component {
    goProfileHandler = () => {
        this.props.navigation.navigate('Profile', {
            userName: 'Tecsupiano'
        })
    }
```

4.2. Modificamos Profile.js para renderizar dicho parámetro. Así mismo, le pasamos un valor por defecto (*Usuario* en nuestro caso) para que en caso de no recibir nada, sea utilizado.

```
class ProfileScreen extends React.Component {
  render() {
    const userName = this.props.navigation.getParam('userName', 'Usuario');
    return (
      <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
        <Text>Bienvenido {userName}</Text>
        <Button
```

4.3. Podemos hacer que nuestra aplicación tenga un menú lateral. Esto es conocido como Drawer. Modificaremos las siguientes líneas de **App.js** y adjuntamos un gif del comportamiento.

```
import { createDrawerNavigator, createAppContainer } from "react-navigation";

import HomeScreen from './src/screens/Home';
import DetailsScreen from './src/screens/Details';
import ProfileScreen from './src/screens/Profile';

const AppNavigator = createDrawerNavigator({
  Home: {
    screen: HomeScreen,
    path: 'home/',
    navigationOptions: {
      title: 'Esta es la Home'
    }
```

4.4. De igual manera podemos tener una navegación por Tabs inferiores. Modificamos **App.js** y adjuntamos un gif del comportamiento.

```
import { createBottomTabNavigator, createAppContainer } from "react-navigation";

import HomeScreen from './src/screens/Home';
import DetailsScreen from './src/screens/Details';
import ProfileScreen from './src/screens/Profile';

const AppNavigator = createBottomTabNavigator({
  Home: {
    screen: HomeScreen,
    path: 'home/',
    navigationOptions: {
```

5. Ejercicio Propuesto

5.1. Utilizando los conocimientos del anterior laboratorio (Flatlist y Axios), crear una vista inicial con un listado de películas (filtradas por una barra de búsqueda) que al hacer click sobre una, navegue a una vista de detalles, donde muestre el título, descripción e imagen de la película.

6. **Finalizar la sesión**

6.1. Apagar el equipo virtual

6.2. Apagar el equipo

Conclusiones:

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