GraphQl :

myRepo : https://github.com/VinayakWarad/GraphQLImpl

**Key Features of GraphQL:**

* **Declarative Data Fetching:** Clients specify what data they need in a single query.
* **Single Endpoint:** Unlike REST APIs, which often require multiple endpoints, GraphQL uses one endpoint for all queries, mutations, and subscriptions.
* **Strongly Typed Schema:** The structure of the data is defined in a schema, ensuring consistency and validation.
* **Real-Time Data with Subscriptions:** GraphQL supports subscriptions to receive real-time updates.

**Basic Structure:**

1. **Query** – Used to fetch data. (QueryMapping is the annotation)
2. **Mutation** – Used to modify data (create, update, delete). (MutationMapping)
3. **Subscription** – Used to subscribe to real-time updates.

Youtube Java Techie: https://www.youtube.com/watch?v=GJc36OKFQ\_M

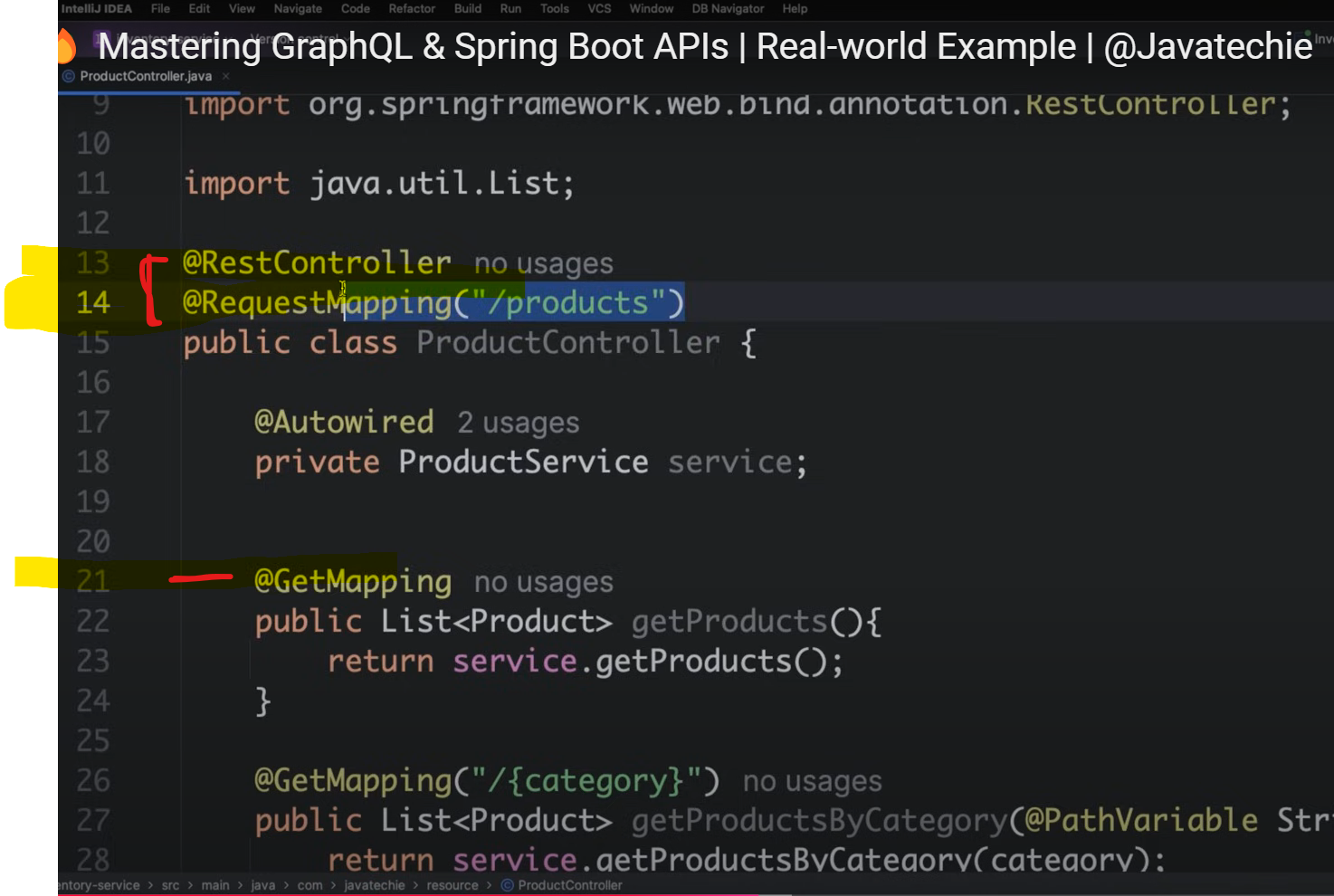
Consider you have Entity   
  
Product{  
id, name, category,price,stock}

Now you have API to get student details (“/get/{studentId}”)

But tis API returns all student details (id, name, address, contactNum) but our requirement is that I just want id and contactNum. Here comes the picture of GraphQl.

Note : Through rest we can make call “/get/{studentId}” and get all fields and later map result DTO (contains required fields only), But we have fetched all details of student and mapped to DTO even though we needed just DTO fields. So to overcome this issue we makeuse of **GraphQl**

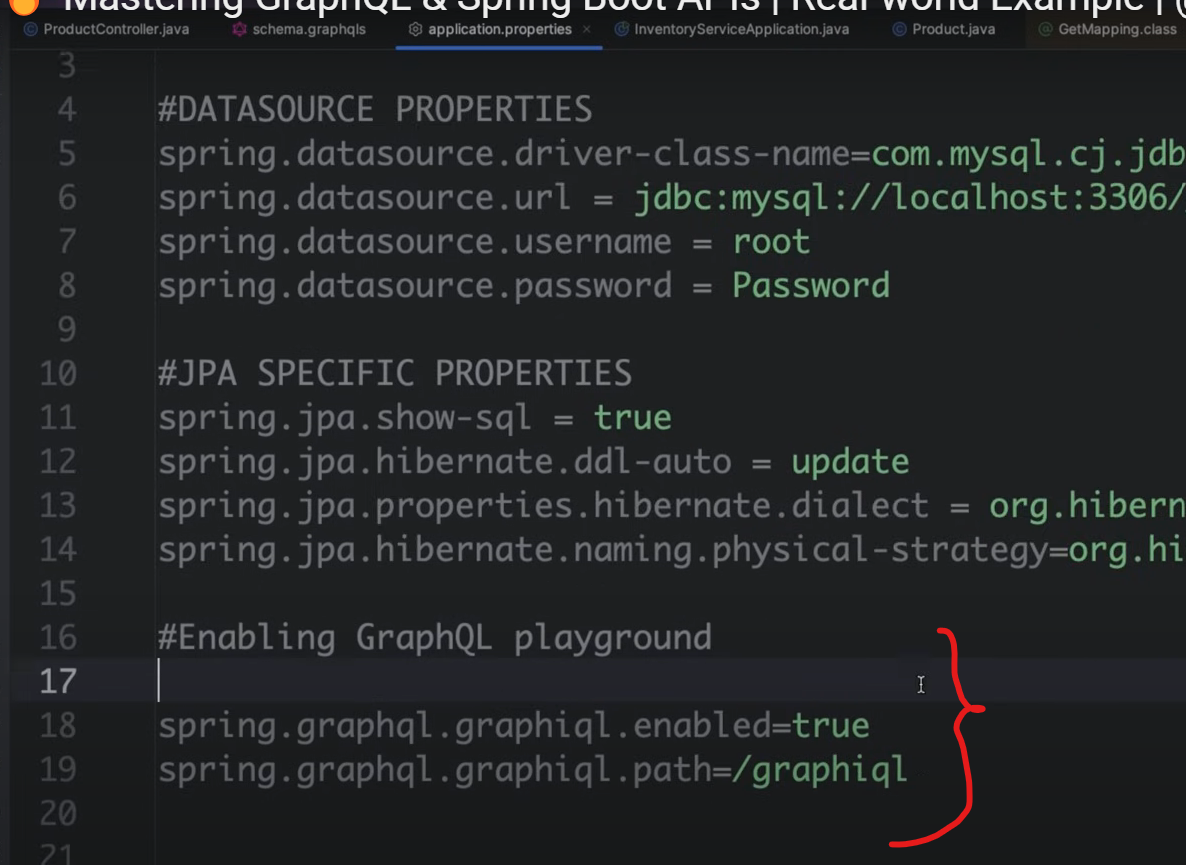
Below is ex of controller class of REST

****

**Implementation :**

Dependencies : “Spring GraphQl”

enable GraphQl playground application.yaml properties needs to be updated :



Let’s convert above REST controller class to graphQl controller format :

A screen shot of a computer

Description automatically generated

Now we will not have URL(Get,post,put,delete) to make call instead we will make use of graphqlQuery.

So where to write our queries?

Note : GraphQl does not support double values instead we use Float for decimals



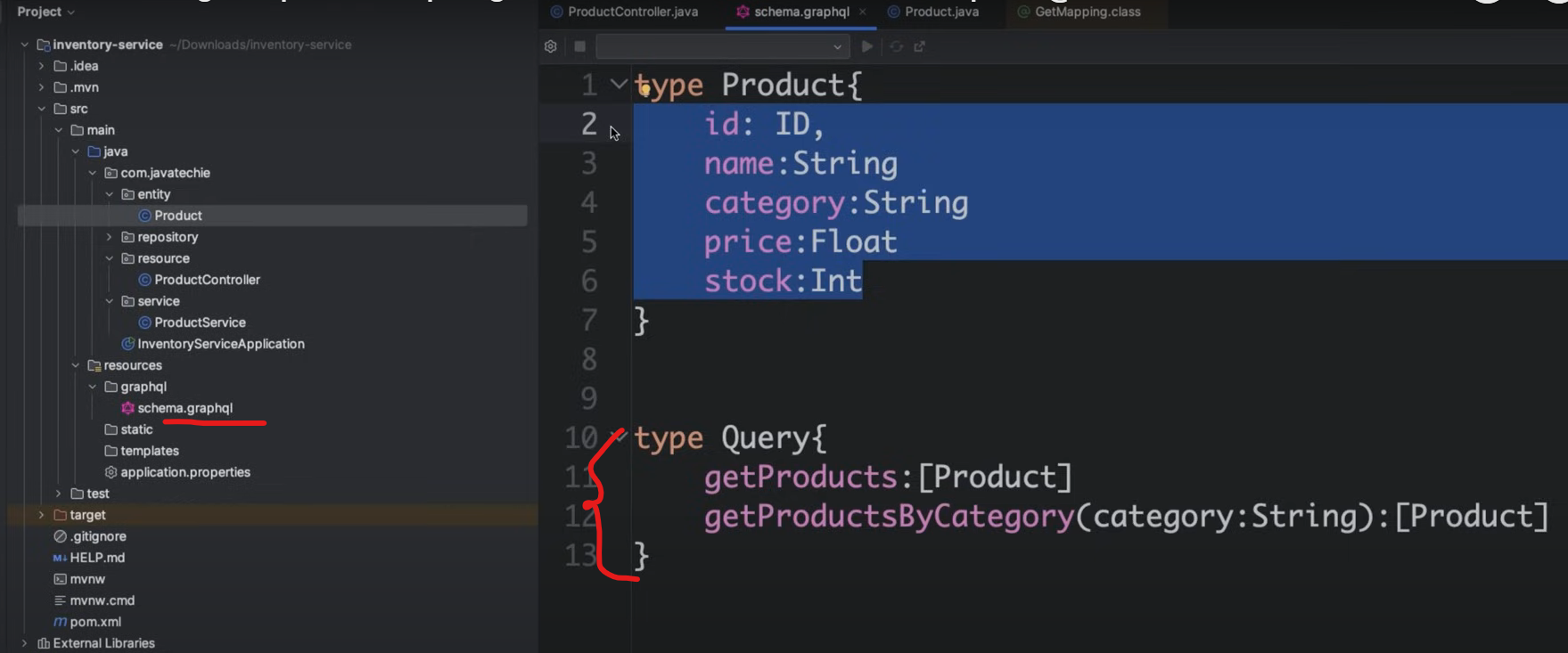
As now we don’t have URL’s to make calls how to make this service call then ?

Ex : getProducts() and getProductsByCategory()

A screen shot of a computer

Description automatically generated

So now to fetch getProducts() and ProductsByCategory() we have to define query in **schema.graphqls**



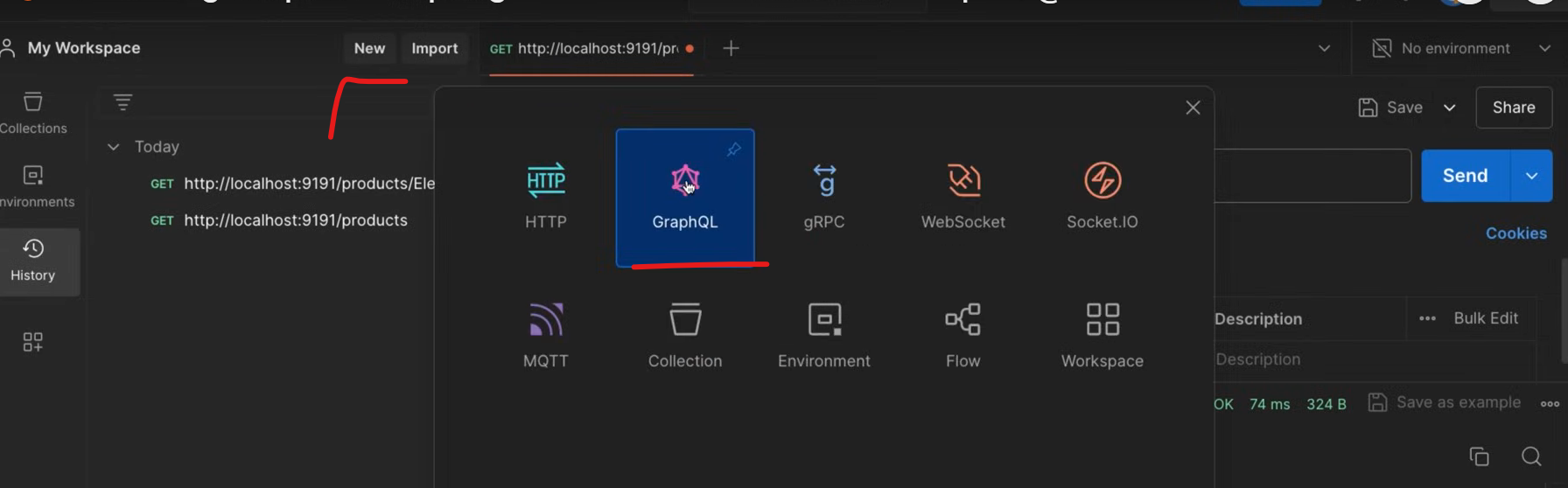
Run above application

Testing :

2 approaches : Postman and trough GraphQl ground

Postman -> New -> Graphql

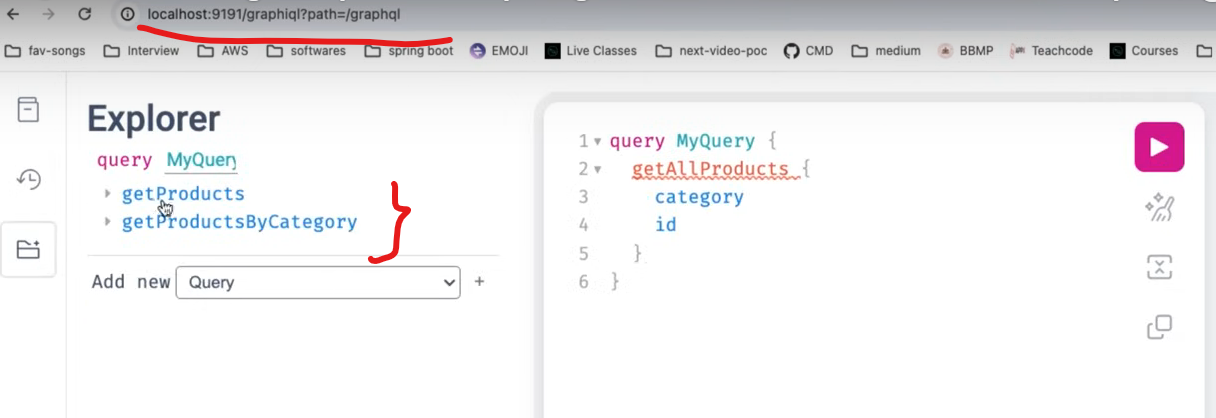
If you Just mention URL postman automatically shows querymethods available for our app



A screenshot of a computer

Description automatically generated

Run app : and open GraphQl ground



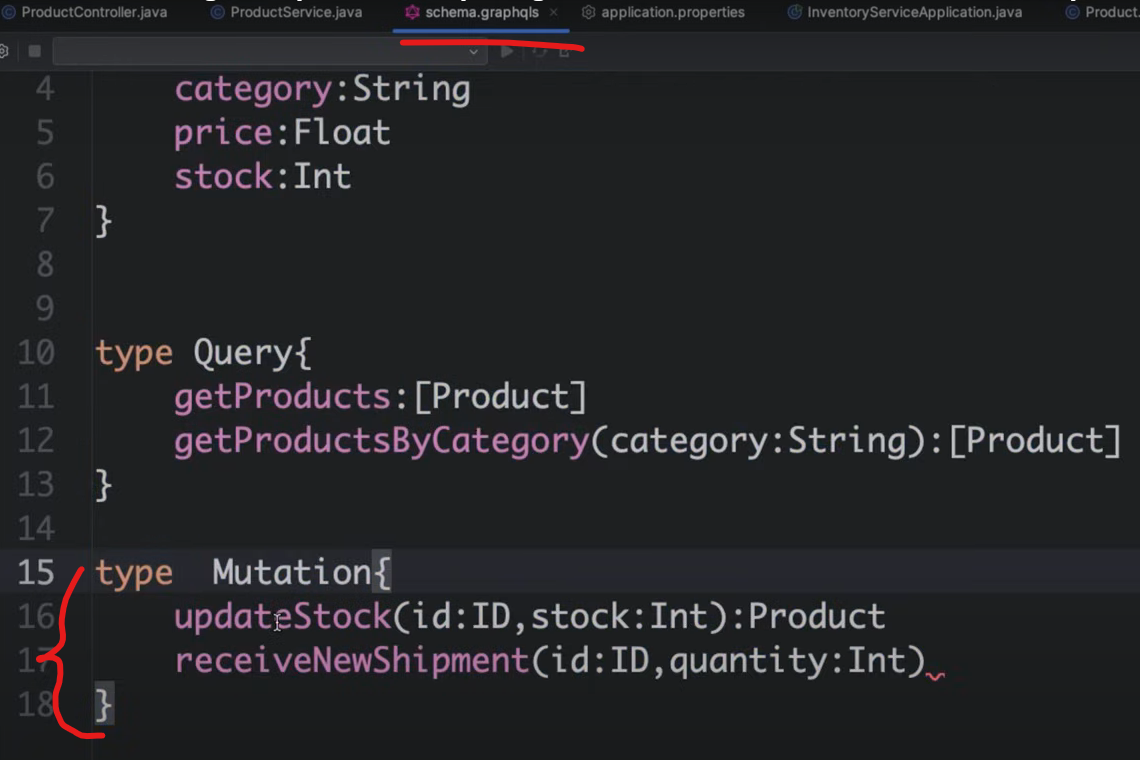
GraphQL is not only for retrieving but also saving data also can be performed.

QueryMapping to fetch data and Mutationmapping to update the data

To fetch these methods we have to write mutationQuery as shown below

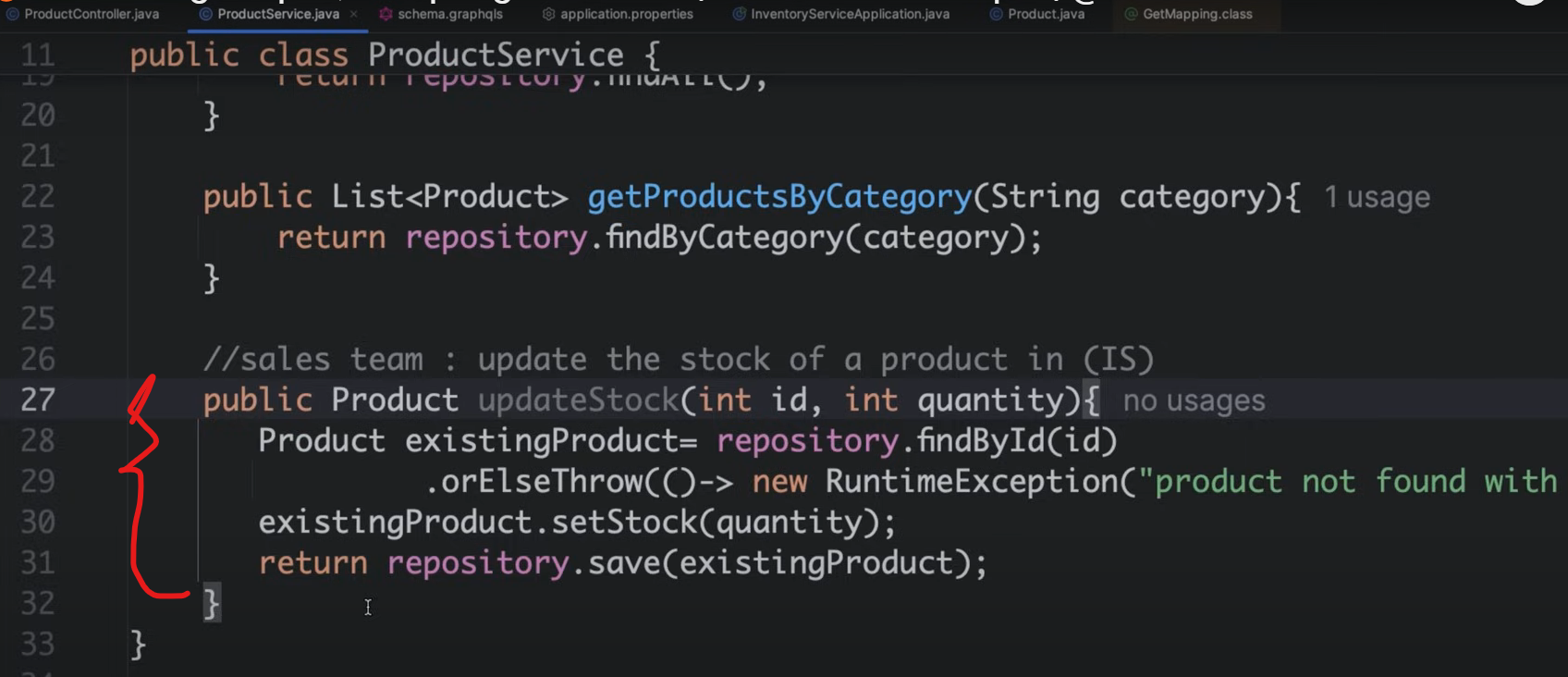
type “Query” to fetch data

type “Mutatiom” to update data

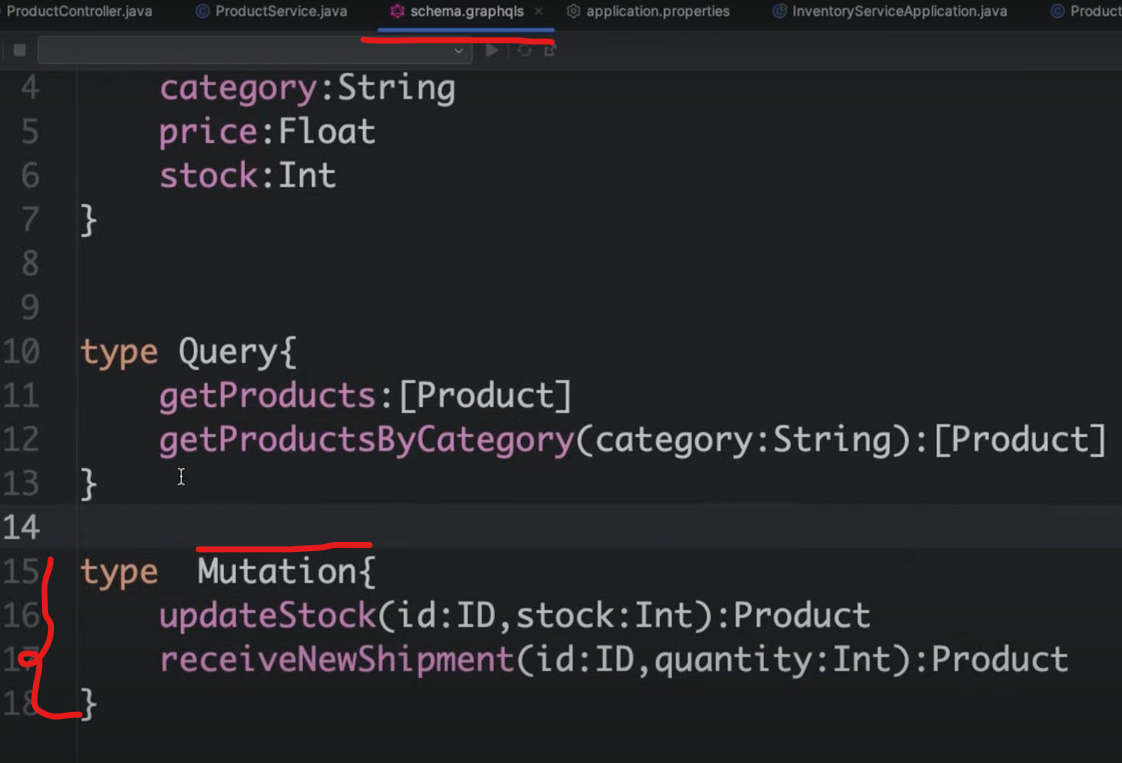


Example : Update stock in inventory updateStock()

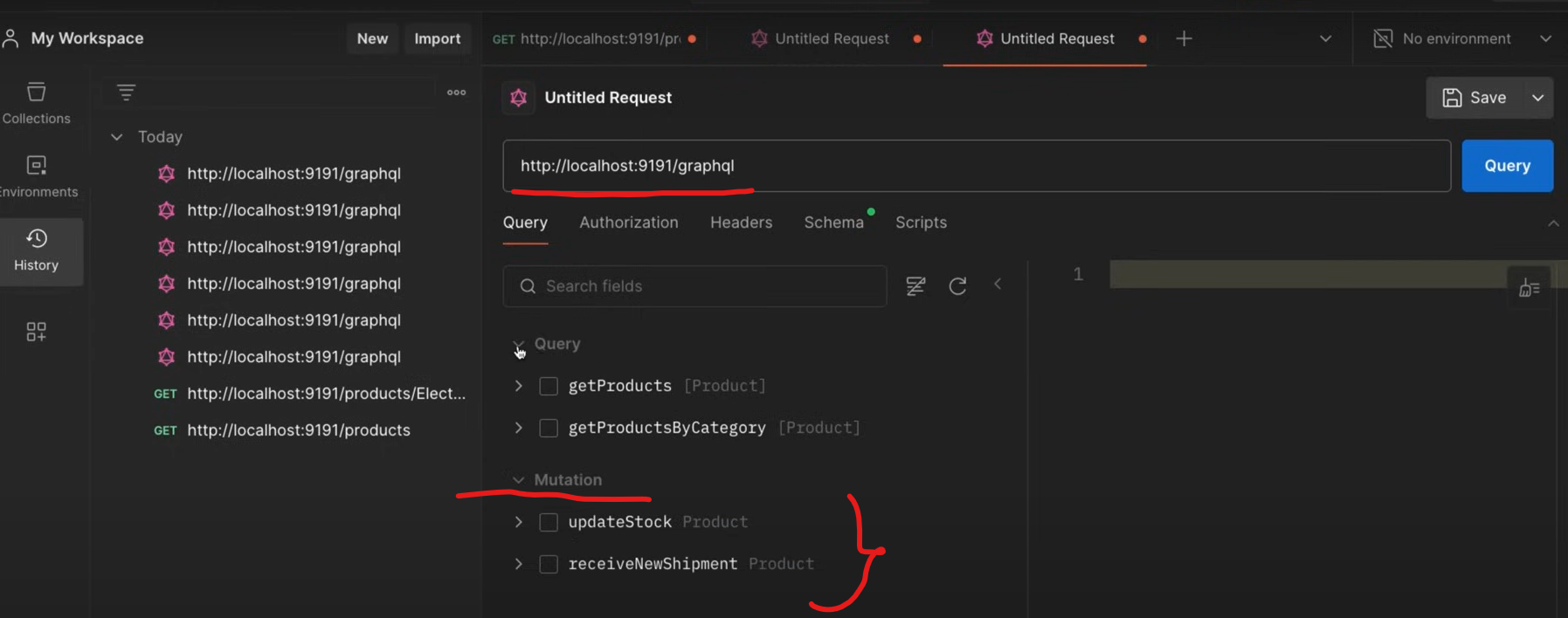
Below is service method



Let’s update controller :



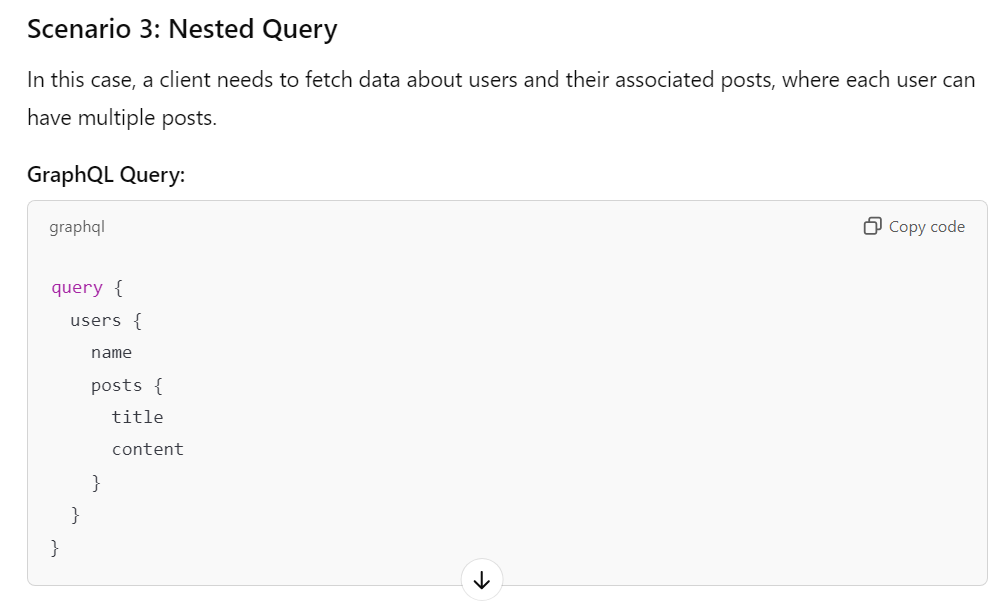
Rerun the project to check mutation:



Youtube end :

Chatgpt :

Example 1 :



A screenshot of a computer

Description automatically generated



Example 2 :

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Example 3 :

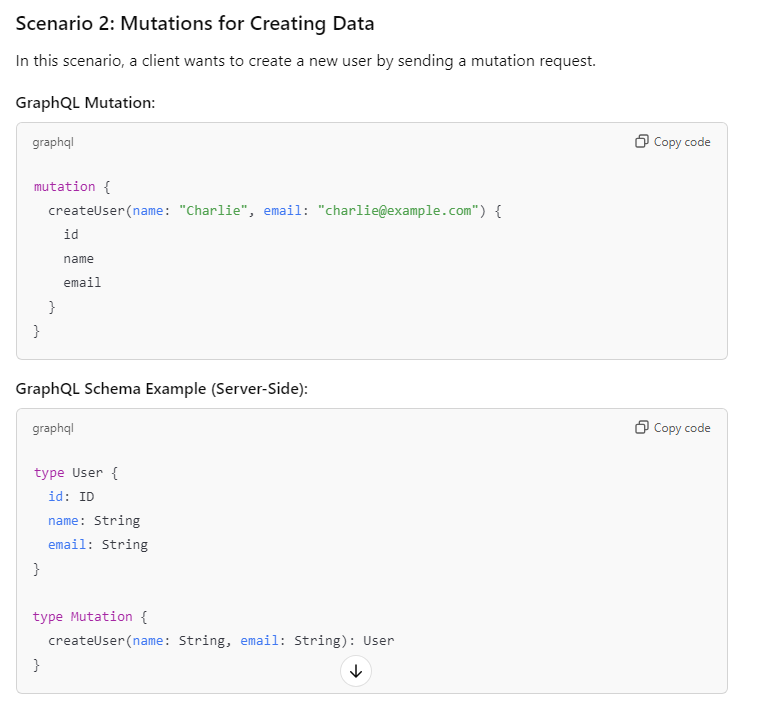
A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

Example 4 :



A white rectangular object with green text

Description automatically generated

Exaple 5 :

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Conclusion**

GraphQL provides a flexible and efficient way to interact with APIs. It allows clients to request only the data they need, supports real-time updates, and provides strong typing through schemas. With the ability to handle queries, mutations, and subscriptions in a unified way, it offers a compelling alternative to traditional REST APIs. The examples above show different ways you can interact with GraphQL, from basic queries to real-time subscriptions and error handling.