

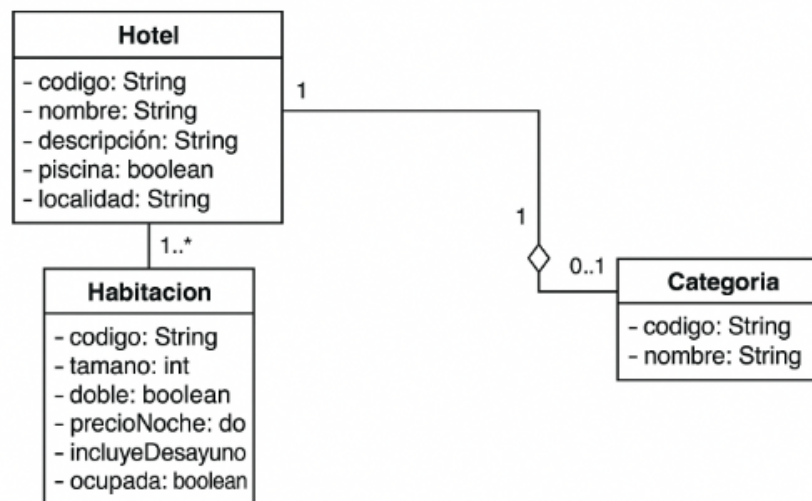
SIMULACRO DE EXAMEN

1. API REST de Gestión de Hoteles y Habitaciones

Se desea construir una **API REST** para gestionar hoteles y habitaciones.

Se proporciona:

- Las entidades **Hotel**, **Categoría** y **Habitación**.



Un hotel pertenece a una única categoría

Una categoría puede estar asociada a múltiples hoteles

Un hotel puede tener una o más habitaciones asociadas

Cada habitación pertenece a un único hotel.

- Un archivo **data.sql** que carga datos iniciales en una base de datos H2 en memoria.

SELECT * FROM HOTEL;

PISCINA	CATEGORIA_ID	ID	CODIGO	DESCRIPCION	LOCALIDAD	NOMBRE
TRUE	3	1	HOT-001	Frente al mar	Valencia	Hotel Playa
FALSE	2	2	HOT-002	En el centro histórico	Madrid	Hotel Centro
TRUE	1	3	HOT-003	Entorno natural	Asturias	Hotel Rural
TRUE	4	4	HOT-004	Exclusivo y elegante	Sevilla	Hotel Lujo
FALSE	3	5	HOT-005	Cerca del aeropuerto	Barcelona	Hotel Negocios
TRUE	2	6	HOT-006	Ideal para familias	Bilbao	Hotel Familiar

SELECT * FROM CATEGORIA;

ID	CODIGO	DESCRIPCION	NOMBRE
1	CAT-2EST	Servicios básicos, ideal para presupuestos limitados	2 estrellas
2	CAT-3EST	Buena relación calidad-precio	3 estrellas
3	CAT-4EST	Alta calidad y confort	4 estrellas
4	CAT-5EST	Lujo y servicios exclusivos	5 estrellas

SELECT * FROM HABITACION;

DOBLE	INCLUYE_DESAYUNO	OCUPADA	PRECIO_NOCHE	TAMANO	HOTEL_ID	ID	CODIGO
FALSE	TRUE	FALSE	50.0	20	1	1	HAB-001-A
TRUE	TRUE	TRUE	90.0	30	1	2	HAB-002-A
TRUE	FALSE	FALSE	100.0	25	1	3	HAB-003-A
FALSE	FALSE	FALSE	45.0	15	2	4	HAB-004-A
TRUE	TRUE	FALSE	80.0	25	2	5	HAB-005-A
TRUE	TRUE	TRUE	110.0	35	2	6	HAB-006-A
FALSE	FALSE	FALSE	30.0	10	3	7	HAB-007-A
TRUE	FALSE	FALSE	60.0	20	3	8	HAB-008-A
TRUE	TRUE	TRUE	95.0	30	3	9	HAB-009-A
TRUE	TRUE	FALSE	200.0	40	4	10	HAB-010-A
TRUE	TRUE	FALSE	180.0	35	4	11	HAB-011-A
FALSE	TRUE	TRUE	150.0	20	4	12	HAB-012-A
FALSE	FALSE	FALSE	55.0	15	5	13	HAB-013-A
TRUE	TRUE	FALSE	85.0	25	5	14	HAB-014-A
TRUE	FALSE	TRUE	95.0	30	5	15	HAB-015-A
FALSE	TRUE	FALSE	60.0	20	6	16	HAB-016-A
TRUE	TRUE	FALSE	100.0	30	6	17	HAB-017-A
TRUE	FALSE	TRUE	90.0	25	6	18	HAB-018-A

(18 rows, 1 ms)

ENDPOINTS: Formato de entrada y salida

1. GET /hoteles/buscar

Buscar todos los hoteles u hoteles por localidad o código de categoría.

GET <http://localhost:8082/hoteles/buscar>

GET <http://localhost:8082/hoteles/buscar?codigoCategoria=CAT-3EST>

GET <http://localhost:8082/hoteles/buscar?localidad=Valencia>

Salida (ejemplo GET <http://localhost:8082/hoteles/buscar?localidad=Valencia>):

HTTP/1.1 200

> (Headers) ...Content-Type: application/json...

```
[
  {
    "codigo": "HOT-001",
    "nombre": "Hotel Playa",
    "descripcion": "Frente al mar",
    "piscina": true,
    "localidad": "Valencia",
    "categoria": {
      "codigo": "CAT-4EST",
      "nombre": "4 estrellas"
    }
  }
]
```

2. GET /habitaciones/{codigoHotel}/buscar

Buscar habitaciones libres de un hotel, filtrando por tamaño mínimo y precio mínimo/máximo.

GET <http://localhost:8082/habitaciones/HOT-001/buscar?>

[tamanoMinimo=20&](#)

[precioMinimo=100&](#)

[precioMaximo=200](#)

Ejemplos:

<pre> GET http://localhost:8082/habitaciones/HOT-001/buscar? tamanoMinimo=20& precioMinimo=100& precioMaximo=200 </pre> <p>Submit Request Open in Editor</p> <pre> HTTP/1.1 200 > (Headers) ...Content-Type: application/json... [{ "codigo": "HAB-003-A", "tamano": 25, "doble": true, "precioNoche": 100.0, "incluyeDesayuno": false, "ocupada": false }] </pre>	<pre> GET http://localhost:8082/habitaciones/HOT-001/buscar </pre> <p>Submit Request Open in Editor</p> <pre> [{ "codigo": "HAB-001-A", "tamano": 20, "doble": false, "precioNoche": 50.0, "incluyeDesayuno": true, "ocupada": false }, { "codigo": "HAB-003-A", "tamano": 25, "doble": true, "precioNoche": 100.0, "incluyeDesayuno": false, "ocupada": false }] </pre>
---	--

3. POST /hoteles

Registrar un nuevo hotel.

<pre> POST http://localhost:8082/hoteles Content-Type: application/json Authorization: Bearer eyJhbGciOiJIUzI1Ni. </pre> <pre> { "codigo": "HOT-007", "nombre": "Nuevo Hotel", "descripcion": "Hotel de prueba", "piscina": false, "localidad": "Granada", "codigoCategoria": "CAT-3EST" } </pre>	<pre> HTTP/1.1 201 > (Headers) ...Content-Type: application/json... { "codigo": "HOT-007", "nombre": "Nuevo Hotel", "descripcion": "Hotel de prueba", "piscina": false, "localidad": "Granada", "categoria": { "codigo": "CAT-3EST", "nombre": "3 estrellas" } } </pre>
---	--

4. POST /habitaciones/{codigoHotel}

Registrar una nueva habitación en un hotel.

```
POST http://localhost:8082/habitaciones/HOT-001
Content-Type: application/json
Authorization: Bearer eyJhbGciOiJIUzI1NiJ9.eyJyb2xlci
```

```
{
  "codigo": "HAB-019-A",
  "tamano": 25,
  "doble": true,
  "precioNoche": 85.0,
  "incluyeDesayuno": false
}
```

mit Request

Open in Editor

HTTP/1.1 201

(Headers) ...Content-Type: application/json...

```
{
  "codigo": "HAB-019-A",
  "tamano": 25,
  "doble": true,
  "precioNoche": 85.0,
  "incluyeDesayuno": false,
  "ocupada": false
}
```

5. DELETE /habitaciones/{codigoHabitacion}

Eliminar una habitación.

```
DELETE http://localhost:8082/habitaciones/HAB-019-A
Authorization: Bearer eyJhbGciOiJIUzI1NiJ9.eyJyb2xlcyI6WyJST
```

Submit Request

Open in Editor

DELETE http://localhost:8082/habitaciones/HAB-019-A

Show Request

HTTP/1.1 204

X-Content-Type-Options: nosniff

X-XSS-Protection: 0

Cache-Control: no-cache, no-store, max-age=0, must-revalidate

Pragma: no-cache

Expires: 0

Date: Tue, 06 May 2025 20:51:59 GMT

6. PATCH /habitaciones/{codigoHabitacion}/ocupar

```
PATCH http://localhost:8082/habitaciones/HAB-001-A/ocupar
Authorization: Bearer eyJhbGciOiJIUzI1NiJ9.eyJyb2xlcyI6WyJ
```

mit Request

Open in Editor

```
PATCH http://localhost:8082/habitaciones/HAB-001-A/ocupar
```

Show Request

```
HTTP/1.1 200
```

```
> (Headers) ...Content-Type: application/json...
```

```
{
  "codigo": "HAB-001-A",
  "tamano": 20,
  "doble": false,
  "precioNoche": 50.0,
  "incluyeDesayuno": true,
  "ocupada": true
}
```

CONTROL DE EXCEPCIONES:

Manejo global de errores: excepciones personalizadas:

- HotelNotFoundException
- HabitacionNotFoundException
- CategoriaNotFoundException

Impleméntalas siempre que proceda (en los endpoints que usen el código de hotel, código de habitación y código de categoría). **Ejemplos:**

```
GET http://localhost:8082/habitaciones/HOT-666/buscar
```

Show Request

```
HTTP/1.1 404
```

```
> (Headers) ...Content-Type: application/json...
```

```
{
  "timestamp": "2025-05-06T22:09:14.7980417",
  "error": "Hotel no encontrado con código: HOT-666"
}
```

```
POST http://localhost:8082/habitaciones/HOT-666
```

```
Content-Type: application/json
```

```
Authorization: Bearer eyJhbGciOiJIUzI1NiJ9.eyJyb2xlcyI6WyJ
```

```
{
  "codigo": "",
  "tamano": 0,
  "doble": false,
  "precioNoche": 0,
  "incluyeDesayuno": false
}
```

	<pre> HTTP/1.1 404 > (Headers) ...Content-Type: application/json... { "error": "Hotel no encontrado con código: HOT-666", "timestamp": "2025-05-06T22:40:30.7689306" } </pre>
<pre> DELETE http://localhost:8082/habitaciones/HAB-666 Authorization: Bearer eyJhbGciOiJIUzI1NiJ9.eyJyb2xlcyI6WyJ </pre> <p>mit Request Open in Editor</p> <pre> DELETE http://localhost:8082/habitaciones/HAB-666 Show Request HTTP/1.1 404 > (Headers) ...Content-Type: application/json... { "error": "Habitación no encontrada con código: HAB-666", "timestamp": "2025-05-06T22:42:39.5019339" } </pre>	<pre> PATCH http://localhost:8082/habitaciones/HAB-666/ocupar Authorization: Bearer eyJhbGciOiJIUzI1NiJ9.eyJyb2xlcyI6WyJS </pre> <p>mit Request Open in Editor</p> <pre> PATCH http://localhost:8082/habitaciones/HAB-666/ocupar Show Request HTTP/1.1 404 (Headers) ...Content-Type: application/json... { "error": "Habitación no encontrada con código: HAB-666", "timestamp": "2025-05-06T22:43:34.7972157" } </pre>
<pre> POST http://localhost:8082/hoteles Content-Type: application/json Authorization: Bearer eyJhbGciOiJIUzI1NiJ9.eyJyb2xlcyI6WyJ </pre> <pre> { "codigo": "HOT-007", "nombre": "Nuevo Hotel", "descripcion": "Hotel de prueba", "piscina": false, "localidad": "Granada", "codigoCategoria": "CAT-666" } </pre> <p>mit Request Open in Editor</p> <pre> HTTP/1.1 404 > (Headers) ...Content-Type: application/json... { "error": "Categoría no encontrada con código: CAT-666", "timestamp": "2025-05-06T22:45:53.8791803" } </pre>	<p>Observa que en la salida se devuelve:</p> <ul style="list-style-type: none"> • Estado HTTP 404 • JSON con el mensaje de error de la excepción y el timestamp

<h2>Seguridad</h2> <p>Usuario: admin / Contraseña: admin / Rol: ADMIN</p> <p>Usuario: user / Contraseña: user /</p> <p>Rol: USER</p> <p>Reglas de acceso:</p> <p>Rol ADMIN puede acceder a todos los endpoints (GET, POST, PATCH, DELETE).</p> <p>Todos los endpoints GET son públicos.</p>	<h2>Validaciones obligatorias</h2> <ul style="list-style-type: none"> El nombre del hotel no puede ser nulo ni vacío. El precio por noche de una habitación debe ser mayor que 0. El tamaño de una habitación debe ser mayor que 10 metros cuadrados. El código (codigo) de hotel, habitación y categoría debe ser único y no nulo. No se debe permitir registrar una habitación en un hotel inexistente. <p>Ejemplo:</p> <pre>POST http://localhost:8082/hoteles Content-Type: application/json Authorization: Bearer eyJhbGciOiJIUzI1NiJ9.eyJyb2x7 { "codigo": "HOT-008", "nombre": "", "descripcion": "Hotel de prueba", "piscina": false, "localidad": "Granada", "codigoCategoria": "CAT-666" }</pre> <p>mit Request Open in Editor</p> <pre>HTTP/1.1 400 (Headers) ...Content-Type: application/json... { "timestamp": "2025-05-06T23:14:52.073421", "error": "no debe estar vacío" }</pre>
--	---

Reglas obligatorias:

- Uso de DTOs:** Todos los datos de entrada y salida deben gestionarse a través de objetos DTO.
- Uso de Servicios (@Service):** No se permite inyectar Repositorios (@Repository) directamente en los Controladores (@RestController).
- Uso de ResponseEntity:** Todos los controladores deben devolver ResponseEntity como respuesta con códigos de estado Http adecuados.

2. Spring MVC HOTELES

La app muestra una **pantalla de login simple** (usuario y contraseña). No modifiques la vista **login.html**.

Se puede entrar como:

Usuario: invitado / da igual la contraseña que se introduzca.

Usuario: admin / Contraseña: admin

Usuario: user / Contraseña: user

La autenticación se realizará contra el API Rest de hoteles.

- Si no eres invitado, se conectará al API Rest de Hoteles a través de (JWT) para obtener el token.
- La url estará configurada en application.properties:
`api.hoteles.url=http://localhost:8082`
- Muestra el token por consola.

A mockup of a login form titled "Iniciar Sesión en el simulacro". It features two input fields: "Usuario" and "Contraseña". Below the fields is a blue button labeled "Entrar".

Si falla la autenticación contra el API Rest, se mostrará el mensaje de la excepción obtenida al realizar la conexión con WebClient.

En login.html se pinta dicho error:

```
<div th:if="${error}" class="alert alert-danger" th:text="${error}"></div>
```

A mockup of the login form titled "Iniciar Sesión en el simulacro" showing an error state. The "Usuario" and "Contraseña" fields are present. Below them is a red alert box containing the text: "ERROR: 500 Internal Server Error from POST http://localhost:8082/auth/login". At the bottom is a blue button labeled "Entrar".

Tanto si hace login como invitado, admin o user, obtendrás una lista de todos los hoteles obtenidos a través del endpoint del API Rest (**GET /hoteles/buscar**). Modifica la vista buscarHoteles.html

Buscar Hoteles

Hotel Playa

Localidad: Valencia

Descripción: Frente al mar

Piscina: Sí

Categoría: 4 estrellas

Ver Habitaciones

Hotel Centro

Localidad: Madrid

Descripción: En el centro histórico

Piscina: No

Categoría: 3 estrellas

Ver Habitaciones

Hotel Rural

Localidad: Asturias

Descripción: Entorno natural

Piscina: Sí

Categoría: 2 estrellas

Ver Habitaciones

Hotel Lujo

Localidad: Sevilla

Descripción: Exclusivo y elegante

Piscina: Sí

Categoría: 5 estrellas

Ver Habitaciones

Hotel Negocios

Localidad: Barcelona

Descripción: Cerca del aeropuerto

Piscina: No

Categoría: 4 estrellas

Ver Habitaciones

Hotel Familiar

Localidad: Bilbao

Descripción: Ideal para familias

Piscina: Sí

Categoría: 3 estrellas

Ver Habitaciones

Al seleccionar un hotel, muestra las habitaciones disponibles a través del endpoint del API Rest

(**GET /habitaciones/{codigoHotel}/buscar**)

Modifica la vista buscarHoteles.html

localhost:8080/habitaciones/HOT-001/buscar

Habitaciones disponibles

Habitación HAB-001-A

Tamaño: 20 m²

Doble: No

Precio por noche: 50.0 €

Incluye desayuno: Sí

Ocupada: No

Habitación HAB-003-A

Tamaño: 25 m²

Doble: Sí

Precio por noche: 100.0 €

Incluye desayuno: No

Ocupada: No

Volver a Hoteles

Si el usuario es “admin”, muestra un botón para eliminar la habitación seleccionada.

(**REMOVE** /habitaciones/{codigoHabitacion}).

Habitaciones disponibles

Habitación HAB-010-A

Tamaño: 40 m²

Doble: Sí

Precio por noche: 200.0 €

Incluye desayuno: Sí

Ocupada: No

Eliminar

Habitación HAB-011-A

Tamaño: 35 m²

Doble: Sí

Precio por noche: 180.0 €

Incluye desayuno: Sí

Ocupada: No

Eliminar

Volver a Hoteles

Se vuelve a mostrar las habitaciones disponibles del hotel y un mensaje indicando que se ha eliminado correctamente.

Habitaciones disponibles

Habitación eliminada correctamente.

Habitación HAB-011-A

Tamaño: 35 m²

Doble: Sí

Precio por noche: 180.0 €

Incluye desayuno: Sí

Ocupada: No

Eliminar

Volver a Hoteles