ThePower business school

Proyecto MERN 7

Enunciado de proyecto

¡Has llegado al último proyecto para convertirte en **Backend Developer de Rock{theCode}**! Ahora tendrás que aplicar todo lo aprendido a lo largo del módulo para demostrar que puedes crear una API completa ya resiliente..

- Para preparar el proyecto, repite lo que hiciste en los proyecto anteriores del mismo módulo.
- En este proyecto, tendrás que limitar tu API con `rate-limiter` para que no se puedan hacer más de 50 peticiones en 3 minutos.
- Activarás CORS en tu servidor, para permitir acceso desde cualquier dominio..
- Habrá protección de rutas por medio de JWTs, así que no olvides repasar autenticación.
- Utilizarás subida de imágenes, no olvides registrarte en Cloudinary previamente y usar tu Token.
- Desplegarás tu base de datos MongoDB en Mongo Atlas y tu servidor el Render (preferiblemente) o en Fly. Para esto, no olvides ver los videos correspondientes que se encuentran en la lección de**: CI/CD** ya que cubren este proceso, al final de este modulo

En este proyecto, crearás nuevamente una API que incluya todo lo del proyecto anterior, es decir, tenga dos modelos distintos de datos relacionados entre si. Adicionalmente, crearás un modelo `User` que te permitirá tener guardados usuarios registrados que podrán hacer **Login** en tu aplicación.

Protegerás los otros dos modelos de datos por medio de un middleware y JWT. De forma que un usuario **no logeado** no pueda crear, editar o eliminar elementos, pero si leerlos.

También deberías crear dos endpoints, uno para añadir un campo `avatar` a los documentos del modelo `User` subiendo una imagen a Cloudinary, y otro para hacer lo mismo y añadir una imagen a uno de los otros dos modelos que deben estar relacionados entre si.

Criterio de aceptación

Estos criterios son generales y estarán referidos al servidor y base de datos. Cumplirlos es obligatorio para considerar el proyecto completado y que podamos así certificarte como Backend Developer una vez acabez todos los proyectos.

- [ ] **El servidor y la base de datos deben estar desplegados**, no necesitamos que corran en ningún puerto en específico ya que debes proveernos de una URL para probar el servidor.
- [ ] Tendremos acceso al código en Github, asegúrate de nos subir variables de entorno a la plataforma.
- [ ] Debes tener un modelo `User` que cuente con endpoints para:
  - [ ] Un enpoint `/auth/register` que permita registrar un nuevo usuario. El email no puede estar repetido y la contraseña debe tener al menos 6 caracteres, una minúscula y una mayúscula.
  - [ ] Un endpoint `/auth/login` que permita logearse con un usuario ya registrado y devuelva un JWT.
  - [ ] El **JWT** debe tener una caducidad de una hora.
- [ ] Deben existir dos modelos relacionados entre si, que cumplan por completo con los requerimientos del **proyecto 8:**
  - [ ] Estos documentos deben poder leerse (endpoints de tipo GET) sin estar autenticado, debe ser pública la información.
  - [ ] Estos documentos solo podrán trabajarse (POST, PUT, DELETE) si el **usuario está autenticado.** En caso contrario, devolverán un status 401 y un mensaje adecuado.
- [ ] La API debe estar limitada a 50 peticiones cada 3 minutos.

- [ ] Existirá un endpoint `/auth/avatar` de tipo POST que permita añadir un avatar a un usuario que obtendrás por medio de un JWT válido. De forma que el usuario que hace la petición pueda subir una imagen a través de Thunder Client y subirla a Cloudinary, editando su documento correspondiente en MongoDB.
- [ ] Existirá un endpoint para uno de los modelos relacionados entre si que permita, dada su id, subir una imagen para dicho documento en Cloudinary y edite el documento con un nuevo campo `image: 'url_de_la_imagen'`.