

Proyecto Final Full Stack

Desafío Latam

Gustavo Parada – Andrés Villarroel

Diseño y Prototipo

Tienda de Juegos de Nintendo Switch

Games Switch Store



Juegos

Filtros

Genero del Juego

- ☐ Acción
- ☐ Aventuras
- ☐ Deporte
- ☐ Estrategia
- ☐ Lucha

Precios del Juego

- ☐ Gratuito
- ☐ \$1 - \$9.99912
- ☐ \$10.000 - \$29.999
- ☐ \$30.000 - \$49.999
- ☐ \$50.000+



Pokémon™ Scarlet

18/11/22

\$58.990

| Nintendo Switch



Pokémon™ Violet

18/11/22

\$58.990

| Nintendo Switch



Paquete doble de
Pokémon™ Scarlet y...

18/11/22

\$117.890

| Nintendo Switch



Xenoblade Chronicles™ 3

29/7/22

\$58.990

| Nintendo Switch



Kirby's Dream Buffet™

17/8/22

\$11.263 ~~\$16.000~~ -30%

| Nintendo Switch

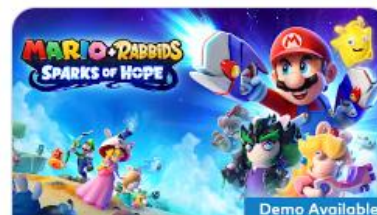


Mario Strikers™: Battle
League

10/6/22

\$39.523 ~~\$58.000~~ -33%

| Nintendo Switch



MARIO + RABBIDS SPARKS
OF HOPE

20/10/22

\$15.490 ~~\$46.000~~ -67%

| Nintendo Switch



LIVE A LIVE

22/7/22

\$48.990

| Nintendo Switch





Registrarse

Ingresar tus Datos

Nombre:

Dirección:

Correo:

Rol:

Elige tu Avatar :





Ingreso al Sistema

Usuario :

Contraseña :



Mi Perfil



Mi Información

Mi Nombre :

Mi Apellido :

Mi Dirección :

Mi Correo:

Rol : Cliente

Editar Carrito

Mi Historial de Preferencias:



The Legend of Zelda™:
Echoes of Wisdom

| Nintendo Switch



The Legend of Zelda™:
Tears of the Kingdom

| Nintendo Switch



The Legend of Zelda™:
Breath of the Wild

| Nintendo Switch



The Legend of Zelda:
Breath of the Wild...

| Nintendo Switch



The Legend of Zelda™:
Skyward Sword HD

| Nintendo Switch



The Legend of Zelda™:
Link's Awakening

| Nintendo Switch

















Mi Carrito



Método de Pago: Tarjeta

Nº Tarjeta : XXXX-XXXX-XXXX-XXXX

Detalle del Pedido:

	Metroid Prime™ 4: Beyond Nintendo Switch	 	\$45.000
	Metroid Prime™ Remastered Nintendo Switch	 	\$29.000
	Metroid™ Dread Nintendo Switch	 	\$30.000
	Metroidvania Bundle Nintendo Switch	 	\$10.000

Total : \$114.000

Completar pedido



Mi Perfil



Mi Información

Mi Nombre :

Mi Apellido :

Mi Dirección :

Mi Correo:

Rol : Administrador

Ultimas Publicaciones

Administrar Publicaciones



Morbid: The Lords of Ire
17/05/24

\$15.427



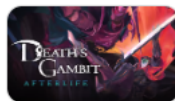
MARVEL ULTIMATE
ALLIANCE 3: The Black
Order
19/07/19

\$58.990



STAR WARSTM
Episode I Racer
12/05/22

\$3.777 ~~\$10.793~~ -65%



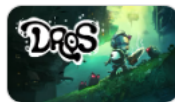
Death's Gambit:
Afterlife
30/09/21

\$7.916 ~~\$14.393~~ -45%



Mother Russia
Bleeds
29/08/23

\$1.600 ~~\$8.300~~ -80%



DROS
12/07/24

\$22.925

Pedidos por dar de alta :



Cliente 1



Cliente 3



Cliente 2



Administracion de Publicaciones

Nueva Publicación

Buscar Por :



Mario & Luigi™:
Brothership

| Nintendo Switch



Super Mario Bros.™
Wonder

| Nintendo Switch



Paper Mario™: The
Thousand-Year Door

| Nintendo Switch



Super Mario RPG™

| Nintendo Switch



Mario vs. Donkey Kong™

| Nintendo Switch



Super Mario Party™
Jamboree

| Nintendo Switch



Beat 'Em Up Archives
(QUByte Classics)

| Nintendo Switch



Beat Them Up - Street
Fight Band Simulator

| Nintendo Switch



Beat Souls

| Nintendo Switch



Sonar Beat

| Nintendo Switch



Last Beat Enhanced

| Nintendo Switch



Bullet Beat

| Nintendo Switch





Administracion de Publicaciones


Nueva Publicación de Juego

Título:

Descripción :

Precio :

Stock :

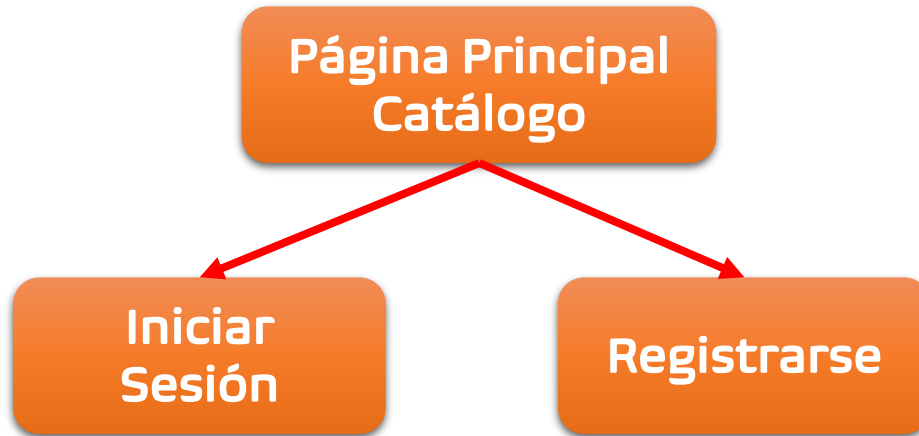
 

Url Imagen:

Navegación entre las vistas

Tienda de Juegos de Nintendo Switch
Games Switch Store

Vistas Públicas



Vistas Privadas



Listado de Dependencias

Tienda de Juegos de Nintendo Switch
Games Switch Store

Dependencias Frontend

Nombre	Detalle
React	Biblioteca de JavaScript para construir interfaces de usuario.
React Bootstrap	Componentes de Bootstrap para React.
React Dom	Métodos específicos del DOM para aplicaciones React.
React Router Dom	Enrutador para aplicaciones React en el navegador.
SweetAlert2	Biblioteca para mostrar alertas elegantes y responsivas.
Font Awesome	Conjunto de iconos para usar en la web.
Axios	Cliente HTTP basado en promesas para hacer peticiones a servidores.

Dependencias Backend

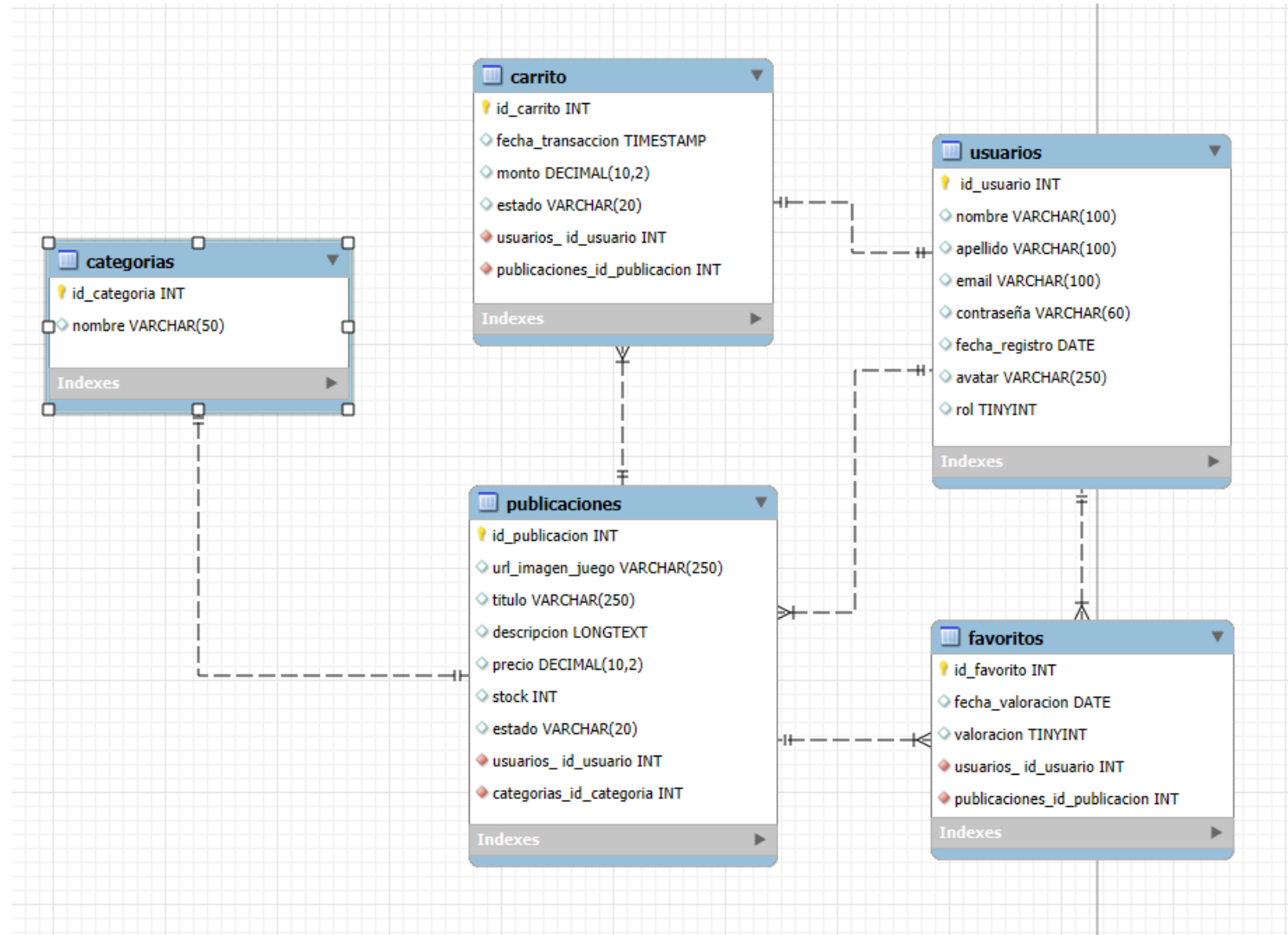
Nombre	Detalle
Node JS	Entorno de ejecución para JavaScript del lado del servidor.
Express	Framework web para Node.js.
Express-Validator	Middleware para validación de datos en Express.
JSON Web Token	Estándar para la creación de tokens de acceso.
Cors	Middleware para habilitar CORS (Cross-Origin Resource Sharing).
Dotenv	Cargar variables de entorno desde un archivo .env.
Logger-express	Middleware para logging en aplicaciones Express.
Bcrypt	Biblioteca para encriptar contraseñas.
Pg	Cliente de PostgreSQL para Node.js.
Pg-format	Biblioteca para formatear consultas SQL en PostgreSQL.

Modelo Base de Datos

Tienda de Juegos de Nintendo Switch

Games Switch Store

Modelo Base de datos



Script PostgreSQL

Base de datos

```
CREATE TABLE usuarios (  
    id_usuario SERIAL PRIMARY KEY,  
    nombre VARCHAR(100),  
    apellido VARCHAR(100),  
    email VARCHAR(100) UNIQUE,  
    contraseña VARCHAR(60),  
    fecha_registro DATE,  
    avatar VARCHAR(250),  
    rol SMALLINT CHECK (rol IN (0, 1)) -- 0 para cliente, 1 para administrador  
);
```

```
CREATE TABLE categorias (  
    id_categoria SERIAL PRIMARY KEY,  
    nombre VARCHAR(50)  
);
```

```
CREATE TABLE publicaciones (  
    id_publicacion SERIAL PRIMARY KEY,  
    id_usuario INT REFERENCES usuarios(id_usuario),  
    url_imagen_juego VARCHAR(250),  
    titulo VARCHAR(250),  
    descripcion TEXT,  
    precio DECIMAL(10, 2),  
    stock INT,  
    estado VARCHAR(10) CHECK (estado IN ('disponible', 'vendido')),  
    id_categoria INT REFERENCES categorias(id_categoria)  
);
```

```
CREATE TABLE carrito (  
    id_usuario INT PRIMARY KEY REFERENCES usuarios(id_usuario),  
    id_publicacion INT REFERENCES publicaciones(id_publicacion),  
    fecha_transaccion TIMESTAMPTZ,  
    monto DECIMAL(10, 2),  
    estado VARCHAR(20) CHECK (estado IN ('pendiente', 'completado', 'cancelado'))  
);
```

```
CREATE TABLE favoritos (  
    id_favorito SERIAL PRIMARY KEY,  
    id_usuario INT REFERENCES usuarios(id_usuario),  
    id_publicacion INT REFERENCES publicaciones(id_publicacion),  
    fecha_valoracion DATE,  
    valoracion BOOLEAN  
);
```

Contrato de Datos de API REST

Tienda de Juegos de Nintendo Switch
Games Switch Store

A. URL Base: <https://api.gameswitchstore.cl/v1>

B. Recursos

1. Usuarios

- o GET /usuarios: Obtiene la lista de todos los usuarios.
- o POST /usuarios: Crea un nuevo usuario.
- o GET /usuarios/{id}: Obtiene un usuario por ID.
- o PUT /usuarios/{id}: Actualiza un usuario existente.
- o DELETE /usuarios/{id}: Elimina un usuario por ID.

2. Categorías

- o GET /categorias: Obtiene la lista de todas las categorías.
- o POST /categorias: Crea una nueva categoría.
- o GET /categorias/{id}: Obtiene una categoría por ID.
- o PUT /categorias/{id}: Actualiza una categoría existente.
- o DELETE /categorias/{id}: Elimina una categoría por ID.

3. Publicaciones

- o GET /publicaciones: Obtiene la lista de todas las publicaciones.
- o POST /publicaciones: Crea una nueva publicación.
- o GET /publicaciones/{id}: Obtiene una publicación por ID.
- o PUT /publicaciones/{id}: Actualiza una publicación existente.
- o DELETE /publicaciones/{id}: Elimina una publicación por ID.

- 4. Carrito

- o GET /carrito: Obtiene la lista de todos los carritos.
- o POST /carrito: Crea un nuevo carrito.
- o GET /carrito/{id}: Obtiene un carrito por ID.
- o PUT /carrito/{id}: Actualiza un carrito existente.
- o DELETE /carrito/{id}: Elimina un carrito por ID.

5. Favoritos

- o GET /favoritos: Obtiene la lista de todos los favoritos.
- o POST /favoritos: Crea un nuevo favorito.
- o GET /favoritos/{id}: Obtiene un favorito por ID.
- o PUT /favoritos/{id}: Actualiza un favorito existente.
- o DELETE /favoritos/{id}: Elimina un favorito por ID.

C. Detalles de los Endpoints

1. Usuarios

GET /usuarios

Descripción: Retorna la lista de todos los usuarios.

Respuesta:

```
json
[
  {
    "id_usuario": 1,
    "nombre": "Juan Pérez",
    "email": "juan.perez@ejemplo.com",
    "contraseña": "hashedpassword",
    "fecha_registro": "2024-01-01",
    "avatar": "https://ejemplo.com/avatar.jpg",
    "rol": 1
  }
]
```

POST /usuarios

Descripción: Crea un nuevo usuario.

Solicitud:

```
json
{
  "nombre": "Juan Pérez",
  "email": "juan.perez@ejemplo.com",
  "contraseña": "contraseñaSegura",
  "avatar": "https://ejemplo.com/avatar.jpg",
  "rol": 1
}
```

Respuesta:

```
json
{
  "id_usuario": 1,
  "nombre": "Juan Pérez",
  "email": "juan.perez@ejemplo.com",
  "fecha_registro": "2024-01-01",
  "avatar": "https://ejemplo.com/avatar.jpg",
  "rol": 1
}
```


GET /usuarios/{id}

Descripción: Retorna un usuario por ID.

Respuesta:

```
json
{
  "id_usuario": 1,
  "nombre": "Juan Pérez",
  "email": "juan.perez@ejemplo.com",
  "fecha_registro": "2024-01-01",
  "avatar": "https://ejemplo.com/avatar.jpg",
  "rol": 1
}
```

PUT /usuarios/{id}

Descripción: Actualiza un usuario existente.

Solicitud:

```
json
{
  "nombre": "Juan Pérez",
  "email": "juan.perez@ejemplo.com",
  "contraseña": "nuevaContraseñaSegura",
  "avatar": "https://ejemplo.com/avatar.jpg",
  "rol": 1
}
```

Respuesta:

```
json
{
  "id_usuario": 1,
  "nombre": "Juan Pérez",
  "email": "juan.perez@ejemplo.com",
  "fecha_registro": "2024-01-01",
  "avatar": "https://ejemplo.com/avatar.jpg",
  "rol": 1
}
```

DELETE /usuarios/{id}

Descripción: Elimina un usuario por ID.

Respuesta:

```
json
{
  "message": "Usuario eliminado correctamente"
}
```

2. Categorías

GET /categorias

Descripción: Retorna la lista de todas las categorías.

Respuesta:

```
json
[
  {
    "id_categoria": 1,
    "nombre": "Juegos de Acción"
  }
]
```

POST /categorias

Descripción: Crea una nueva categoría.

Solicitud:

```
json
{
  "nombre": "Juegos de Acción"
```

Respuesta:

```
json
{
  "id_categoria": 1,
  "nombre": "Juegos de Acción"
```

GET /categorias/{id}

Descripción: Retorna una categoría por ID.

Respuesta:

```
json
{
  "id_categoria": 1,
  "nombre": "Juegos de Acción"
}
```


PUT /categorias/{id}

Descripción: Actualiza una categoría existente.

Solicitud:

```
json
{
  "nombre": "Juegos de Aventura"
}
```

Respuesta:

```
json
{
  "id_categoria": 1,
  "nombre": "Juegos de Aventura"
}
```

DELETE /categorias/{id}

Descripción: Elimina una categoría por ID.

Respuesta:

```
json
{
  "message": "Categoría eliminada correctamente"
}
```

3. Publicaciones

GET /publicaciones

Descripción: Retorna la lista de todas las publicaciones.

Respuesta:

```
json
[
  {
    "id_publicacion": 1,
    "url_imagen_juego": "https://ejemplo.com/imagen.jpg",
    "titulo": "Juego de Acción",
    "descripcion": "Descripción del juego de acción",
    "precio": 59.99,
    "stock": 10,
    "estado": "disponible",
    "usuarios_id_usuario": 1,
    "categorias_id_categoria": 1
  }
]
```

POST /publicaciones

Descripción: Crea una nueva publicación.

Solicitud:

```
json
{
  "url_imagen_juego": "https://ejemplo.com/imagen.jpg",
  "titulo": "Juego de Acción",
  "descripcion": "Descripción del juego de acción",
  "precio": 59.99,
  "stock": 10,
  "estado": "disponible",
  "usuarios_id_usuario": 1,
  "categorias_id_categoria": 1
}
```

Respuesta:

```
json
{
  "id_publicacion": 1,
  "url_imagen_juego": "https://ejemplo.com/imagen.jpg",
  "titulo": "Juego de Acción",
  "descripcion": "Descripción del juego de acción",
  "precio": 59.99,
  "stock": 10,
  "estado": "disponible",
  "usuarios_id_usuario": 1,
  "categorias_id_categoria": 1
}
```

GET /publicaciones/{id}

Descripción: Retorna una publicación por ID.

Respuesta:

```
json
{
  "id_publicacion": 1,
  "url_imagen_juego": "https://ejemplo.com/imagen.jpg",
  "titulo": "Juego de Acción",
  "descripcion": "Descripción del juego de acción",
  "precio": 59.99,
  "stock": 10,
  "estado": "disponible",
  "usuarios_id_usuario": 1,
  "categorias_id_categoria": 1
}
```

PUT /publicaciones/{id}

Descripción: Actualiza una publicación existente.

Solicitud:

```
json
{
  "url_imagen_juego": "https://ejemplo.com/imagen.jpg",
  "titulo": "Juego de Aventura",
  "descripcion": "Descripción del juego de aventura",
  "precio": 49.99,
  "stock": 5,
  "estado": "disponible",
  "usuarios_id_usuario": 1,
  "categorias_id_categoria": 1
}
```

Respuesta:

```
json
{
  "id_publicacion": 1,
  "url_imagen_juego": "https://ejemplo.com/imagen.jpg",
  "titulo": "Juego de Aventura",
  "descripcion": "Descripción del juego de aventura",
  "precio": 49.99,
  "stock": 5,
  "estado": "disponible",
  "usuarios_id_usuario": 1,
  "categorias_id_categoria": 1
}
```

DELETE /publicaciones/{id}

Descripción: Elimina una publicación por ID.

Respuesta:

```
json
{
  "message": "Publicación eliminada correctamente"
}
```

4. Carrito

GET /carrito

Descripción: Retorna la lista de todos los carritos.

Respuesta:

```
json
[
  {
    "id_carrito": 1,
    "fecha_transaccion": "2024-08-04T12:00:00Z",
    "monto": 100.00,
    "estado": "completado",
    "usuarios_id_usuario": 1,
    "publicaciones_id_publicacion": 1
  }
]
```


POST /carrito

Descripción: Crea un nuevo carrito.

Solicitud:

```
json
{
  "monto": 100.00,
  "estado": "pendiente",
  "usuarios_id_usuario": 1,
  "publicaciones_id_publicacion": 1
}
```

Respuesta:

```
json
{
  "id_carrito": 1,
  "fecha_transaccion": "2024-08-04T12:00:00Z",
  "monto": 100.00,
  "estado": "pendiente",
  "usuarios_id_usuario": 1,
  "publicaciones_id_publicacion": 1
}
```

GET /carrito/{id}

Descripción: Retorna un carrito por ID.

Respuesta:

```
json
{
  "id_carrito": 1,
  "fecha_transaccion": "2024-08-04T12:00:00Z",
  "monto": 100.00,
  "estado": "completado",
  "usuarios_id_usuario": 1,
  "publicaciones_id_publicacion": 1
}
```

PUT /carrito/{id}

Descripción: Actualiza un carrito existente.

Solicitud:

```
json
{
  "monto": 150.00,
  "estado": "completado",
  "usuarios_id_usuario": 1,
  "publicaciones_id_publicacion": 1
}
```

Respuesta:

```
json
{
  "id_carrito": 1,
  "fecha_transaccion": "2024-08-04T12:00:00Z",
  "monto": 150.00,
  "estado": "completado",
  "usuarios_id_usuario": 1,
  "publicaciones_id_publicacion": 1
}
```

DELETE /carrito/{id}

Descripción: Elimina un carrito por ID.

Respuesta:

```
json
{
  "message": "Carrito eliminado correctamente"
}
```

5. Favoritos

GET /favoritos

Descripción: Retorna la lista de todos los favoritos.

Respuesta:

```
json
[
  {
    "id_favorito": 1,
    "fecha_valoracion": "2024-08-04",
    "valoracion": 5,
    "usuarios_id_usuario": 1,
    "publicaciones_id_publicacion": 1
  }
]
```

POST /favoritos

Descripción: Crea un nuevo favorito.

Solicitud:

```
json
{
  "fecha_valoracion": "2024-08-04",
  "valoracion": 5,
  "usuarios_id_usuario": 1,
  "publicaciones_id_publicacion": 1
}
```

Respuesta:

```
json
{
  "id_favorito": 1,
  "fecha_valoracion": "2024-08-04",
  "valoracion": 5,
  "usuarios_id_usuario": 1,
  "publicaciones_id_publicacion": 1
}
```

GET /favoritos/{id}

Descripción: Retorna un favorito por ID.

Respuesta:

```
json
{
  "id_favorito": 1,
  "fecha_valoracion": "2024-08-04",
  "valoracion": 5,
  "usuarios_id_usuario": 1,
  "publicaciones_id_publicacion": 1
}
```

PUT /favoritos/{id}

Descripción: Actualiza un favorito existente.

Solicitud:

```
json
{
  "fecha_valoracion": "2024-08-04",
  "valoracion": 4,
  "usuarios_id_usuario": 1,
  "publicaciones_id_publicacion": 1
}
```

Respuesta:

```
json
{
  "id_favorito": 1,
  "fecha_valoracion": "2024-08-04",
  "valoracion": 4,
  "usuarios_id_usuario": 1,
  "publicaciones_id_publicacion": 1
}
```


DELETE /favoritos/{id}

Descripción: Elimina un favorito por ID.

Respuesta:

```
json
{
  "message": "Favorito eliminado correctamente"
}
```