



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE OAXACA

INTEGRACION DE PROCESOS DE DESARROLLO DE SOFTWARE

GRUPO: 9SA

HORARIO: 07:00-08:00

DOCENTE: Espinosa Pérez Jacob

ALUMNOS:

Méndez Mendoza Luisa Michel

Martínez García Yahir Omar

Pérez de Jesús Edith

López García Lourdes Gloria

PRESENTA:

MANUAL DE APIS

VISIÓN GENERAL DE LA ARQUITECTURA API

Frontend: Aplicación React desplegada en Vercel, utiliza variables de entorno para consumir servicios del backend y APIs externas (YouTube, Google Maps).

Backend: Node.js/Express alojado en Render. Expone endpoints RESTful para autenticación, usuarios, videos y ubicaciones.

APIs Externas: Integración de YouTube Data API (consulta información de videos) y Google Maps Platform (visualización y geolocalización en el frontend).

ENDPOINTS DEL BACKEND EXPRESS (RENDER)

- Autenticación

POST `/api/auth/login` : Valida credenciales de usuario y devuelve token JWT.

POST `/api/auth/register` : Registra nuevo usuario.

GET `/api/auth/me` : Obtiene info básica del usuario autenticado (requiere JWT).

- Usuarios

GET `/api/users` : Lista todos los usuarios (requiere permisos).

GET `/api/users/:id` : Consulta por usuario específico.

PUT `/api/users/:id` : Actualiza información de usuario (restringido).

DELETE `/api/users/:id` : Elimina un usuario (restringido).

- Videos

GET `/api/videos` : Lista los videos disponibles en la base de datos.

GET `/api/videos/:id` : Detalles de un video por su ID.

Implementa consumo indirecto de YouTube API mediante lógica personalizada en el backend

- Ubicaciones (Locations)

GET `/api/locations` : Devuelve lista de ubicaciones almacenadas.

GET `/api/locations/:id` : Información detallada de una ubicación.

- YouTube API (Data API v3)

GET `/api/youtube/search?q=PALABRA` : Busca videos en YouTube a partir de un término y devuelve datos seleccionados. El backend procesa la petición, consulta YouTube y filtra la respuesta.

GET `/api/youtube/video/:id` : Detalles extendidos de un video de YouTube por su ID.

Ambos requieren haber configurado la API Key de YouTube como variable de entorno en Render*

- Geocodificación

GET `/api/geocode?address=DIRECCION` : Traduce una dirección en coordenadas usando la Geocoding API de Google Maps.

Requiere API Key de Google Maps configurada en Render

CONSUMO DE APIS EN EL FRONTEND REACT (VERCEL)

El frontend está diseñado con componentes que interactúan con el backend y con APIs externas:

- Configuración:

- Variables de entorno en Vercel como `REACT_APP_API_URL` (URL de Render) y `REACT_APP_GOOGLE_MAPS_API_KEY`.

Usuarios, Autenticación: Formularios que envían datos mediante `fetch`/`Axios` al backend, gestionan tokens y sesiones.

Videos: El componente de biblioteca carga datos de `/api/videos`, e implementa funcionalidades como búsqueda que pueden consumir video info de YouTube usando los endpoints backend que acceden a la YouTube API.

Google Maps: Uso de la biblioteca `@vis.gl/react-google-maps`, integrando mapas interactivos con marcadores de ubicaciones y rutas.

Geocodificación: Formularios en el frontend pueden consumir `/api/geocode` para transformar direcciones en coordenadas, usando la respuesta para posicionar marcadores en el mapa.

CONFIGURACIÓN DE VARIABLES DE ENTORNO

Backend (Render):

- `YOUTUBE_API_KEY`: Clave para consumir YouTube Data API desde el servidor.
- `GOOGLE_MAPS_API_KEY`: Clave para consultas de Geocoding y otros servicios Maps.

Frontend (Vercel):

- `REACT_APP_API_URL`: URL pública del backend en Render.
- `REACT_APP_GOOGLE_MAPS_API_KEY`: Clave para visualizar Google Maps en el frontend.

INTEGRACIÓN Y CONSUMO: EJEMPLO GENERAL

Buscar videos en YouTube desde el frontend: Formulario en React envía la búsqueda al backend `/api/youtube/search?q=palabra``; el backend consulta la YouTube API y devuelve resultados filtrados que la interfaz muestra.

Visualizar ubicaciones: Componente de mapa en React obtiene coordenadas desde `/api/locations`` (backend), las coloca como marcadores usando Google Maps API. Si el usuario ingresa una dirección, se consume `/api/geocode`` y se añade un nuevo marcador en el mapa.

Autenticación y sesiones: El frontend utiliza endpoints de login y registro, almacena el token JWT, y muestra contenido protegido según el estado de la sesión.

DESPLIEGUE

Render (backend):

- Proyecto conectado a GitHub, configurado como Web Service. Claves de APIs y otras variables de entorno gestionadas en el panel de Render.
- Prueba de endpoints mediante Postman o llamados desde el frontend.

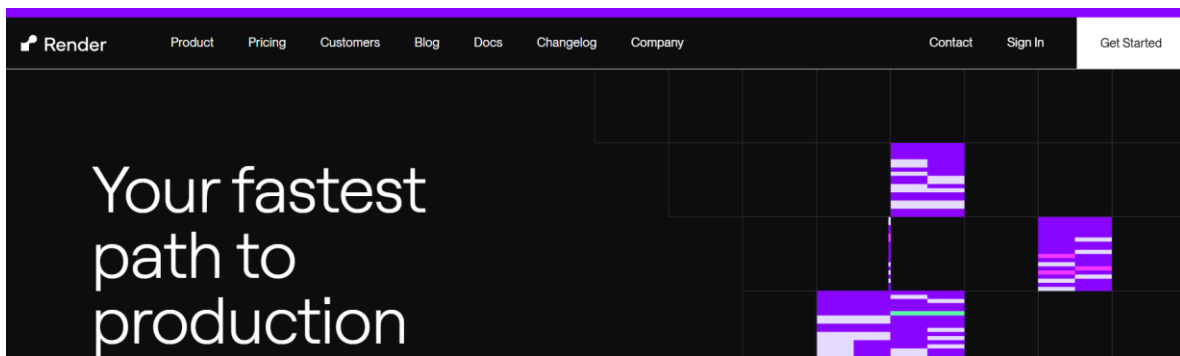
Vercel (frontend):

- Proyecto React vinculado a GitHub, variables de entorno configuradas en el dashboard.
- Pruebas directas en navegador y mediante herramientas de desarrollo de React.

PROCESO DE DESPLIEGUE

CONFIGURAR EL BACKEND EN RENDER.COM

Ingresa a render.com, haz clic en “Get started for free” o “Sign up”, y regístrate con GitHub.



Creamos un nuevo Web Service haciendo clic en "New" > "Web Service". Conectamos nuestro repositorio de GitHub donde tenemos nuestro backend.

🔗 Projects

Overview

Projects

My project

✓ All services are up and running

+ Create new project

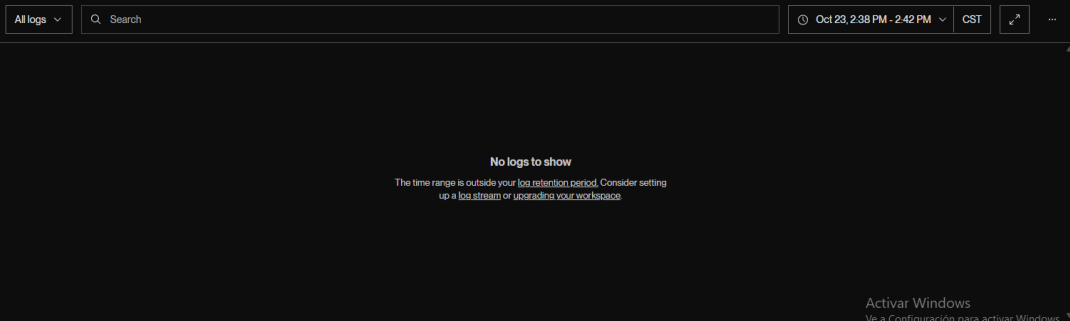
Si el backend está dentro de una carpeta específica (por ejemplo, "backend"), lo especificamos en Root Directory.

🔔 Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.

[Upgrade now](#)

October 23, 2025 at 2:39 PM ✓ Live

858cfa6 Logs en back funcional

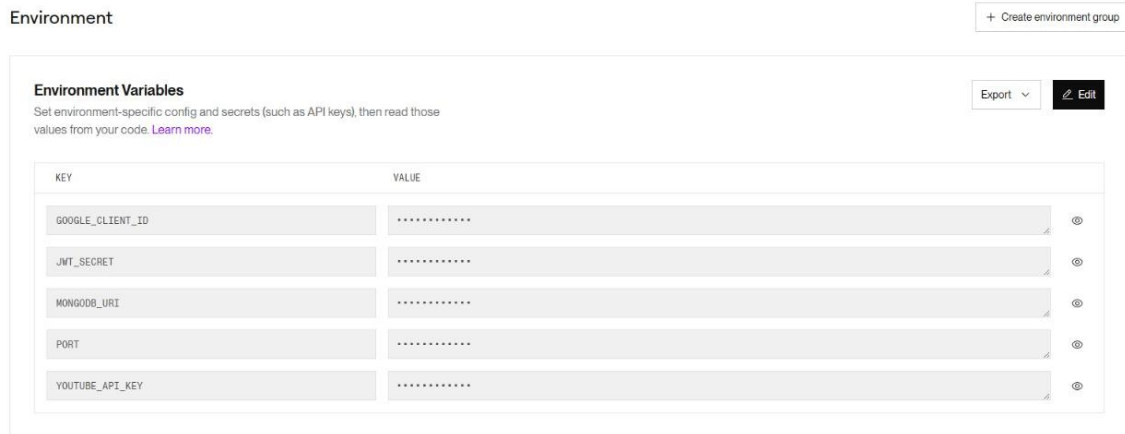


Need better ways to work with logs? Try the [Render CLI](#), [Render MCP Server](#), or set up a [log stream integration](#).

	Ignored Paths Changes that match these paths will not trigger a new build. Add ignored path
Build Command Render runs this command to build your app before each deploy.	Backend/ \$ npm install Edit
Pre-Deploy Command <small>Optional</small> Render runs this command before the start command. Useful for database migrations and static asset uploads.	Backend/ \$ Edit
Start Command Render runs this command to start your app with each deploy.	Backend/ \$ node server.js Edit
Auto-Deploy By default, Render automatically deploys your service whenever you update its code or configuration. Disable to handle deploys manually. Learn more .	On Control Edit
Deploy Hook Your private URL, to trigger a deploy for this server. Remember to keep this a secret.	<div></div> Regenerate hook

Llenamos los datos de configuración: Name, Region, Branch, Runtime (Node), Build Command (npm install), Start Command (node server.js o npm start si se tiene el script en package.json).

Configuramos las Variables de Entorno importantes (PORT, MONGODB_URI, JWT_SECRET, YOUTUBE_API_KEY, GOOGLE_CLIENT_ID).



The screenshot shows the 'Environment' tab in Vercel. At the top right is a button '+ Create environment group'. Below it is the 'Environment Variables' section with a description: 'Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more](#)'. To the right of this section are 'Export' and 'Edit' buttons. Below is a table with two columns: 'KEY' and 'VALUE'. The table contains five rows of environment variables, each with a key, a masked value (dots), and an eye icon to toggle visibility.

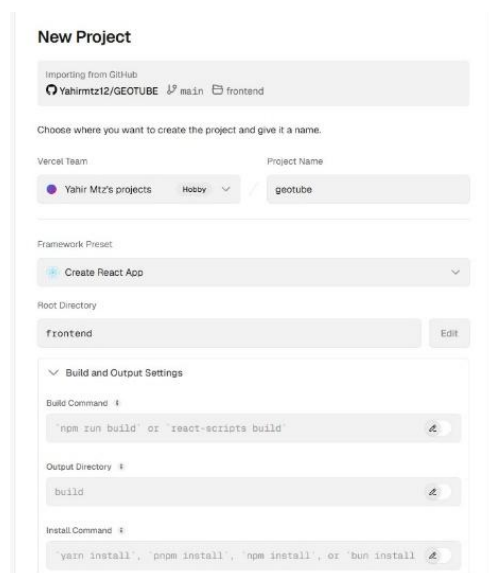
KEY	VALUE
GOOGLE_CLIENT_ID
JWT_SECRET
MONGODB_URI
PORT
YOUTUBE_API_KEY

Seleccionamos el plan “Free” en Instance Type para evitar costos.
Hacemos clic en “Create Web Service”. Render instalará dependencias y desplegará la aplicación.

Cuando el servicio sea “Live”, tendremos una URL como <https://geotube-back.onrender.com>. Copiamos esta URL, será nuestra nueva API_BASE_URL.

DESPLIEGAR EL FRONTEND EN VERCEL

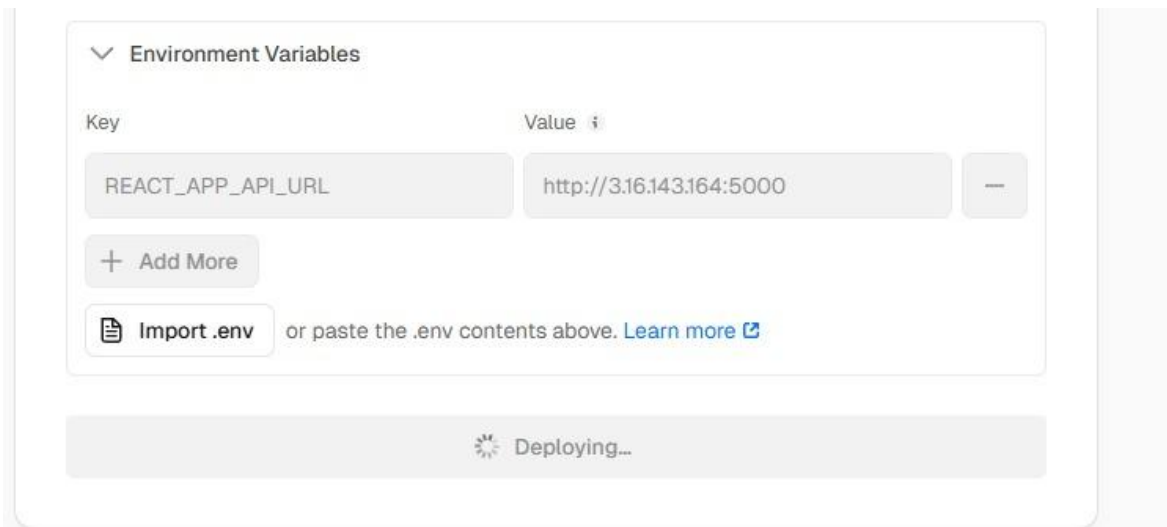
Inicia sesión en Vercel, haz clic en "Add New..." > "Project", buscamos y seleccionamos el repositorio de GitHub.



The screenshot shows the 'New Project' page in Vercel. It starts with 'Importing from GitHub' and shows the repository 'Yahirmtz12/GEOTUBE' with branches 'main' and 'frontend'. Below is a prompt to 'Choose where you want to create the project and give it a name.' The 'Vercel Team' is set to 'Yahir Mtz's projects' and the 'Project Name' is 'geotube'. The 'Framework Preset' is 'Create React App'. The 'Root Directory' is 'frontend'. Under 'Build and Output Settings', the 'Build Command' is 'npm run build' or 'react-scripts build', the 'Output Directory' is 'build', and the 'Install Command' is 'yarn install', 'pnpm install', 'npm install', or 'bun install'.

Vercel detecta automáticamente que es una app de React (create-react-app). Normalmente no necesitamos modificar la configuración de build.

Vamos a “Environment Variables” y definimos la variable REACT_APP_API_URL para indicar la URL de tu backend (puede ser la IP de nuestra instancia o el enlace HTTPS de Render).
Hacemos clic en “Deploy”.




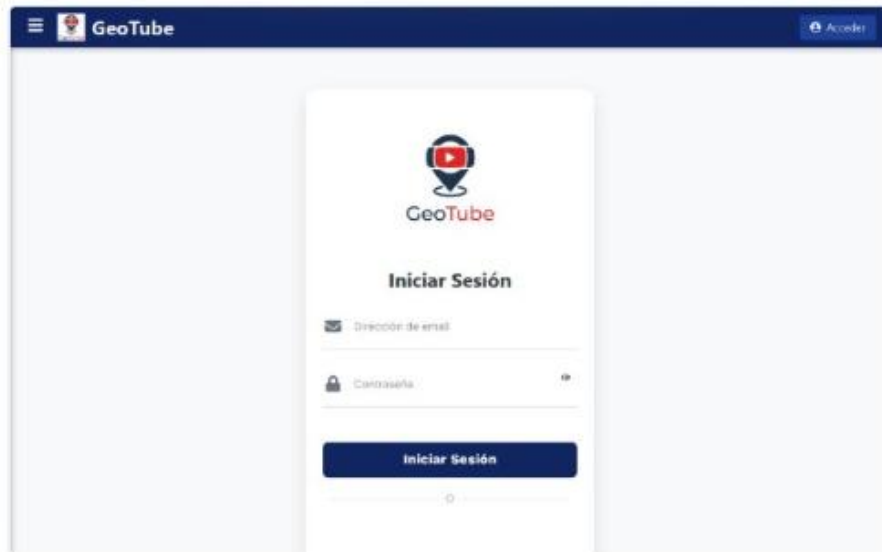
Actualizamos nuestro código local para que use la variable de entorno en vez de <http://localhost:5000>.

Hacemos git add ., git commit y git push para subir este cambio. Vercel detecta el push y despliega automáticamente tu frontend.

Una vez finalizado el despliegue, obtenemos el enlace público de tu frontend como <https://geotube-delta.vercel.app/>

Congratulations!

You just deployed a new project to  Yahir Mtz's projects.



Next Steps

-  **Instant Previews**
Push a new branch to preview changes instantly
-  **Add Domain**
Add a custom domain to your project →
-  **Enable Speed Insights**
Track how users experience your site over time →

[Continue to Dashboard](#)