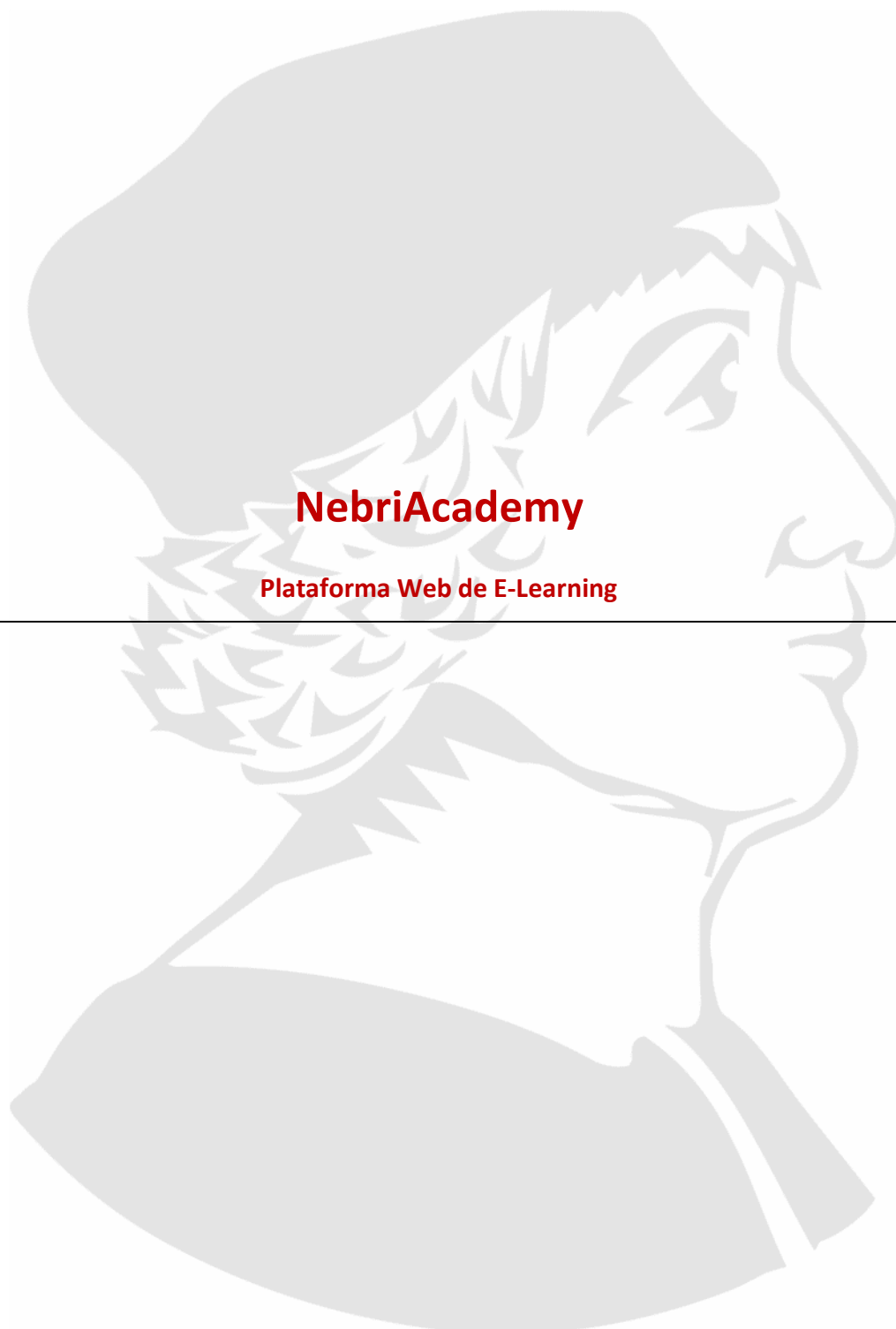


PORTADA



Nombres del equipo: Nicolás García-Sampedro Docampo, Álvaro Santos Calvo y Chao

An Alarcón Chen

Curso académico: 2º DAW

Tutora/Tutor del proyecto: Rubén Gonzalez Martín

ÍNDICE PAGINADO

JUSTIFICACIÓN DEL PROYECTO.....	2
INTRODUCCIÓN.....	3
OBJETIVOS.....	3
• OBJETIVO GENERAL.....	3
• OBJETIVOS ESPECÍFICOS.....	3
DESARROLLO	4
1. FUNDAMENTACIÓN TEÓRICA:.....	4
2. MATERIALES Y MÉTODOS:.....	5
3. RESULTADOS Y ANÁLISIS.....	6
CONCLUSIONES	6
LÍNEAS DE INVESTIGACIÓN FUTURAS	6
BIBLIOGRAFÍA.....	6
ANEXOS.....	7
OTROS PUNTOS	10

JUSTIFICACIÓN DEL PROYECTO

El proyecto **NebriAcademy** se concibe como una plataforma de E-learning, inspirada en servicios como Coursera o Udemy, orientada a la gestión de contenidos educativos, usuarios y relaciones académicas. En esta fase del proyecto, el desarrollo se ha centrado exclusivamente en la implementación del backend, dejando el frontend para futuras etapas.

El objetivo principal de este backend es proporcionar una API REST sólida y escalable que sirva como base para una plataforma educativa online, permitiendo la consulta estructurada de información relacionada con cursos, alumnos, profesores, materiales y evaluaciones.

Este proyecto permite aplicar de forma práctica los conocimientos adquiridos en el desarrollo de aplicaciones web del lado servidor, utilizando Node.js y Express, así como una correcta organización del código mediante una arquitectura modular.

INTRODUCCIÓN

NebriAcademy API es una aplicación backend desarrollada con Node.js y Express.js, cuyo objetivo es servir como núcleo funcional de una plataforma de E-learning. Actualmente, la API permite consultar información académica en formato JSON mediante diferentes endpoints accesibles por URL.

En esta etapa del desarrollo, la API se centra en la lectura de datos (GET), permitiendo obtener listados completos o registros individuales de las distintas entidades del sistema. La arquitectura del proyecto está preparada para futuras ampliaciones que incluyan autenticación, autorización y operaciones CRUD completas.

La API dispone de una página principal informativa desde la cual se explica cómo acceder a los diferentes recursos disponibles.

Para consultar los datos:

- Para obtener todos los registros de una tabla:
`http://localhost:3000/nombreTabla`
- Para obtener un registro concreto por ID:
`http://localhost:3000/nombreTabla/id`

OBJETIVOS

• OBJETIVO GENERAL

Desarrollar el backend de una plataforma de E-learning mediante una API REST que permita gestionar y exponer la información académica necesaria para una aplicación similar a Coursera o Udemy.

• OBJETIVOS ESPECÍFICOS

- Implementar una API REST con Node.js y Express.js.
- Organizar el proyecto mediante una estructura modular.
- Crear modelos para cada entidad de la base de datos.
- Definir rutas específicas para cada recurso académico.
- Devolver la información en formato JSON.
- Facilitar la integración futura con aplicaciones frontend.
- Aplicar buenas prácticas en la estructuración del backend.

DESARROLLO

1. FUNDAMENTACIÓN TEÓRICA:

El backend del proyecto se fundamenta en el uso de APIs REST, siguiendo principios de simplicidad, escalabilidad y separación de responsabilidades. Se emplea una arquitectura modular basada en Node.js y Express, donde cada recurso académico dispone de su propio modelo y conjunto de rutas.

En esta fase del proyecto, la API se limita a la consulta de información, utilizando endpoints GET que devuelven respuestas en formato JSON, lo que facilita su consumo por aplicaciones frontend o clientes externos.

MODELOS:

Los modelos representan las entidades principales del sistema académico y se encargan de interactuar con la base de datos:

- Administradores
- Alumnos
- Apuntes
- Cursos
- CursosAlumnos
- Ejercicios
- Incidencias
- Profesores
- ProfesoresCursos
- PuntuacionesEjercicios
- Usuarios
- Videos

RUTAS:

Las rutas gestionan las peticiones y devuelven las respuestas en formato JSON. Cada entidad cuenta con su propio archivo de rutas:

/administradores

/alumnos

/apuntes

/cursos

/cursosalumnos

/ejercicios

/incidencias
/index
/profesores
/profesoresrecursos
/puntuacionesejercicios
/usuarios
/videos

ESTRUCTURA DEL PROYECTO:

El proyecto sigue una estructura clara y organizada:

src/database → Conexión a la base de datos
src/models → Modelos de datos
src/routes → Definición de endpoints
app.js → Archivo principal de la aplicación

Esta estructura favorece la escalabilidad y el mantenimiento del código.

2. MATERIALES Y MÉTODOS:

Tecnologías utilizadas:

- Node.js → Entorno de ejecución
- Express.js → Framework backend
- SQL → Base de datos relacional
- JavaScript → Lenguaje principal
- JSON → Formato de intercambio de datos

Metodología de trabajo:

- Desarrollo modular.
- Pruebas constantes de los endpoints.
- Organización clara de rutas y modelos.
- Uso de herramientas de control de versiones.

3. RESULTADOS Y ANÁLISIS

Como resultado del proyecto se ha obtenido:

- Backend funcional para una plataforma de E-learning.
- API REST estructurada y modular.
- Endpoints operativos para la consulta de datos académicos.
- Respuestas JSON claras y consistentes.
- Base técnica preparada para la futura integración de un frontend.

La API permite simular el funcionamiento interno de plataformas educativas online, gestionando relaciones entre cursos, usuarios, profesores y recursos.

CONCLUSIONES

El desarrollo del backend de NebriAcademy ha permitido sentar las bases técnicas de una plataforma de E-learning similar a Coursera o Udemy. En esta fase inicial se ha logrado construir una API REST organizada y funcional, capaz de exponer información académica de forma estructurada.

El proyecto ha reforzado conocimientos clave sobre Node.js, Express y el diseño de APIs REST, así como la importancia de una correcta separación entre modelos y rutas. Aunque actualmente la API se limita a operaciones de consulta, su diseño permite ampliar fácilmente la funcionalidad en futuras fases del desarrollo.

Este backend constituye una base sólida sobre la que implementar autenticación, gestión de usuarios y un frontend completo.

LÍNEAS DE INVESTIGACIÓN FUTURAS

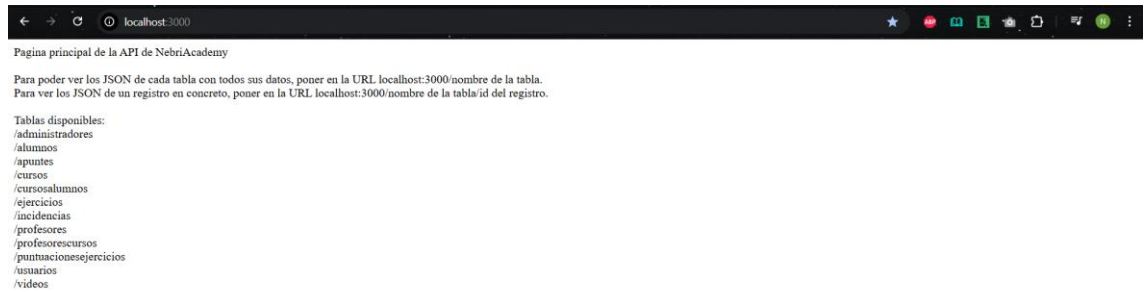
- Implementar operaciones CRUD completas (POST, PUT, DELETE).
- Incorporar validación de datos.
- Añadir paginación y filtros avanzados.

BIBLIOGRAFÍA

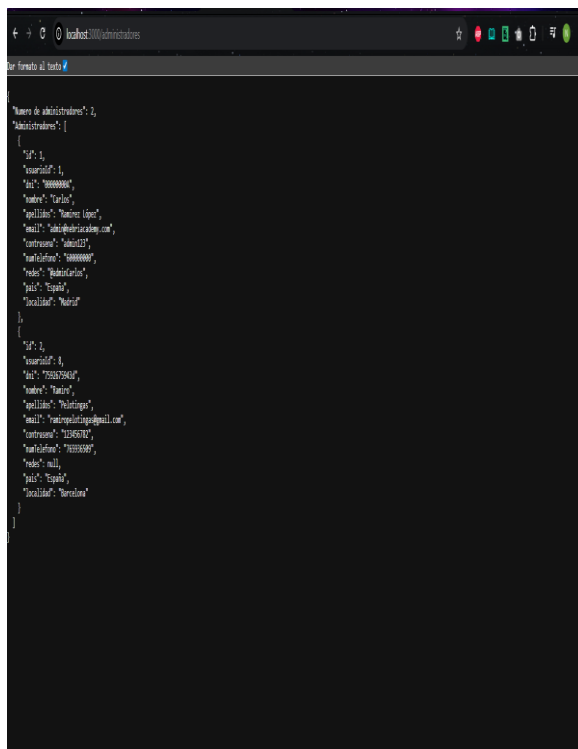
- Documentación oficial de Node.js ⇒ [Index | Node.js v25.2.1 Documentation](#)
- Documentación de Express.js ⇒ [Express routing](#)

ANEXOS

- **Página principal:**



- **Tabla Administradores:**

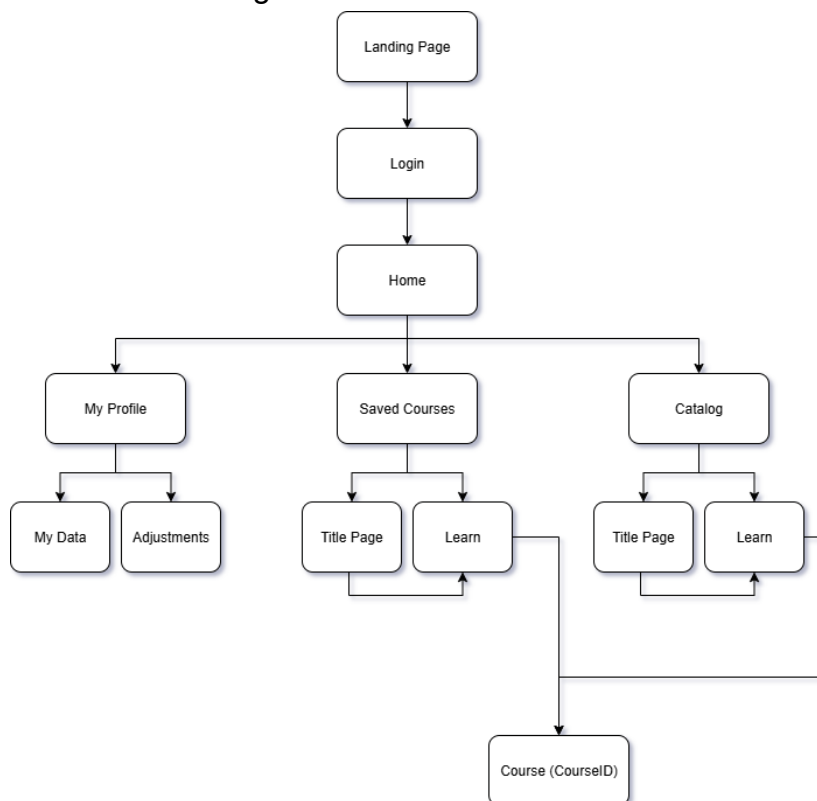


- Tabla Administradores filtrada por id:

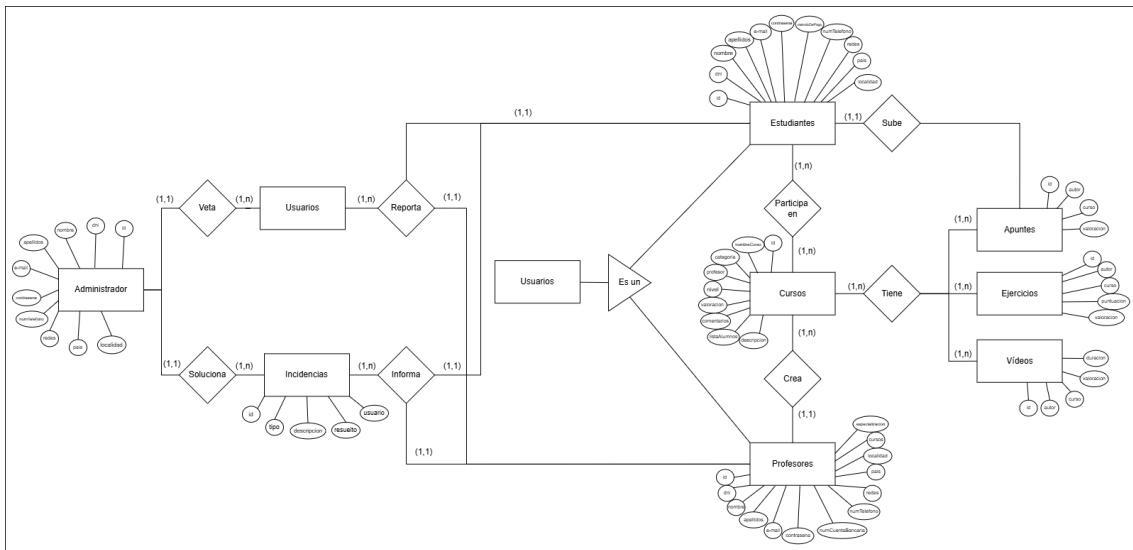
```

localhost:3000/administradores/1
Dar formato al texto
{
  "id": 1,
  "usuarioId": 1,
  "dni": "99999999A",
  "nombre": "Carlos",
  "apellidos": "Ramirez López",
  "email": "admin@nebracademy.com",
  "contrasena": "admin123",
  "numTelefono": "600000000",
  "roles": "AdminCarlos",
  "pais": "España",
  "localidad": "Madrid"
}
    
```

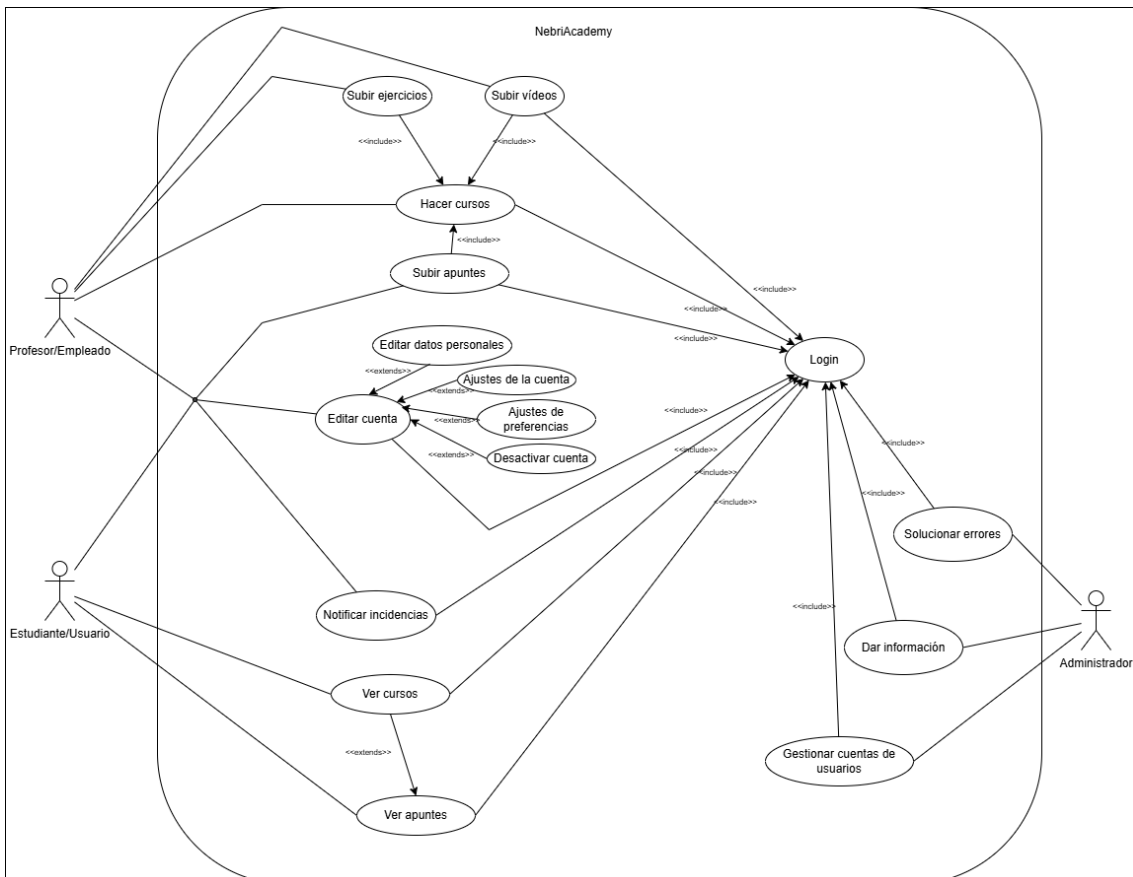
- Primer diagrama:



- Diagrama ER



- Diagrama de casos de uso:



OTROS PUNTOS

Agradecimientos:

Queríamos agradecer a nuestros profesores Rubén Gonzalez Martín y Francisco Albiar Jiménez por aportarnos los conocimientos necesarios para este proyecto y habernos guiado durante todo el proceso de creación de este.