

1078346748 Omar Roa

Crear categoría

The screenshot displays a development environment with two main windows. The top window is a web browser at `localhost:5001/graphql` showing the GraphiQL interface. The left pane contains a GraphQL mutation query:

```
1 mutation{
2   createCategory
3   (category:{
4     name: "C1",
5     description: "Categoria 1"
6   }){
7
8     name
9     description
10  }
11 }
```

The right pane shows the JSON response:

```
{
  "data": {
    "createCategory": {
      "name": "C1",
      "description": "Categoria 1"
    }
  }
}
```

The bottom window is MySQL Workbench, titled "Diplomad - Warning - not supported". The "SCHEMAS" sidebar on the left lists various databases and tables, including `supermarket_ms_category`. The main query editor shows the SQL query:

```
1 • SELECT * FROM supermarket_db.supermarket_ms_category;
```

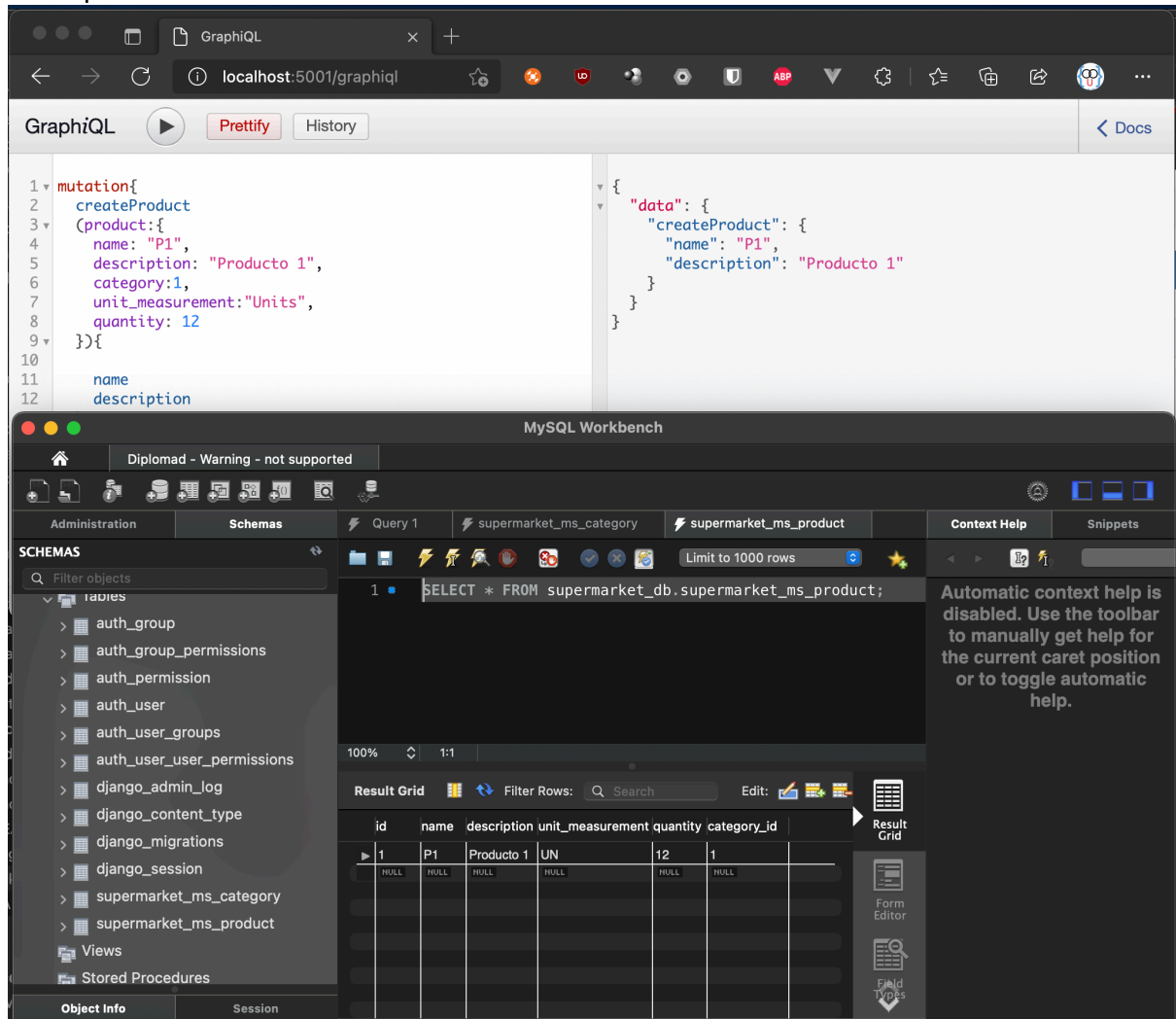
The "Result Grid" at the bottom displays the query results in a table format:

id	name	description
1	C1	Categoria 1
	NULL	NULL
	NULL	NULL
	NULL	NULL
	NULL	NULL
	NULL	NULL
	NULL	NULL
	NULL	NULL
	NULL	NULL
	NULL	NULL

On the right side of the MySQL Workbench window, a message states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

[illegible]

Crear producto



Crear inventario

The screenshot shows two applications. The top application is GraphQL IDE, displaying a mutation query to create an inventory item. The query is as follows:

```
1 mutation{
2   createInventario
3   (inventario:{
4     cantidad: 13,
5     producto: 1
6   }){
7     cantidad
8   }
9 }
10 }
```

The response shows the created item with a quantity of 13 and product ID 1. The bottom application is MongoDB Compass, showing the 'supermarket_db.inventarios' collection with one document:

```
{
  "_id": ObjectId("613a2cbcc759f6b778e949da"),
  "producto": 1,
  "cantidad": 13,
  "__v": 0
}
```

Obtener inventario

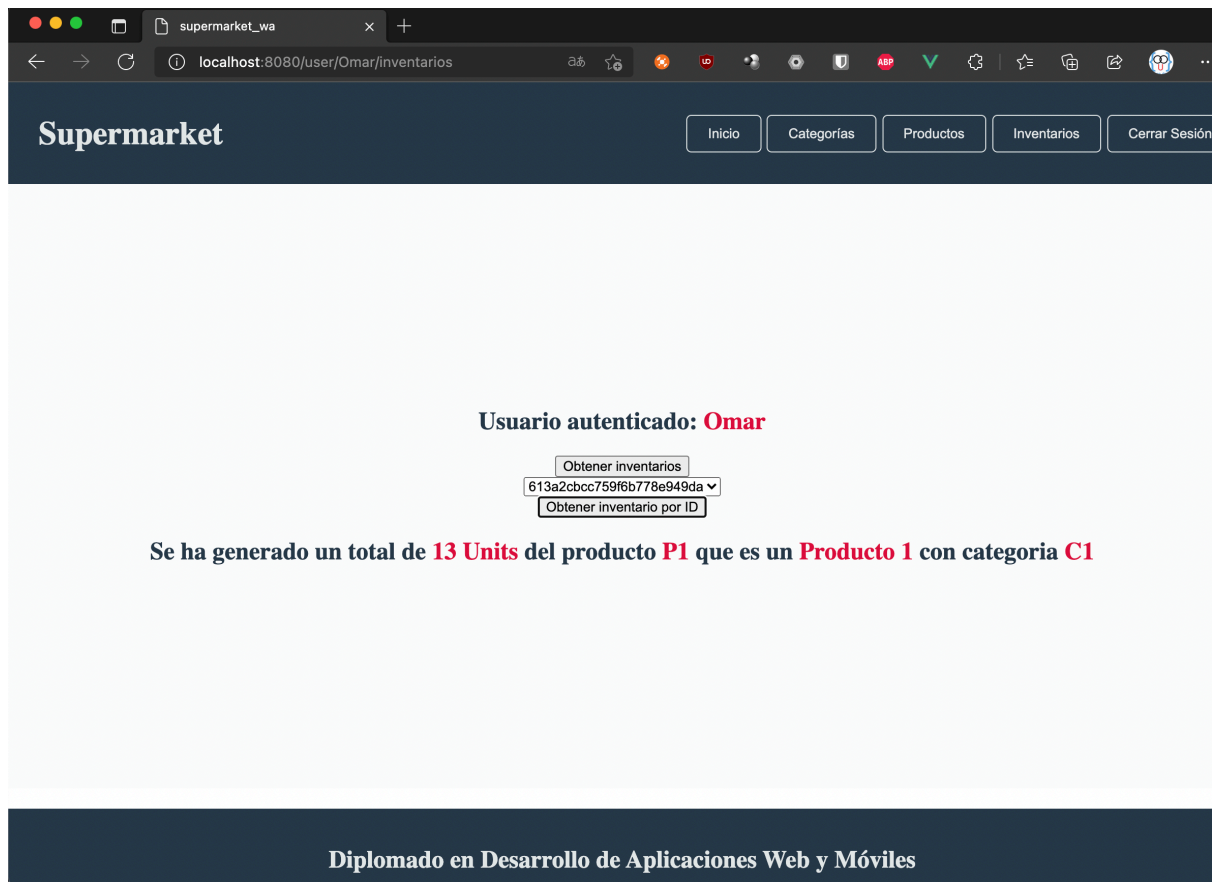
GrapQL

The screenshot shows the GraphQL IDE with a query to retrieve an inventory item by ID. The query is as follows:

```
1 query{
2   inventarioById(id: "613a2cbcc759f6b778e949da"){
3     _id
4     cantidad
5     producto{
6       name
7       description
8       unit_measurement
9       quantity
10    category{
11      name
12      description
13    }
14  }
15 }
16 }
```

The response shows the retrieved item with its details, including product name, description, unit measurement, quantity, and category.

Interface grafica:



Código Fuente orquestador:

```
const resolvers = {
  Query: {
    allInventarios: (_, { id }) => {
      getRequest(URL, ''),
    },
    inventarioById: async (_, { id }) => {
      const inventario = await generalRequest(`${URL}/${id}`, 'GET');
      console.log(inventario.producto)
      inventario.producto = await Product.default.Query.productById(_, { id: inventario.producto });
      return inventario;
    },
  },
}
```

Se obtiene el inventario que contiene el id del producto en el micro servicio inventario y después se llama al micro servicio de products para obtener el detalle del producto

Código Fuente: <https://github.com/oeroaq/DAWM-M1>