



# CENTRO DEPORTIVO GRUPO 4

Ingeniería Web

Aurora Andreu Mateo, Rubén del Castillo Fuentes, Hugo  
Huertas Blasco y Mario Davó Candela

## Tabla de contenido

Usuarios de la web .....	2
Funcionalidades .....	3
Tecnologías .....	6
Interoperación .....	7
Mockups .....	7
Diagrama de datos .....	15
Metodología .....	16
Implementación .....	17
Detalles del código .....	18
Problemas encontrados .....	21
Mejoras y futuras ampliaciones .....	22

Hemos desarrollado una web nueva para un centro deportivo donde puedan ofrecer información del mismo. Desde la web es posible que los visitantes que deseen ser socios tendrán enviar una solicitud desde la web pública y esperar a ser aceptados. En nuestra web, el formulario de registro es la solicitud de socio. Desde la web, es posible realizar reservas del centro deportivo.

Desde la cuenta de socio, se podrá reservar instalaciones, que son las diferentes pistas o salas disponibles en el centro deportivo. Los socios deben tener saldo en la cuenta del centro. La reserva le cargará un importe a su cuenta del club. El socio podrá cargar dinero en la misma desde un TPVV con su tarjeta. Si el socio no dispone de saldo, no podrá realizar la reserva.

Desde la cuenta de recepcionista, podrá realizar reservas de las instalaciones para socios o no socios del club, entre otras cosas.

El webmaster o administrador es el usuario que tiene acceso a toda la web y por ello tiene unas funcionalidades distintas.

## Usuarios de la web

Los usuarios que podrán acceder a la web, aunque con vistas diferentes son:

- No socios
- Socios
- Recepcionistas
- Webmaster
- Administrador

Aunque lo hemos detallado en las funcionalidades que para los usuarios existen los privilegios anteriores. Estos privilegios solo pueden ser modificados o usados por un webmaster. Esto quiere decir que un recepcionista tiene que crearlo un webmaster, no se puede crear desde la web. Cabe destacar que tanto estados como privilegios son 2 atributos de los usuarios. Para los usuarios de tipo socio existen diferentes estados. Los estados posibles son: pendiente, activo, bloqueado, de baja. Los socios en

estado pendiente son aquellos que han completado el formulario de registro y no han sido aceptados por un recepcionista o un webmaster. En cambio, aquellos socios que, si han sido aceptados, cambian de estado pendiente a activo. Un webmaster puede bloquear socios, por ello, el estado de un socio cambiaría a bloqueado. Los socios también se pueden dar de baja, y en ese caso su estado pasaría a de baja.

La diferencia entre webmaster y administrador en nuestra web es que el webmaster sería el cliente que ha encargado la web y el administrador seríamos nosotros.

## Funcionalidades

Las funcionalidades que hemos desarrollado son las siguientes:

### 001 Ver tipos de instalaciones

- Se visualizarán todas las instalaciones que hay en el club deportivo.
  - Tipo de instalación
  - Descripción
  - Precio
  - Estado: disponible, temporalmente fuera de servicio,...
- El horario disponible solo será visible para los socios

### 002 Crear usuario (webmaster)

- El webmaster podrán crear usuarios para el club deportivo. Estos usuarios incluyen: socios, recepcionistas y webmaster.
- De los usuarios se guardarán los siguientes datos:
  - Identificador, que, aunque no se mostrará, esta recopilado en la base de datos
  - Nombre
  - Apellidos
  - Email
  - Teléfono
  - Privilegios: socio, recepcionista, webmaster, no socio.
  - Estado: pendiente, activo, bloqueado, de baja.

### 003 Crear usuarios (repcionista)

- El recepcionista podrá crear cuentas de socios.
- De los usuarios se guardarán los siguientes datos:
  - Identificador, que, aunque no se mostrará, esta recopilado en la base de datos
  - Nombre
  - Apellidos
  - Email
  - Teléfono
  - Estado: pendiente, activo, bloqueado, de baja.
- Los privilegios por defecto serán de socio

### 004 Crear cuenta de socio (socio)

- Un socio podrá crear una cuenta como socio enviando una solicitud. Esta cuenta será creada, pero con estado "pendiente" hasta que un recepcionista o webmaster acepte esta solicitud. Una vez aceptada, el estado del socio cambia a "activo".

### 005 Dar privilegios a un usuario (webmaster)

- Un webmaster puede cambiar los privilegios de un usuario para poder asignar privilegios a los recepcionistas u otro webmaster.
- Por defecto, desde la web pública será posible solo crear cuentas de socio. En este caso, enviar solicitud para crear una cuenta de socio. Solo podrá cambiar el privilegio de un socio los recepcionistas o los webmaster. Solo el webmaster podrá crear usuarios con privilegios diferentes de socio. Es decir, puede crear otro webmaster o un recepcionista.

### 006 Cambiar estado del socio (repcionista/webmaster)

- Por defecto, desde la web pública será posible solo crear cuentas de socio en estado pendiente. Una vez enviada la solicitud para crear una cuenta de socio, un recepcionista o un web master podrá cambiar el estado.
- Estados: pendiente, activo, bloqueado, de baja.

- Un recepcionista o webmaster recibirá la solicitud de un socio y puede aceptar esta solicitud. De esta forma, un usuario cambiará de pendiente a activo.

#### 007 Darse de baja como socio (socio)

- Un socio podrá darse de baja del club deportivo.
- El estado del socio cambiará de "activo" a "de baja" cuando el usuario lo solicite.

#### 008 Crear una reserva (socio)

- Un socio puede reservar una pista.
- De las reservas guardaremos:
  - Identificador
  - Identificador del usuario de la reserva
  - Identificador del usuario que realiza la reserva
  - Identificador de la pista reservada
  - Horario reservado
  - Fecha reservada

#### 009 Crear una reserva para un socio o no socio (recepcionista)

- Este tipo de reservas serán hechas únicamente por recepcionistas
- De los usuarios de tipo "no socio" solo guardaremos nombre, email y teléfono de contacto
- El identificador del usuario que realiza la reserva seria el del recepcionista para indicar que esta reserva se ha hecho presencialmente o por teléfono

#### 010 Ver reservas (socio)

- El socio podrá ver todas las reservas que tiene activas actualmente y los detalles de la misma.

#### 011 Ver reservas de los socios (recepcionista/webmaster)

- Podrán ver las reservas activas y quien ha realizado la reserva.
- Podrán cancelar reservas o realizarlas, para socios y para no socios.

#### 012 Cambiar el estado de una pista (repcionista/webmaster)

- Se podrán añadir instalaciones, eliminarlas o bloquearlas temporalmente por un administrador o un recepcionista. Un socio podrá realizar una reserva.
- Un webmaster o recepcionista puede:
  - Añadir una nueva pista
  - Cambiar el estado de la pista
  - Bloquear una pista para que no se puedan hacer reservas
  - Eliminar una pista

#### 013 Cancelar reserva (socio)

- El socio podrá cancelar las reservas que tiene hechas, como máximo el día de antes de la misma. En ese caso, se devolvería el dinero a la tarjeta del club.

#### 014 Cancelar reserva (repcionista)

- Podrán cancelar reservas que han realizado los socios o reservas que han hecho ellos mismos para socios o no socios.

#### 015 Consultar saldo (socio)

- El socio podrá consultar el saldo que tiene disponible en la tarjeta del club

#### 016 Recargar saldo (socio)

- El socio podrá solicitar recargar su saldo
- Se indicaría la cantidad que se desea recargar

#### 017 Ver socios (repcionista/webmaster)

- Podrán ver un listado de todos los socios.

#### 018 Mostrar tienda (socio)

- Se mostrará la tienda.

## Tecnologías

Hemos empleado las siguientes tecnologías para desarrollar nuestra web:

- Para la base de datos hemos usado SQL Postgres
- Para el backend hemos usado Typescript con NestJS
- Para el frontend hemos usado VueJS
- Para la autenticación hemos usado JWT

## Interoperación

Como hemos detallado en las funcionalidades, nuestra web interopera con otros proyectos. En nuestro caso, la interoperación es con el pago por TPVV y mostrar una tienda en la web del centro a partir de categorías o productos de una de las tiendas online.

El TPVV en nuestra parte de la web lo utilizamos para:

Recargar saldo a la cuenta del centro. Dentro de la cuenta del socio, es posible recargar saldo a tu cuenta del centro de forma automática, introduciendo la cantidad que desees. De la misma forma que puedes consultar tu saldo.

Una vez con el saldo del socio, podemos:

Realizar reservas por parte del socio. Antes de finalizar la reserva es necesario tener saldo en la cuenta del centro. Por ello, se realizará la comprobación del saldo y una vez comprobado el saldo, se reservaría la instalación.

Devolución del saldo por la cancelación de la reserva. Si un socio cancela una reserva, automáticamente se le devuelve a la cuenta del centro el precio de reserva de la instalación.

## Mockups

Los mockups en los que hemos basado el diseño son los siguientes:

<https://www.figma.com/design/bevpB14qk964jhro8kTpiN/centro-deportivo?node-id=0-1&t=ez7aKUYHGjdxRa8Z-1>

Pero vamos a destacar los principales flujos de la web, entre ellos, realizar reservas por un socio. Contamos que el socio ha iniciado sesión



correctamente y accede a reservar. La primera pantalla que le aparece es la de seleccionar el tipo de pista que desea reservar:



Una vez seleccionada la pista o la instalación que el socio desea reservar, seleccionamos el día y la franja horaria:

**CENTRO DEPORTIVO**

Inicio Instalaciones Tienda **Reservar** Mi Cuenta

**Pista de Pádel** Selecciona la fecha y la franja horaria

**May 2024**

Sun	Mon	Tue	Wed	Thu	Fri	Sat
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Franja Horaria:

- 10:00 - 11:30
- 11:30 - 13:00
- 16:00 - 17:30
- 17:30 - 19:00**
- 19:00 - 21:30

**Siguiente**

Una vez hecho esto, seleccionamos la pista. De la fecha seleccionada y la franja horaria que se ha seleccionado en la pantalla anterior, aparecen todas las pistas disponibles. Cada pista aparece con su precio.

Al pulsar el botón de finalizar, se cobraría la pista que se ha seleccionado automáticamente. Se comprobaría con el TPVV que el saldo de la cuenta del socio sea suficiente para abonar la pista, y se cobraría automáticamente.

**Pista de Pádel**

14 de Mayo de 2024  
17:30 - 19:00

Selecciona la pista que deseas entre las disponibles en la fecha  
y horario seleccionado



Pista 1  
13 €



Pista 2  
15€



Pista 5  
17€

Finalizar

Por último, nos aparece un resumen. En esta pantalla nos muestra el  
localizados de la reserva, los detalles:

- Tipo de pista
- Fecha
- Franja horaria
- Número de pista
- Comprobante del pago

## Reservar Pista

Resumen de la reserva

Localizador de la reserva:  
00000000000000

Pista de Pádel  
14 de Mayo de 2024  
17:30 - 19:00  
Pista 2

Pago completado:  
15€



Desde la cuenta del socio, también se puede consultar el saldo actual, además de poder recargar el saldo desde ahí.

Mi cuenta

Saldo

Saldo actual: 8,00€

Cantidad

Recargar

Para los recepcionistas, el flujo de las reservas es un poco diferente. En primer lugar, tienen que introducir los datos del cliente. Esto es porque un recepcionista no puede realizar reservas para si mismo. Por ello, tiene que ingresar esos datos mínimos para realizar la reserva de un socio o no socio. Con esos datos, comprobaremos si el cliente que desea realizar la reserva es socio o no.



The screenshot shows the 'Reservar Pista' (Reserve Court) interface. At the top, there is a navigation bar with the logo 'CENTRO DEPORTIVO' on the left and links for 'Inicio', 'Instalaciones', 'Tienda', 'Reservar' (highlighted), and 'Mi Cuenta' on the right. The main heading is 'Reservar Pista' with the subtitle 'Introduce los datos del cliente'. Below this, there are three input fields labeled 'Nombre', 'Email', and 'Teléfono'. A dark 'Continuar' button is positioned below the input fields. The background of the form is a faded image of a sports facility with multiple tennis courts.

Una vez introducidos los datos del cliente, las pantallas para realizar la reserva son prácticamente iguales a las del socio:



## Reservar Pista

Selecciona el tipo de pista que desaea reservar

Pista de Pádel

Pista de Tenis

Piscina

Pista de Fútbol

Pista de Baloncesto

Pista de Voleibol

Gimnasio

Pista de Atletismo

Pista de Frontón

Siguiente

Pista de Pádel

Selecciona la fecha y la franja horaria

### May 2024

Sun	Mon	Tue	Wed	Thu	Fri	Sat
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Franja Horaria:

10:00 - 11:30

11:30 - 13:00

16:00 - 17:30

17:30 - 19:00

19:00 - 21:30

Siguiente

Pista de Pádel

14 de Mayo de 2024  
17:30 - 19:00

Selecciona la pista que deseas entre las disponibles en la fecha y horario seleccionado



Pista 1  
13 €



Pista 2  
15€



Pista 5  
17€

Finalizar

Por último, nos aparece un resumen. A diferencia del resumen del socio, aquí nos aparece la información del cliente que ha realizado la reserva.

## Reservar Pista

Resumen de la reserva

Localizador de la reserva:  
00000000000000

Pista de Pádel  
14 de Mayo de 2024  
17:30 - 19:00  
Pista 2

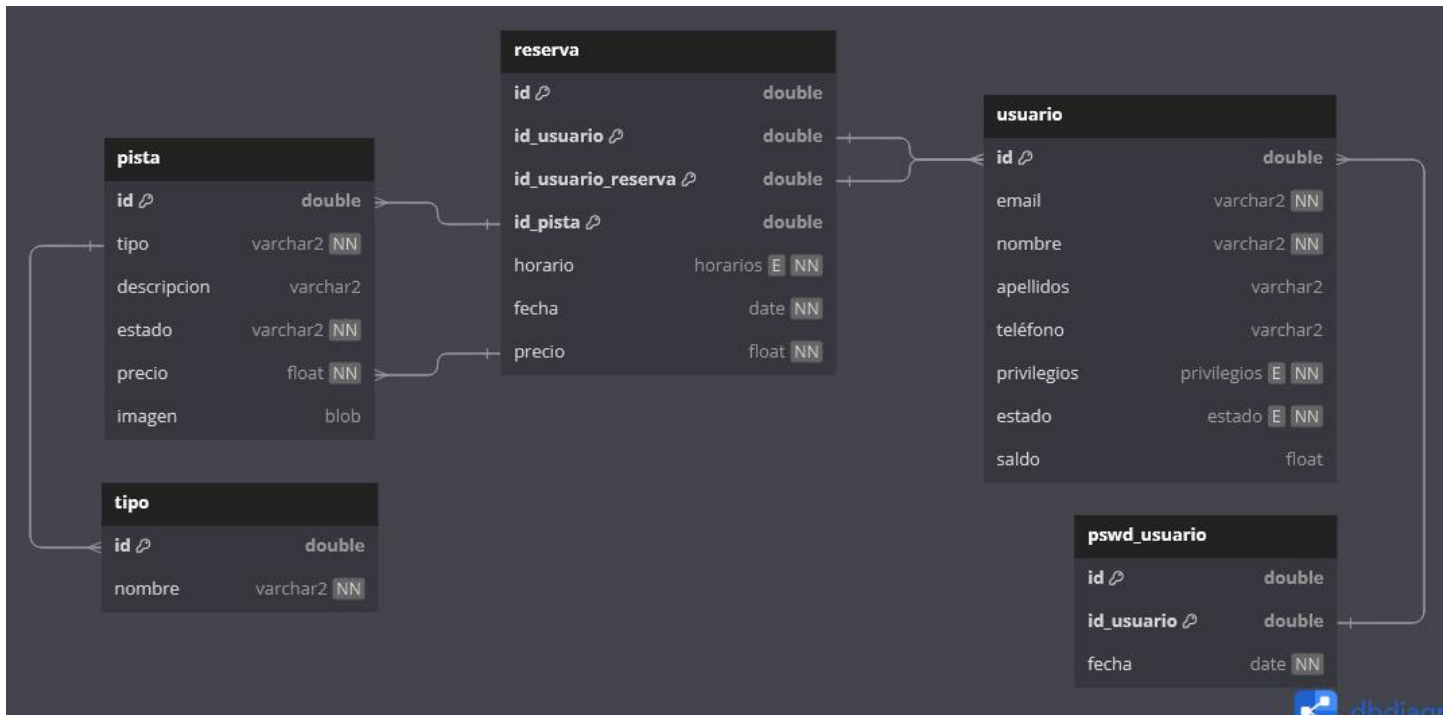
Pago completado:  
15€

Datos del cliete:  
Nombre: Aurora  
Email: aam216@alu.ua.es  
Teléfono: 600 000 000



## Diagrama de datos

El diagrama de datos que hemos empleado para desarrollar la web ha sido:

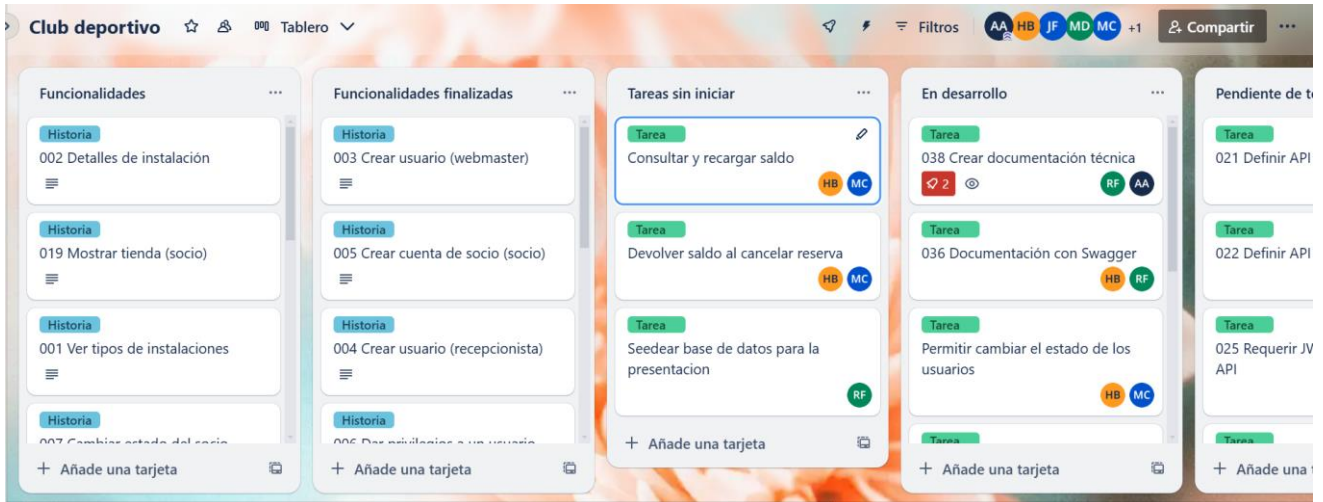


Cabe destacar que, en las reservas, hay 2 id, aparte del identificador propio de la reserva. Está el "id\_usuario" y el "id\_usuario\_reserva". Estos campos se usan para las reservas realizadas por un recepcionista. Como hemos comentado anteriormente, un recepcionista no puede realizar reservar para si mismo, por lo que, una reserva contendrá el identificador del usuario que reserva la instalación y el identificador del recepcionista que gestiona la reserva.



## Metodología

Para organizar nuestro trabajo hemos usado un tablero de trello.



El tablero de trello lo hemos organizado con las siguientes columnas:

- Funcionalidades
- Funcionalidades finalizadas
- Tareas sin iniciar
- En desarrollo
- Pendiente de testeo
- Tareas finalizadas

Hemos etiquetado cada tarjeta con una de las siguientes etiquetas:

- Historia
- Tareas
- Error

Además de usar el trello, a lo largo de las clases de prácticas de la asignatura, realizábamos reuniones para planear el proyecto y para ver la evolución del mismo.

## Implementación

Para la implementación de la web, la hemos desarrollado en local, ejecutándolo con imágenes de Docker, para de esta manera que el proyecto tuviera una fácil instalación y así poder cooperar entre nosotros de forma más sencilla. También pensando en el despliegue en clase, decidimos que sería más sencillo hacerlo con Docker.

El link al repositorio es:

<https://github.com/aam216-ua/iweb-centro-deportivo.git>

## Detalles del código

Para el código que hemos desarrollado, destacamos las siguientes implementaciones:

```
public async signIn(
  authCredentialsDto: AuthCredentialsDto
): Promise<{ token: string; user: User }> {
  const user = await this.userRepository.findOne({
    where: {
      email: authCredentialsDto.email,
    },
  });

  if (!user) throw new UnauthorizedException('invalid credentials');

  if (user.status == UserStatus.BLOCKED)
    throw new UnauthorizedException('blocked user');

  if (user.status == UserStatus.PENDING)
    throw new UnauthorizedException('pending user');

  const password = await this.passwordRepository.findOneBy({
    user: { id: user.id },
  });

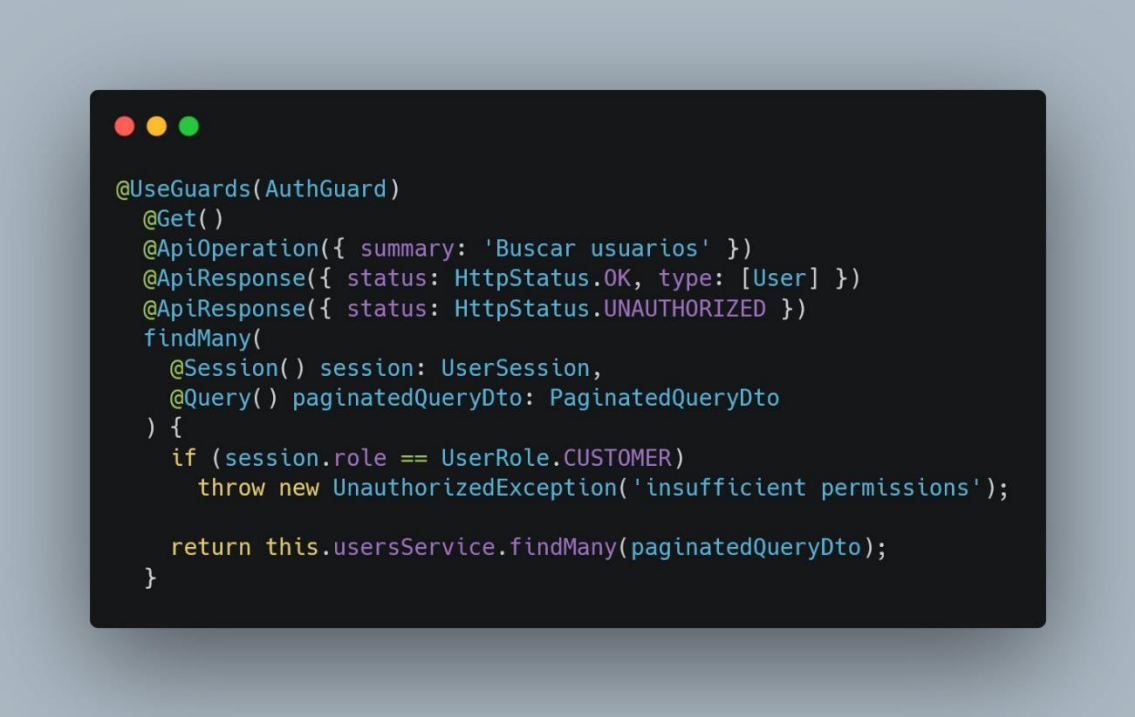
  if (!password) throw new UnauthorizedException('invalid credentials');

  if (await verify(password.password, authCredentialsDto.password)) {
    return {
      token: await this.jwtService.signAsync({
        id: user.id,
        role: user.role,
        status: user.status,
      }),
      user,
    };
  }

  throw new UnauthorizedException('invalid credentials');
}
```

Permite a los usuarios iniciar sesión con su contraseña y devuelve un access token de JWT en el caso de que sea correcto. Este token es necesario para la mayoría de llamadas de la api, que fallarán si no se proporciona. Además, en él va incluido el rol del usuario, lo que nos permite bloquear a usuarios

de realizar operaciones para las que no tienen suficientes privilegios. Los token no se pueden falsificar debido al checksum que tienen al final.



```
@UseGuards(AuthGuard)
@Get()
@ApiOperation({ summary: 'Buscar usuarios' })
@ApiResponse({ status: HttpStatus.OK, type: [User] })
@ApiResponse({ status: HttpStatus.UNAUTHORIZED })
findMany(
    @Session() session: UserSession,
    @Query() paginatedQueryDto: PaginatedQueryDto
) {
    if (session.role == UserRole.CUSTOMER)
        throw new UnauthorizedException('insufficient permissions');

    return this.usersService.findMany(paginatedQueryDto);
}
```

En este ejemplo uso el token mencionado anteriormente. Primero, uso el decorador `@UseGuards(AuthGuard)`, que comprueba que el token es correcto. El decorador `@Session` extrae los datos del token, entre los que están el id de usuario y su rol. Podemos usar este rol para determinar si tiene acceso o no a la operación. Por ejemplo, los socios no deberían tener acceso a la lista de todos los usuarios.

```

public async findMany(
  queryBookingDto: QueryBookingDto
): Promise<PaginatedResult<Booking>> {
  const {
    page = 0,
    size = 10,
    appointeeId = undefined,
    appointerId = undefined,
    venueId = undefined,
    after = undefined,
    before = undefined,
    sort = 'DESC',
  } = queryBookingDto;

  const query = this.bookingRepository
    .createQueryBuilder('booking')
    .leftJoinAndSelect('booking.venue', 'venue')
    .leftJoinAndSelect('booking.appointer', 'user.booked')
    .leftJoinAndSelect('booking.appointee', 'user.bookings')
    .orderBy('booking.date', sort);

  if (appointeeId)
    query.andWhere({
      appointee: await this.userService.findOne(appointeeId),
    });

  if (appointerId)
    query.andWhere({
      appointer: await this.userService.findOne(appointerId),
    });

  if (venueId)
    query.andWhere({ venue: await this.venueService.findOne(venueId) });

  if (after) query.andWhere('booking.date >= :after', after);

  if (before) query.andWhere('booking.date <= :before', before);

  const [data, total] = await query
    .skip(page * size)
    .take(size)
    .getManyAndCount();

  return {
    data,
    meta: { page, size, total },
  } as PaginatedResult<Booking>;
}

```

```

export class QueryBookingDto extends PaginatedQueryDto {
  @IsOptional()
  @IsUUID()
  appointerId: string;

  @IsOptional()
  @IsUUID()
  appointeeId: string;

  @IsOptional()
  @IsUUID()
  venueId: string;

  @IsOptional()
  @IsDate()
  @Transform(({ value }) => (value as Date).setHours(0, 0, 0, 0))
  after: Date;

  @IsOptional()
  @IsDate()
  @Transform(({ value }) => (value as Date).setHours(0, 0, 0, 0))
  before: Date;

  @IsOptional()
  @IsString()
  @IsIn(['ASC', 'DESC'])
  sort: 'ASC' | 'DESC';
}

```

## Problemas encontrados

A lo largo del proyecto nos han ido surgiendo varios problemas, sobre todo, problemas de comunicación. Ya que en muchas ocasiones teníamos opiniones dispares sobre como plantear partes del proyecto o sobre como desarrollarlo. De igual forma, hemos podido solventar estas discrepancias en las diferentes reuniones que hemos ido realizando a lo largo del proyecto.

También hemos tenido un problema con la interconexión con los distintos trabajos de nuestros compañeros. En cuanto al TPVV, nuestro proyecto tenía que hacer uso de este para aumentar el saldo de la tarjeta del centro.

El problema que hemos tenido es que, para realizar las reservas, el TPVV hace una llamada a un URL nuestro para confirmar el pago del cliente. Pero, como nuestro proyecto esta en local, no podemos proporcionarle la URL a la que llaman.

## Mejoras y futuras ampliaciones

Alguna de las mejoras que se podrían plantear para nuestro proyecto son las siguientes.

Poder realizar reservas periódicas. Por ejemplo, para un equipo de baloncesto, que tenga un horario fijo de enteramiento, poder realizar una reserva de uno o varios días a la semana en una franja a la vez. Es decir, no tener que realizar varias reservas, con una única reserva sobraría. Una forma de gestionar este tipo de reservas seria a través de los recepcionistas.

Otra de las mejoras que podríamos añadir a nuestro proyecto podría ser colaborar con alguna tienda para mostrar sus productos en nuestro centro deportivo, de esta forma, tendríamos una web más completa.

Como hemos comentado, el desarrollo de la practica ha sido en local, por lo que, una de las mejoras seria hacerlo funcional. Por ello, para hacer que nuestro proyecto fuera funcional tendríamos que subirlo a un hosting.