GraphQL with NestJS

Node.js Accelerator – Jan 23

Agenda

1. Review
2. Code first approach
3. Data Loader
4. Authentication
5. Federation

Toptal®

Node.js Accelerator

Oct 2022

It’s ideally that you finished the Learn Graphql for Express and React course in Udemy with a combination of the official (and free) course from Apollo

- Mutation: modifies data
- Query: get the data you need
- Resolvers: responsible for populating the data for a single field in your schema

NestJS offers both code first and schema first approach for building a GraphQL API.

With Typescript you can get benefits on coding once and already having the schema ready, without a need to get the Schema and then map in Typescript with the required types.

Today we are going to use code first approach to build our GraphQL api

```
@Resolver( classType: () => Task)
export class TasksResolver {
  constructor(private readonly tasksService: TasksService) {
  }

  @Query( typeFunc: () => [Task], options: {name: 'tasks'})
  findAll() {
    return this.tasksService.findAll();
  }
```

```
@ObjectType()
export class Task {
  @Field( returnTypeFunction: () => ID)
  id: number;
  @Field( returnTypeFunction: () => Int)
  exampleField: number;
}
```

More on schema first and code first here

Avoiding N+1 problem with dataloaders in NestJS

Add the library with: npm i dataloader, configure a data loader and a resolve field for it

```
@Injectable( options: { scope: Scope.REQUEST })
export class ProductsFromWishlistLoader extends DataLoader<string, Product> {
  constructor(
    @InjectModel(Product.name)
    private readonly productModel: Model<Product>,
  ) {
    super( batchLoadFn: (keys : readonly string[] ) => this.batchLoad(keys));
  }
```

```
  @ResolveField( propertyName: 'products', typeFunc: () => [Product])
  async getUserAddresses(@Parent() wishlist: Wishlist) {
    return this.productsFromWishlistLoader
      .loadMany(wishlist.products.toString().split( separator: ','));
  }
```

Good code explanation without NestjS here

Let’s see how we can authenticate with GraphQL and NestJS in our code

- npm i @nestjs/passport passport passport-jwt @nestjs/jwt

It is also a common practice to auto generate the code you need in your frontend application.

Auto generate tutorial with nx.dev here

Toptal®

ACTION ITEMS

Week 6

1. We strongly recommend you to explore below module through Udemy this week and learn about Node.js essentials

Module	Topic	Description	Udemy Link	
Self-paced learning on Udemy	Testing	Learn Unit Testing	Click here	1 hrs 42 min


Toptal®