

Informe Proyecto

Semana: 10

Integrantes:

Esther Hernández	12211292
Jorge López	12141356
Josue Ham	12141190
Oliver Iraheta	12151028
Victor Romero	12211079

Sede de estudio:

UNITEC TGU

Docente:

Ing. Claudia Cortez

Sección:

1561

Fecha de entrega:

Domingo 23 de junio de 2024.

a. Tecnologías Utilizadas

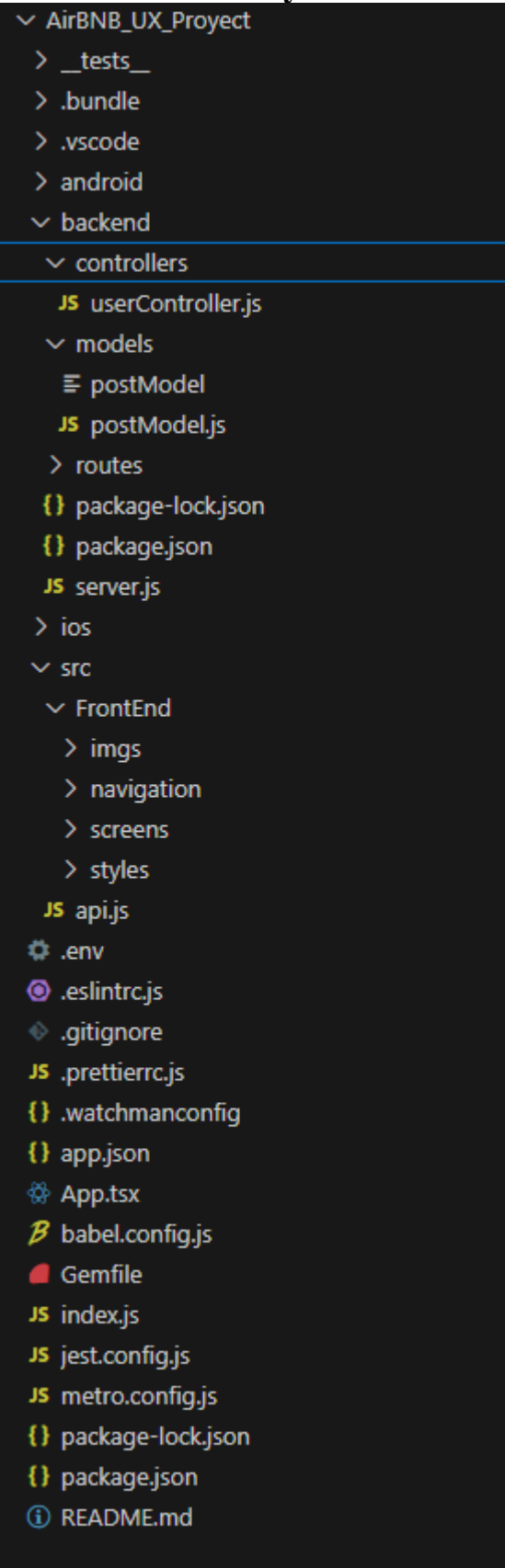
- Android Studio 2024.1.1
- Express 4.19.2
- MongoDB 7.0.11
- Node.js 6.7.0
- Firebase 13.8.0
- Postman v11
- Git
- Github
- React Native 0.74
- Metro 0.80.9
- Visual Studio Code
- Ngrok 3.11.0

b. Servicios De Terceros

- Firebase Authentication
- Mongo Object ID

Anexos

Estructura del Proyecto



Sign Up

A screenshot of the Airbnb mobile app's sign-up screen. The screen has a solid green background. At the top, there's a white header bar with a back arrow and the text "Signup". Below this, the Airbnb logo (a white stylized 'A') is centered, followed by the word "airbnb" in white lowercase letters. Underneath the logo, there are two light green rounded rectangular input fields. The first field is labeled "Email" and the second is labeled "Password". Below these fields is a white rounded rectangular button with the text "SIGNUP" in green. At the bottom of the screen, there is a line of white text that reads "Welcome to this Amazing Experience!". The phone's status bar at the very top shows the time "4:17" and various icons for battery, signal, and Wi-Fi.

4:17

← Signup



airbnb

Email

Password

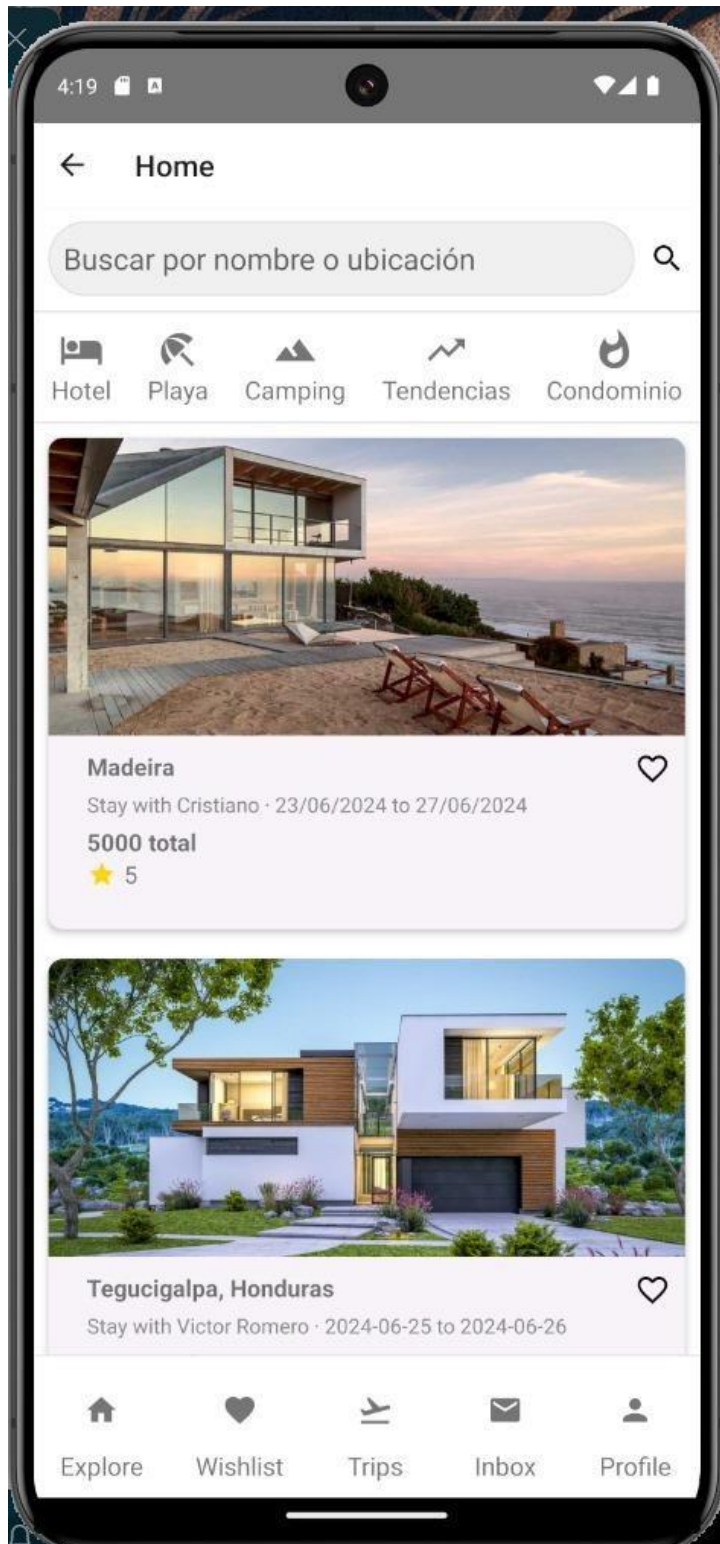
SIGNUP

Welcome to this Amazing Experience!

Log In

A screenshot of the Airbnb mobile app's login screen. The screen has a solid coral-red background. At the top, a white header bar contains the word "Login" in black. Below the header, the Airbnb logo (a white stylized 'A' shape) is centered, with the word "airbnb" in white lowercase letters directly beneath it. Further down, there are two light pink rounded rectangular input fields. The first field is labeled "Email" in black text, and the second field is labeled "Password" in black text. Below these fields is a white rounded rectangular button with the text "LOG IN" in red, uppercase letters. At the bottom of the screen, the text "Don't have an account? Sign up" is displayed in white, with "Sign up" being a link. The entire screen is framed by a black border representing the phone's bezel. At the very top of the phone's display, a status bar shows the time "4:18" and various system icons (signal, Wi-Fi, battery).

Página Principal



BackEnd

Sign Up

```
app.post('/signUpFirebase', async (req, res) => {
  const auth = getAuth(firebaseApp);
  const email = req.body.correo;
  const password = req.body.contrasena;
  try {
    const userCredential = await createUserWithEmailAndPassword(
      auth,
      email,
      password,
    );
    res.status(200).send({
      descripcion: 'usuario creado con exito',
      resultado: userCredential,
    });
  } catch (error) {
    console.error('Hubo un error al crear el usuario', error);
    res.status(500).send({
      descripcion: 'No se pudo crear el usuario en firebase',
      resultado: error,
    });
  }
});
```

Login

```
app.post('/logInFirebase', async (req, res) => {
  const auth = getAuth(firebaseApp);
  const { email, password } = req.body;

  try {
    const userCredential = await signInWithEmailAndPassword(
      auth,
      email,
      password,
    );
    res.json({
      success: true,
      user: userCredential.user,
    });
  } catch (error) {
    res.status(401).json({
      success: false,
      error: 'Invalid email or password',
    });
  }
});
```

LogOut

```
app.post('/logOutFirebase', async (req, res) => {
  const auth = getAuth(firebaseApp);
  try {
    await signOut(auth);
    res.status(200).send({
      descripcion: 'Sesion cerrada con exito',
    });
  } catch (error) {
    res.status(500).send({
      descripcion: 'No se pudo cerrar sesion',
      resultado: error,
    });
  }
});
```

```
app.post('/logIn', (req, res) => {
  console.log('body-parseado', req.body);
  res.status(200).send('Info recibida!');
});

app.post('/signUp', (req, res) => {
  res.status(200).send({
    mensaje: 'Usuario creado con exito',
  });
});

app.put('/update', (req, res) => {
  res.status(200).send({
    mensaje: 'Usuario actualizado con exito',
  });
});

app.delete('/delete', (req, res) => {
  res.status(200).send({
    mensaje: 'Usuario eliminado con exito',
  });
});
```


Crear Lugar

```
app.post('/createLugar', async (req, res)=>{
  await client.connect();
  const owner = req.body.owner
  const nombre = req.body.nombre
  const categoria = req.body.categoria
  const ubicacion = req.body.ubicacion
  const url = req.body.url
  const precio = req.body.precio
  const horario = req.body.horario
  const rating = req.body.rating
  const cantidadPersonas = req.body.cantidadPersonas
  const fechaEntrada = req.body.fechaEntrada
  const fechaSalida = req.body.fechaSalida
  const lugar = {owner, nombre, categoria, ubicacion, url, precio, horario, rating, cantidadPersonas, fechaEntrada, fechaSalida}
  const result = await posts.insertOne(lugar);
  if (!result.insertedId) {
    res.status(500).send({
      msg: "No se pudo crear el lugar",
    });
  }
  res.status(200).send({
    msg: "Lugar creado exitosamente",
    data: result.insertedId,
  });
  await client.close();
})
```

Listar Lugares

```
app.get('/listLugares', async (req, res)=>{
  await client.connect();
  if ((await posts.countDocuments()) === 0) {
    res.status(200).send({
      msg: "No hay lugares guardados",
    });
  }
  const query = posts.find();
  let arrPosts = [];
  for await (const doc of query) {
    arrPosts.push(doc);
  }
  res.status(200).send({
    documentos: arrPosts,
  });
  await client.close();
})
```

Eliminar Lugar

```
app.delete('/deleteLugar', async (req, res)=>{
  await client.connect();
  if ((await posts.countDocuments()) === 0) {
    res.status(200).send({
      msg: "No hay lugares guardados",
    });
  }

  if (await !posts.findOne({ _id: new ObjectId(req.params.id) })) {
    return res.status(500).send({
      msg: `No se encontró ningún lugar con id ${res.body.id}`,
    });
  }

  const filter = { _id: new ObjectId(req.params.id) };
  const result = await posts.deleteOne(filter);
  res.status(200).send("El lugar fue eliminado exitosamente");
  await client.close();
})
```

Editar Lugar

```
app.put('/editLugar', async (req, res)=>{
  await client.connect();
  if ((await posts.countDocuments()) === 0) {
    res.status(200).send({
      msg: "No hay lugares guardados",
    });
  }
  if (await !posts.findOne({ _id: new ObjectId(req.params.id) })) {
    return res.status(500).send({
      msg: `No se encontró ningún lugar con id ${res.body.id}`,
    });
  }

  const filter = { _id: new ObjectId(req.params.id) };
  const update = { $set: { owner: req.body.owner,
    nombre: req.body.nombre,
    categoria: req.body.categoria,
    ubicacion: req.body.ubicacion,
    precio: req.body.precio,
    horario: req.body.horario,
    rating: req.body.rating,
    cantidadPersonas: req.body.cantidadPersonas,
    fechaEntrada: req.body.fechaEntrada,
    fechaSalida: req.body.fechaSalida } };
  const options = { upsert: false };

  const result = await posts.updateOne(filter, update, options);

  res.status(200).send("El lugar fue editado exitosamente");
  await client.close();
})
```

FrontEnd

```
AirBNB_UX_Project > App.tsx > ...
1  import React from 'react';
2  import {NavigationContainer} from '@react-navigation/native';
3  import {createStackNavigator} from '@react-navigation/stack';
4  import HomeScreen from './src/FrontEnd/screens/HomeScreen';
5  import DetailsScreen from './src/FrontEnd/screens/DetailsScreen';
6  import LoginScreen from './src/FrontEnd/screens/LoginScreen';
7  import SignupScreen from './src/FrontEnd/screens/SignupScreen';
8
9  const Stack = createStackNavigator();
10
11  const AppNavigator: React.FC = () => (
12    <NavigationContainer>
13      <Stack.Navigator initialRouteName="Login">
14        <Stack.Screen name="Login" component={LoginScreen} />
15        <Stack.Screen name="Signup" component={SignupScreen} />
16        <Stack.Screen name="Home" component={HomeScreen} />
17        { /*<Stack.Screen name="Details" component={DetailsScreen} />*/ }
18      </Stack.Navigator>
19    </NavigationContainer>
20  );
21
22  export default AppNavigator;
23
```

