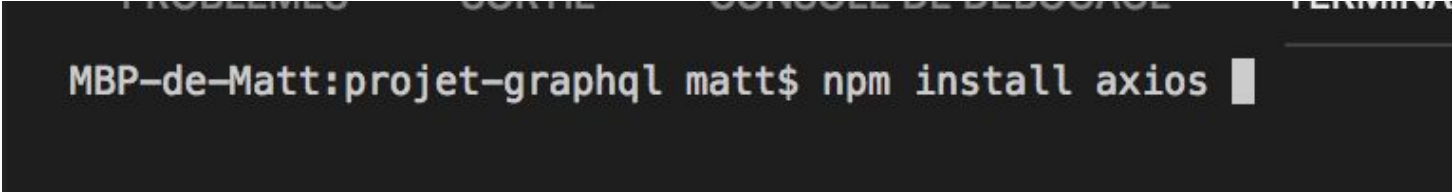


Le module axios nous permet de faire des requêtes http vers notre serveur

A screenshot of a terminal window with a dark background. At the top, there are four tabs labeled 'PROBLEMES', 'SOURCE', 'CONSOLE DE DEBUTAGE', and 'TERMINAL'. The 'TERMINAL' tab is active. The terminal shows the command 'npm install axios' being entered in a prompt that reads 'MBP-de-Matt:projet-graphql matt\$'. A white cursor is at the end of the command.

```
MBP-de-Matt:projet-graphql matt$ npm install axios
```

JS schema.js schem...

{} db.json

PROJET-GRAPHQL

node_modules

schemas

JS schema.js

{} db.json

{} package-lock.json

{} package.json

JS server.js

```

2  const graphql = require('graphql');
3  const axios = require('axios');
4  const{ // récupération des type en destructurant graphql
5      GraphQLObjectType,
6      GraphQLString,
7      GraphQLInt,
8      GraphQLSchema
9  } = graphql;
10
11
12  //création d'un type pour le user composé de trois type
13  const UserType = new GraphQLObjectType({
14      name : 'User',
15      fields : {
16          id : { type : GraphQLString },//il est important de faire attention a la case des propriété par rapport a la data
17          firstName : { type : GraphQLString },
18          age : { type : GraphQLInt}
19      }
20  });
21
22  //création d'une Root Query qui est notre point d'entrée
23  const RootQuery = new GraphQLObjectType({
24      name : 'RootQuery',
25      fields : { // le champs
26          user : { // le nom du champs de notre Root Query USER EST EGALEMENT UNE FONCTION.
27              type : UserType, // on fournit le type de user qui est notre UserType
28              args : {id : { type : GraphQLString}}, // <- on fournis un argument qui précise quel information peut recevoir notre user.
29              resolve(parentValue,args){//<-et ensuite on lui fournis une promesse qui signifie qu'est ce qu'il doit faire quand il a reçu l'id.
30                  return axios.get(`http://localhost:3000/users/${args.id}`).then((response)=>{ // la requête est une promesse, .then = quand tu a fini.
31                      return response.data;
32                  })
33              }
34          }
35      }
36  });
37  module.exports = new GraphQLSchema({ //permet d'exporter tout le schema
38      query : RootQuery // a partir de ce moment là Rootquery contient tout les informations de UserType
39  })

```

←

→

↺

localhost:4000/salutGraphQL?query=query%7B%0A%20%20%20%20user(id%3A"3")%20%7B%0A%20%20%20%20%20%20firstName%2C%0A%20%20%20%20%7D%0...

☆

GraphiQL

▶

Prettify

History

1 query{

2 user(id:"3") {

3 firstName,

4 }

5 }

6 }

▼ {

▼ "data": {

"user": {

"firstName": "Sarah"

}

}

}

Documentation Explorer

×

Q Search Schema...

A GraphQL schema provides a root type for each kind of operation.

ROOT TYPES

query: RootQuery