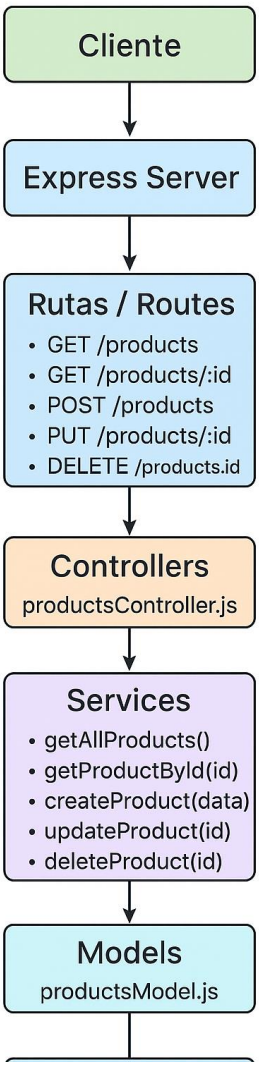
Ambientes

Tipo	Link
Código	https://github.com/contesl/C25256SLC.git
Datos Firebase en Firestore	https://console.firebase.google.com/project/slc-api/overview?hl=es-419&fb_gclid=Cj0KCQjwrJTGBhCbARIsANFBfgv4c2gsVx9X57Kmg5xKfG-JFBR1z72xsEZPvdRExly-eHO5tKO6aBMaAk3KEALw_wcB
Deployment Vercel	https://c25256-slc.vercel.app/ Linkeado con el repositorio github https://vercel.com/scontes-projects/c25256-slc es el directorio no productivo

Estructura de Directorios

Nombre	Descripcion y Contenido
/	Directorio raíz Contiene el modulo principal: server.js Contiene archivos json necesarios: package.json package-lock.json vercel.json Tambien contiene achivo .env con las variables de entorno necesarias. Este archivo no se mueve al repositorio github.
docs	Contiene Diagrama de arquitectura Tambien C25256SLC-app-api-vercel.postman_collection.json que se puede exportar a Postman para hacer las pruebas de cada llamada.
src/utils	Contiene el Modulo generador de tokens
src/data	Contiene el archivo data.js con todo lo necesario para conectarse a Firebase en Firestore. Este archivo es utilizado por los modelos que existen en src/models.
src/middlewares	Contiene el Modulo middleware de autenticación
src/routes	Contiene los Modulos que administran los pedidos de autenticación y de productos llamando al controller correspondiente
src/controllers	Contiene el Modulo controlador de autenticación y de productos. En el caso de productos el controller extrae la información del body del request y llama al service correspondiente
src/services	Contiene el módulo de servicio de productos. Interactua con el modelo ejecutando la función que corresponde.
src/models	Contiene el módulo que iteractua directamente con la base Firebase en Firestore realizando el CRUD de datos
src/collections	Contiene la descripción de las collections existentes en Firebase para independizar el desarrollo de las modificaciones en Firebase

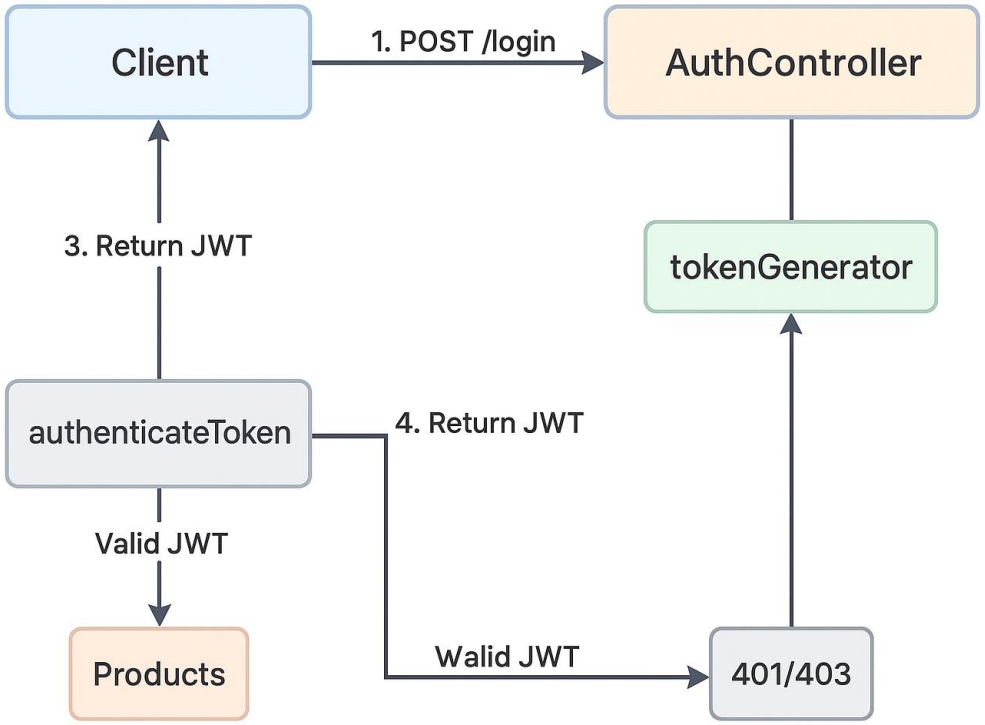
Diagramas de Arquitectura



🔷 **Flujo de una petición (ej. POST /products)**

1. Cliente hace POST /products desde Postman.
2. La **ruta** (productsRoutes.js) recibe la solicitud y llama al **controller**.
3. El **controller** extrae datos del body y llama al **service** correspondiente.
4. El **service** llama a la función del **modelo** (saveProduct) que interactúa con **Firestore**.
5. Firestore guarda el producto y devuelve el documento creado.
6. La respuesta sube por la misma cadena hasta el **controller**, que devuelve el JSON al cliente.

Al agregar jwt el flujo de un requerimiento es:



🔷 **Flujo resumido**

1. Cliente envía credenciales al endpoint /login.
2. Backend valida y genera **JWT**.
3. Cliente recibe el token y lo guarda.
4. Cada request a rutas protegidas envía **Authorization: Bearer <token>**.
5. Middleware authenticateToken verifica JWT:
    - Si válido → permite acceder al recurso.
    - Si inválido/expirado → devuelve 401 o 403.
6. Controlador maneja la petición usando req.user si necesitas info del usuario.

Instructivo para probar la API con Postman

Este instructivo explica paso a paso cómo probar la API Node.js desplegada en Vercel utilizando **Postman**.

🚀 1. Requisitos previos

- Tener **Postman** instalado
  👉 Descargar Postman
- Conexión a Internet
- URL base de la API: http://c25256-slc.vercel.app

(Si querés probar localmente, reemplazá la URL base por http://localhost:3000 cuando tengas el servidor corriendo en tu máquina).

📦 2. Importar la colección en Postman

1. Asegurate de estar en **Collections (arriba a la izquierda)**
2. Haz clic en **Import**
3. Pega el contenido del archivo .json de la colección llamado
   C25256SLC-app-api-vercel.postman_collection.json
   que se incluye en este repositorio en el directorio \docs).

Aparecerá una colección llamada **app-api-vercel** con las siguientes requests:

- ✅ POST Autenticacion
- ✅ GET PRODUCTS
- ✅ POST CREATE PRODUCTS
- ✅ PUT UPDATE PRODUCTS
- ✅ DELETE PRODUCTS

⚙️ 3. Configurar las variables del entorno

1. Asegurate de estar en **Environments (arriba a la izquierda)**
2. Haz clic en **Import**
3. Pega el contenido del archivo .json del ambiente llamado
   C25256SLC-API_Vercel.postman_environment.json
   que se incluye en este repositorio en el directorio \docs).

Aparecerá en Environment **API_Vercel** conteniendo las variables base_url y token.

.

🔑 4. Autenticación (obtener token JWT)

1. Seleccionar **app_api_vercel en Collections**.
2. Abre la request **POST Autenticacion**
3. Asegurate de que esté seleccionado el **Environment API_Vercel.**
4. **Haz clic en Send.**

    - Si la autenticación es correcta, obtendrás una respuesta 200 OK con un token.
    - Postman guardará ese token automáticamente en la variable token.

🧪 5. Probar los endpoints

A continuación, ejecutá cada request en orden:

🟢 1. GET PRODUCTS

- Método: GET
- URL: {{base_url}}/products
- Debe devolver un array de productos.

🟡 2. POST CREATE PRODUCTS

- Método: POST
- URL: {{base_url}}/products
- Body → **raw** → **JSON** debe contender este formato:

```
{
  "price": 3300,
  "name": "Producto Nuevo"
}
```

- Debe responder con el producto creado o un mensaje de éxito.

🟠 3. PUT UPDATE PRODUCTS

- Método: PUT
- URL: {{base_url}}/products/<ID_DEL_PRODUCTO>
  (Reemplazá <ID_DEL_PRODUCTO> con los datos de body)
- Body → **raw** → **JSON** debe contener este formato:

```
{
  "nombre": "Producto actualizado vercel",
  "precio": 5500
}
```

- Debe devolver el producto actualizado.

🔴 4. DELETE PRODUCTS

- Método: DELETE
- URL: {{base_url}}/products/<ID_DEL_PRODUCTO>

  (Eliminará el <ID_DEL_PRODUCTO> que se provee)

- Elimina el producto indicado (revisa la respuesta o status 200/204).

⚡ 7. Flujo completo sugerido

1. **POST Autenticacion** → obtiene y guarda el token.
2. **GET PRODUCTS** → lista los productos.
3. **POST CREATE PRODUCTS** → agrega uno nuevo.
4. **GET PRODUCTS** → verifica que el producto aparezca.
5. **PUT UPDATE PRODUCTS** → actualiza un producto existente.
6. **DELETE PRODUCTS** → elimina un producto.
7. **GET PRODUCTS** → confirma que ya no esté.

🧩 8. Prueba local (opcional)

Si querés probarlo en tu máquina:

1. Cloná el proyecto.
2. Ejecutá:
3. npm install
4. npm start
5. Asegurate de tener el .env con tu SECRET_KEY y configuración Firebase.
6. En Postman, cambiá base_url → http://localhost:3000.
7. Ejecutá las mismas pruebas que antes.

□ 9. Posibles errores y soluciones

Error	Posible causa	Solución
403 Forbidden	Falta el token o está vencido	Repetí el paso de login
500 Internal Server Error	Variables .env faltantes o error en servidor	Revisá logs en consola o en Vercel
404 Not Found	ID inexistente o ruta incorrecta	Verificá el ID en Firestore o el endpoint
CORS Error (desde navegador)	CORS no habilitado	Solo afecta si consumís desde frontend; en Postman no aplica

---

✅ 10. Confirmación de instalación correcta

Sabés que la API funciona correctamente si:

- Podés hacer login y obtener un JWT válido.
- GET /products devuelve un listado JSON.
- Podés crear, modificar y eliminar productos sin errores.