Cambiando datos en el servidor con MUTATIONS

https://youtu.be/3vldzoNRz-8

Mutación

Vamos a cambiar los datos.

Antes de crear una mutación , debemos de definirla.

Mutación para crear una persona:

```
type Mutation {
    createPerson(name: String!, age: Int!, country: String!): Person
  }
```

Instalar uuid

https://github.com/uuidjs/uuid

```
npm install uuid
```

Importar uuid

```
import { v1 as uuid } from "uuid";
```

Una vez creada la Mutación, nos vamos al Resolver

```
//Esta es una mutacion
  Mutation: {
    createPerson: (root, args) => {
      //const { name, age, country } = args;
      const person = { ...args, id: uuid() };

      persons.push(person);
      return person;
    },
  },
```

Al agregar un campo String en nuestra mutation, no se puede agregar con comillas simples. Debe ser con comillas dobles.

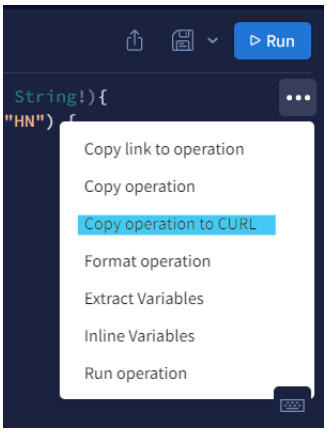
Creación de Mutación en el PlayGround de GraphQL y Selección de Campos

La mutación createPerson le hemos dicho que devuelva una persona, por lo tanto debemos darle los campos a extraer de esa persona.

```
mutation($name: String!, $age: Int!, $country: String!){
  createPerson(name: "Ana", age: 12, country: "HN") {
    name
    age
    country
    info {
      county
      name
    }
    id
  }
}
```

En GraphQL la URL para
hacer las distintas peticiones
siempre sera la misma.

SOLO TENEMOS UN
ENDPOINT



En mi caso es:

```
curl --request POST \
    --header 'content-type: application/json' \
    --url http://localhost:4000/ \
    --data '{"query":"mutation($name: String!, $age: Int!, $country: String!){\r\n  createPerson(name: \"Ana\", age: 12, country: \"HN\") {
```

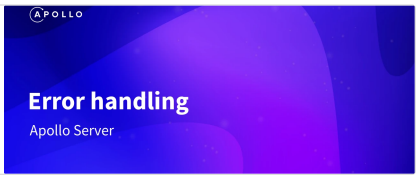
Validaciones en las Mutaciones

vamos a crear validaciones para que no se agreguen personas con el mismo nombre

Error handling

Whenever Apollo Server encounters errors while processing a GraphQL operation, its response to the client includes an errors array that contains each error that occurred. Each error in the array has an extensions field that provides additional useful information, including an error code and (while in development mode) an

https://www.apollographql.com/docs/apollo-server/data/errors/



Para ello iremos al resolver de la mutacion:

```
Mutation: {
  createPerson: (root, args) => {
    if (persons.find((person) => person.name === args.name)) {
      throw new Error("Person already exists");
    }
    //const { name, age, country } = args;
    const person = { ...args, id: uuid() };

    persons.push(person); //update dabase
    return person;
  },
},
```

El resultado al hacer los cambios en la mutación es el siguiente

Ahora vamos importar el UserInputError

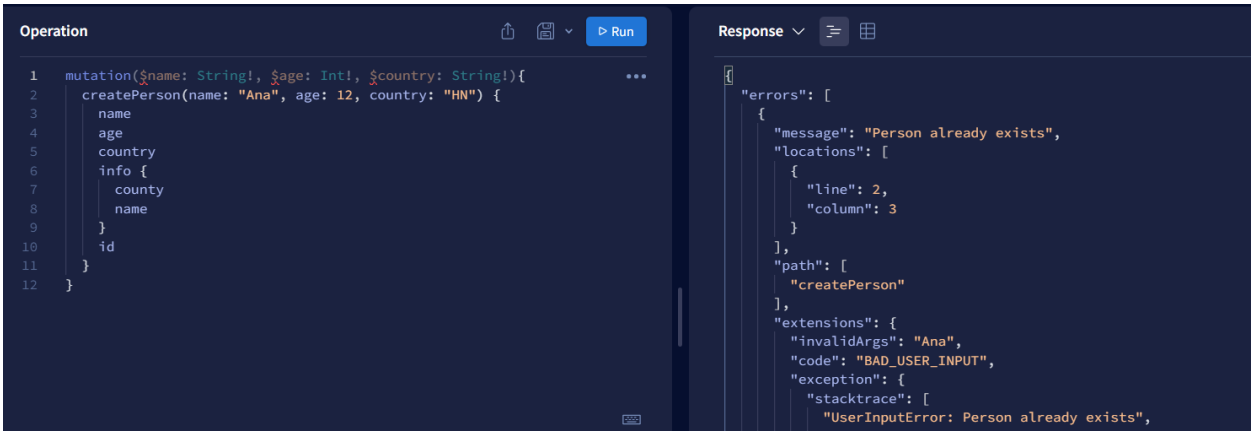
```
import { ApolloServer, UserInputError, gql } from "apollo-server";
```

En lugar de usar throw new Error en la mutación usaremos el throw new UserInputError

```
Mutation: {
    createPerson: (root, args) => {
      if (persons.find((person) => person.name === args.name)) {
        throw new UserInputError("Person already exists", {
          invalidArgs: args.name,
        });
      }
      //const { name, age, country } = args;
      const person = { ...args, id: uuid() };

      persons.push(person); //update dabase
      return person;
    },
  },
```

De esta forma nos da un poco mas de información en el error



GraphQL/Cambiando datos en el servidor con MUTATIONS at main · TiffMonique/GraphQL

Iniciando en GraphQL *-*. Contribute to TiffMonique/GraphQL development by creating an account on GitHub.

https://github.com/TiffMonique/GraphQL/tree/main/Cambiando%20datos%20en%20el%20servidor%20con%20MUTATIONS