Software Requirements Specification

for

Reservation system of equipment and rooms management

Version 2.0 approved

Prepared by

Garcia Reyes Jose Adan

Gómez Cueva Elías Jair

Martínez Bustamante Daniel

Martínez Pérez Diana

18/March/2025

Document information

Date	Review	Author	Verified by quality control.
20/January/2025	**Review 1**	Garcia Reyes Jose Adan Gómez Cueva Elías Jair Martinez Bustamante Daniel Martinez Perez Diana	Ray Brunett Parra Galaviz
18/March/2025	**Review 2**	Garcia Reyes Jose Adan Gómez Cueva Elías Jair Martinez Bustamante Daniel Martinez Perez Diana	Ray Brunett Parra Galaviz

By the client	By the Supplier Company
Ray Brunett Parra Galaviz	Garcia Reyes Jose Adan Gómez Cueva Elías Jair Martinez Bustamante Daniel Martinez Perez Diana

Table of Contents

1 Introduction

This program is designed to streamline the booking and management of resources within an organization, ensuring efficient utilization of equipment and spaces. Built using Java for robust backend functionality and React Native for a responsive and user-friendly interface, the system will provide a smooth experience on mobile. Additionally, it will leverage a database connected through APIs to securely store and manage critical information, enabling reliable and efficient system operation.

1.1 Purpose

The aim of this project is to develop software that facilitates the booking and management of business resources by efficiently organising bookings and equipment availability. This system will help companies to keep track of their equipment and resources, optimising their use and availability. It will also provide tools to improve coordination between employees who need resources to carry out activities, facilitating better internal organisation.

The software will include features such as an intuitive interface that allows bookings to be managed in a simple and efficient manner, integrating specific modules for the administration of equipment and rooms.

1.2 Product Scope

We will develop a business resource management and reservation software called (project name). This system will optimize the management of equipment and rooms in companies, ensuring their availability in an efficient manner.

The software will include the following modules and functionalities:

User Management, Availability Module, Resource Management, Reports and Histories:

The main objective of the software is to improve the management of business resources, facilitate their availability and promote more effective communication between employees who require equipment and rooms for their daily activities.

1.3 Document Conventions

Style and Document Format

- **Titles:** Times New Roman font, size 14, bold, black color.
- **Subtitles:** Times New Roman font, size 12, bold, black color.
- **Text:** Times New Roman font, size 12, black color.

Personnel involved

Name	Gómez Elias	Martinez Daniel	Martínez Pérez Diana	Garcia Adan
Role	Member in charge	Member in charge	Member in charge	The leader
Professional Category	Review and elaboration	Review and elaboration	Review and elaboration	Review and elaboration
Responsibilities	Service-oriented web applications	Mobile applications	Internet of Things	Tester
Contact	664 215 9161	664 135 2020	664 309 6515	663 364 3953

1.4 Intended Audience and Suggestion

The target audience is mainly companies and organizations that require efficient management of their shared resources, such as technological equipment, work tools, and physical spaces.

This solution is especially useful for companies that value transparency, accountability, and better communication between employees for good team and room management.

1.5 References

Reference	Title	Link	Date	Author
● ***TSU***	Higher university technician TSU	https://uvp.mx/uvpblog/tecnico-superior-universitario-tsu/	2023, 1 July	Universidad del Valle de Puebla

2 Overall Description

2.1 Product Perspective

The Reservation and Resource Management System will be developed as a central platform for efficiently managing the booking and usage of equipment and rooms within an organization.

The system will be designed to cater to companies, educational institutions that need to optimize the use of shared resources. By utilizing React Native for the mobile interface, the software ensures reliability and responsiveness. The product will connect to a database via APIs to securely store and manage data related to user profiles, equipment availability, reservation details, and usage histories. This integration will enable real-time availability tracking and ensure the smooth operation of the system.

2.2 Product Functionality

Equipment Inventory Module:

- Equipment Registration (serial number, name, type or category, characteristics, brand, model).
- Equipment Status (available, in use, out of service, removed).
- Equipment Categories (laptops, projectors, machines, etc.).

Room Module:

- Room Registration (number, name, location).
- Room Status (available, in use, out of service).

User Management Module:

- User Registration and Authentication (login/logout).
- User Roles (administrator, regular users).
- Role-based Permission Management.
- User Profile (personal data, reservation history).

Availability Module

- Display boxes showing available equipment and rooms.

Reservation Module

- Report with.
    - Responsible person's name.
    - Requested date.
    - Date of use.
    - Request time.
    - Reservation time.
    - Time used.
    - Requested.
        - Requested room.
        - Equipment used.
    - Description of use.

History Module

- Equipment and rooms used by the user (for the user) and accessible by the admin for all users.

2.3 User Classes and Characteristics

Regular User:

- Can reserve equipment and rooms as needed.
- Can view the availability of resources and make reservations based on that availability.
- Can manage their own profile and view their own reservation history.

Administrator:

- Can register, update, and delete equipment, rooms, and users.
- Can assign roles and manage user permissions.
- Can view the reservation history for all users within the system.
- Can generate reports on resource usage and reservation statistics.

2.4 Operating Environment

The web application will be developed using standard technologies such as Java, Spring boot and the mobile application will be developed using React native and Expo Go, integrated with a database API to manage the information. The system will be designed to be accessible from a mobile device.

2.5 Design and Implementation Constraints

- The system will only be accessible with a login, which means there will be authentication.
- The permission structure must be strict, making sure only authorized users can access certain functions based on their roles.
- The data storage and management must follow security and data protection rules.

2.6 User Documentation

- User Manual: Instructions for using the system, explaining how to manage employees, create tasks, assign roles, and create reports.
- Administrator Guide: Document for system administrators on how to configure the system.

2.7 Assumptions and Dependencies

2.7.1 Assumptions

- Resource Availability
  - It is assumed that equipment and rooms are correctly registered in the system with current and accurate information.
- User Roles and Responsibilities:
  - Users adhere to assigned permissions and roles, and administrators manage resources according to company policies.
- Internet Connectivity:
  - The system requires a stable Internet connection to synchronize reservations and perform real-time queries, especially for the web and mobile version.
- Device Compatibility:
  - It is assumed that the devices used (mobiles and computers) meet the minimum requirements to run the application (such as an updated browser or compatible operating system).
- Data Accuracy:
  - Users enter correct data when registering reservations, equipment, and users to avoid conflicts or errors in the information.

2.7.2 Dependencies

- Database System:
  - The system relies on a database that stores information about equipment, rooms, users, and reservations. The connection to the database is made through an API designed specifically for the system.
- External Libraries and Frameworks:
  - For the development of the system, tools such as:
  - Java: Backend of the system.
  - React Native: For the mobile interface.
  - CSS/HTML: For the web interface.

  - Firebase: Database of the system.
- Authentication Service:
  - Service dependency to authenticate users.
- Hardware Infrastructure:
  - The availability of servers to host the system and the database is critical for the continued operation of the system.
- Operating Environment:
  - The system must work correctly on computers and mobile devices.

3 External Interface Requirements

3.1 User Interfaces

3.1.1 General Design

We seek to appeal to as many people as possible through a simple, beautiful and eye-pleasing interface.

3.1.2 Menu Options

Users will be able to select an option using by pressing on the corresponding button on the graphical interface. Each option will take the user to a new screen or section where they can perform the specific actions related to that category.

3.2 Hardware Interfaces

It will be necessary to have a mobile device that is in good condition and fully functional, it will be required for the execution of the program.

3.3 Software Interfaces

3.3.1 Software Compatibility

In order for the product to be fully functional, without any error or risk of failure, it is necessary for it to use the Android operating system.

3.4 Communications Interfaces

Object-oriented programming interface, network communication interface, hardware communication interface.

Describe the requirements for communication interfaces if there are communications with other systems, and explain which communication protocols will be used.

4 Internal requirements

Requirement number	FR 1
Name	Equipment inventory management
Type	[✓ Requirement] ☐ Constraint
Description	In this module you can register, modify or consult the equipment.
Priority	[✓ High/Essential] ☐ Medium/wished ☐ Low/Optional

Requirement number	FR 2
Name	Room management
Type	[✓ Requirement] ☐ Constraint
Description	In this module you can register a new one or modify an existing one.
Priority	[✓ High/Essential] ☐ Medium/wished ☐ Low/Optional

Requirement number	FR 3
Name	Users management
Type	[✓ ☐ Requirement] ☐ Constraint
Description	In this module you can register, modify or consult the users.
Priority	[✓ ☐ High/Essential] ☐ Medium/wished ☐ Low/Optionall

Requirement number	FR 4
Name	Availability of rooms and equipment management
Type	[✓ ☐ Requirement] ☐ Constraint
Description	In this module you can verify which room or equipment is available.
Priority	[✓ ☐ High/Essential] ☐ Medium/wished ☐ Low/Optional

Requirement number	FR 5
Name	Reservations management
Type	[✓ Requirement] ☐ Constraint
Description	In this module equipment reservations will be managed, a report about the requested equipment will be created, including all the necessary information, like user's name, or reservation date.
Priority	[✓ High/Essential] ☐ Medium/wished ☐ Low/Optional

Requirement number	FR 6
Name	History management
Type	[✓ Requirement] ☐ Constraint
Description	In this module you can check your own history of reservations.
Priority	[✓ High/Essential] ☐ Medium/wished ☐ Low/Optional

4.1 Functional requirements

1. Equipment Inventory Module
2. Room Module
3. User Management Module
4. Availability Module
5. Reservation Module
6. History Module

4.1.1 Functional requirements 1: Equipment Inventory Module

In this module is where we will keep information about the equipment in the company. The administrator can register, delete, update the inventory of the equipment that can be reserved, this helps to organize the equipment used in a company and its availability.

The equipment will have a status, which will be: available, in use, out of service, retired.

The equipment will be separated into types or categories such as: laptops, projectors, machines, etc.

4.1.2 Functional requirements 2: Room Module

This module is where we will store the information of the rooms that are in the company. The administrator will be able to add, delete, and update the inventory of the equipment that can be reserved. This helps to organize the rooms that are used in a company and their availability.

The rooms will have a status, which will be: available, in use, out of service, retired.

4.1.3 Functional requirements 3: User Management Module

There will be only 2 types of users in the system, which will have different permissions in the system. The users will be: administrator and regular user.

This module is where we will keep the records and information of the users who are in the company. The administrator will be able to add, delete, and update users in the system.

Users must be authenticated to log in.

The administrator will have permissions to manage the equipment and rooms in case he requires it, as well as view the reservation history of each one of them.

The regular user will have permissions to make reservations for equipment and rooms, as well as review their reservation history.

4.1.4 Functional requirements 4: Availability Module

In this module the user will be able to:

Check all the registered rooms (number, name, location) and also they will be able to view the availability of it (available, in use or out of service).

Check the information about the registered equipment (ID, name, category, brand, model) and also they will be able to view the availability of it (available, in use or out of service).

4.1.5 Functional requirements 5: Reservation Module

In this module the user will create the reservation for the required equipment, to do that it needs to fill a report with the name of the user that is making the reservation, the date of the day that is being make the request, the date of the day when it will be used, the requested room and the requested equipment.

Also, when the equipment is returned it will be added to a report the time it was returned, the time of use, and the description of the use.

4.1.6 Functional requirements 6: History Module

Here the user will be able to view their own reservation history. The administrator will be able to view all the users reservation history. This will help to improve equipment and room safety by keeping track of who is the possible responsible for a damaged equipment or room.

5 Nonfunctional Requirements

5.1 Security

Total security is guaranteed in the use of the software, safeguarding all stored information and only managed by authorized people without the administration affecting the security of our system.

5.2 Reliability

User interfaces will be easy to access, streamlining their understanding, usefulness and functionality.

5.3 Availability

The system will be available 24 hours a day, every day of the week.

5.4 Maintainability

The developed software will have a user manual that offers all the information necessary for its use and will have monthly updates made by our developers.

6 Other Requirements

6.1 Appendix A: Glossary

Introduction

This appendix provides a glossary of terms used in the document to help clarify technical and specific language for this project

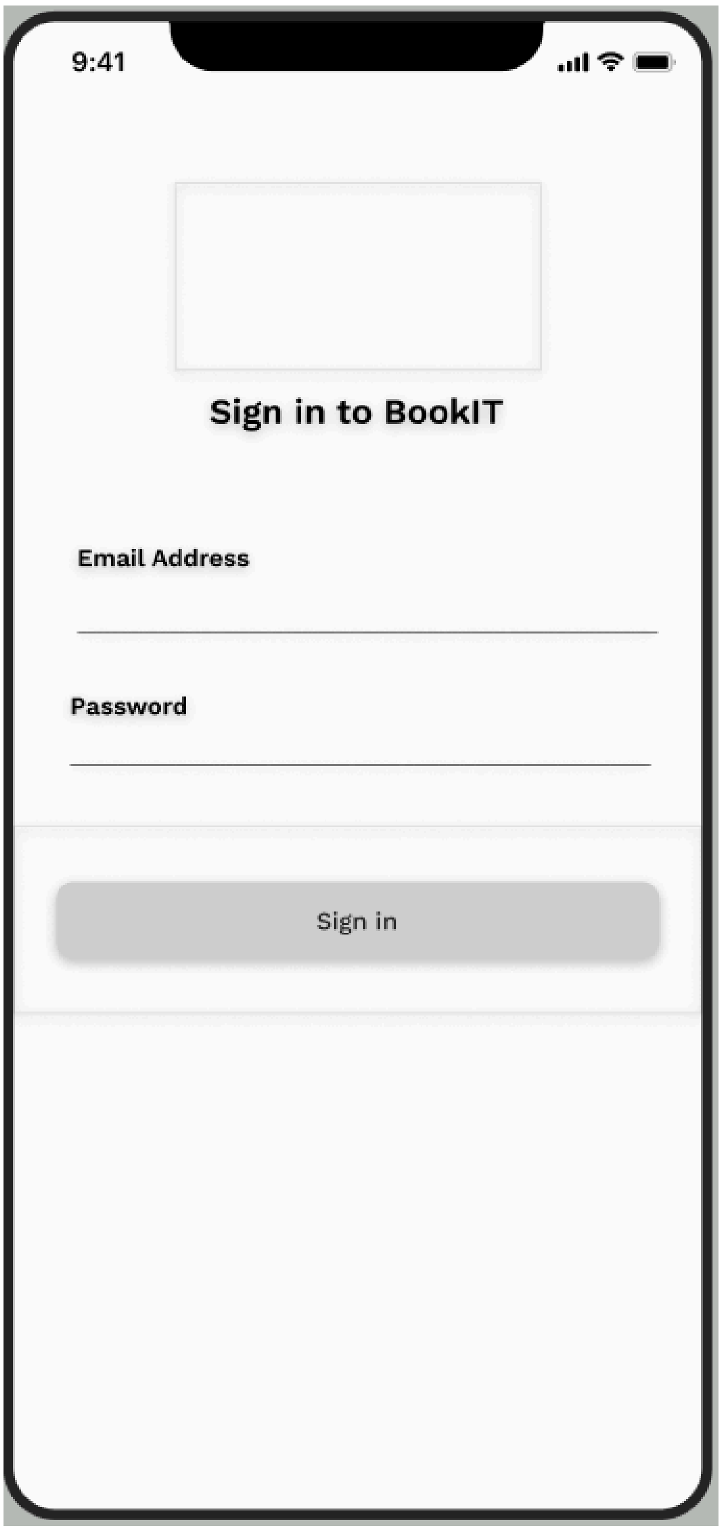
Terms

- **Mockup:**A mockup serves as a basis or guide for how the screen will look in the final application. It's a static design that helps visualize the structure and layout of UI elements before implementing it in code. It's useful for getting a clear idea of the final product and for communicating the design with developers.
- **Screen:** Each of the views or pages within the app. It's the visual part that the user sees and interacts with. It's where the code is actually implemented within the app.

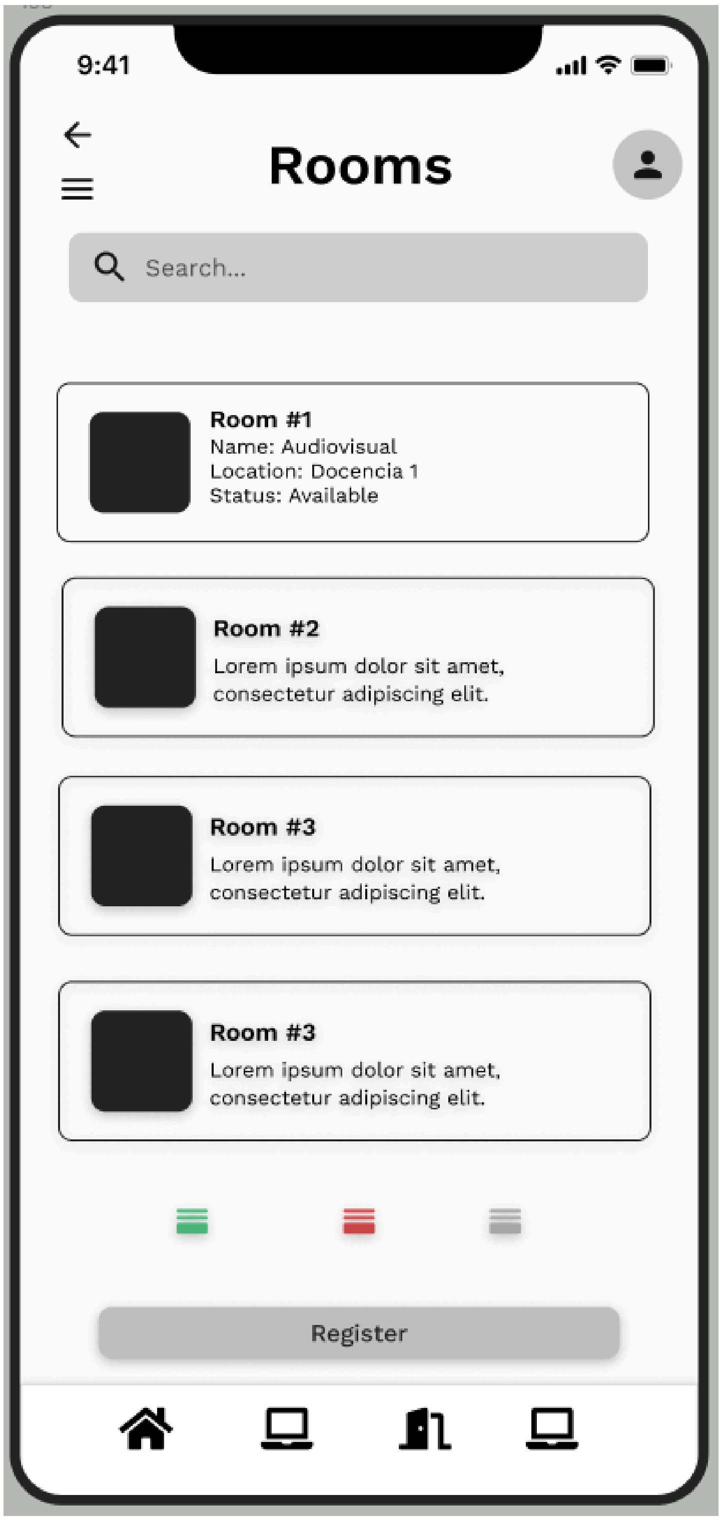
6.2 Appendix B: Analysis Models

6.2.1 Mockups

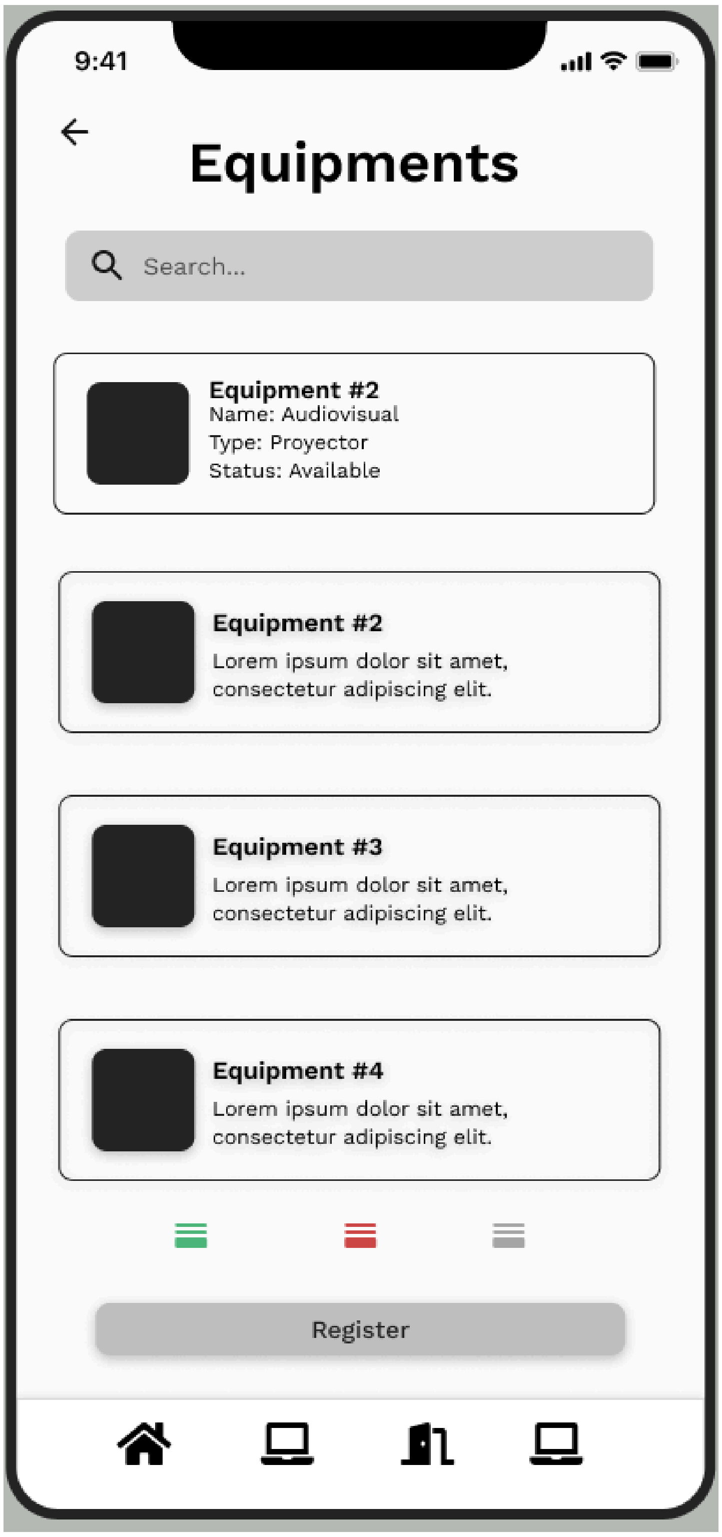
6.2.1.1 Login Mockup



6.2.1.2 Rooms Mockup


9:41
Rooms
Search...
Room #1
Name: Audiovisual
Location: Docencia 1
Status: Available
Room #2
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Room #3
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Room #3
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Register

6.2.1.3 Equipment Mockup



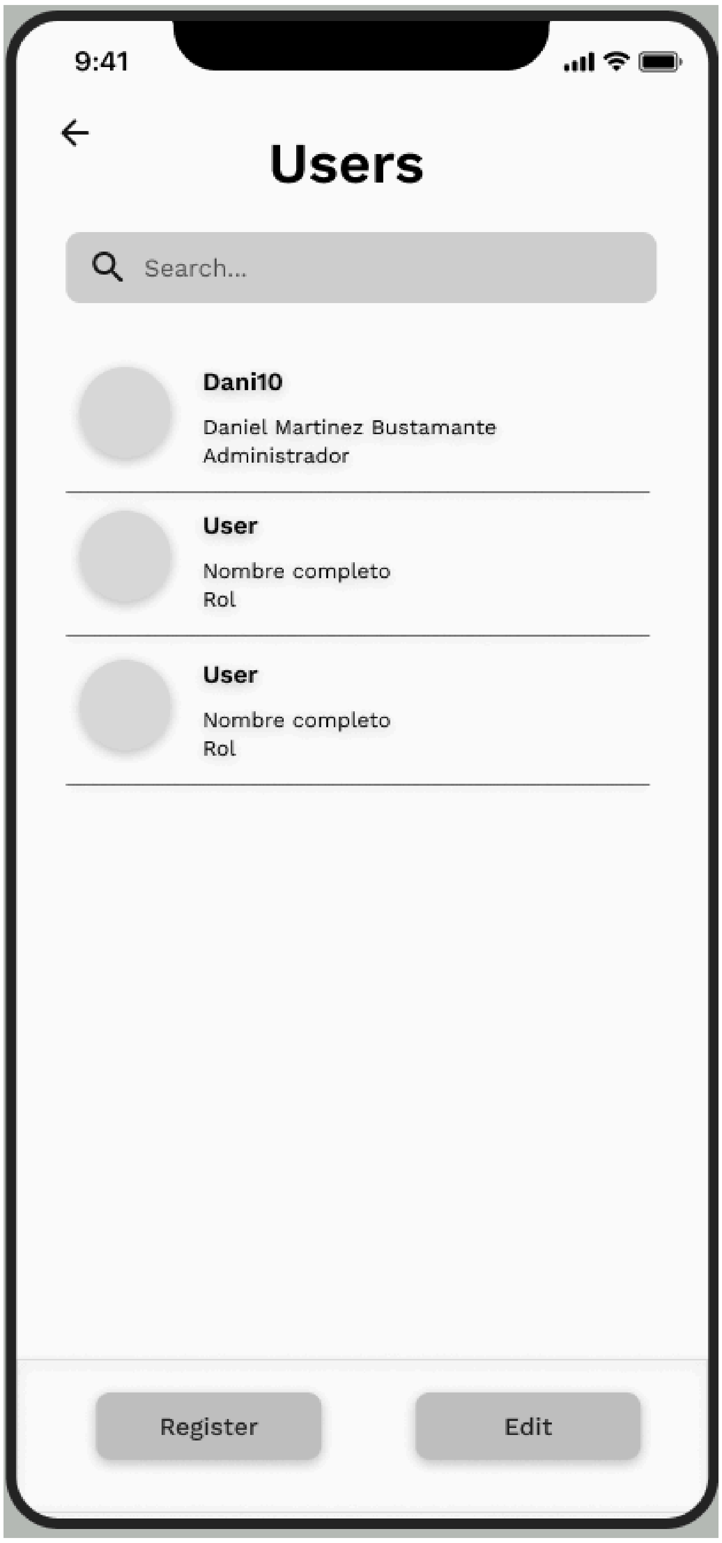
Equipment #2
Name: Audiovisual
Type: Proyector
Status: Available

Equipment #2
Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Equipment #3
Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Equipment #4
Lorem ipsum dolor sit amet, consectetur adipiscing elit.

6.2.1.4 User Mockup

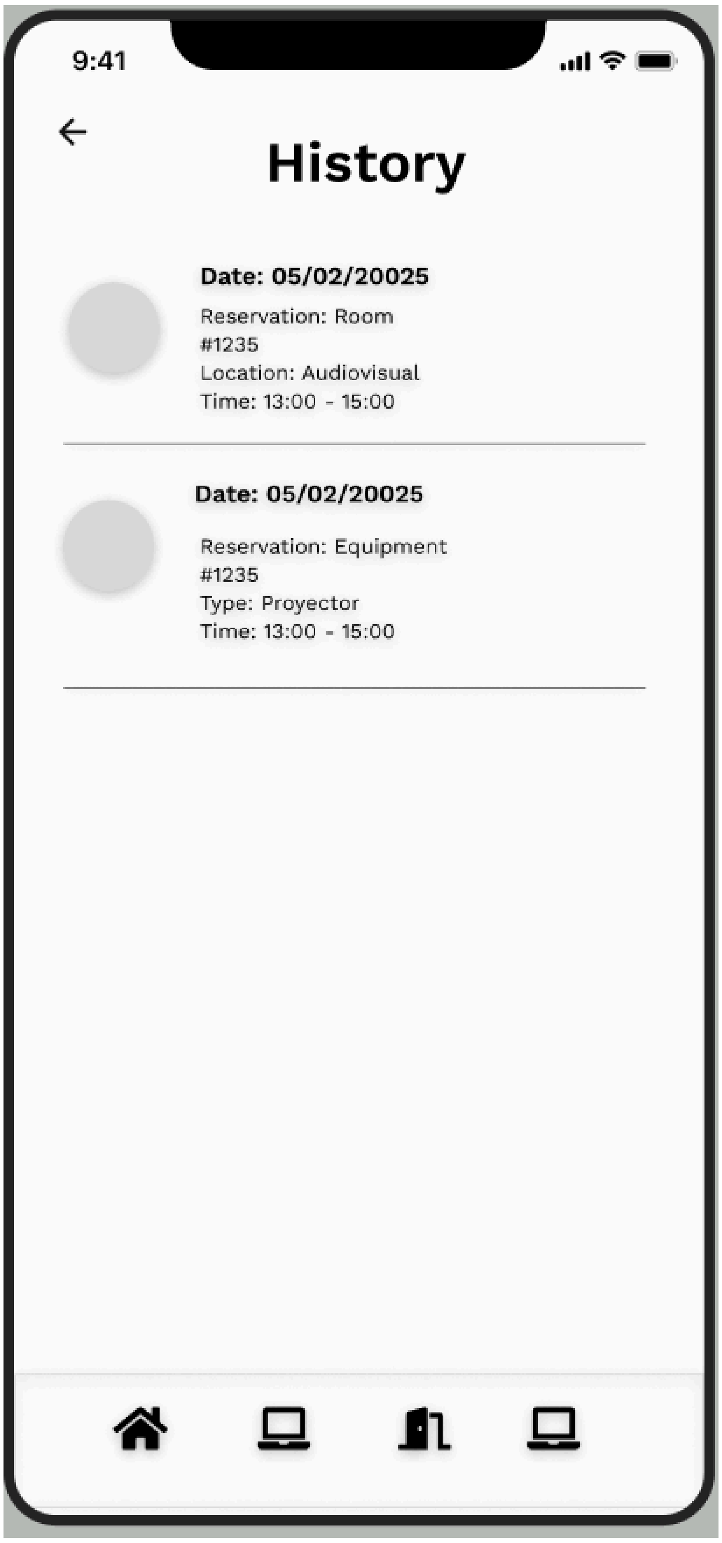


Dani10
Daniel Martinez Bustamante
Administrador

User
Nombre completo
Rol

User
Nombre completo
Rol

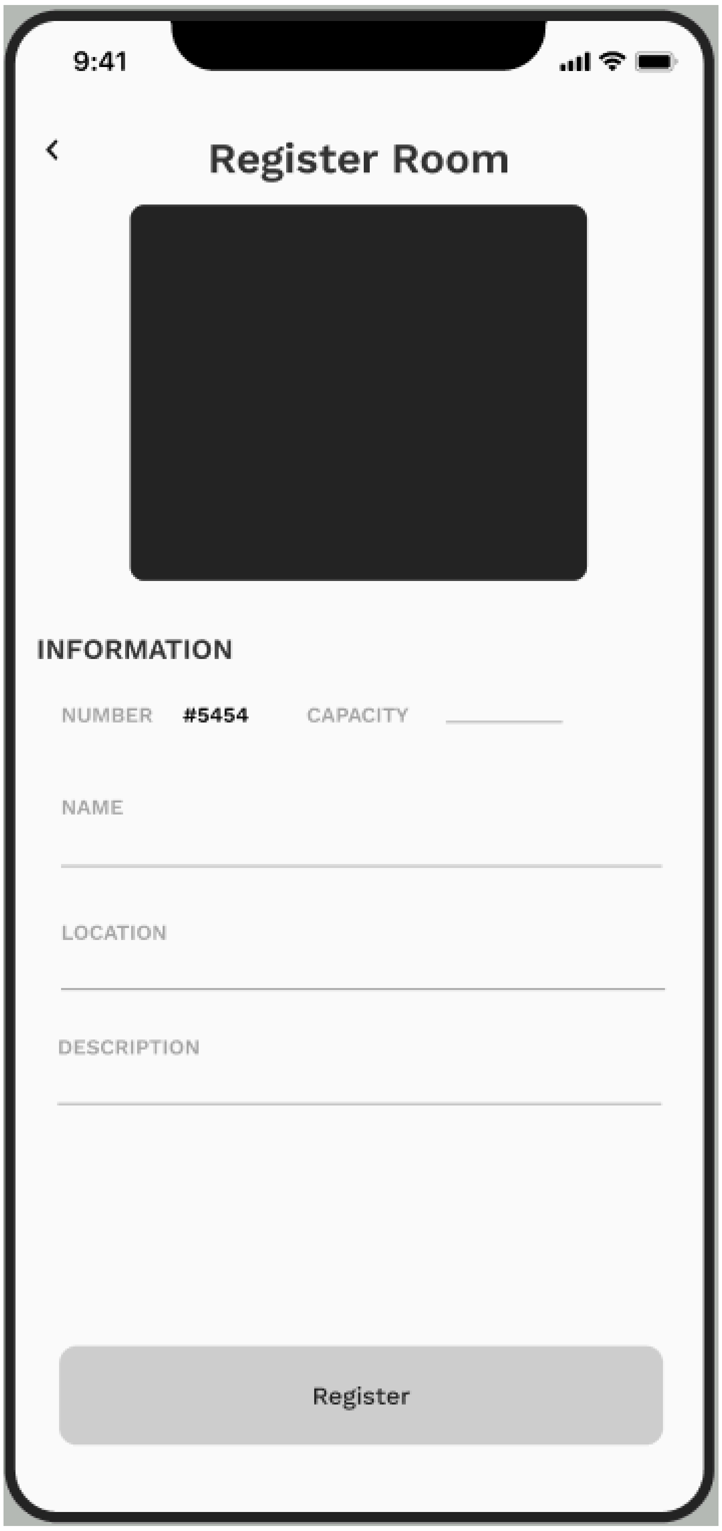
6.2.1.5 History Mockup



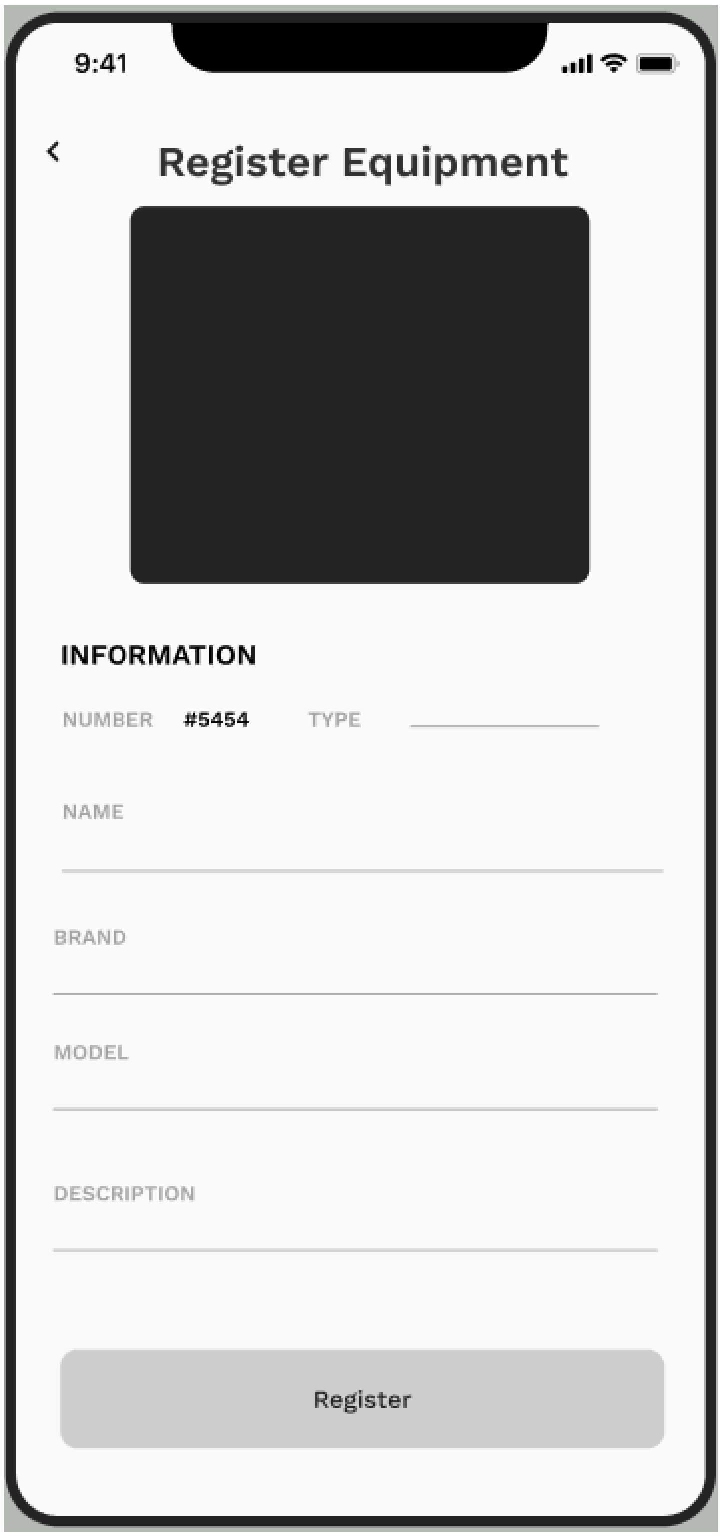
Date: 05/02/20025
Reservation: Room
#1235
Location: Audiovisual
Time: 13:00 - 15:00

Date: 05/02/20025
Reservation: Equipment
#1235
Type: Proyector
Time: 13:00 - 15:00

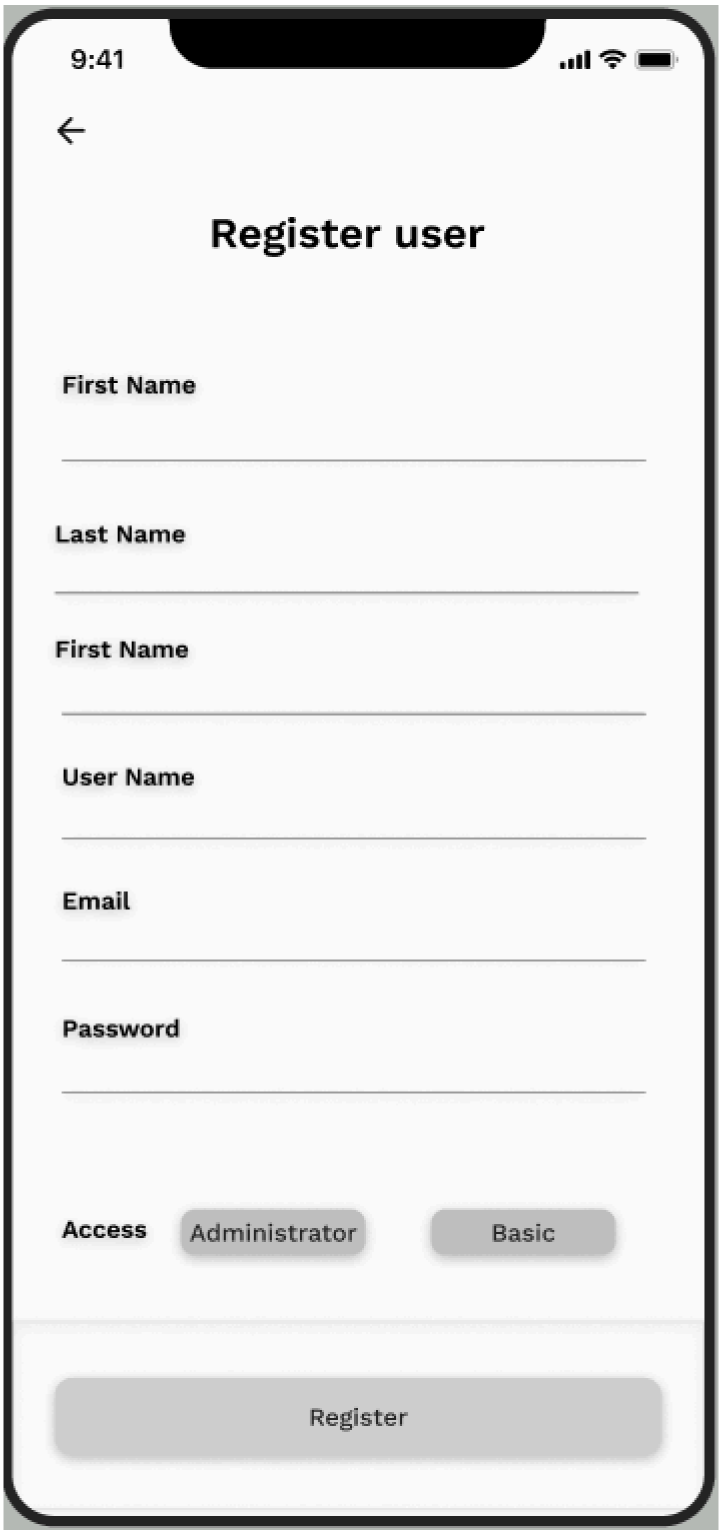
6.2.1.6 Add New Room Mockup



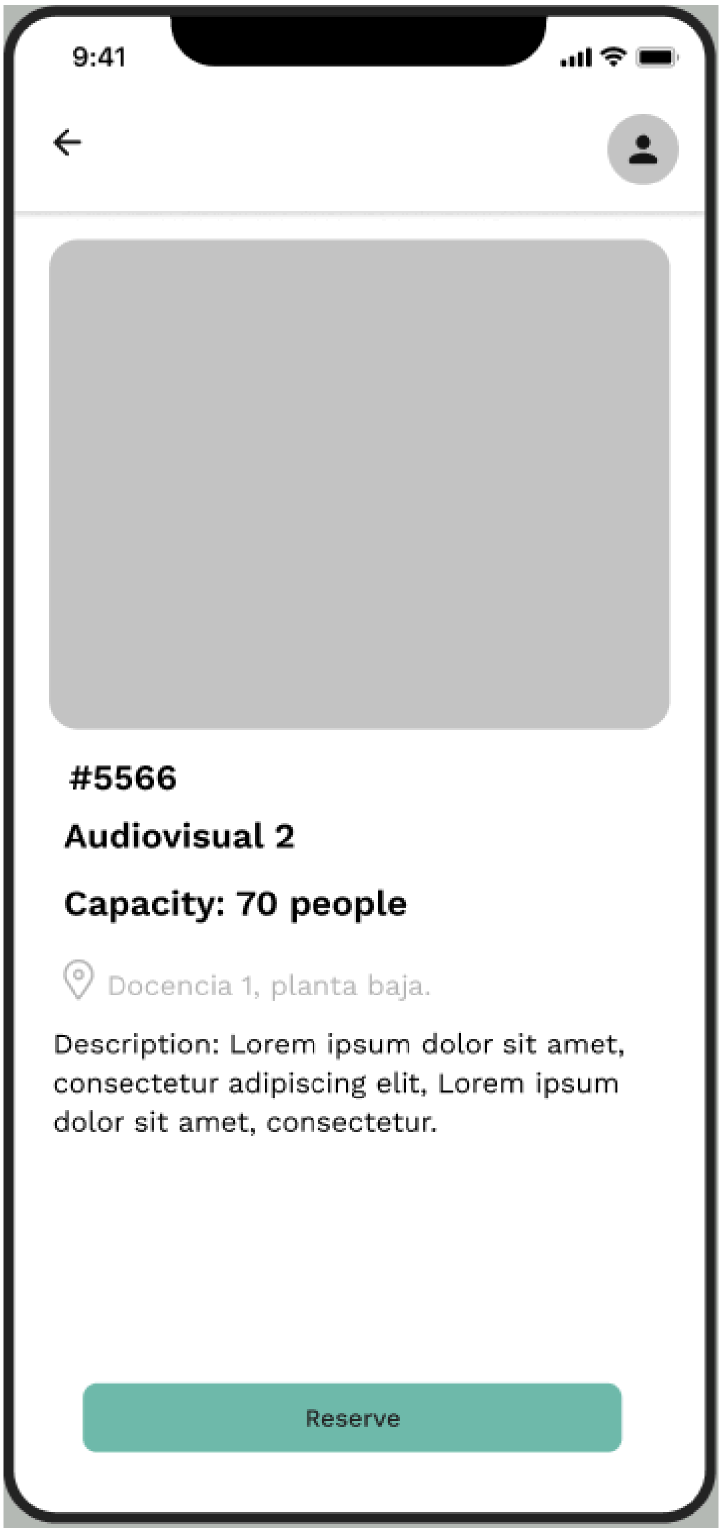
6.2.1.7 Add New Equipment Mockup



6.2.1.8 Add New User Mockup



6.2.1.9 Room Reservation Mockup



9:41

#5566

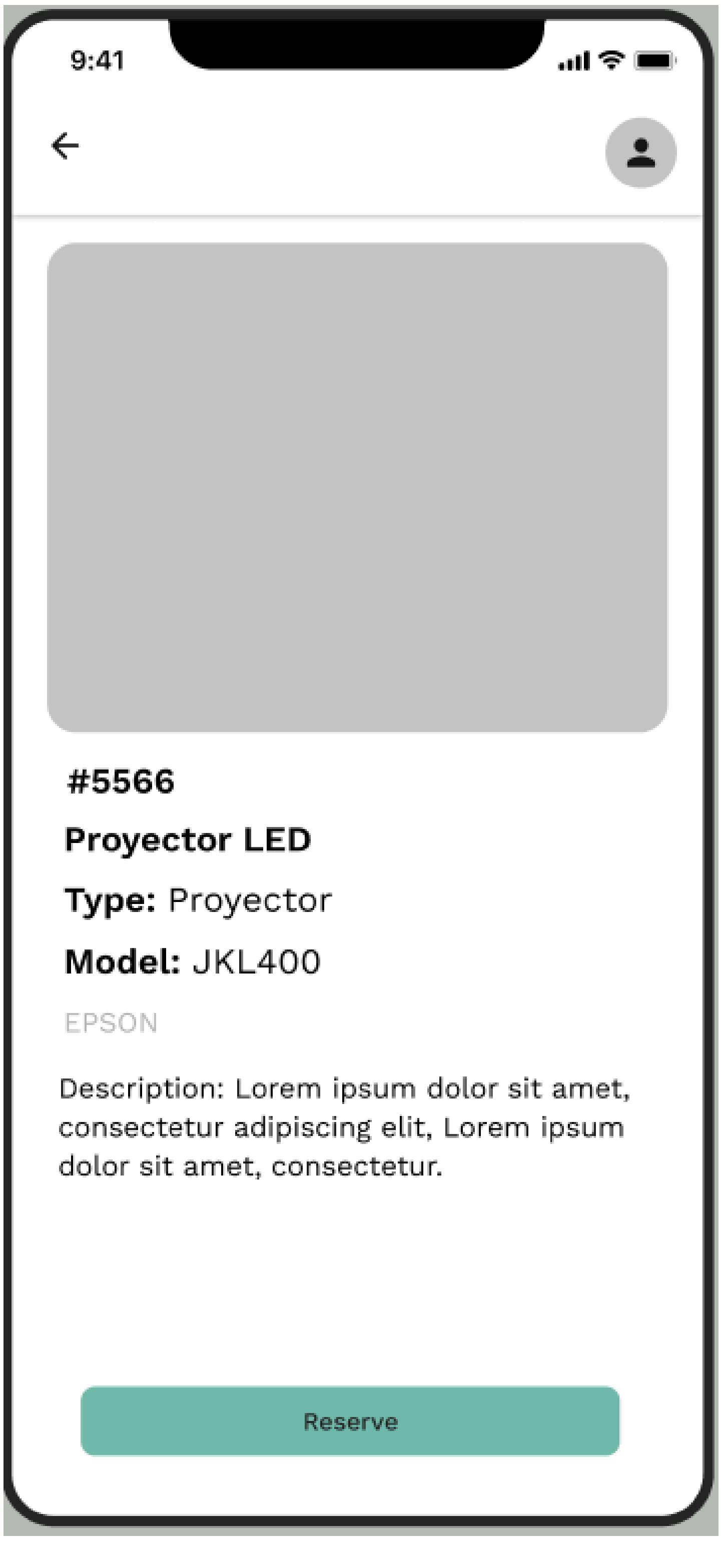
Audiovisual 2

Capacity: 70 people

Docencia 1, planta baja.

Description: Lorem ipsum dolor sit amet, consectetur adipiscing elit, Lorem ipsum dolor sit amet, consectetur.

6.2.1.10 Equipment Reservation Mockup



#5566

Proyector LED

Type: Proyector

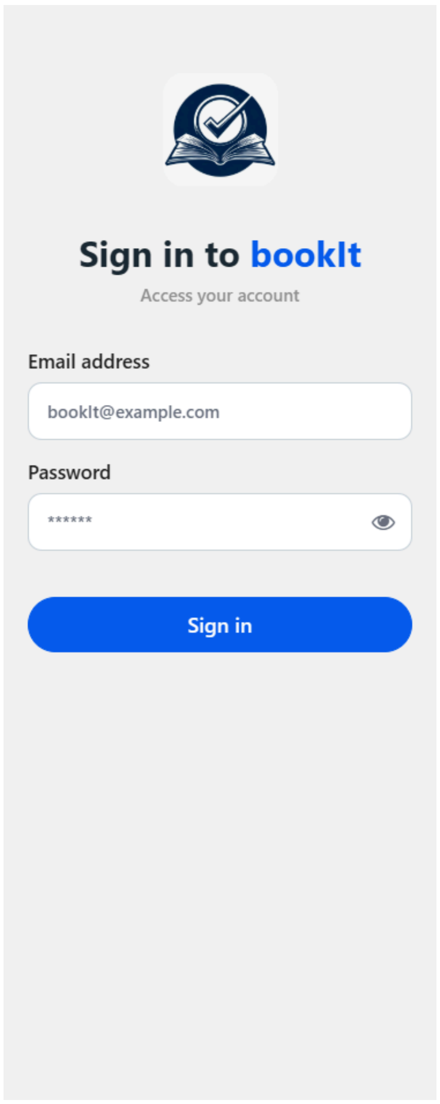
Model: JKL400

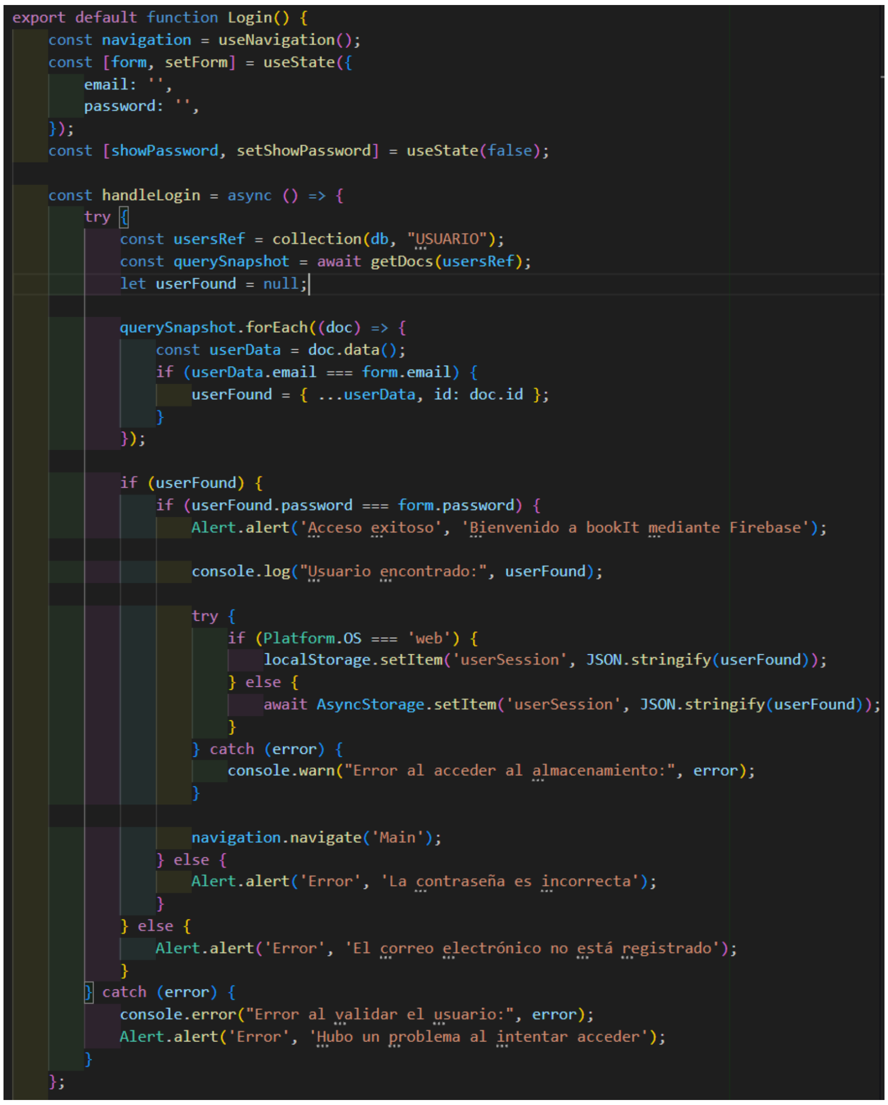
EPSON

Description: Lorem ipsum dolor sit amet, consectetur adipiscing elit, Lorem ipsum dolor sit amet, consectetur.

6.2.2 Screens

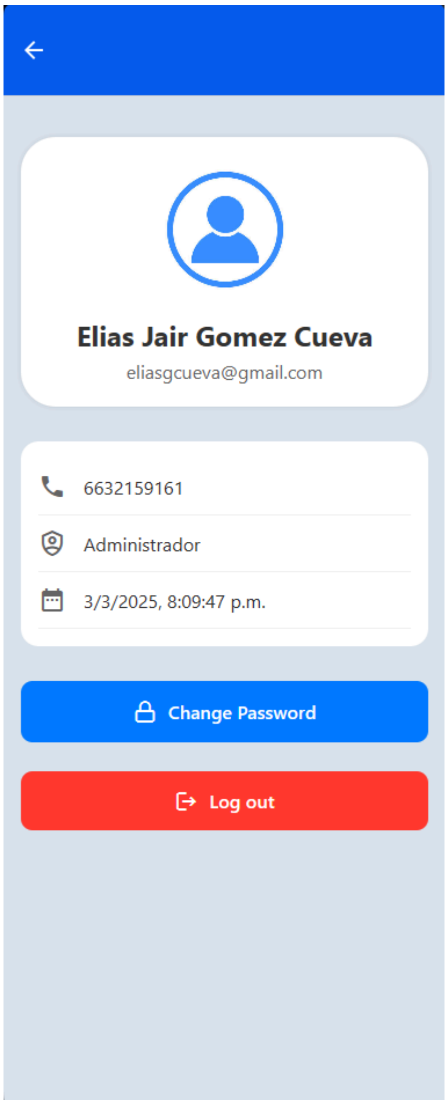
6.2.2.1 Login Screen





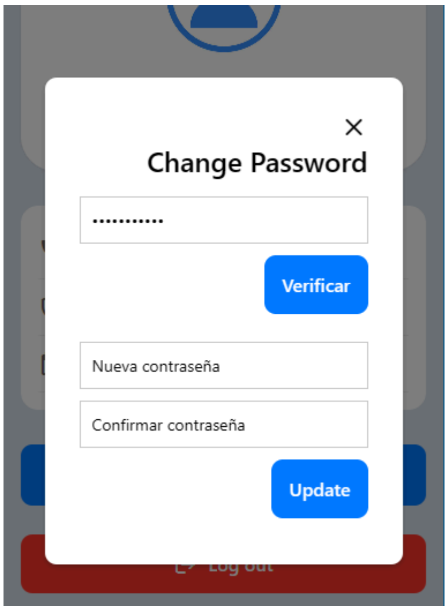
```
export default function Login() {
    const navigation = useNavigation();
    const [form, setForm] = useState({
        email: '',
        password: '',
    });
    const [showPassword, setShowPassword] = useState(false);

    const handleLogin = async () => {
        try {
            const usersRef = collection(db, "USUARIO");
            const querySnapshot = await getDocs(usersRef);
            let userFound = null;

            querySnapshot.forEach((doc) => {
                const userData = doc.data();
                if (userData.email === form.email) {
                    userFound = { ...userData, id: doc.id };
                }
            });

            if (userFound) {
                if (userFound.password === form.password) {
                    Alert.alert('Acceso exitoso', 'Bienvenido a bookIt mediante Firebase');

                    console.log("Usuario encontrado:", userFound);

                    try {
                        if (Platform.OS === 'web') {
                            localStorage.setItem('userSession', JSON.stringify(userFound));
                        } else {
                            await AsyncStorage.setItem('userSession', JSON.stringify(userFound));
                        }
                    } catch (error) {
                        console.warn("Error al acceder al almacenamiento:", error);
                    }

                    navigation.navigate('Main');
                } else {
                    Alert.alert('Error', 'La contraseña es incorrecta');
                }
            } else {
                Alert.alert('Error', 'El correo electrónico no está registrado');
            }
        } catch (error) {
            console.error("Error al validar el usuario:", error);
            Alert.alert('Error', 'Hubo un problema al intentar acceder');
        }
    };
```

6.2.2.2 Profile Screen

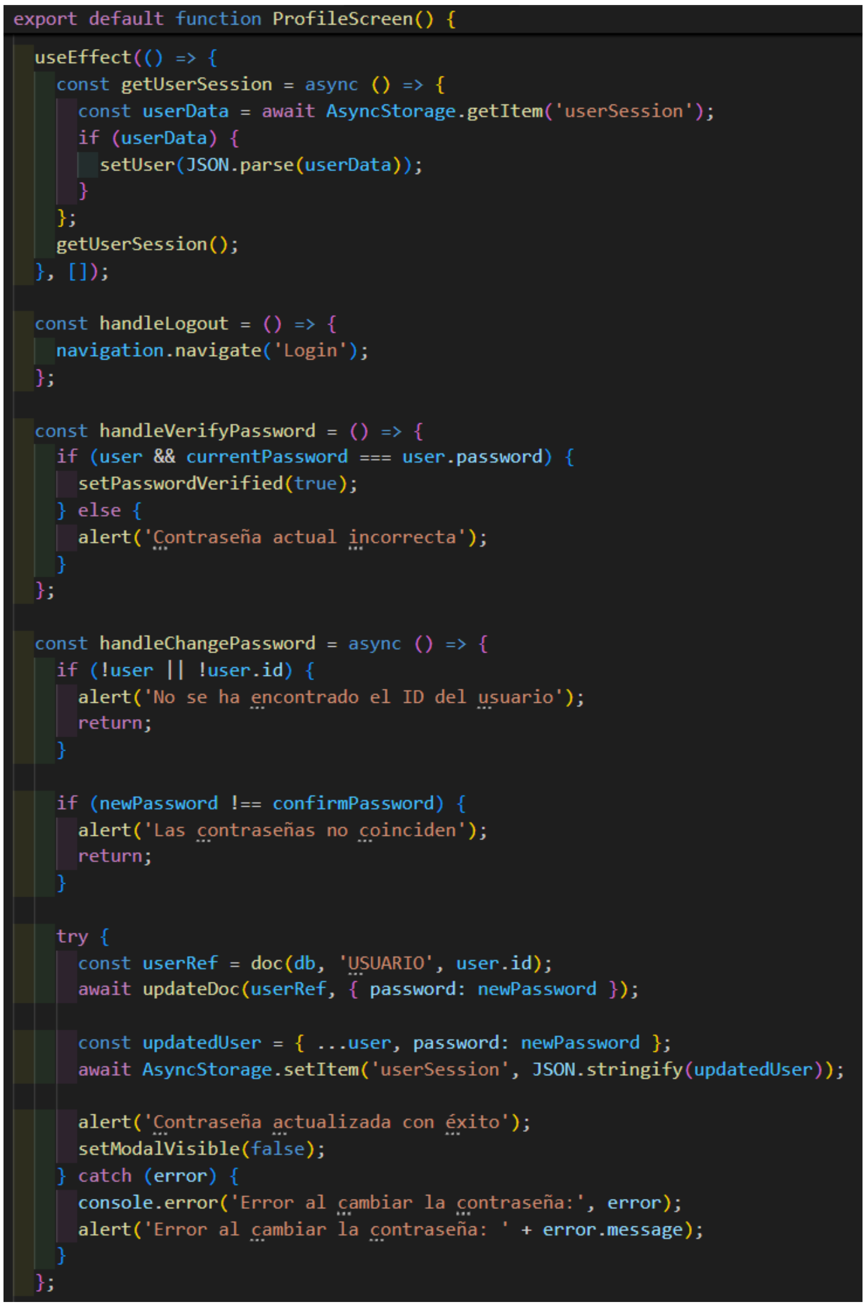


Elias Jair Gomez Cueva
eliasgcueva@gmail.com

6632159161
Administrador
3/3/2025, 8:09:47 p.m.

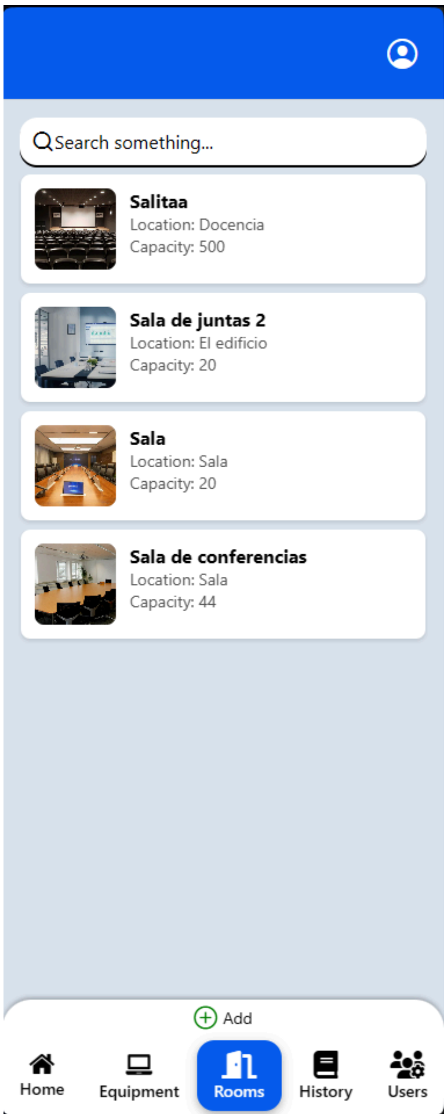


Change Password



```
export default function ProfileScreen() {

  useEffect(() => {
    const getUserSession = async () => {
      const userData = await AsyncStorage.getItem('userSession');
      if (userData) {
        setUser(JSON.parse(userData));
      }
    };
    getUserSession();
  }, []);

  const handleLogout = () => {
    navigation.navigate('Login');
  };

  const handleVerifyPassword = () => {
    if (user && currentPassword === user.password) {
      setPasswordVerified(true);
    } else {
      alert('Contraseña actual incorrecta');
    }
  };

  const handleChangePassword = async () => {
    if (!user || !user.id) {
      alert('No se ha encontrado el ID del usuario');
      return;
    }

    if (newPassword !== confirmPassword) {
      alert('Las contraseñas no coinciden');
      return;
    }

    try {
      const userRef = doc(db, 'USUARIO', user.id);
      await updateDoc(userRef, { password: newPassword });

      const updatedUser = { ...user, password: newPassword };
      await AsyncStorage.setItem('userSession', JSON.stringify(updatedUser));

      alert('Contraseña actualizada con éxito');
      setModalVisible(false);
    } catch (error) {
      console.error('Error al cambiar la contraseña:', error);
      alert('Error al cambiar la contraseña: ' + error.message);
    }
  };
```

6.2.2.3 Room Screen

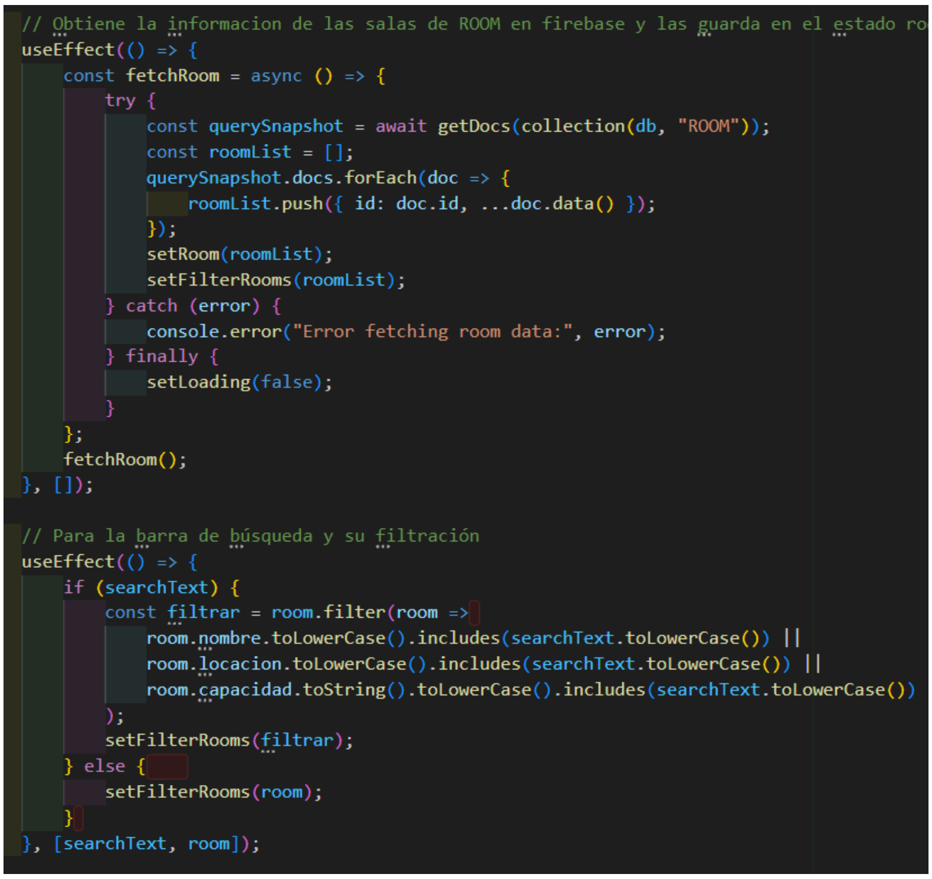


Salitaa
Location: Docencia
Capacity: 500

Sala de juntas 2
Location: El edificio
Capacity: 20

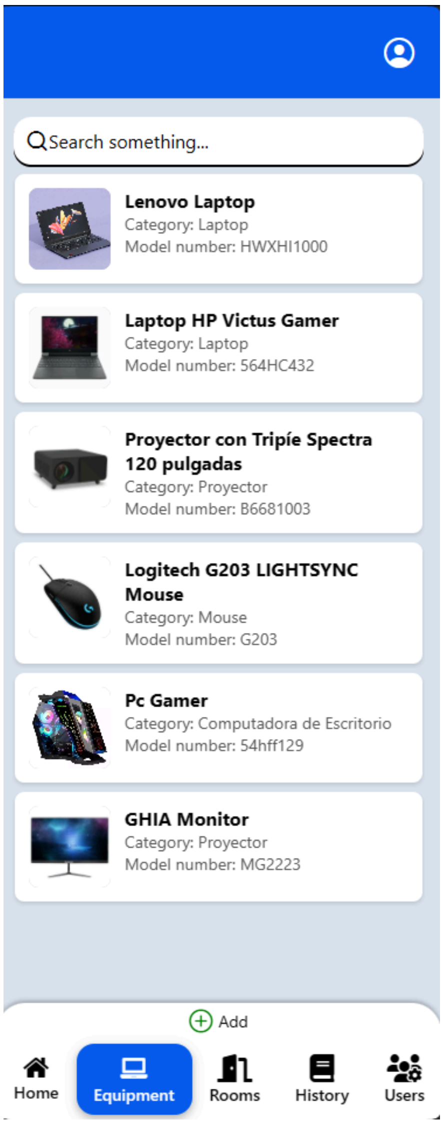
Sala
Location: Sala
Capacity: 20

Sala de conferencias
Location: Sala
Capacity: 44



```
// Obtiene la informacion de las salas de ROOM en firebase y las guarda en el estado ro
useEffect(() => {
    const fetchRoom = async () => {
        try {
            const querySnapshot = await getDocs(collection(db, "ROOM"));
            const roomList = [];
            querySnapshot.docs.forEach(doc => {
                roomList.push({ id: doc.id, ...doc.data() });
            });
            setRoom(roomList);
            setFilterRooms(roomList);
        } catch (error) {
            console.error("Error fetching room data:", error);
        } finally {
            setLoading(false);
        }
    };
    fetchRoom();
}, []);

// Para la barra de búsqueda y su filtración
useEffect(() => {
    if (searchText) {
        const filtrar = room.filter(room =>
            room.nombre.toLowerCase().includes(searchText.toLowerCase()) ||
            room.locacion.toLowerCase().includes(searchText.toLowerCase()) ||
            room.capacidad.toString().toLowerCase().includes(searchText.toLowerCase())
        );
        setFilterRooms(filtrar);
    } else {
        setFilterRooms(room);
    }
}, [searchText, room]);
```

6.2.2.4 Equipment Screen



Search something...

Lenovo Laptop
Category: Laptop
Model number: HWXHI1000

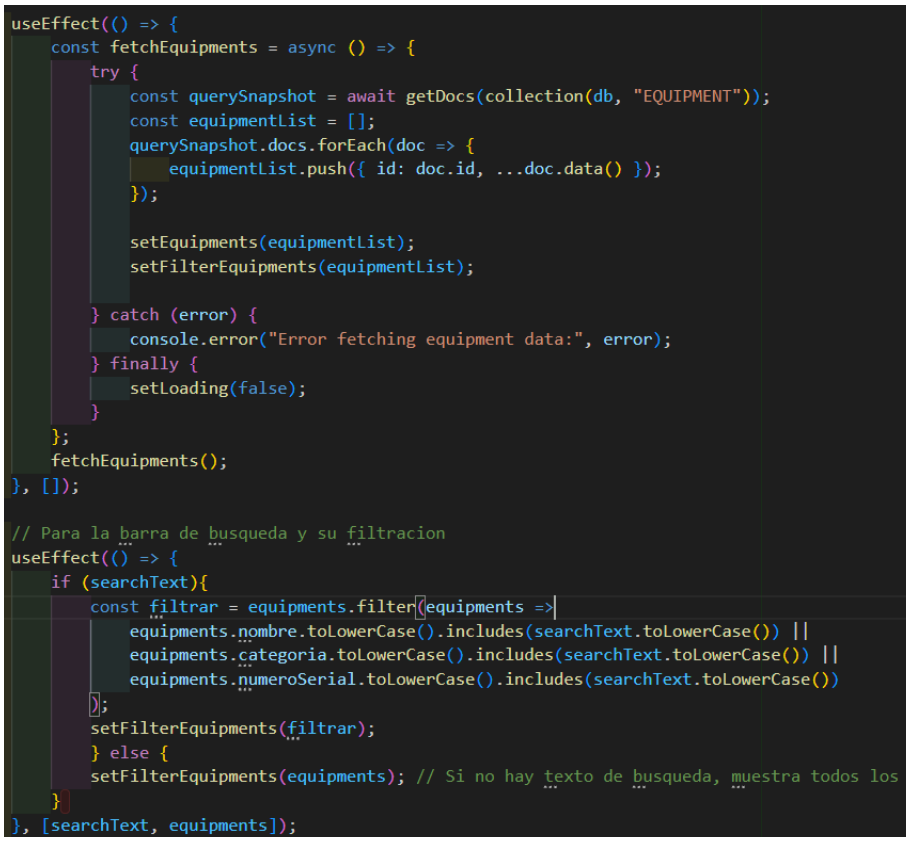
Laptop HP Victus Gamer
Category: Laptop
Model number: 564HC432

Proyector con Tripíe Spectra 120 pulgadas
Category: Proyector
Model number: B6681003

Logitech G203 LIGHTSYNC Mouse
Category: Mouse
Model number: G203

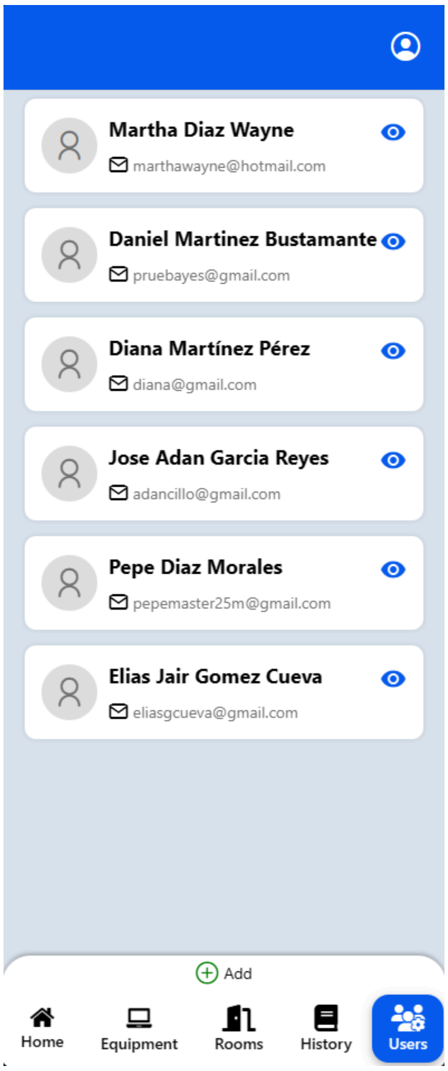
Pc Gamer
Category: Computadora de Escritorio
Model number: 54hff129

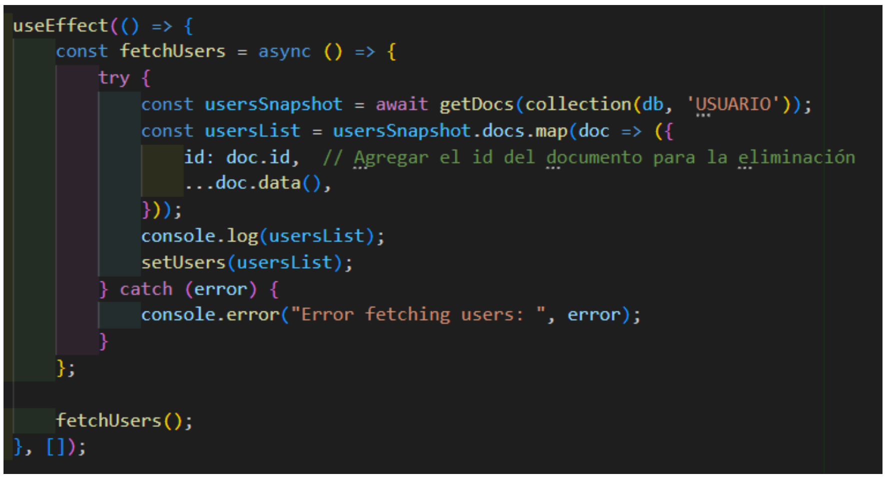
GHIA Monitor
Category: Proyector
Model number: MG2223



```
useEffect(() => {
    const fetchEquipments = async () => {
        try {
            const querySnapshot = await getDocs(collection(db, "EQUIPMENT"));
            const equipmentList = [];
            querySnapshot.docs.forEach(doc => {
                equipmentList.push({ id: doc.id, ...doc.data() });
            });

            setEquipments(equipmentList);
            setFilterEquipments(equipmentList);

        } catch (error) {
            console.error("Error fetching equipment data:", error);
        } finally {
            setLoading(false);
        }
    };
    fetchEquipments();
}, []);

// Para la barra de busqueda y su filtracion
useEffect(() => {
    if (searchText){
        const filtrar = equipments.filter(equipments =>
            equipments.nombre.toLowerCase().includes(searchText.toLowerCase()) ||
            equipments.categoria.toLowerCase().includes(searchText.toLowerCase()) ||
            equipments.numeroSerial.toLowerCase().includes(searchText.toLowerCase())
        );
        setFilterEquipments(filtrar);
        } else {
        setFilterEquipments(equipments); // Si no hay texto de busqueda, muestra todos los
    }
}, [searchText, equipments]);
```

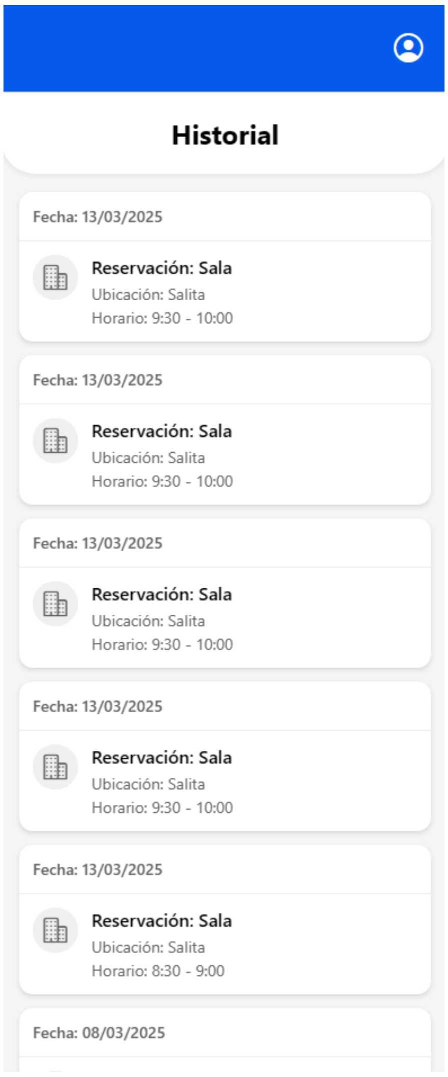
6.2.2.5 Users Screen





```
useEffect(() => {
    const fetchUsers = async () => {
        try {
            const usersSnapshot = await getDocs(collection(db, 'USUARIO'));
            const usersList = usersSnapshot.docs.map(doc => ({
                id: doc.id,  // Agregar el id del documento para la eliminación
                ...doc.data(),
            }));
            console.log(usersList);
            setUsers(usersList);
        } catch (error) {
            console.error("Error fetching users: ", error);
        }
    };

    fetchUsers();
}, []);
```

6.2.2.6 Administrator History Screen



Historial

Fecha: 13/03/2025

Reservación: Sala
Ubicación: Salita
Horario: 9:30 - 10:00

Fecha: 13/03/2025

Reservación: Sala
Ubicación: Salita
Horario: 9:30 - 10:00

Fecha: 13/03/2025

Reservación: Sala
Ubicación: Salita
Horario: 9:30 - 10:00

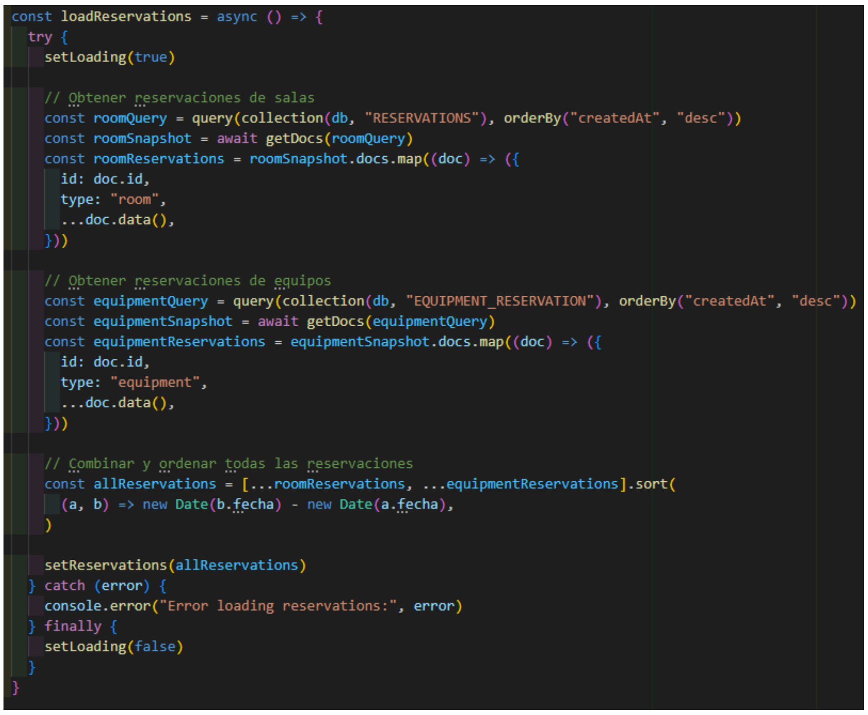
Fecha: 13/03/2025

Reservación: Sala
Ubicación: Salita
Horario: 9:30 - 10:00

Fecha: 13/03/2025

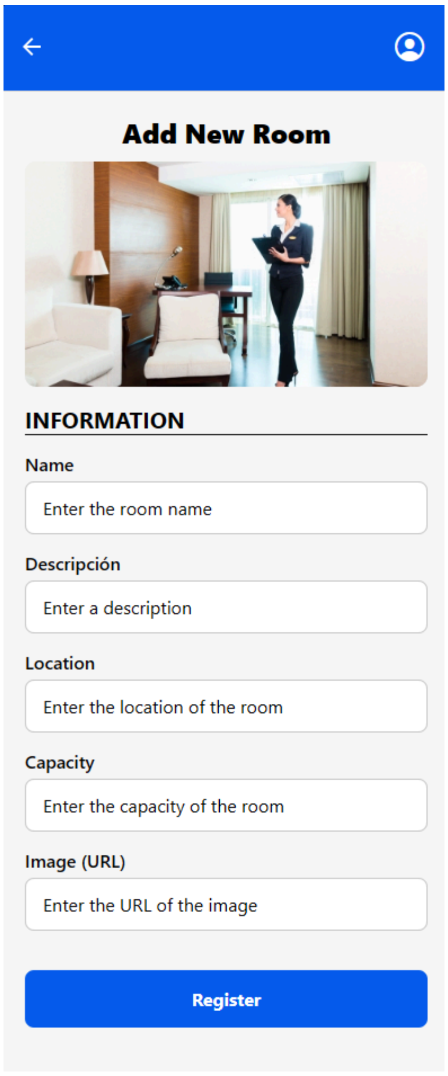
Reservación: Sala
Ubicación: Salita
Horario: 8:30 - 9:00

Fecha: 08/03/2025



```
const loadReservations = async () => {
  try {
    setLoading(true)

    // Obtener reservaciones de salas
    const roomQuery = query(collection(db, "RESERVATIONS"), orderBy("createdAt", "desc"))
    const roomSnapshot = await getDocs(roomQuery)
    const roomReservations = roomSnapshot.docs.map((doc) => ({
      id: doc.id,
      type: "room",
      ...doc.data(),
    }))

    // Obtener reservaciones de equipos
    const equipmentQuery = query(collection(db, "EQUIPMENT_RESERVATION"), orderBy("createdAt", "desc"))
    const equipmentSnapshot = await getDocs(equipmentQuery)
    const equipmentReservations = equipmentSnapshot.docs.map((doc) => ({
      id: doc.id,
      type: "equipment",
      ...doc.data(),
    }))

    // Combinar y ordenar todas las reservaciones
    const allReservations = [...roomReservations, ...equipmentReservations].sort(
      (a, b) => new Date(b.fecha) - new Date(a.fecha),
    )

    setReservations(allReservations)
  } catch (error) {
    console.error("Error loading reservations:", error)
  } finally {
    setLoading(false)
  }
}
```

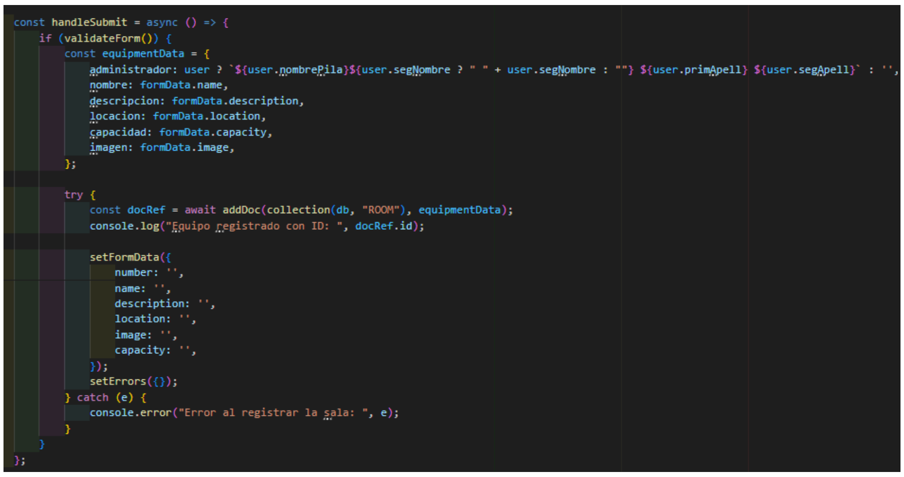
6.2.2.7 Add New Room Screen



Add New Room

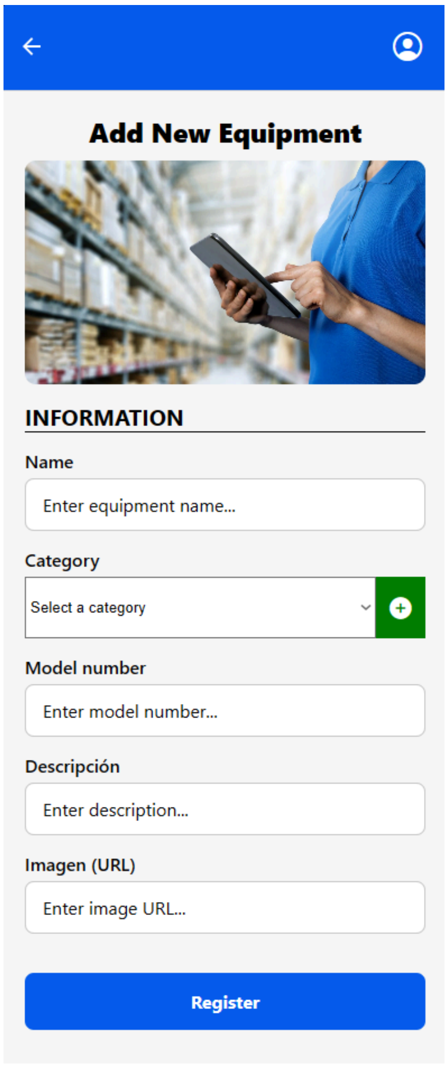
INFORMATION

Name	Enter the room name
Descripción	Enter a description
Location	Enter the location of the room
Capacity	Enter the capacity of the room
Image (URL)	Enter the URL of the image



```
const handleSubmit = async () => {
    if (validateForm()) {
        const equipmentData = {
            administrador: user ? `${user.nombrePila}${user.segNombre ? " " + user.segNombre : ""} ${user.primApell} ${user.segApell}` : '',
            nombre: formData.name,
            descripcion: formData.description,
            locacion: formData.location,
            capacidad: formData.capacity,
            imagen: formData.image,
        };

        try {
            const docRef = await addDoc(collection(db, "ROOM"), equipmentData);
            console.log("Equipo registrado con ID: ", docRef.id);

            setFormData({
                number: '',
                name: '',
                description: '',
                location: '',
                image: '',
                capacity: '',
            });
            setErrors({});
        } catch (e) {
            console.error("Error al registrar la sala: ", e);
        }
    }
};
```

6.2.2.8 Add New Equipment Screen



Add New Equipment

INFORMATION

Name

Enter equipment name...

Category

Select a category

Model number

Enter model number...

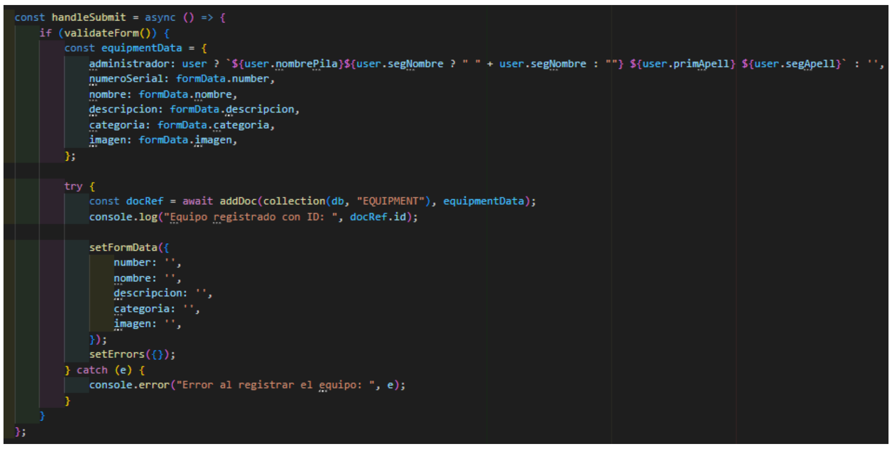
Descripción

Enter description...

Imagen (URL)

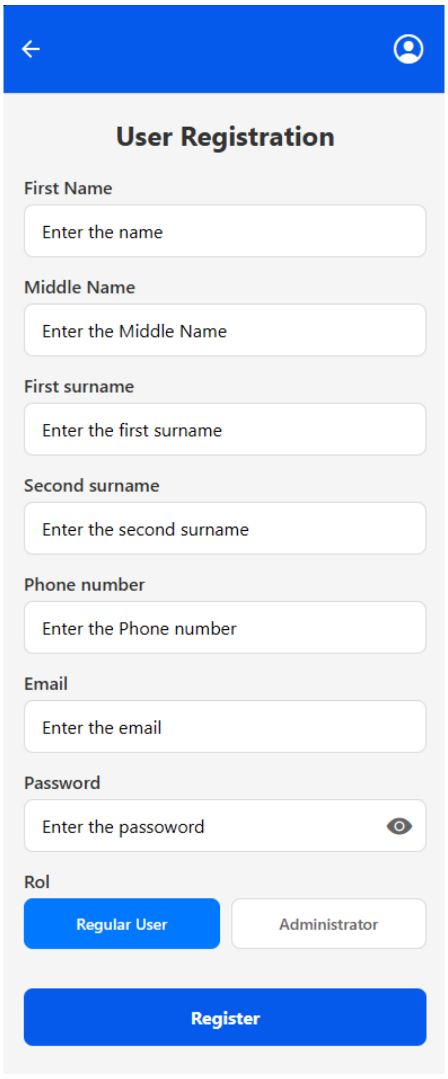
Enter image URL...

Register



```
const handleSubmit = async () => {
    if (validateForm()) {
        const equipmentData = {
            administrador: user ? `${user.nombrePila}${user.segNombre ? " " + user.segNombre : ""} ${user.primApell} ${user.segApell}` : '',
            numeroSerial: formData.number,
            nombre: formData.nombre,
            descripcion: formData.descripcion,
            categoria: formData.categoria,
            imagen: formData.imagen,
        };

        try {
            const docRef = await addDoc(collection(db, "EQUIPMENT"), equipmentData);
            console.log("Equipo registrado con ID: ", docRef.id);

            setFormData({
                number: '',
                nombre: '',
                descripcion: '',
                categoria: '',
                imagen: '',
            });
            setErrors({});
        } catch (e) {
            console.error("Error al registrar el equipo: ", e);
        }
    }
};
```

6.2.2.8 Add New User Screen



User Registration

First Name
Enter the name

Middle Name
Enter the Middle Name

First surname
Enter the first surname

Second surname
Enter the second surname

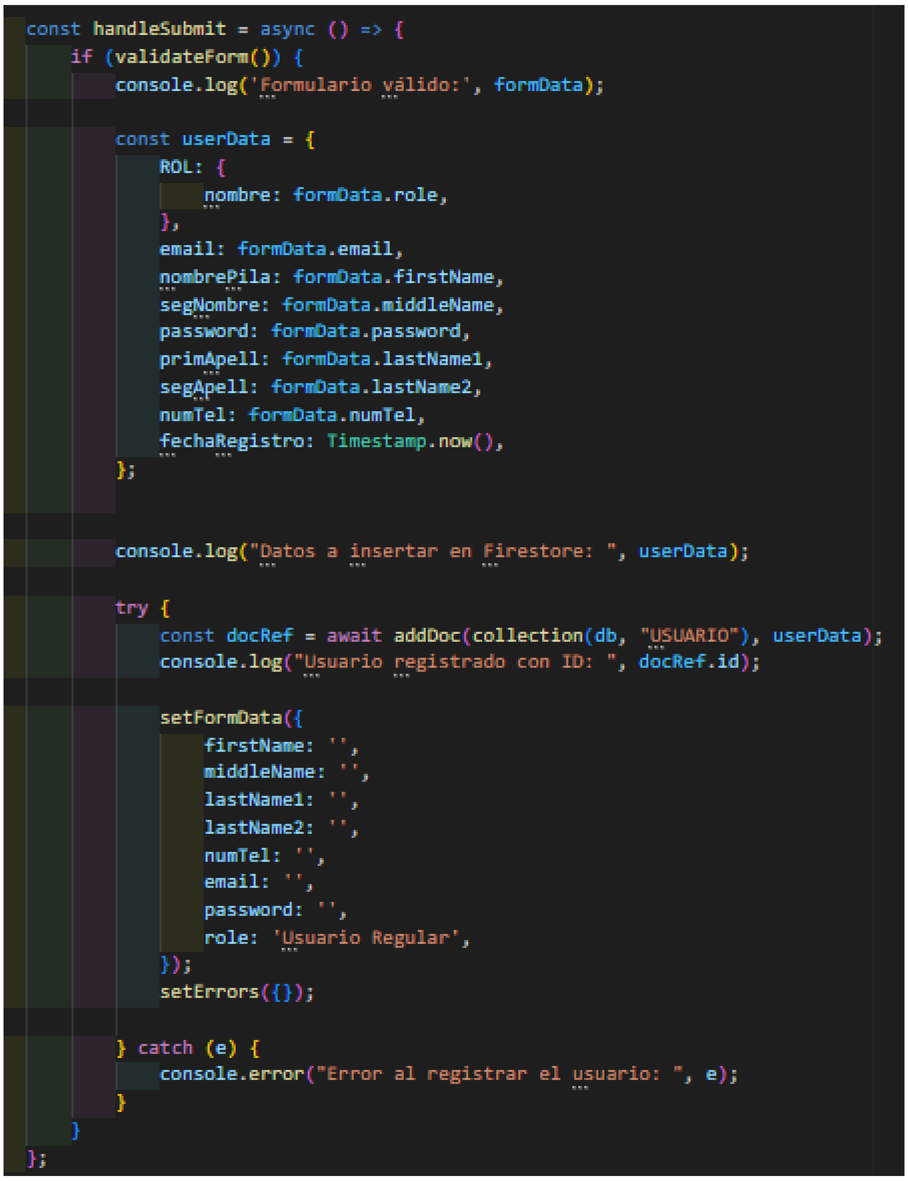
Phone number
Enter the Phone number

Email
Enter the email

Password
Enter the passoword

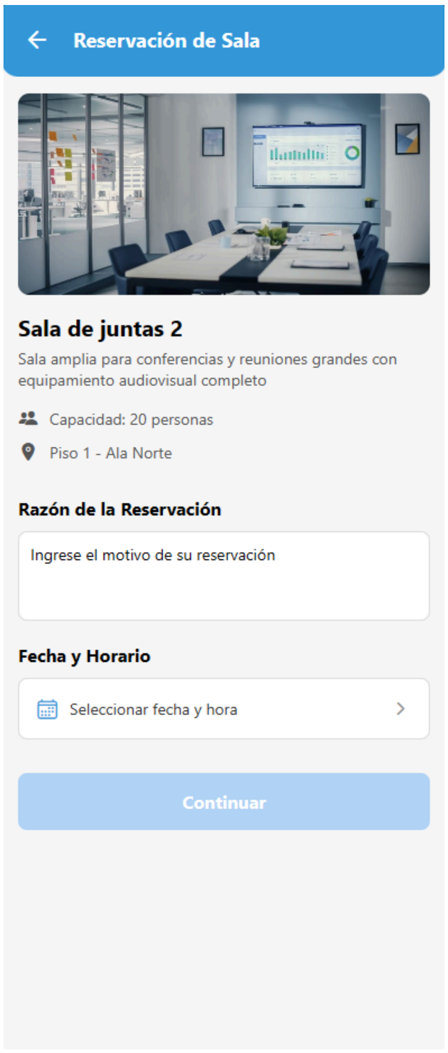
Rol
Regular User | Administrator

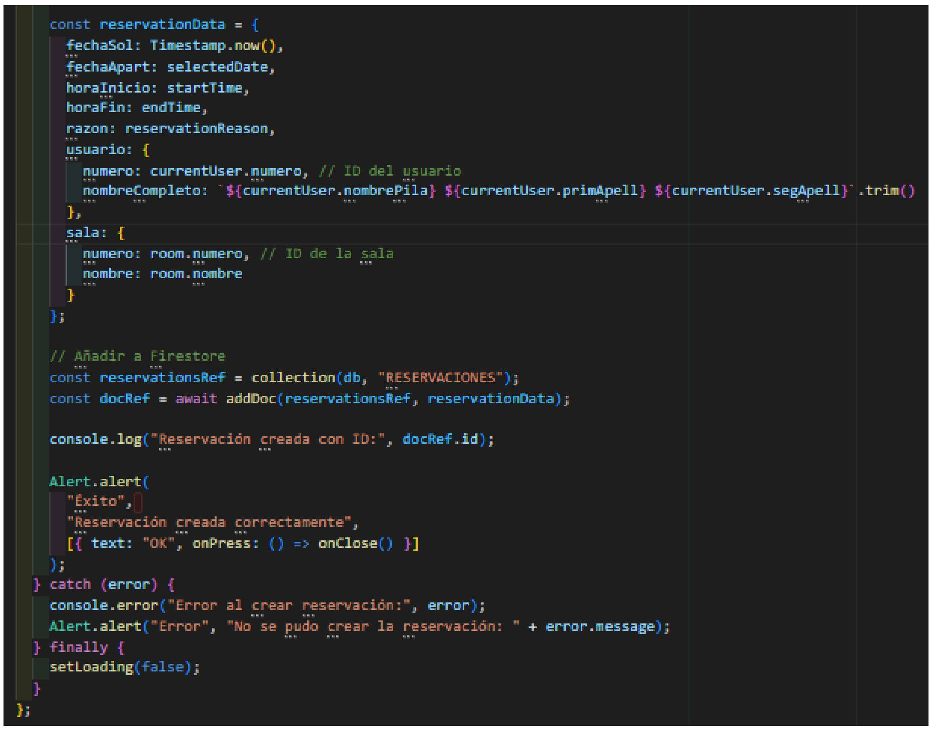
Register



```
const handleSubmit = async () => {
  if (validateForm()) {
    console.log('Formulario válido:', formData);

    const userData = {
      ROL: {
        nombre: formData.role,
      },
      email: formData.email,
      nombrePila: formData.firstName,
      segNombre: formData.middleName,
      password: formData.password,
      primApell: formData.lastName1,
      segApell: formData.lastName2,
      numTel: formData.numTel,
      fechaRegistro: Timestamp.now(),
    };

    console.log("Datos a insertar en Firestore: ", userData);

    try {
      const docRef = await addDoc(collection(db, "USUARIO"), userData);
      console.log("Usuario registrado con ID: ", docRef.id);

      setFormData({
        firstName: '',
        middleName: '',
        lastName1: '',
        lastName2: '',
        numTel: '',
        email: '',
        password: '',
        role: 'Usuario Regular',
      });
      setErrors({});

    } catch (e) {
      console.error("Error al registrar el usuario: ", e);
    }
  }
};
```

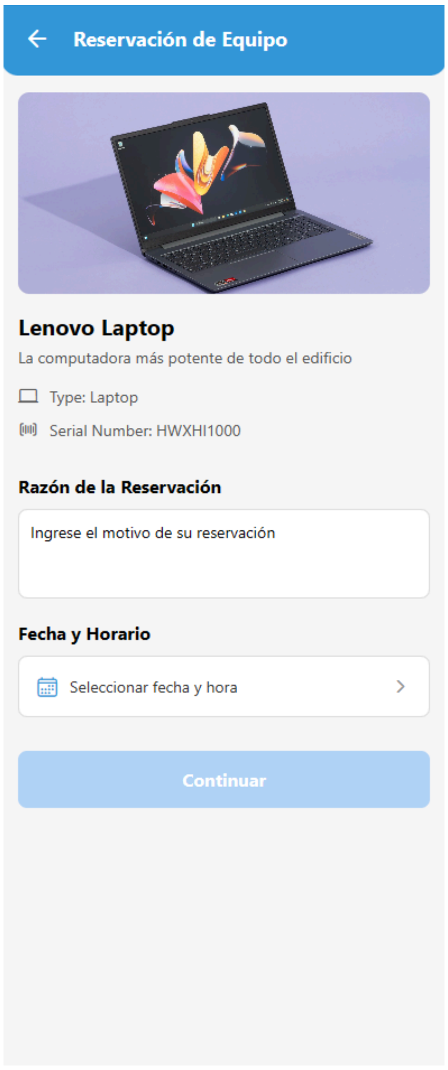
6.2.2.9 Room Reservation Screen





```
    const reservationData = {
      fechaSol: Timestamp.now(),
      fechaApart: selectedDate,
      horaInicio: startTime,
      horaFin: endTime,
      razon: reservationReason,
      usuario: {
        numero: currentUser.numero, // ID del usuario
        nombreCompleto: `${currentUser.nombrePila} ${currentUser.primApell} ${currentUser.segApell}`.trim()
      },
      sala: {
        numero: room.numero, // ID de la sala
        nombre: room.nombre
      }
    };

    // Añadir a Firestore
    const reservationsRef = collection(db, "RESERVACIONES");
    const docRef = await addDoc(reservationsRef, reservationData);

    console.log("Reservación creada con ID:", docRef.id);

    Alert.alert(
      "Éxito",
      "Reservación creada correctamente",
      [{ text: "OK", onPress: () => onClose() }]
    );
  } catch (error) {
    console.error("Error al crear reservación:", error);
    Alert.alert("Error", "No se pudo crear la reservación: " + error.message);
  } finally {
    setLoading(false);
  }
};
```

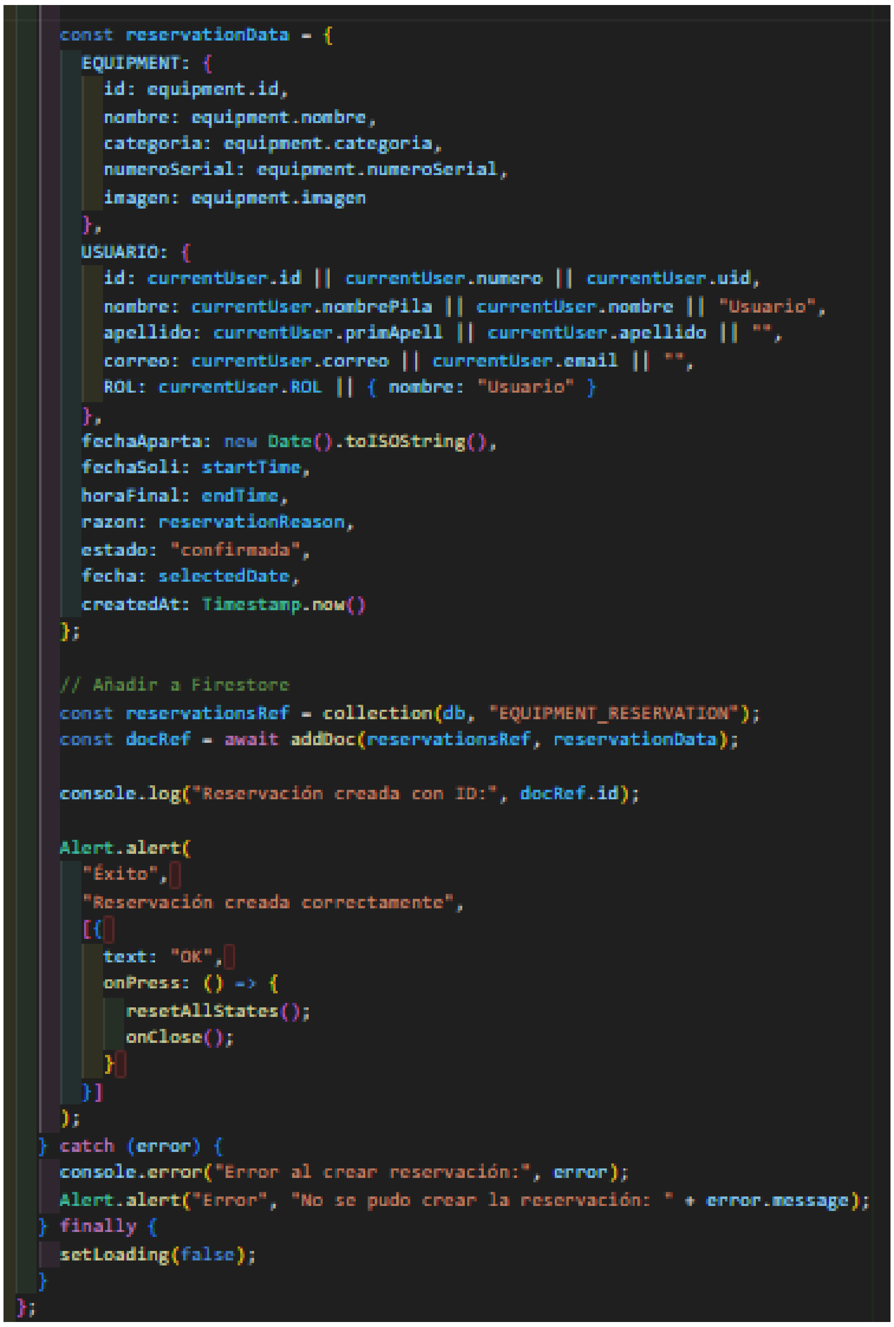
6.2.2.10 Equipment Reservation Screen



Lenovo Laptop
La computadora más potente de todo el edificio
Type: Laptop
Serial Number: HWXHI1000

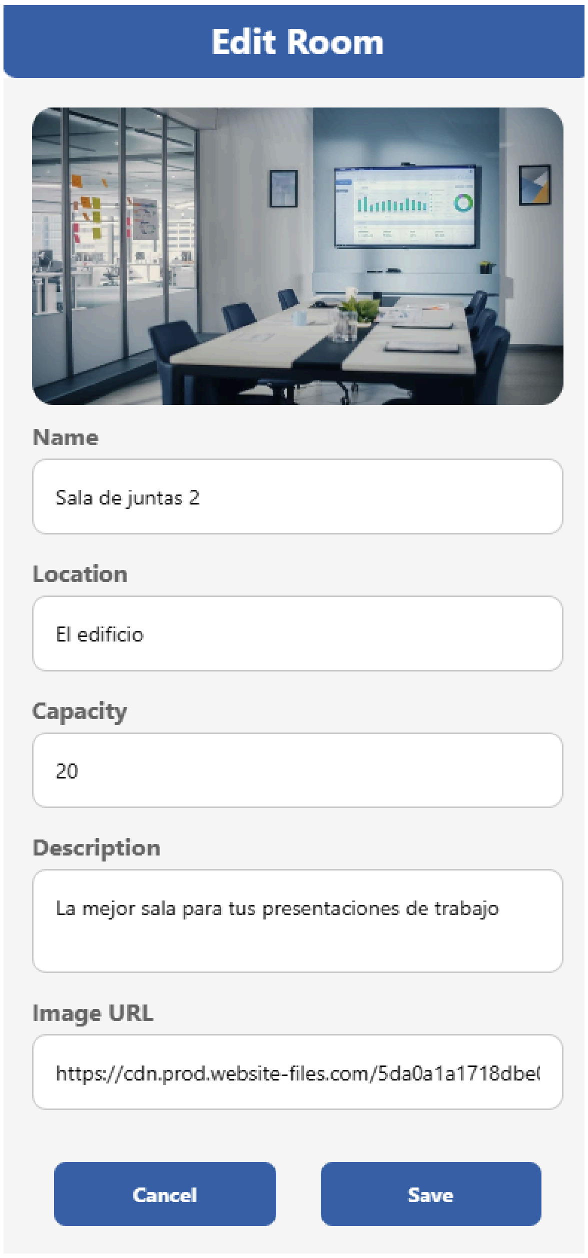
Razón de la Reservación
Ingrese el motivo de su reservación

Fecha y Horario
Seleccionar fecha y hora



```
    const reservationData = {
      EQUIPMENT: {
        id: equipment.id,
        nombre: equipment.nombre,
        categoria: equipment.categoria,
        numeroSerial: equipment.numeroSerial,
        imagen: equipment.imagen
      },
      USUARIO: {
        id: currentUser.id || currentUser.numero || currentUser.uid,
        nombre: currentUser.nombrePila || currentUser.nombre || "Usuario",
        apellido: currentUser.primApell || currentUser.apellido || "",
        correo: currentUser.correo || currentUser.email || "",
        ROL: currentUser.ROL || { nombre: "Usuario" }
      },
      fechaAparta: new Date().toISOString(),
      fechaSoli: startTime,
      horaFinal: endTime,
      razon: reservationReason,
      estado: "confirmada",
      fecha: selectedDate,
      createdAt: Timestamp.now()
    };

    // Añadir a Firestore
    const reservationsRef = collection(db, "EQUIPMENT_RESERVATION");
    const docRef = await addDoc(reservationsRef, reservationData);

    console.log("Reservación creada con ID:", docRef.id);

    Alert.alert(
      "Éxito",
      "Reservación creada correctamente",
      [{
        text: "OK",
        onPress: () => {
          resetAllStates();
          onClose();
        }
      }]
    );
  } catch (error) {
    console.error("Error al crear reservación:", error);
    Alert.alert("Error", "No se pudo crear la reservación: " + error.message);
  } finally {
    setLoading(false);
  }
};
```

6.2.2.11 Room Edit Screen


Edit Room

Name

Sala de juntas 2

Location

El edificio

Capacity

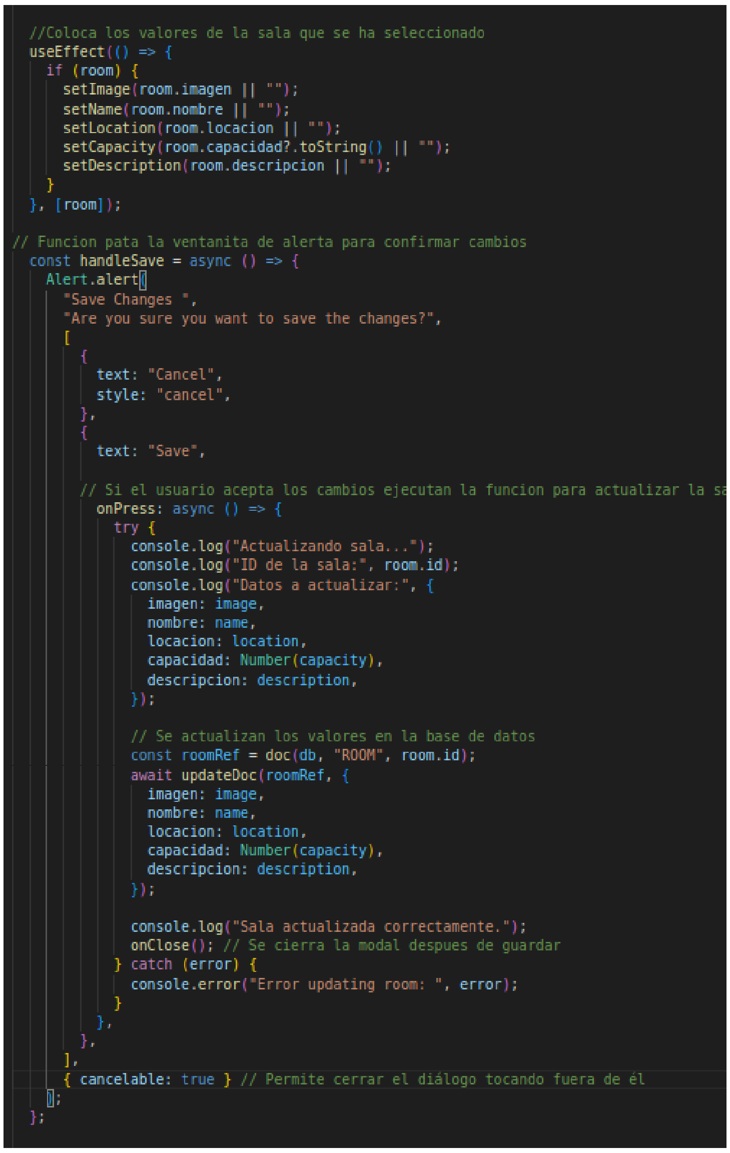
20

Description

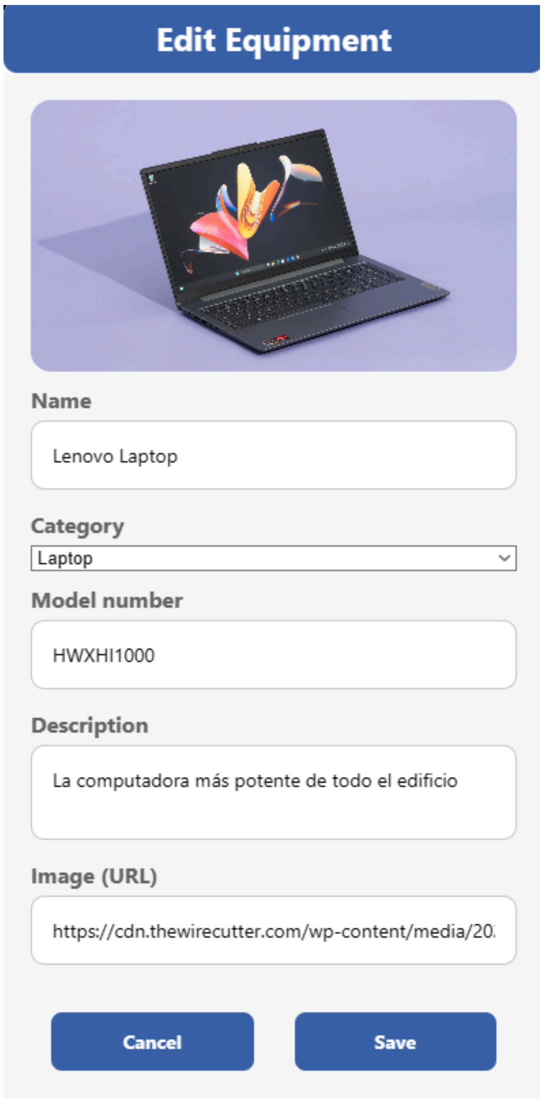
La mejor sala para tus presentaciones de trabajo

Image URL

https://cdn.prod.website-files.com/5da0a1a1718dbe(


```
  //Coloca los valores de la sala que se ha seleccionado
  useEffect(() => {
    if (room) {
      setImage(room.imagen || "");
      setName(room.nombre || "");
      setLocation(room.locacion || "");
      setCapacity(room.capacidad?.toString() || "");
      setDescription(room.descripcion || "");
    }
  }, [room]);

// Funcion pata la ventanita de alerta para confirmar cambios
  const handleSave = async () => {
    Alert.alert(
      "Save Changes ",
      "Are you sure you want to save the changes?",
      [
        {
          text: "Cancel",
          style: "cancel",
        },
        {
          text: "Save",

        // Si el usuario acepta los cambios ejecutan la funcion para actualizar la sa
          onPress: async () => {
            try {
              console.log("Actualizando sala...");
              console.log("ID de la sala:", room.id);
              console.log("Datos a actualizar:", {
                imagen: image,
                nombre: name,
                locacion: location,
                capacidad: Number(capacity),
                descripcion: description,
              });

              // Se actualizan los valores en la base de datos
              const roomRef = doc(db, "ROOM", room.id);
              await updateDoc(roomRef, {
                imagen: image,
                nombre: name,
                locacion: location,
                capacidad: Number(capacity),
                descripcion: description,
              });

              console.log("Sala actualizada correctamente.");
              onClose(); // Se cierra la modal despues de guardar
            } catch (error) {
              console.error("Error updating room: ", error);
            }
          },
        },
      ],
      { cancelable: true } // Permite cerrar el diálogo tocando fuera de él
    );
  };
```

6.2.2.12 Equipment Edit Screen



Edit Equipment

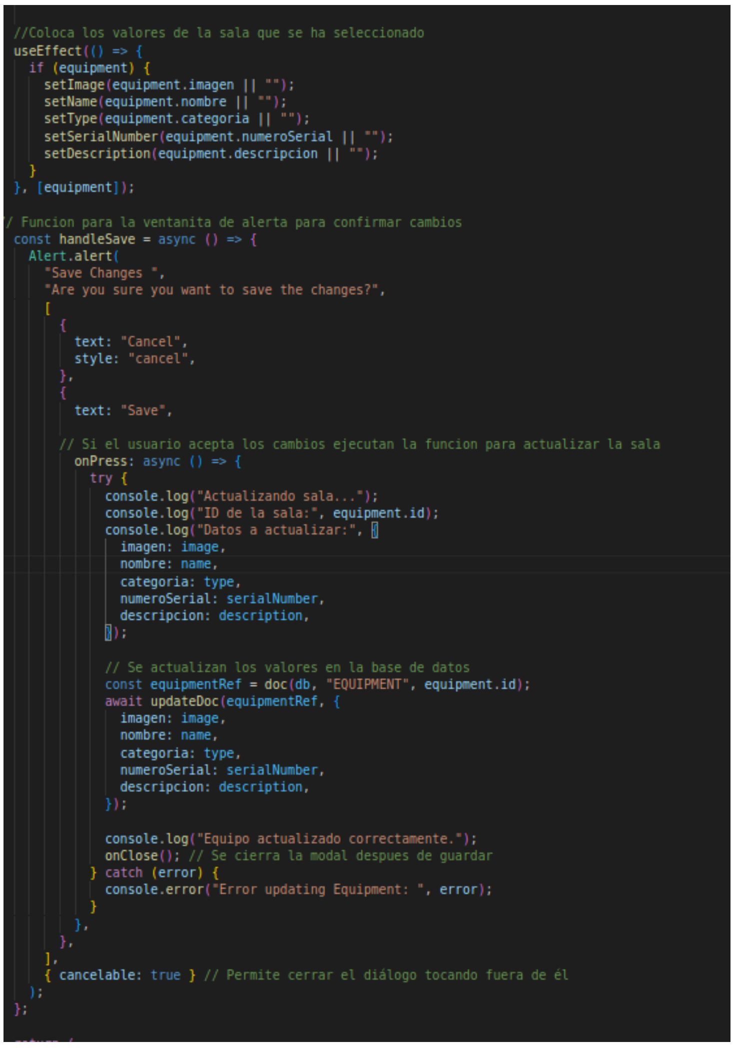
Name: Lenovo Laptop

Category: Laptop

Model number: HWXHI1000

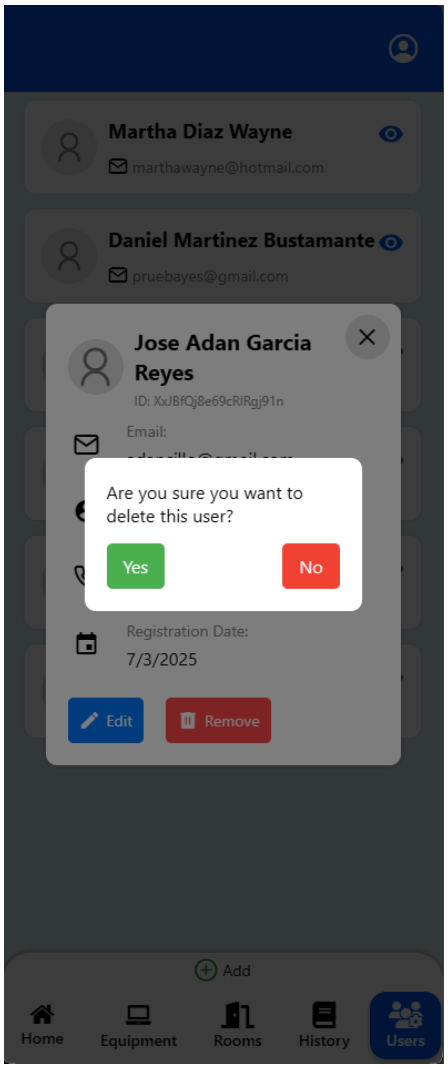
Description: La computadora más potente de todo el edificio

Image (URL): https://cdn.thewirecutter.com/wp-content/media/20



```
//Coloca los valores de la sala que se ha seleccionado
useEffect(() => {
  if (equipment) {
    setImage(equipment.imagen || "");
    setName(equipment.nombre || "");
    setType(equipment.categoria || "");
    setSerialNumber(equipment.numeroSerial || "");
    setDescription(equipment.descripcion || "");
  }
}, [equipment]);

/ Funcion para la ventanita de alerta para confirmar cambios
const handleSave = async () => {
  Alert.alert(
    "Save Changes ",
    "Are you sure you want to save the changes?",
    [
      {
        text: "Cancel",
        style: "cancel",
      },
      {
        text: "Save",

      // Si el usuario acepta los cambios ejecutan la funcion para actualizar la sala
        onPress: async () => {
          try {
            console.log("Actualizando sala...");
            console.log("ID de la sala:", equipment.id);
            console.log("Datos a actualizar:", {
              imagen: image,
              nombre: name,
              categoria: type,
              numeroSerial: serialNumber,
              descripcion: description,
            });

            // Se actualizan los valores en la base de datos
            const equipmentRef = doc(db, "EQUIPMENT", equipment.id);
            await updateDoc(equipmentRef, {
              imagen: image,
              nombre: name,
              categoria: type,
              numeroSerial: serialNumber,
              descripcion: description,
            });

            console.log("Equipo actualizado correctamente.");
            onClose(); // Se cierra la modal despues de guardar
          } catch (error) {
            console.error("Error updating Equipment: ", error);
          }
        },
      },
    ],
    { cancelable: true } // Permite cerrar el diálogo tocando fuera de él
  );
};
```

6.2.2.13 User Delete Screen



Martha Diaz Wayne
marthawayne@hotmail.com

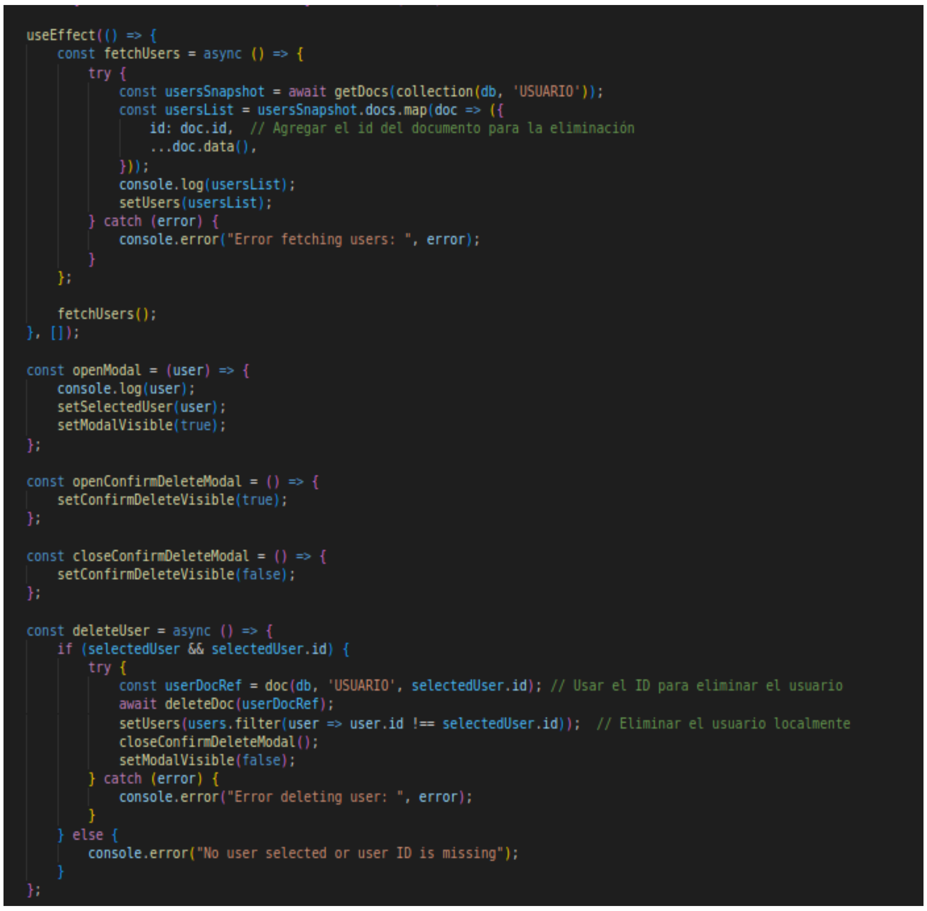
Daniel Martinez Bustamante
pruebayes@gmail.com

Jose Adan Garcia Reyes
ID: XxJBfQj8e69cRIRgj91n
Email:

Are you sure you want to delete this user?

Yes No

Registration Date:
7/3/2025



```
useEffect(() => {
    const fetchUsers = async () => {
        try {
            const usersSnapshot = await getDocs(collection(db, 'USUARIO'));
            const usersList = usersSnapshot.docs.map(doc => ({
                id: doc.id,  // Agregar el id del documento para la eliminación
                ...doc.data(),
            }));
            console.log(usersList);
            setUsers(usersList);
        } catch (error) {
            console.error("Error fetching users: ", error);
        }
    };

    fetchUsers();
}, []);

const openModal = (user) => {
    console.log(user);
    setSelectedUser(user);
    setModalVisible(true);
};

const openConfirmDeleteModal = () => {
    setConfirmDeleteVisible(true);
};

const closeConfirmDeleteModal = () => {
    setConfirmDeleteVisible(false);
};

const deleteUser = async () => {
    if (selectedUser && selectedUser.id) {
        try {
            const userDocRef = doc(db, 'USUARIO', selectedUser.id); // Usar el ID para eliminar el usuario
            await deleteDoc(userDocRef);
            setUsers(users.filter(user => user.id !== selectedUser.id));  // Eliminar el usuario localmente
            closeConfirmDeleteModal();
            setModalVisible(false);
        } catch (error) {
            console.error("Error deleting user: ", error);
        }
    } else {
        console.error("No user selected or user ID is missing");
    }
};
```