

CursosApp

Documentación de la API REST

Plataforma de Cursos Online — Backend .NET 8 + PostgreSQL

1. Introducción

Esta API REST proporciona el backend completo para la plataforma CursosApp, una aplicación de cursos online desarrollada con .NET 8 y PostgreSQL. La API gestiona la autenticación de usuarios, el catálogo de cursos y categorías, las matrículas de alumnos y el dashboard de estadísticas.

URL base de la API:

```
http://localhost:8080
```

Documentación interactiva (Swagger):

```
http://localhost:8080/swagger
```

2. Autenticación

La API utiliza JWT (JSON Web Tokens) para autenticar las peticiones. Para acceder a los endpoints protegidos es necesario:

- Obtener un token mediante POST /api/auth/login
- Incluir el token en la cabecera Authorization de cada petición

Formato de la cabecera:

```
Authorization: Bearer {token}
```

2.1 Roles disponibles

Existen tres roles con distintos permisos:

- admin — Acceso total: puede gestionar cursos, categorías y usuarios
- instructor — Puede crear y editar cursos, ver el dashboard
- student — Puede consultar cursos públicos, matricularse y ver su progreso

2.2 Usuarios de prueba

Rol	Email	Contraseña
admin	admin@cursosapp.com	Admin1234!
instructor	instructor@cursosapp.com	Admin1234!
student	student@cursosapp.com	Admin1234!

3. Endpoints de Autenticación

Gestiona el registro, login y perfil de los usuarios de la aplicación.

Método	Endpoint	Auth	Descripción
POST	/api/auth/login	No	Autentica al usuario con email y contraseña. Devuelve un token JWT con los datos del usuario.
POST	/api/auth/register	No	Registra un nuevo usuario. El rol asignado siempre es 'student'. Devuelve token JWT.
GET	/api/auth/me	JWT	Devuelve el perfil del usuario que está actualmente autenticado.

POST /api/auth/login

Cuerpo de la petición:

```
{ "email": "admin@cursosapp.com", "password": "Admin1234!" }
```

Respuesta exitosa (200):

```
{ "token": "eyJ...", "role": "admin", "name": "Admin Principal", "email": "admin@cursosapp.com", "userId": 1 }
```

POST /api/auth/register

Cuerpo de la petición:

```
{ "name": "Nuevo Usuario", "email": "nuevo@email.com", "password": "MiPassword1!" }
```

4. Endpoints de Cursos

Gestiona el catálogo completo de cursos de la plataforma. Los endpoints de consulta son públicos, los de modificación requieren rol de admin o instructor.

Método	Endpoint	Auth	Descripción
GET	/api/courses	Opcional	Busca y lista cursos con filtros, paginación y ordenación. Los usuarios no autenticados solo ven cursos publicados.
GET	/api/courses/{id}	Opcional	Devuelve el detalle completo de un curso. Si el usuario está autenticado, indica si está matriculado y su progreso.
POST	/api/courses	admin/instructor	Crea un nuevo curso. El curso se crea sin publicar por defecto.
PUT	/api/courses/{id}	admin/instructor	Actualiza todos los datos de un curso existente, incluyendo si está publicado o no.
PATCH	/api/courses/{id}/publish	admin	Alterna el estado de publicación de un curso (publicado ↔ no publicado).
DELETE	/api/courses/{id}	admin	Elimina un curso permanentemente de la base de datos.
POST	/api/courses/{id}/enroll	JWT	Matricula al usuario autenticado en el curso especificado. No se puede matricular dos veces en el mismo curso.

Parámetros de búsqueda — GET /api/courses

El endpoint de búsqueda acepta los siguientes parámetros en la URL:

Parámetro	Ejemplo	Descripción
query	vue	Texto libre: busca en título, descripción e instructor
categoryId	1	Filtra por ID de categoría
level	Beginner	Nivel del curso: Beginner, Intermediate, Advanced
minPrice	0	Precio mínimo
maxPrice	100	Precio máximo
dateFrom	2024-01-01	Fecha de creación desde (formato ISO)
dateTo	2024-12-31	Fecha de creación hasta (formato ISO)
isPublished	true	Solo admin/instructor: filtra por estado de publicación
sortBy	rating	Criterio de ordenación: createdAt, price, rating, enrollments, title
sortOrder	desc	Dirección: asc o desc

Parámetro	Ejemplo	Descripción
page	1	Número de página (empieza en 1)
pageSize	10	Resultados por página (por defecto 10)

Ejemplo de URL completa:

```
GET
/api/courses?query=vue&level=Beginner&sortBy=rating&sortOrder=desc&page=1&pageSize=5
```

Estructura de respuesta paginada

Todos los listados devuelven el siguiente formato:

```
{ "items": [...], "totalCount": 42, "page": 1, "pageSize": 10, "totalPages": 5 }
```

5. Endpoints de Categorías

Gestiona las categorías a las que pertenecen los cursos. Los endpoints de lectura son públicos y no requieren autenticación.

Método	Endpoint	Auth	Descripción
GET	/api/categories	No	Lista todas las categorías con el número de cursos publicados que tiene cada una.
GET	/api/categories/{id}	No	Devuelve el detalle de una categoría específica.
POST	/api/categories	admin	Crea una nueva categoría con nombre, descripción e icono.
PUT	/api/categories/{id}	admin	Actualiza los datos de una categoría existente, incluido su estado activo/inactivo.
DELETE	/api/categories/{id}	admin	Elimina una categoría. No se puede eliminar si tiene cursos asociados.

POST /api/categories — Cuerpo de la petición

```
{ "name": "Programación", "description": "Cursos de desarrollo de software", "icon": "📖" }
```

6. Endpoints de Dashboard

Proporciona los datos estadísticos para el panel de administración. Requiere rol admin o instructor.

Método	Endpoint	Auth	Descripción
GET	/api/dashboard	admin/instructor	Devuelve KPIs generales y datos para 4 gráficas distintas. Acepta filtros de fecha opcionales.

Parámetros opcionales

Parámetro	Ejemplo	Descripción
dateFrom	2024-01-01	Inicio del periodo para los datos de la gráfica de ingresos por mes
dateTo	2024-12-31	Fin del periodo para los datos de la gráfica de ingresos por mes

Estructura de la respuesta

El endpoint devuelve un objeto con dos secciones principales:

KPIs (indicadores clave):

- totalCourses — Total de cursos en la plataforma
- publishedCourses — Cursos publicados y visibles
- totalCategories — Número de categorías activas
- totalUsers — Usuarios activos registrados
- totalEnrollments — Total de matrículas realizadas
- totalRevenue — Ingresos estimados (precio × matrículas)
- averageRating — Valoración media de los cursos publicados
- averageDurationHours — Duración media de los cursos en horas

Datos de gráficas:

- enrollmentsByCategory — Número de matrículas agrupadas por categoría (gráfica de barras o tarta)
- coursesByLevel — Distribución de cursos por nivel: Beginner, Intermediate, Advanced (gráfica de tarta)
- revenueByMonth — Ingresos estimados mes a mes en el periodo seleccionado (gráfica de líneas)
- topCoursesByRating — Los 5 cursos mejor valorados (gráfica de barras horizontal)

7. Endpoints de Usuarios

Gestiona los usuarios de la plataforma. La administración de usuarios requiere rol admin. Los endpoints de perfil y progreso son accesibles para cualquier usuario autenticado.

Método	Endpoint	Auth	Descripción
GET	/api/users	admin	Lista todos los usuarios con su rol, estado y número de matrículas.
POST	/api/users	admin	Crea un nuevo usuario desde el panel de administración. Permite asignar cualquier rol.
PATCH	/api/users/{id}	admin	Actualiza el rol o el estado activo/inactivo de un usuario.
DELETE	/api/users/{id}	admin	Elimina un usuario permanentemente.
GET	/api/users/me/enrollments	JWT	Devuelve la lista de cursos en los que está matriculado el usuario autenticado, con su progreso.
POST	/api/users/me/progress	JWT	Actualiza el porcentaje de progreso del usuario autenticado en un curso. Si llega al 100%, marca el curso como completado.

POST /api/users — Cuerpo de la petición

```
{ "name": "Nuevo Instructor", "email": "instructor2@email.com", "password": "Pass1234!", "role": "instructor" }
```

PATCH /api/users/{id} — Cuerpo de la petición

```
{ "role": "instructor", "isActive": true }
```

Ambos campos son opcionales. Se puede enviar solo uno de ellos.

POST /api/users/me/progress — Cuerpo de la petición

```
{ "courseId": 1, "progressPercent": 75 }
```

El valor de progressPercent debe estar entre 0 y 100. Al llegar a 100 se marca automáticamente como completado.

8. Códigos de Respuesta HTTP

Código	Significado
200 OK	La petición se ha procesado correctamente y se devuelven datos
201 Created	El recurso se ha creado correctamente
204 No Content	La operación se ha realizado correctamente pero no hay datos que devolver (ej: DELETE)
400 Bad Request	Los datos enviados son incorrectos o incompletos

Código	Significado
401 Unauthorized	No se ha proporcionado token o el token es inválido/expirado
403 Forbidden	El usuario está autenticado pero no tiene el rol necesario
404 Not Found	El recurso solicitado no existe
409 Conflict	El recurso ya existe (ej: email duplicado) o hay un conflicto de estado
500 Internal Server Error	Error interno del servidor

9. Conexión desde Vue 3

Para conectar el frontend Vue 3 con esta API, configura la URL base en el archivo de entorno:

Archivo .env:

```
VITE_API_URL=http://localhost:8080
```

Ejemplo de llamada al login con axios:

```
const response = await
axios.post(`${import.meta.env.VITE_API_URL}/api/auth/login`, { email, password
});
const token = response.data.token;
```

Incluir el token en las peticiones protegidas:

```
axios.defaults.headers.common['Authorization'] = `Bearer ${token}`;
```

El token JWT caduca a las 24 horas. Al caducar, el servidor devuelve 401 y el usuario debe volver a hacer login.