EXAMEN 3

Miguel Jurado Vázquez

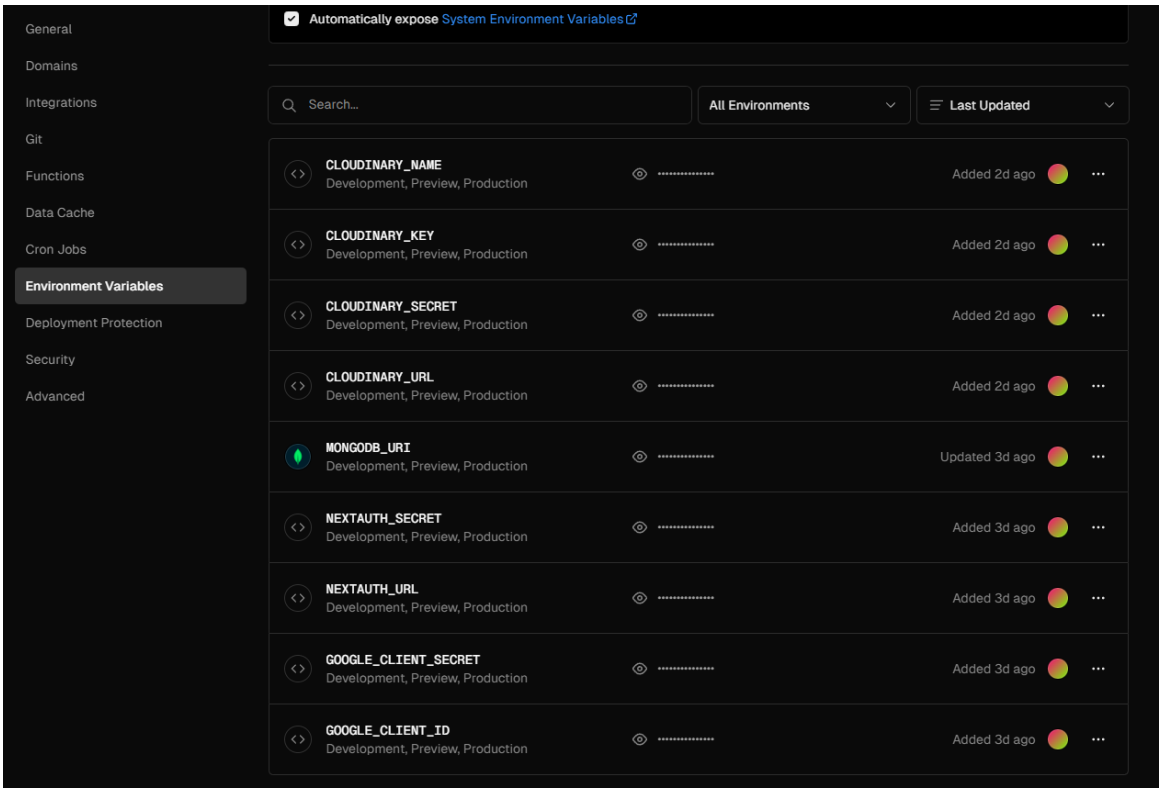
Instrucciones - local

1. Descomprimir proyecto y asegurarse que se encuentran las variables en .env
2. Instalación de Dependencias: ejecuta el siguiente comandos para instalar las dependencias: *npm install*
3. Ejecuta el siguiente comando para iniciar el servidor en modo desarrollo: *npm run dev.* El proyecto se desplegará en http://localhost:3000/

Se adjunta proyecto comprimido para poderlo correr en local. El archivo .env se encuentra adjunto en el comprimido, además lo introduzco aquí:

```
MONGODB_URI=mongodb+srv://mxke17:exam3@exam3.ko51hys.mongodb.net/
GOOGLE_CLIENT_ID=444015143951-bs4449st7ubvfl172jcqtrnivdrapjg9.apps.googleusercontent.com
GOOGLE_CLIENT_SECRET=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
NEXTAUTH_URL=http://localhost:3000/
NEXTAUTH_SECRET=bcc272549864d5a16b15957db204967b
CLOUDINARY_NAME=db32trinj
CLOUDINARY_KEY=716654641157954
CLOUDINARY_SECRET=9h2el4JYeLU4SmcG_71GO8pioso
CLOUDINARY_URL=cloudinary://716654641157954:9h2el4JYeLU4SmcG_71GO8pioso
```

Estas mismas variables se encuentran adjuntas, modificadas con el link de mi proyecto y cifradas como variables de entorno en Vercel:



Microservicios Externos y librerías

OpenStreetMap se utiliza para obtener información detallada de las direcciones almacenadas en la base de datos. Se usa la api nominatim para buscar las direcciones a partir de sólo algunos datos. De la mano de OpenStreetMap usaremos la librería Leaflet para su integración en NextJs.

Cloudinary facilita la gestión de imágenes, permitiendo cargar, obtener y eliminar fotografías de manera eficiente. Como base de datos no relacional se ha usado MongoDB.

Streamifier facilita la creación de flujos de datos (streams) a partir de diferentes fuentes, como buffers.

Vercel

He hospedado nuestro backend en Vercel. La integración de Next.js con Vercel ha resultado ser sencilla y eficiente. La plataforma ofrece características que facilitan significativamente el proceso de implementación y mantenimiento de nuestro backend. Una de las funcionalidades más destacadas es la capacidad de realizar un re-deploy automáticamente cada vez que realizamos un commit en nuestro repositorio. Este enfoque de implementación continua no solo asegura una integración suave de los cambios, sino que también garantiza que nuestro sistema esté siempre activo, proporcionando continuidad en el servicio.

Sin embargo, como en cualquier tecnología, no todo ha sido perfecto. Recientemente, Next.js experimentó una importante actualización, lo que resultó en un desafío para nosotros, ya que gran parte de la información disponible en la web se encontraba desactualizada.

Cloudinary

Al principio tuve problemas ya que hacía una copia temporal en el sistema de ficheros y hacíamos la subida con un método que ofrece Cloudinary pero Versel no permite crear archivos en sus servidores. Además, existen problemas de incompatibilidad respecto a los buffer de carga que los desarrolladores de Versel y Cloudinary no terminan de solucionar.

Finalmente, hemos optado por subir las imágenes en otro formato. Hemos integrado la biblioteca streamifier para gestionar el flujo de carga de manera eficiente.. streamifier.createReadStream(buffer) toma el buffer de la imagen y lo convierte en un flujo de lectura. Este flujo de lectura se utiliza para alimentar el flujo de carga de Cloudinary.

NextAuth.js | Google

Al hacer uso de google next auth no es necesario token de identificación, se almacena la información en cookies que son manejadas por el navegador.

Middleware

Importamos el middleware proporcionado por NextAuth para gestionar la autenticación de sesión de manera efectiva.

Utilizamos la propiedad config para especificar las rutas que deben estar protegidas. En este caso, las rutas "/logRegistrations", “/newEvent”, “myEvents” están designadas como áreas que requieren autenticación.
Este middleware se integra en nuestro flujo de manejo de sesiones, garantizando que solo los usuarios autenticados tengan acceso a las pestañas críticas para el funcionamiento adecuado de la aplicación.

Al implementar este middleware, estamos fortaleciendo la seguridad de nuestra aplicación al prevenir el acceso no autorizado a áreas sensibles y asegurando que solo aquellos con sesiones válidas puedan utilizar las funciones asociadas.

Entidades

Existen las siguientes entidades:

- events (que representan los eventos que existen)
  nombre
  timestamp
  lugar
  organizador
  imagen

  Se tendrían que haber ampliado los campos de lugar para que openStreetMap pudiese encontrar bien las localizaciones

- registrations
  timestamp (fecha inicio sesion)
  caducidad
  email

Se ha creado un CRUD completo de la entidad events (GET, POST, PUT, DELETE) pudiendo hacer GET de un evento concreto y pudiendo obtener la lista de eventos de un usuario (identificador por su email). Se han dejado los métodos preparados para ir actualizando la página de “myEvents” y la page por defecto.

TAREAS EN PROCESO (WIP)

- Se ha dejado la página principal con un mapa con las coordenadas fijas por defecto. Además, se han insertado algunos marcadores como ejemplo… Haría falta un form al principio de lo que se muestra para fijar la dirección proporcionada por el usuario.

- Existe un componente ImageForm con la que puedes subir fotos perfectamente a Cloudinary (muestra de ejemplo de funcionamiento)...