Estructura de proyecto Back-End con Node.js (Arquitectura de Microservicios)

Este documento muestra la estructura recomendada de carpetas y archivos para un proyecto Back-End con Node.js usando una arquitectura de microservicios. Cada servicio es independiente y se comunica mediante API REST o mensajería.

Esquema de carpetas y archivos (Microservicios)

```
Back-End/
■   docker-compose.yml   (orquestación de microservicios)
■   .env
■   .gitignore
■
■■■ api-gateway/         (punto de entrada único para clientes)
■   ■■■ index.js
■   ■■■ routes/
■   ■■■ middlewares/
■
■■■ services/
■   ■■■ usuarios/
■   ■   ■■■ index.js
■   ■   ■■■ controllers/
■   ■   ■■■ routes/
■   ■   ■■■ models/
■   ■   ■■■ helpers/
■   ■   ■■■ validaciones/
■   ■
■   ■■■ profesores/
■   ■   ■■■ index.js
■   ■   ■■■ controllers/
■   ■   ■■■ routes/
■   ■   ■■■ models/
■   ■   ■■■ validaciones/
■   ■
■   ■■■ alumnos/
■   ■   ■■■ index.js
■   ■   ■■■ controllers/
■   ■   ■■■ routes/
■   ■   ■■■ models/
■   ■   ■■■ validaciones/
■   ■
■   ■■■ notificaciones/
■       ■■■ index.js
■       ■■■ services/ (ej. correo, WhatsApp)
■       ■■■ helpers/
■
■■■ shared/              (código compartido entre servicios)
■   ■■■ utils/
■   ■■■ middlewares/
■
■■■ node_modules/        (por servicio, si no se usa monorepo)
```

Pasos básicos de instalación

1. Crear la carpeta Back-End como contenedor del proyecto.
2. Crear un directorio por cada microservicio dentro de /services (ej: usuarios, alumnos, profesores, notificaciones).
3. Dentro de cada microservicio: ejecutar npm init -y para inicializar package.json.
4. Instalar dependencias por servicio: npm i express dotenv cors helmet bcrypt body-parser.
5. Instalar mongoose o mysql2 según la base de datos elegida por cada microservicio.
6. Crear el servicio api-gateway que centraliza las peticiones del cliente.
7. Configurar docker-compose.yml para levantar todos los servicios y la base de datos.
8. Usar variables de entorno (.env) separadas por servicio.

9. Implementar comunicación entre servicios: REST (http) o mensajería (ej. RabbitMQ, Kafka).
10. Opcional: usar un repositorio monorepo (ej: con Nx o Lerna) para centralizar dependencias.