**Aplicando GRAPHQL**

Documento el primer ejemplo que hice donde aplique Graphql para que se pueda ver de forma sencilla como lo implemente y funcionó. En el código se van a ver más cosas ya que agregue más funciones.

Archivo graphqlMiddleware.js

import { buildSchema } from 'graphql'

import { graphqlHTTP } from 'express-graphql'

import {obtenerProductos} from './controller/ProductosControllerGraphql.js'

const schema = buildSchema(`

  input ProductoInput {

    nombre: String

    descripcion: String

    precio: Int

    imagenURL: String

    stock: Int

    categoria: String

  }

  type Producto {

    id: ID!

    nombre: String

    descripcion: String

    precio: Int

    imagenURL: String

    stock: Int

    categoria: String

  }

  type Query {

    obtenerProductos: [Producto]

  }

`)

export const graphqlMiddleware = graphqlHTTP({

  schema: schema,

  rootValue: {

    obtenerProductos,

  },

  graphiql: true,

})

Archivo server.js agregue

import { graphqlMiddleware } from './graphqlMiddleware.js'

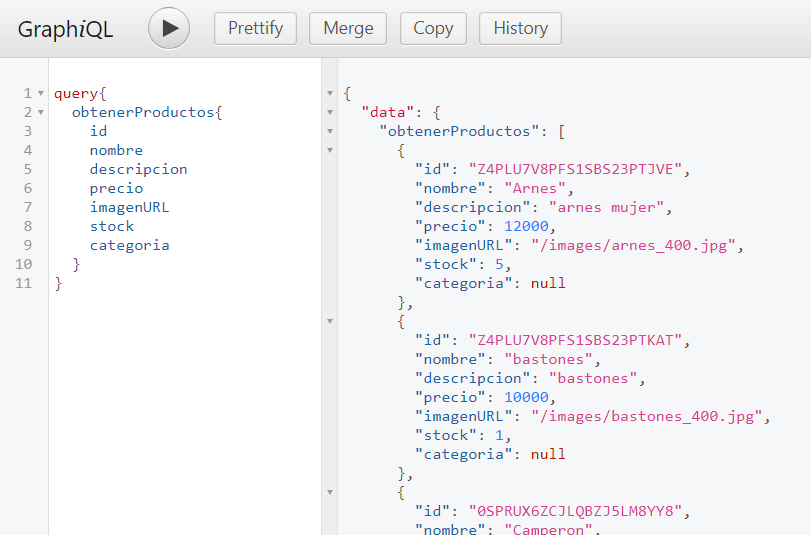
y

app.use('/graphql', graphqlMiddleware)

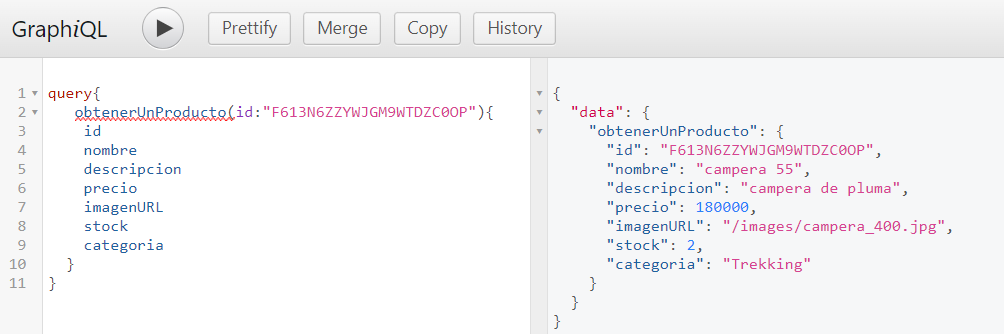
cree archivo ProductosControllerGraphql.js

Tuve que crear el archivo ProductosControllerGraphql.js, que es igual a ProductosController.js, solo que en vez de devolver res. Devuelve directo lo que me da la función.

http://localhost:8080/graphql

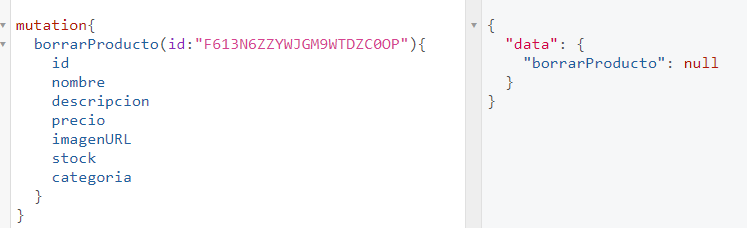


**Otros ejemplos de GRAPHQL**

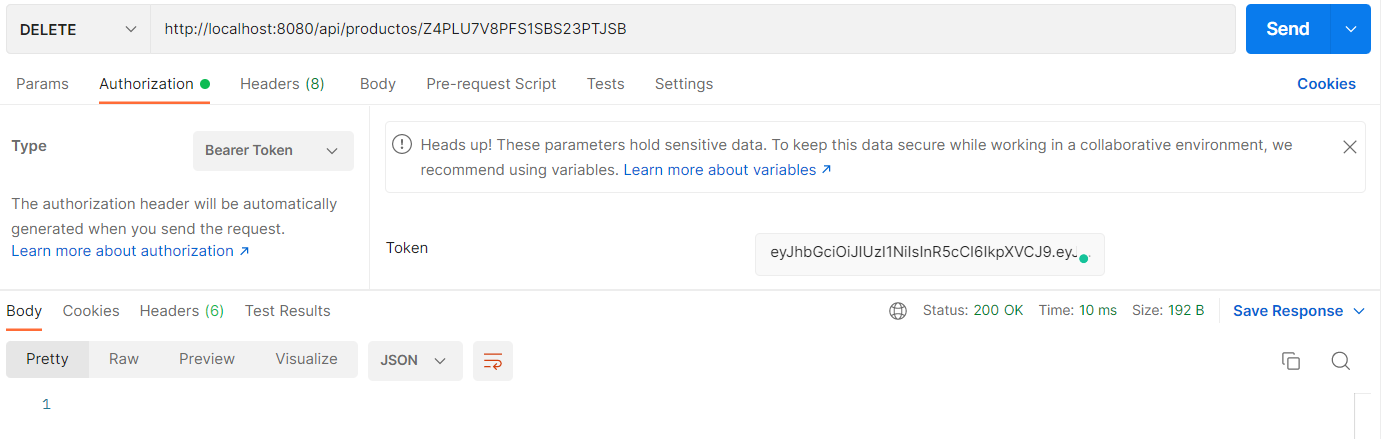
**obtenerUnProducto**

**borrarProducto**

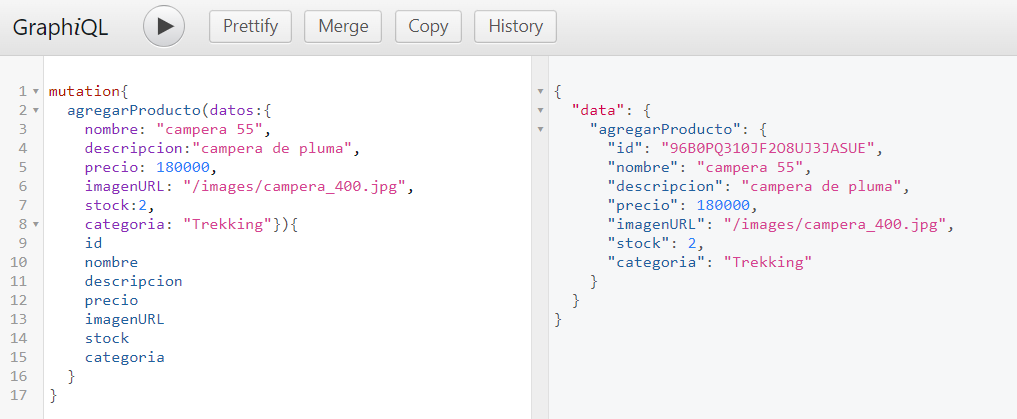
Borro el mismo producto que busque anteriormente, y luego lo vuelvo a buscar y se observa que no lo encuentra , muestra el error. La api no devuelve nada por eso el null



Ejemplo con el postman donde se observa que no devuelve nada por eso el null



**agregarProducto**



Quedan finalmente el código graphqlMiddleware.js

import { buildSchema } from 'graphql'

import { graphqlHTTP } from 'express-graphql'

import {obtenerProductos, obtenerUnProducto, agregarProducto, borrarProducto} from './controller/ProductosControllerGraphql.js'

const schema = buildSchema(`

  input ProductoInput {

    nombre: String

    descripcion: String

    precio: Int

    imagenURL: String

    stock: Int

    categoria: String

  }

  type Producto {

    id: ID!

    nombre: String

    descripcion: String

    precio: Int

    imagenURL: String

    stock: Int

    categoria: String

  }

  type Query {

    obtenerProductos: [Producto]

    obtenerUnProducto(id: ID!): Producto

  }

  type Mutation {

    agregarProducto(datos: ProductoInput!): Producto

    borrarProducto(id: ID!): Producto

  }

`)

export const graphqlMiddleware = graphqlHTTP({

  schema: schema,

  rootValue: {

    obtenerProductos,

    obtenerUnProducto,

    borrarProducto,

    agregarProducto,

  },

  graphiql: true,

})