

ÉDI... 1 NON ENREGISTRÉ(S)

- JS schema.js schem...
- JS server.js

PROJET-GRAPHQL






- node\_modules
- schemas
  - JS schema.js

{ } package-lock.json

{ } package.json

JS server.js

```
1  const graphql = require('graphql');
2  const lodash = require('lodash'); // lodash est une librairie qui permet de faire des manipulation sur les objets
3  const { // récupération des type en destructurant graphql
4    GraphQLObjectType,
5    GraphQLString,
6    GraphQLInt,
7    GraphQLSchema
8  } = graphql;
9
10 const users = [ // tableau de donné factice pour les tests
11   {id: "1", firstname: "Mathieu", age: "32"}, //
12   {id: "1", firstname: "Lucil", age: "29"}
13 ]
14
15
16 //création d'un type pour le user composer de trois type
17 const UserType = new GraphQLObjectType({
18   name: 'User',
19   fields: {
20     id: { type: GraphQLString }, //il est important de faire attention a la case des propriété par rapport a la data
21     firstname: { type: GraphQLString },
22     age: { type: GraphQLInt }
23   }
24 });
25
26 //création d'une Root Query qui est notre point d'entrée
27 const RootQuery = new GraphQLObjectType({
28   name: 'RootQueryType',
29   fields: { // le champs
30     user: { // le nom du champs de notre Root Query USER EST EGALEMENT UNE FONCTION.
31       type: UserType, // on fournit le type de user qui est notre UserType
32       args: {id: { type: GraphQLString}}, // <- on fournis un argument qui précise quel information peut recevoir notre user.
33       resolve(parentValue, args) { //<-et ensuite on lui fournisse une promesse qui signifie qu'est ce qu'il doit faire quand il a reçu l'id.
34         return lodash.find(users, {id: args.id}) //<-la fonction lodash.find permet de donner le tableau que l'on veut et
35         // le type de la colonne que l'on veut il va nous retrouver la valeur.
36       }
37     }
38   });
39 module.exports = new GraphQLSchema({ //permet d'exporter tout le schema
40   query: RootQuery // a partir de ce moment là Rootquery contient tout les informations de UserType
41 });
```



EXPLORATEUR

ÉDITEURS OUVERTS

PROJET-GRAPHQL

JS schema.js schem...

JS server.js

node\_modules

schemas

JS schema.js

package-lock.json

package.json

JS server.js

JS schema.js

JS server.js

```
1  const express = require("express"); // récupération du module express
2  const expressGraphQL = require("express-graphql"); // récupération du module express-graphql
3  const userSchema = require('./schemas/schema') // on importe notre schema dans le fichier server.js
4  const server = express(); // création du serveur
5
6  server.use("/salutGraphQL", expressGraphQL({ //crée le path pour acceder et on utilise le module expressGraphQL
7    graphiql:true, //pour lier express et graphql
8    schema : userSchema // on fournis notre schema en paramètre
9  }))
10 server.listen(4000,()=>{ // lancement du serveur avec une fonction call back consol.log
11   console.log("Serveur est en écoute sur le port 4000.");
12 })
```

GraphiQL - Google Drive

05\_Root Query - Google Slides

GraphiQL

localhost:4000/salutGraphQL

GraphiQL

▶

Prettify

History

```
1 # Welcome to GraphiQL
2 #
3 # GraphiQL is an in-browser tool for writing, validating, and
4 # testing GraphQL queries.
5 #
6 # Type queries into this side of the screen, and you will see int
7 # typeaheads aware of the current GraphQL type schema and live s
8 # validation errors highlighted within the text.
9 #
10 # GraphQL queries typically start with a "{" character. Lines tha
11 # with a # are ignored.
12 #
13 # An example GraphQL query might look like:
14 #
15 #   {
16 #     field(arg: "value") {
17 #       subField
18 #     }
19 #   }
20 #
21 # Keyboard shortcuts:
22 #
23 #   Prettify Query:  Shift-Ctrl-P (or press the prettify button ab
24 #
25 #   Run Query:      Ctrl-Enter (or press the play button above)
26 #
27 #   Auto Complete:  Ctrl-Space (or just start typing)
28 #
29
30
```

< RootQueryType

User

×

Q Search User...

No Description

FIELDS

id: String

firstname: String

age: Int