Trabajo Práctico: Desarrollo y Deploy de una API REST en TypeScript

Descripción general

A partir del código base proporcionado en los siguientes repositorios:

- Frontend: `https://github.com/GabrielAlberini/frontend-utn`
- Backend: `https://github.com/GabrielAlberini/backend_utn`

El objetivo del trabajo es completar el desarrollo funcional del backend, implementando una API REST robusta, tipada en TypeScript, con arquitectura MVC, validaciones, autenticación y deploy final en Render.com.

Los estudiantes deberán garantizar que el frontend suministrado pueda consumir correctamente la API una vez finalizados los desarrollos.

Objetivos

1. Implementar una API REST funcional en TypeScript siguiendo buenas prácticas.
2. Ejecutar correctamente la aplicación en entornos local y producción.
3. Realizar el deploy del backend en Render.com.
4. Incorporar seguridad, validaciones, logs y estructura profesional de proyecto.

Requerimientos obligatorios

1. Estructura del proyecto

- Implementación completa del patrón MVC en el backend.
- Organización clara de controladores, servicios, modelos, rutas y middlewares.
- Código tipado completamente en TypeScript.

2. Scripts de ejecución

Se deberán agregar en el `package.json` todos los scripts necesarios para:

- Ejecutar el proyecto en desarrollo (TypeScript)
- Compilar a JavaScript (producción)
- Ejecutar en producción (JavaScript compilado)

Ejemplo esperado:

- `dev`
- `build`
- `start`

3. Logger

- Incorporar `morgan` para registrar todas las solicitudes entrantes.
- Registrar método, ruta y status code.

4. Rate limit

- Implementar rate limit únicamente en las rutas de autenticación.

5. Autenticación y autorización

- Crear un `authMiddleware` que restrinja agregar, actualizar y eliminar productos.
- Solo usuarios autenticados pueden realizar operaciones de escritura.

6. Query params

- Deben permitir filtrar la data total de productos desde la base de datos.
- La lógica del filtrado debe ejecutarse en la consulta a la base, no en el backend.

Ejemplo:

- Filtrar por categoría
- Filtrar por precio mínimo y máximo
- Filtrar por nombre (búsqueda parcial)

7. Validación de inputs

- Validar todos los datos que el cliente envía al backend.
- Completar validaciones faltantes respecto al código actual.
- Manejar errores y responder consistentemente.

8. Variables de entorno

- Correcto uso del archivo `.env`.
- No exponer claves ni cadenas de conexión en el repositorio.
- Configurar `dotenv` en el backend.

9. Deploy

- Realizar el deploy del backend en `Render.com`.
- Proveer URL pública de la API funcionando.

Requerimientos opcionales (para sumar valoración extra)

A. Envío de correos

- Implementar integración con `nodemailer` para enviar un correo ante alguna acción de negocio (por ejemplo: creación de usuario).

B. Subida de archivos

- Implementar carga de imágenes de productos utilizando `multer`.
- Guardar los archivos en el servidor o en un proveedor externo (a elección).

Forma de entrega

Los estudiantes deben entregar:

1. **URL online del backend** desplegado en Render, funcionando correctamente.
2. **Repositorio actualizado del backend**, con:
    - Código ordenado, funcional, con commits claros.
    - Archivo README con instrucciones de uso, instalación y endpoints.
3. **Instrucciones de instalación y ejecución local**, incluyendo scripts de desarrollo y producción.
4. **Capturas de Postman o Bruno** demostrando el funcionamiento de los endpoints obligatorios.
5. **Archivo `.env.example`** con las variables esperadas.
6. Video de 2 a 5 minutos mostrando el funcionamiento general del proyecto.