

Área del Conocimiento de Tecnología de la Información y la comunicación

Entregable BackEnd

Diseño de Sistemas de Internet

Elaborado por:

Br. Gabriel
Alejandro Escorcia
Chávez.

Carnet: 2021-0338I

Tutor:

Br. Andreus Enrique
Ramírez Salinas.

Carnet: 2021-0584I

Ing. Cristopher Chávez Larios

09 de noviembre, 2025

Managua, Nicaragua

Realización del modelo relacional de la API, hecho en Strapi y PostgreSQL

The screenshot shows the Strapi Content-Type Builder interface. On the left, a sidebar titled "Content-Type Builder" lists "COLLECTION TYPES" with options: Impuesto (selected), Inventario, Producto, Productolimpuesto, Sucursal (selected), Tienda, and User. A "Save" button and a "..." button are at the top of the sidebar. The main area is titled "Impuesto" and contains two fields: "Nombre*" (Text) and "producto_impuestos" (Relation (oneToMany) with Productolimpuesto). A button at the bottom right says "+ Add another field to this collection type". A "Configure the view" button is in the top right corner.

configuración de los permisos y roles.

Permissions

Only actions bound by a route are listed below.

Impuesto
Define all allowed actions for the api:impuesto plugin. 

Inventario
Define all allowed actions for the api:inventario plugin. 

INVENTARIO	<input checked="" type="checkbox"/> Select all
<input checked="" type="checkbox"/> create	<input checked="" type="checkbox"/> delete
<input checked="" type="checkbox"/> find	<input checked="" type="checkbox"/> findOne
<input checked="" type="checkbox"/> update	

Configuración de permisos y que solo el owner (propietario de la tienda), que se estableció en la relación entre usuario y tienda, pueda ver todas las tiendas del sistema.

```
import { factories } from '@strapi/strapi';

export default factories.createCoreController('api::tienda.tienda', ({ strapi }) => ({
  async create(ctx) {
    const userId = ctx.state.user.id;

    ctx.request.body.data = {
      ...ctx.request.body.data,
      owner: userId,
    };

    const response = await super.create(ctx);
    return response;
  },

  async find(ctx) {
    const userId = ctx.state.user.id;
    ctx.query.filters = {
      ...(typeof ctx.query.filters === 'object' && ctx.query.filters !== null ? ctx.query.filters : {}),
      owner: userId,
    };

    const response = await super.find(ctx);
    return response;
  },

  async findOne(ctx) {
    await this.validateOwner(ctx);
    return super.findOne(ctx);
  },

  async update(ctx) {
    await this.validateOwner(ctx);
    return super.update(ctx);
  }
});
```

Creacion de dueño en postman

The screenshot shows the Postman interface for creating a new owner. The request URL is `http://localhost:1337/api/auth/local/register`. The Body tab is selected, showing a raw JSON payload:

```
1 {
2   "username": "dueno_a",
3   "email": "dueno_a@tienda.com",
4   "password": "Password123"
5 }
```

The response status is 200 OK, with a response time of 1.73 s and a size of 1.36 KB. The response body contains a JWT token and user information:

```
1 {
2   "jwt": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MSwiaWF0IjoxNzYyNzQ3MzA0LCJleHAiOjE3NjUzMzkzMDR9.vYBvBhGnA1sXlQ7chS4sJ0Yl3MgRvRh9cRd-fB-pCtU",
3   "user": {
4     "id": 1,
5     "documentId": "y9oly2xy8k28t36axzzffsp2",
6     "username": "dueno_a",
7     "email": "dueno_a@tienda.com",
8     "provider": "local",
9     "confirmed": true,
10    "blocked": false,
```

Inicio de sesión de dueño a

The screenshot shows the Postman interface for logging in the owner. The request URL is `http://localhost:1337/api/auth/local`. The Body tab is selected, showing a raw JSON payload:

```
1 {
2   "identifier": "dueno_a@tienda.com",
3   "password": "Password123"
4 }
```

The response status is 200 OK, with a response time of 744 ms and a size of 1.36 KB. The response body contains a JWT token and user information:

```
1 {
2   "jwt": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MSwiaWF0IjoxNzYyNzQ3NjQ1LCJleHAiOjE3NjUzMzk2NDV9.jCzVnjWh519wMLWHQ260xeSIWuZ3_xgB0-XXrTvoxQ",
3   "user": {
4     "id": 1,
5     "documentId": "y9oly2xy8k28t36axzzffsp2",
6     "username": "dueno_a",
7     "email": "dueno_a@tienda.com",
8     "provider": "local",
9     "confirmed": true,
10    "blocked": false,
```

Configuracion del token de auth del dueño a

This authorization method will be used for every request in this collection. You can override this by specifying one in the request.

Auth Type

Bearer Token

The authorization header will be automatically generated when you send the request.
Learn more about [Bearer Token](#) authorization.

Token

Registro de tienda a nombre del dueño a

POST http://localhost:1337/api/tiendas

Params Authorization Headers (10) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary

```
1 {
2   "data": {
3     "nombre": "Tienda Dueño A",
4     "ruc": "A12345"
5   }
6 }
```

Body Cookies Headers (17) Test Results

{ } JSON ▾ ▶ Preview Visualize

```
1 {
2   "data": {
3     "id": 2,
4     "documentId": "bqjgjgtzlxzojqaov9gtxqaz",
5     "nombre": "Tienda Dueño A",
6     "ruc": "A12345",
7     "createdAt": "2025-11-10T05:18:20.597Z",
8     "updatedAt": "2025-11-10T05:18:20.597Z",
9     "publishedAt": "2025-11-10T05:18:22.058Z"
10   },
11   "meta": {}
}
```

Find (GET) de las tiendas

Body Cookies Headers (17) Test Results | ⚙️

{ } JSON ▾ Preview Visualize | ↴

```
1   1
2   "data": [
3     {
4       "id": 1,
5       "documentId": "bqjgigtzlxzojqaov9gtxqaz",
6       "nombre": "Tienda Dueño A",
7       "ruc": "A12345",
8       "createdAt": "2025-11-10T05:18:20.597Z",
9       "updatedAt": "2025-11-10T05:18:24.010Z",
10      "publishedAt": null
11    }
```


[GabrielEscoria27/saas-impuestos-backend: Repositorio de Proyecto de DSi de Gabriel Escoria y Andreus Ramírez](#)