laClínica
DENTAL

LA CLINICA DENTAL APP

MANUAL TÉCNICO

Derechos de autor

Usted es libre de:

- **Compartir** — copiar y redistribuir el material en cualquier medio o formato
- **Adaptar** — remezclar, transformar y construir a partir del material.

Bajo los siguientes términos:

Atribución — Usted debe dar crédito de manera adecuada, brindar un enlace a la licencia, e indicar si se han realizado cambios. Puede hacerlo en cualquier forma razonable, pero no de forma tal que sugiera que usted o su uso tienen el apoyo de la licenciante.

- **No Comercial** — Usted no puede hacer uso del material con propósitos comerciales.

- **Compartir Igual** — Si remezcla, transforma o crea a partirdel

No hay restricciones adicionales — No puede aplicar términos legales ni medidas tecnológicas que restrinjan legalmente a otras ahacer cualquier uso permitido por la licencia.

Avisos:

No tiene que cumplir con la licencia para elementos del material en el dominio público ocuando su uso esté permitido por una excepción o limitación aplicable.

No se dan garantías. La licencia podría no darle todos los permisos que necesita para eluso que tenga previsto. Por ejemplo, otros derechos como publicidad, privacidad, o derechos morales pueden limitar la forma en que utilice el material.

Introducción

Con el presente documento se busca explicar al usuario las pautasde configuración necesarias para ejecutar el proyecto de la maneraadecuada y también mostrar la lógica y herramientas utilizadas durante el desarrollo de esta aplicación.

Requerimientos de desarrollo

Visual Studio Code

Se utilizo Visual Studio Code para el desarrollo del front-end de la aplicación. El front-end es básicamente la interfaz que el usuario vey utiliza para navegar dentro de la aplicación móvil.

Expo.dev

También se utilizo Expo.dev como complemento de visual studiocode, para desarrollar algunas pantallas y utilizar el emulador de Android y verificar aspectos gráficos de las pantallas.

Lenguaje de programación Python.

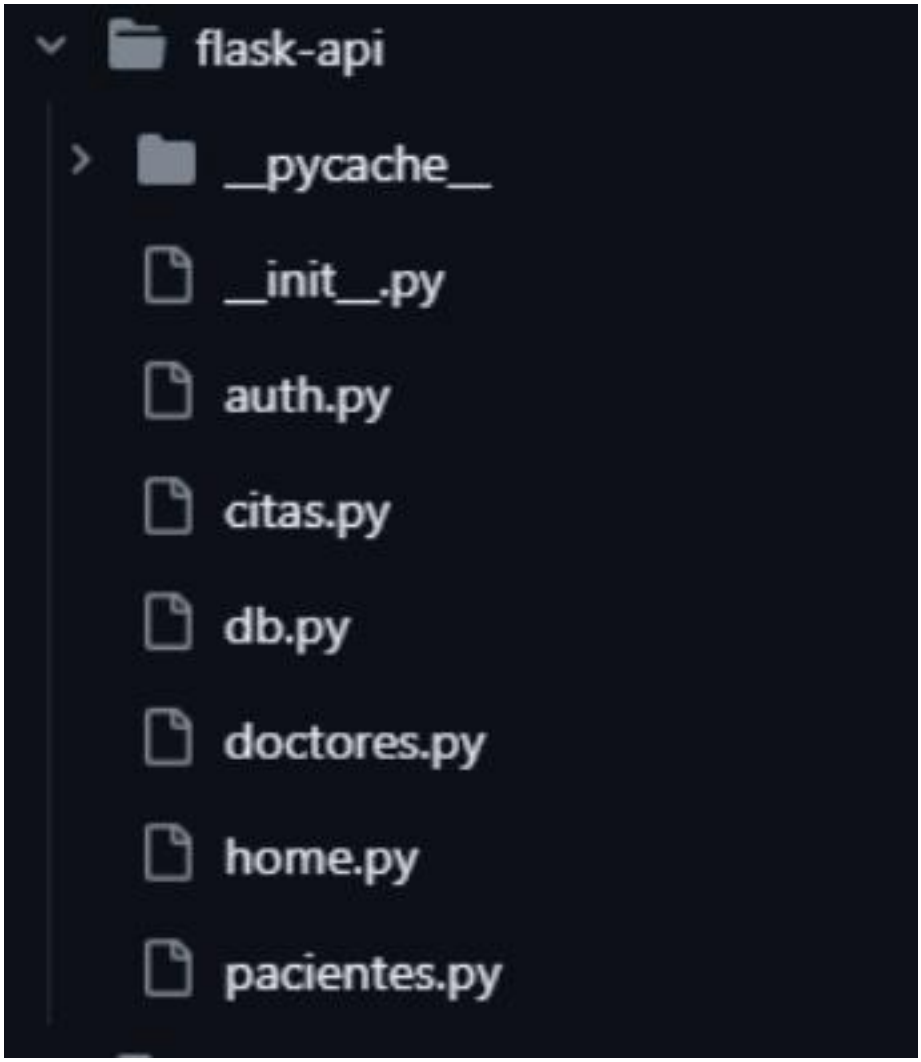
El lenguaje de programación Python se utilizó para configurar el back-end de la aplicación, es decir, todas las instrucciones de código para realizar las acciones que el usuario solicite mediante laspantallas fueron codificadas en este lenguaje.

JavaScript

JavaScript es otra herramienta utiliza en el desarrollo de la aplicación, y su función es la de dar estilos personalizados a laspantallas de la aplicación.

Estructura API

En la siguiente imagen podemos ver para de la estructura del API backend en flask, donde destaca el archivo auth encargado de validar los permisos y cada uno de las entidades principales de la aplicación:



Doctores.py

Como se puede observar todos los requests relacionados al manejo de los doctores se accederan mediante http://url/doctores

doctores.py

flask-api > doctores.py > nuevo_doctor

```python
from flask import (
    Blueprint, jsonify, abort, request
)
from .db import *
from .auth import login_required

bp = Blueprint('doctores', __name__, url_prefix='/doctores')

```

Siendo los siguientes las rutas disponibles:

```python
@bp.route('/')
# @login_required
def lista_doctores():
    try:
        doctores = _obtener_doctor()
        return jsonify(doctores), 200
    except Exception as e:
        print(e, flush=True)
        return abort(500)


@bp.route('/<int:id>')
# @login_required
def obtener_doctor(id):
    try:
        doctores = _obtener_doctor(id)
        return jsonify(doctores), 200
    except Exception as e:
        print(e, flush=True)
        return abort(500)


@bp.route('/eliminar/<int:id>')
# @login_required
def eliminar(id):
    try:
        _eliminar_doctor((id,))
        return jsonify(_obtener_doctor(id)), 200
    except Exception as e:
        return abort(400)
```

```python
@bp.route('/nuevo', methods=('POST',))
# @login_required
def nuevo_doctor():
    try:
        data = request.json
        doc = (data.get('nombre_completo'), data.get('fecha_nac'),
            data.get('domicilio'), data.get('correo'), data.get('telefono'), data.get('espec
            data.get('horario'), data.get('contrasenia'))

        if not all(doc):
            return jsonify('Faltan campos requeridos'), 400

        doctor = _crear_doctor(doc)
        if doctor:
            return jsonify(_obtener_doctor(doctor)), 200
        return abort(400)
    except Exception as e:
        return abort(400)


@bp.route('/modificar/<int:id>', methods=('POST',))
# @login_required
def modificar_doctor(id):
    try:
        data = request.json
        doc = (data.get('nombre_completo'), data.get('fecha_nac'),
            data.get('domicilio'), data.get('correo'), data.get('telefono'), data.get('espec
            data.get('horario'), id)
        if not all(doc):
            return jsonify('Faltan campos requeridos'), 400
```

Citas.py

Accesible via: http://url/citas

```python
bp = Blueprint('citas', __name__, url_prefix='/citas')
```

```python
bp = Blueprint('citas', __name__, url_prefix='/citas')


@bp.route('/')
# @login_required
def lista_citas():
    try:
        citas = _obtener_cita()
        return jsonify(citas), 200
    except Exception as e:
        print(e, flush=True)
        return abort(500)


@bp.route('/<int:id>')
@login_required
def obtener_cita(id):
    try:
        citas = _obtener_cita(id)
        return jsonify(citas), 200
    except Exception as e:
        print(e, flush=True)
        return abort(500)

@bp.route('/eliminar/<int:id>')
# @login_required
def eliminar(id):
    try:
        _eliminar_cita((id,))
        return jsonify(_obtener_cita(id)), 200
    except Exception as e:
        return abort(400)

```

```python

@bp.route('/nuevo', methods=('POST',))
# @login_required
def nuevo_cita():
    try:
        data = request.json
        doc = (data.get('id_paciente'), data.get('id_doctor'), da

        if not all(doc):
            return jsonify('Faltan campos requeridos'), 400

        cita = _crear_cita(doc)
        if cita:
            cita = _obtener_cita(cita)
            return jsonify(cita), 200
        return abort(400)
    except Exception as e:
        return abort(400)


@bp.route('/modificar/<int:id>', methods=('POST',))
# @login_required
def modificar_cita(id):
    try:
        data = request.json
        doc = (data.get('id_paciente'), data.get('id_doctor'), da
        if not all(doc):
            return jsonify('Faltan campos requeridos'), 400
        _actualizar_cita(doc)
        return jsonify(_obtener_cita(id)), 200
    except Exception as e:
        return abort(400)


```

Pacientes.py

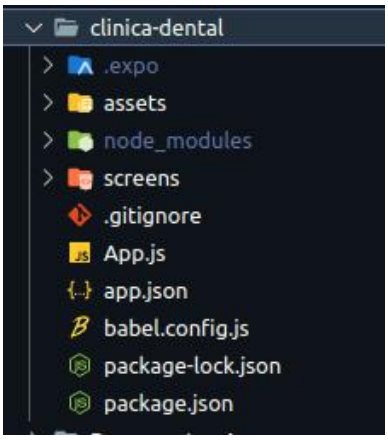
Accesible via: http://url/pacientes

```python
bp = Blueprint('pacientes', __name__, url_prefix='/pacientes')
```

```python
@bp.route('/')
# @login_required
def lista_pacientes():
    try:
        pacientes = _obtener_paciente()
        return jsonify(pacientes), 200
    except Exception as e:
        print(e, flush=True)
        return abort(500)


@bp.route('/<int:id>')
# @login_required
def obtener_paciente(id):
    try:
        pacientes = _obtener_paciente(id)
        return jsonify(pacientes), 200
    except Exception as e:
        print(e, flush=True)
        return abort(500)


@bp.route('/nuevo', methods=('POST',))
# @login_required
def nuevo_paciente():
    try:
        data = request.json
        doc = (data.get('nombre_completo'), data.get('fecha_nac'),
            data.get('domicilio'), data.get('correo'), data.get('telefono'),  data.get('contrasenia'))

        if not all(doc):
            return jsonify('Faltan campos requeridos'), 400

        paciente = _crear_paciente(doc)
        if paciente:
           return jsonify(_obtener_paciente(paciente)), 200
        return abort(400)
    except Exception as e:
        return abort(400)
```

```python
@bp.route('/modificar/<int:id>', methods=('POST',))
# @login_required
def modificar_paciente(id):
    try:
        data = request.json
        doc = (data.get('nombre_completo'), data.get('fecha_nac'),
            data.get('domicilio'), data.get('correo'), data.get('telefono'), i
        if not all(doc):
            return jsonify('Faltan campos requeridos'), 400

        _actualizar_paciente(doc)
        return jsonify(_obtener_paciente(id)), 200
    except Exception as e:
        return abort(400)


@bp.route('/eliminar/<int:id>')
# @login_required
def eliminar(id):
    try:
        _eliminar_paciente((id,))
        return jsonify(_obtener_paciente(id)), 200
    except Exception as e:
        return abort(400)
```

Estructura React App



App.js

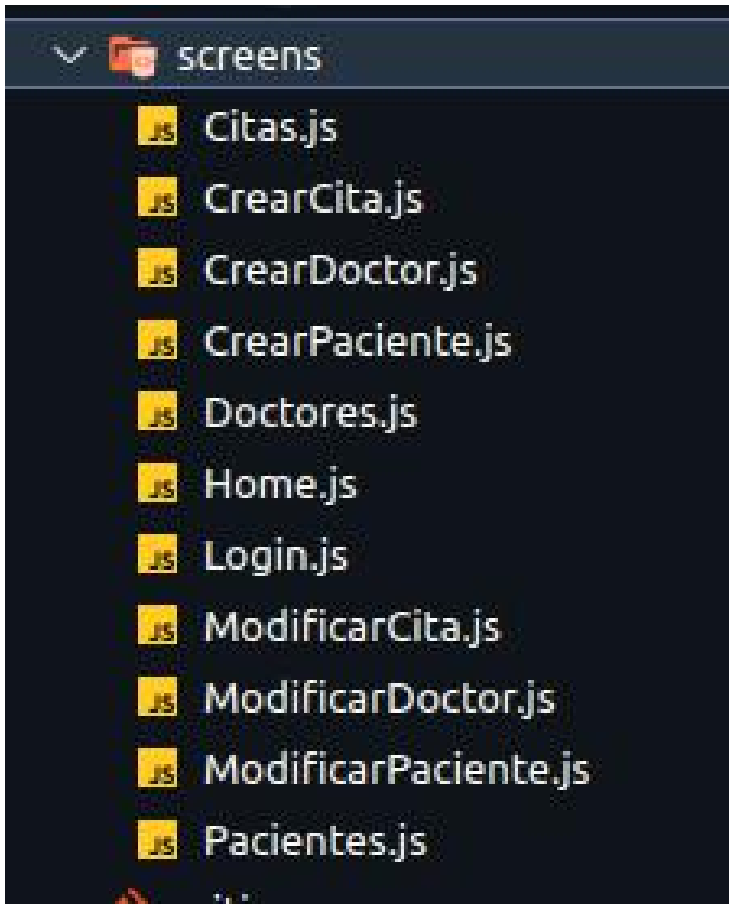
Muestra del import de las pantallas de aplicación y es el entrypoint de la aplicación.

App.js

clinica-dental > App.js > App

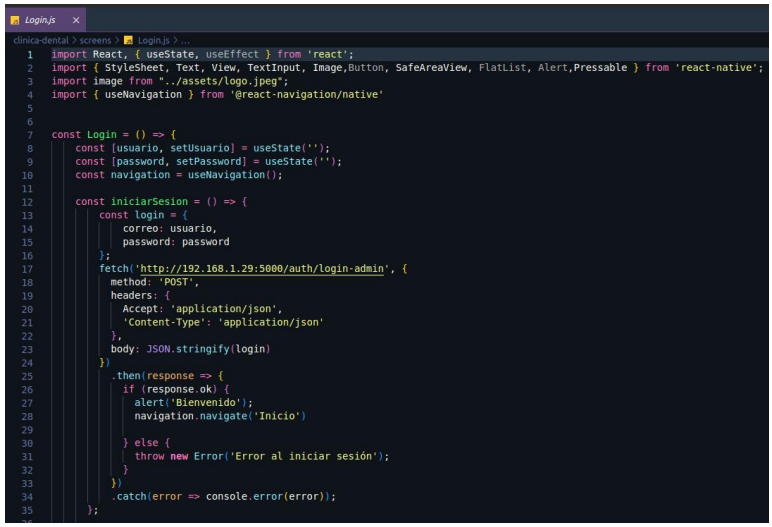
```
import CrearDoctor from './screens/CrearDoctor'
import CrearCita from './screens/CrearCita'
import ModificarCita from './screens/ModificarCita'
import ModificarDoctor from './screens/ModificarDoctor'
import ModificarPaciente from './screens/ModificarPaciente'
import Login from './screens/Login'

import  { NavigationContainer } from '@react-navigation/native';
import  { createStackNavigator } from '@react-navigation/stack';

const Stack = createStackNavigator()

export default function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator>
        <Stack.Screen name ="Login" component={Login}/>
        <Stack.Screen name ="Inicio" component={Home}/>
        <Stack.Screen name ="Doctores" component={Doctores}/>
        <Stack.Screen name ="Pacientes" component={Pacientes}/>
        <Stack.Screen name ="Citas" component={Citas}/>
        <Stack.Screen name ="Crear Paciente" component={CrearPaciente}/>
        <Stack.Screen name ="Crear Doctor" component={CrearDoctor}/>
        <Stack.Screen name ="Crear Cita" component={CrearCita}/>
        <Stack.Screen name ="Modificar Cita" component={ModificarCita}/>
        <Stack.Screen name ="Modificar Doctor" component={ModificarDoctor}/>
        <Stack.Screen name ="Modificar Paciente" component={ModificarPaciente}/>
      </Stack.Navigator>
    </NavigationContainer>
  )
}
```

Screens/
Contiene las diferentes pantallas de navegación dentro del aplicativo.



Para terminos de documentación se explicara el contenido del Home, Login y del objeto Doctor, ya que los demas guardan la misma estructura y contenido lógico.

Login.js
Contiene formulario de usuario y password para iniciar sesión, para ello envia un POST Request al API. En caso obtiene resultado exitoso envia al Home, caso contrario muestra mensaje de error al iniciar sesión y no redirecciona.



```
import React, { useState, useEffect } from 'react';
import { StyleSheet, Text, View, TextInput, Image,Button, SafeAreaView, FlatList, Alert,Pressable } from 'react-native';
import image from "../assets/logo.jpeg";
import { useNavigation } from '@react-navigation/native'


const Login = () => {
    const [usuario, setUsuario] = useState('');
    const [password, setPassword] = useState('');
    const navigation = useNavigation();

    const iniciarSesion = () => {
        const login = {
            correo: usuario,
            password: password
        };
        fetch('http://192.168.1.29:5000/auth/login-admin', {
          method: 'POST',
          headers: {
            Accept: 'application/json',
            'Content-Type': 'application/json'
          },
          body: JSON.stringify(login)
        })
          .then(response => {
            if (response.ok) {
              alert('Bienvenido');
              navigation.navigate('Inicio')

            } else {
              throw new Error('Error al iniciar sesión');
            }
          })
          .catch(error => console.error(error));
      };
```

```
    return (
        <SafeAreaView style={styles.container}>
            <Image style = {styles.logo}
            source={image}>
            </Image>

            <Text style={styles.title}>Inicia Sesión</Text>
            <View style={styles.formulario}>

                <TextInput
                style={styles.input}
                placeholder="Usuario"
                value={usuario}
                onChangeText={text => setUsuario(text)}
                />

                <TextInput
                style={styles.input}
                placeholder="Contraseña"
                value={password}
                onChangeText={text => setPassword(text)}
                />


                <Pressable onPress={iniciarSesion}  style={styles.button}>
                    <Text style={styles.textButton}>Iniciar Sesión</Text>
                </Pressable>

            </View>
        </SafeAreaView>
    );
}

export default Login
```

Home.js

Contiene el menú principal de la aplicación:

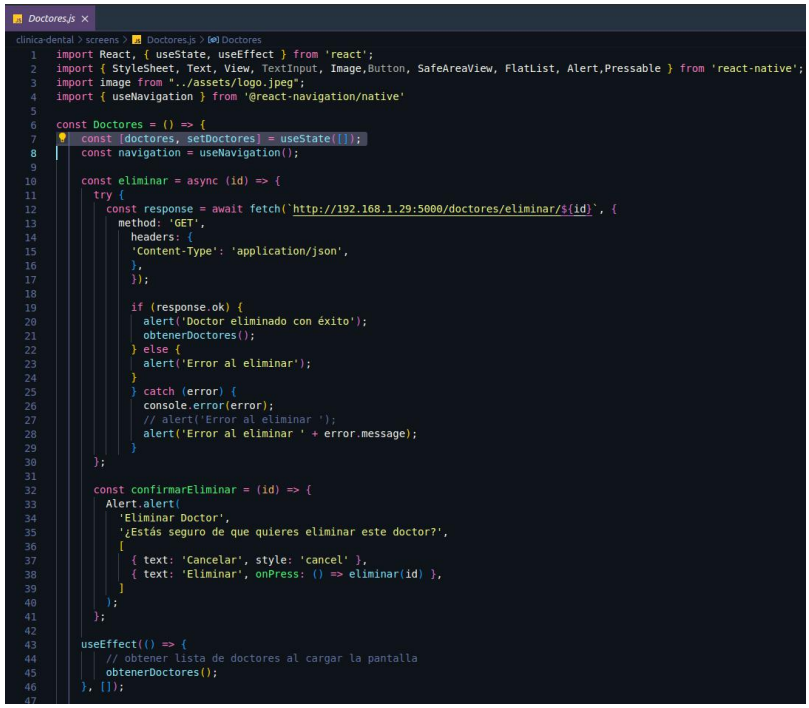
Home.js

clinica-dental > screens > Home.js > Home

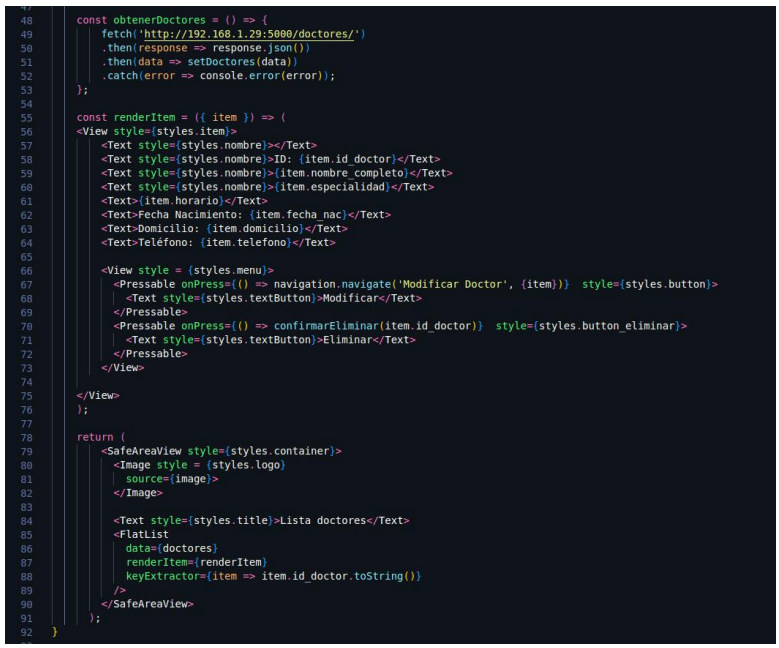
```
1   import React from 'react';
2   import { StyleSheet, Text, View, Image,Button, SafeAreaView, Pressable } from 'react-native';
3   import image from "../assets/logo.jpeg";
4   import { useNavigation } from '@react-navigation/native'
5
6   const Home = () => {
7     const navigation = useNavigation();
8
9       return(
10        <SafeAreaView style={styles.container}>
11        <Image style = {styles.logo} source={image}></Image>
12
13        <View style = {styles.menu}>
14        <Text style = {styles.title}>Pacientes</Text>
15          <Pressable onPress={() => navigation.navigate('Crear Paciente')}  style={styles.butt
16            <Text style={styles.textButton}>Crear</Text>
17          </Pressable>
18          <Pressable onPress={() => navigation.navigate('Pacientes')}  style={styles.button}>
19            <Text style={styles.textButton}>Ver</Text>
20          </Pressable>
21        </View>
22
23        <View style = {styles.menu}>
24        <Text style = {styles.title}>Doctores</Text>
25          <Pressable onPress={() => navigation.navigate('Crear Doctor')}  style={styles.buttor
26            <Text style={styles.textButton}>Crear</Text>
27          </Pressable>
28          <Pressable onPress={() => navigation.navigate('Doctores')}  style={styles.button}>
29            <Text style={styles.textButton}>Ver</Text>
30          </Pressable>
31        </View>
32
33        <View style = {styles.menu}>
34        <Text style = {styles.title}>Citas</Text>
35          <Pressable onPress={() => navigation.navigate('Crear Cita')}  style={styles.button}>
36            <Text style={styles.textButton}>Crear</Text>
37          </Pressable>
38          <Pressable onPress={() => navigation.navigate('Citas')}  style={styles.button}>
39            <Text style={styles.textButton}>Ver</Text>
40          </Pressable>
41        </View>
42
43
44      </SafeAreaView>
45    )
46  }
47
48  export default Home
```

Doctores.js

Contiene la vista de lista, desde la cual podemos eliminar o modificar



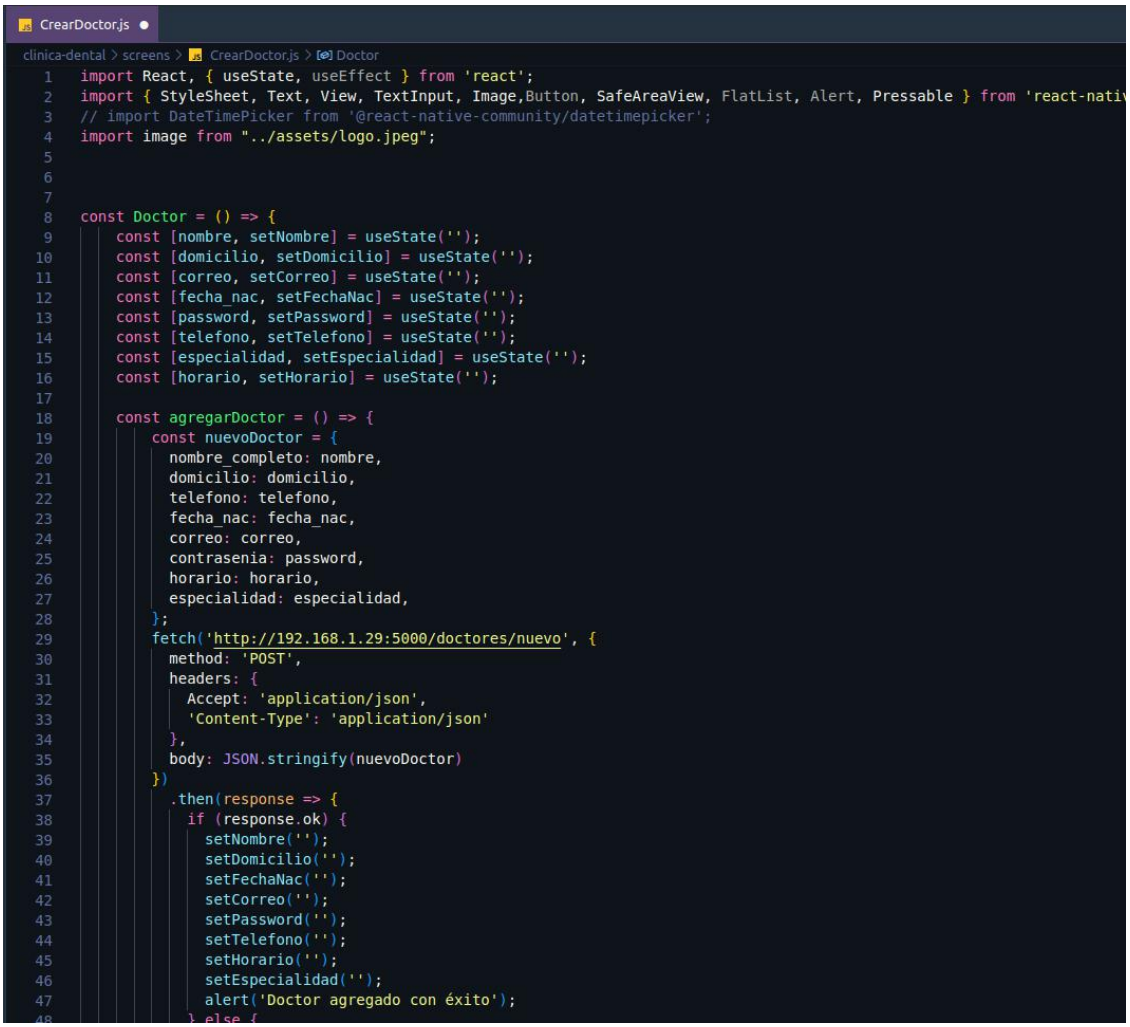
```
import React, { useState, useEffect } from 'react';
import { StyleSheet, Text, View, TextInput, Image,Button, SafeAreaView, FlatList, Alert,Pressable } from 'react-native';
import image from "../assets/logo.jpeg";
import { useNavigation } from '@react-navigation/native'

const Doctores = () => {
  const [doctores, setDoctores] = useState([]);
  const navigation = useNavigation();

  const eliminar = async (id) => {
    try {
      const response = await fetch(`http://192.168.1.29:5000/doctores/eliminar/${id}`, {
        method: 'GET',
          headers: {
          'Content-Type': 'application/json',
          },
          });

          if (response.ok) {
            alert('Doctor eliminado con éxito');
            obtenerDoctores();
          } else {
            alert('Error al eliminar');
          }
          } catch (error) {
            console.error(error);
            // alert('Error al eliminar ');
            alert('Error al eliminar ' + error.message);
          }
    };

    const confirmarEliminar = (id) => {
      Alert.alert(
        'Eliminar Doctor',
        '¿Estás seguro de que quieres eliminar este doctor?',
        [
          { text: 'Cancelar', style: 'cancel' },
          { text: 'Eliminar', onPress: () => eliminar(id) },
        ]
      );
    };

  useEffect(() => {
      // obtener lista de doctores al cargar la pantalla
      obtenerDoctores();
  }, []);
```



```
  const obtenerDoctores = () => {
      fetch('http://192.168.1.29:5000/doctores/')
      .then(response => response.json())
      .then(data => setDoctores(data))
      .catch(error => console.error(error));
  };

  const renderItem = ({ item }) => (
  <View style={styles.item}>
      <Text style={styles.nombre}></Text>
      <Text style={styles.nombre}>ID: {item.id_doctor}</Text>
      <Text style={styles.nombre}>{item.nombre_completo}</Text>
      <Text style={styles.nombre}>{item.especialidad}</Text>
      <Text>{item.horario}</Text>
      <Text>Fecha Nacimiento: {item.fecha_nac}</Text>
      <Text>Domicilio: {item.domicilio}</Text>
      <Text>Teléfono: {item.telefono}</Text>

      <View style = {styles.menu}>
        <Pressable onPress={() => navigation.navigate('Modificar Doctor', {item})}  style={styles.button}>
          <Text style={styles.textButton}>Modificar</Text>
        </Pressable>
        <Pressable onPress={() => confirmarEliminar(item.id_doctor)}  style={styles.button_eliminar}>
          <Text style={styles.textButton}>Eliminar</Text>
        </Pressable>
      </View>

  </View>
  );

  return (
      <SafeAreaView style={styles.container}>
        <Image style = {styles.logo}
          source={image}>
        </Image>

        <Text style={styles.title}>Lista doctores</Text>
        <FlatList
          data={doctores}
          renderItem={renderItem}
          keyExtractor={item => item.id_doctor.toString()}
        />
      </SafeAreaView>
    );
}
```

CrearDoctor.js

Contiene formulario para creación de doctor

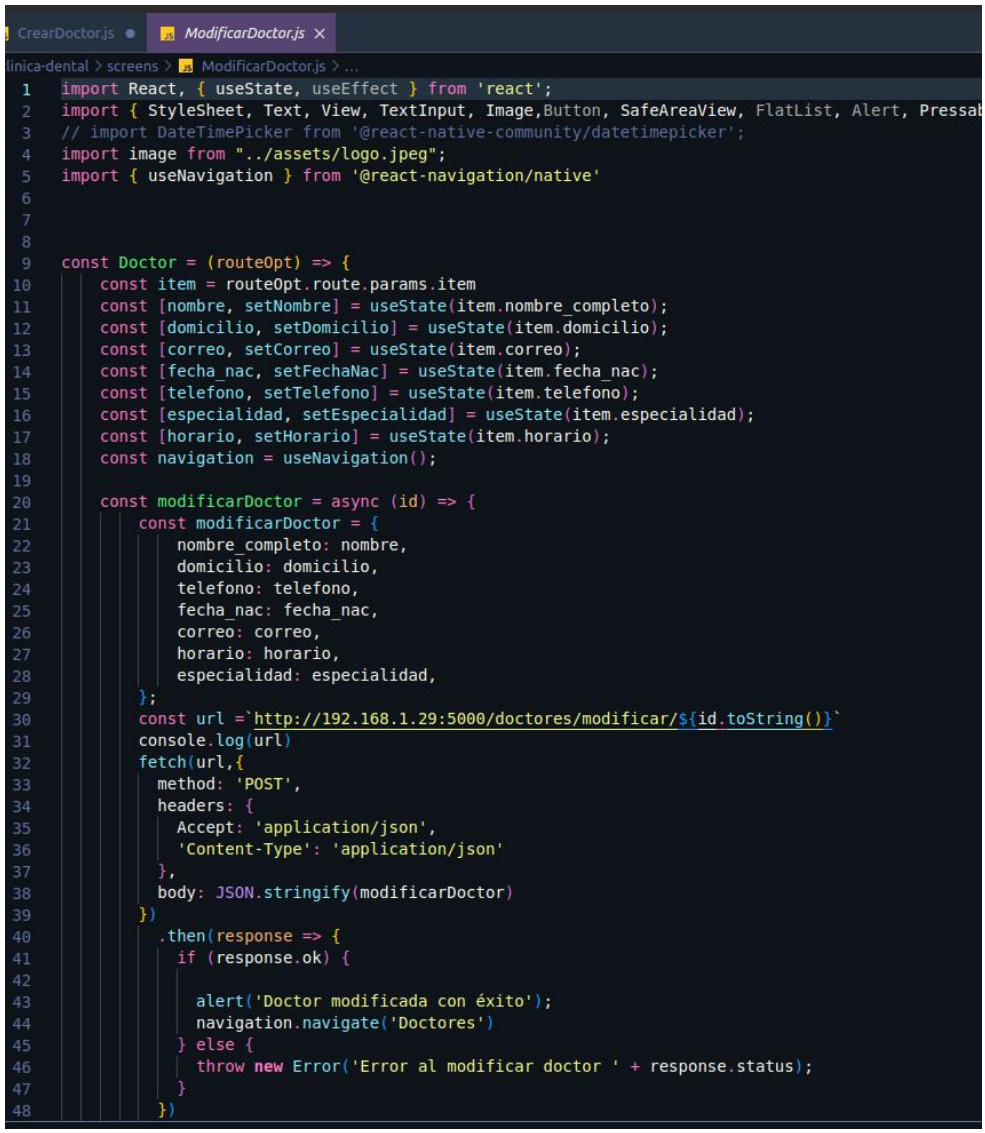


```
        } else {
          throw new Error('Error al agregar Doctor');
        }
      })
      .catch(error => console.error(error));
  };

  return (
    <SafeAreaView style={styles.container}>
      <Image style = {styles.logo}
        source={image}>
      </Image>

      <View style={styles.formulario}>

        <TextInput
        style={styles.input}
        placeholder="Nombre"
        value={nombre}
        onChangeText={text => setNombre(text)}
        />
        <TextInput
        style={styles.input}
        placeholder="Domicilio"
        value={domicilio}
        onChangeText={text => setDomicilio(text)}
        />

        <TextInput
        style={styles.input}
        placeholder="Telefono"
        value={telefono}
        onChangeText={text => setTelefono(text)}
        />


        <TextInput
        style={styles.input}
        placeholder="Correo"
        value={correo}
        onChangeText={text => setCorreo(text)}
        />
```

```
        />

        <TextInput
        style={styles.input}
        placeholder="Fecha Nacimiento"
        value={fecha_nac}
        onChangeText={text => setFechaNac(text)}
        />
        <TextInput
        style={styles.input}
        placeholder="Contraseña"
        value={password}
        onChangeText={text => setPassword(text)}
        />
        <TextInput
        style={styles.input}
        placeholder="Horario"
        value={horario}
        onChangeText={text => setHorario(text)}
        />
        <TextInput
        style={styles.input}
        placeholder="Especialidad"
        value={especialidad}
        onChangeText={text => setEspecialidad(text)}
        />
        <Pressable onPress={agregarDoctor}  style={styles.button}>
            <Text style={styles.textButton}>Agregar</Text>
        </Pressable>

      </View>
    </SafeAreaView>
  );
}

export default Doctor


const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: 'white'
  },
```

ModificarDoctor.js

Contiene formulario con envio POST a modificación de doctor.



```
return (
    <SafeAreaView style={styles.container}>
      <Image style = {styles.logo}
        source={image}>
      </Image>

      <View style={styles.formulario}>
      <TextInput
            style={styles.input}
            placeholder="Nombre"
            value={nombre}
            onChangeText={text => setNombre(text)}
            />

            <TextInput
            style={styles.input}
            placeholder="Domicilio"
            value={domicilio}
            onChangeText={text => setDomicilio(text)}
            />

            <TextInput
            style={styles.input}
            placeholder="Telefono"
            value={telefono}
            onChangeText={text => setTelefono(text)}
            />


            <TextInput
            style={styles.input}
            placeholder="Correo"
            value={correo}
            onChangeText={text => setCorreo(text)}
            />

            <TextInput
            style={styles.input}
            placeholder="Fecha Nacimiento"
            value={fecha_nac}
            onChangeText={text => setFechaNac(text)}
            />
```

```
            <TextInput
            style={styles.input}
            placeholder="Correo"
            value={correo}
            onChangeText={text => setCorreo(text)}
            />

            <TextInput
            style={styles.input}
            placeholder="Fecha Nacimiento"
            value={fecha_nac}
            onChangeText={text => setFechaNac(text)}
            />

            <TextInput
            style={styles.input}
            placeholder="Horario"
            value={horario}
            onChangeText={text => setHorario(text)}
            />
            <TextInput
            style={styles.input}
            placeholder="Especialidad"
            value={especialidad}
            onChangeText={text => setEspecialidad(text)}
            />

            <Pressable onPress={() => modificarDoctor(item.id_doctor)}  style={styles.button}>
                <Text style={styles.textButton}>Modificar</Text>
            </Pressable>

          </View>
      </SafeAreaView>
    );
}

export default Doctor
```