

COMPTE RENDU TP10-NODEJS

Objectif du TP : Création d'une API permettant de gérer une liste de chansons dans une base de données MySQL.

Liste des requêtes implémentées :

- /songs/
 - GET : Récupérer toutes les chansons dans la base ;
 - POST : Ajouter une musique à la base de données ;
 - DELETE : Supprimer toutes les chansons de la BDD.
- /songs/<id_search>
 - GET : Récupérer la chanson avec l'id correspondant ;
 - PUT : Mettre à jour la chanson d'id correspondant ;
 - DELETE : Supprime la chanson.
- /songs/artists?q=<artist_search>
 - GET : Récupérer les musiques qui correspondent à l'artiste.

Architecture du projet :

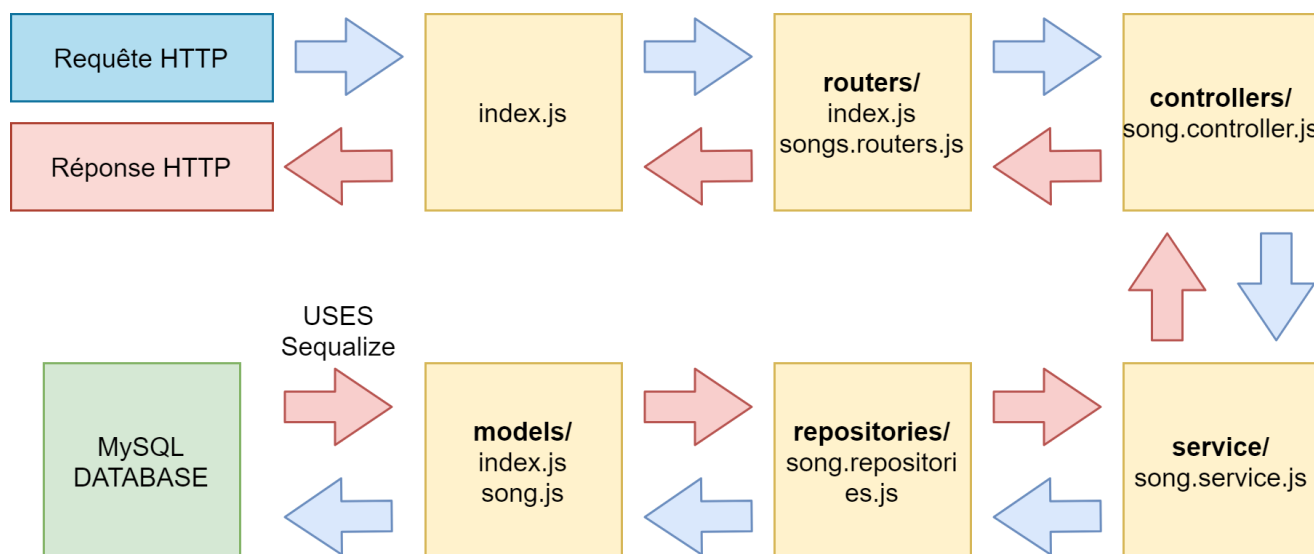


Schéma architecture projet Song

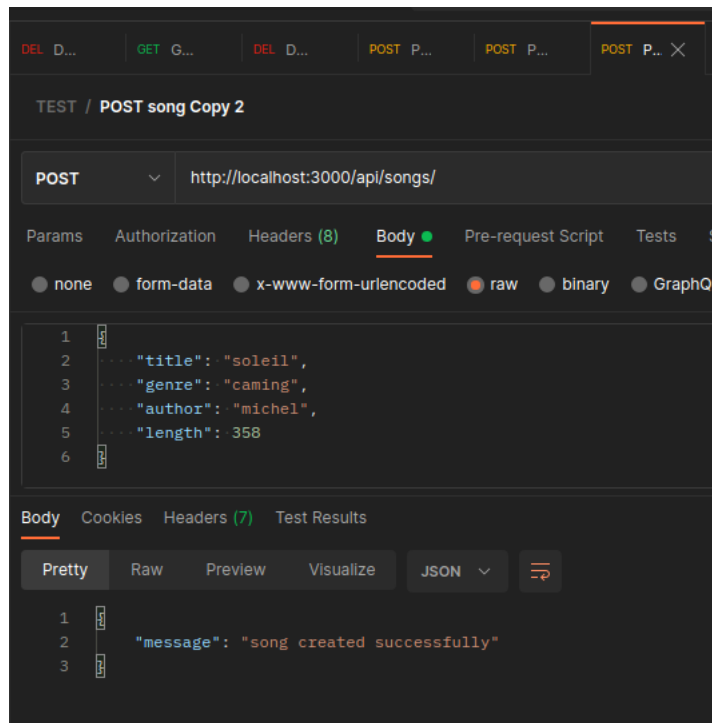
Implémentation d'un module validation :

Permet de vérifier la validité du body de la requête avant l'exécution de celle-ci. Agis lors de la réception dans le router pour une requête POST ou PUT.

Par exemple : Limite de taille de chaîne de caractères, obligation de type, valeur minimale / maximale.

Extraits de requêtes faites avec POSTMAN vers le serveur tournant sur localhost:

POST A SONG



TEST / POST song Copy 2

POST http://localhost:3000/api/songs/

Params Authorization Headers (8) Body Pre-request Script Tests

none form-data x-www-form-urlencoded raw binary GraphQL

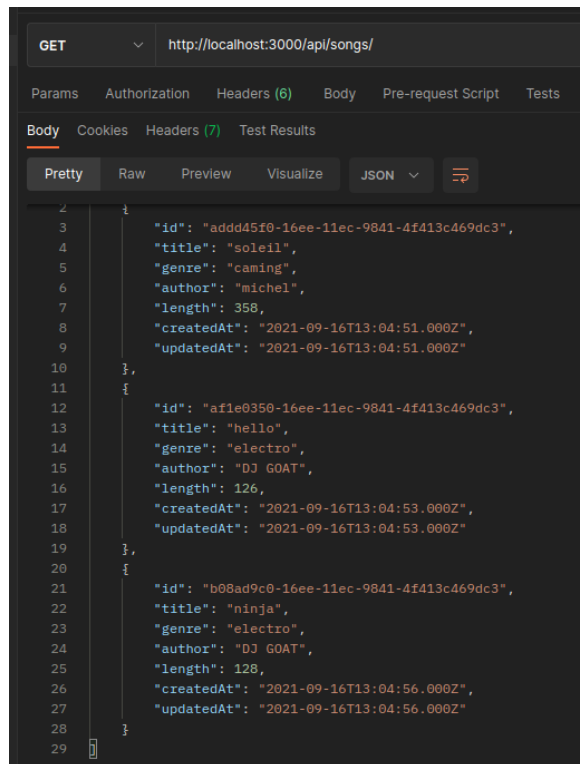
```
1 {
2   "title": "soleil",
3   "genre": "coming",
4   "author": "michel",
5   "length": 358
6 }
```

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "song created successfully"
3 }
```

GET ALL SONGS



GET http://localhost:3000/api/songs/

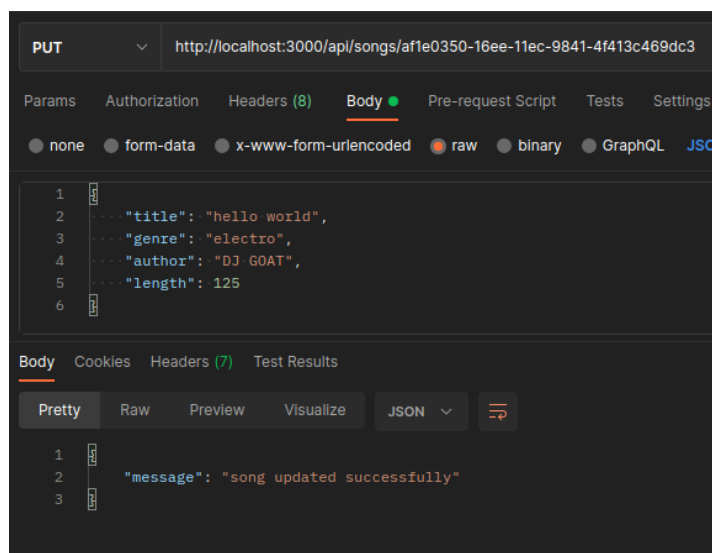
Params Authorization Headers (6) Body Pre-request Script Tests

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
2 {
3   "id": "add45f0-16ee-11ec-9841-4f413c469dc3",
4   "title": "soleil",
5   "genre": "coming",
6   "author": "michel",
7   "length": 358,
8   "createdAt": "2021-09-16T13:04:51.000Z",
9   "updatedAt": "2021-09-16T13:04:51.000Z"
10 },
11 {
12   "id": "af1e0350-16ee-11ec-9841-4f413c469dc3",
13   "title": "hello",
14   "genre": "electro",
15   "author": "DJ GOAT",
16   "length": 126,
17   "createdAt": "2021-09-16T13:04:53.000Z",
18   "updatedAt": "2021-09-16T13:04:53.000Z"
19 },
20 {
21   "id": "b08ad9c0-16ee-11ec-9841-4f413c469dc3",
22   "title": "ninja",
23   "genre": "electro",
24   "author": "DJ GOAT",
25   "length": 128,
26   "createdAt": "2021-09-16T13:04:56.000Z",
27   "updatedAt": "2021-09-16T13:04:56.000Z"
28 }
29 ]
```

UPDATE A SONG



PUT http://localhost:3000/api/songs/af1e0350-16ee-11ec-9841-4f413c469dc3

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSC

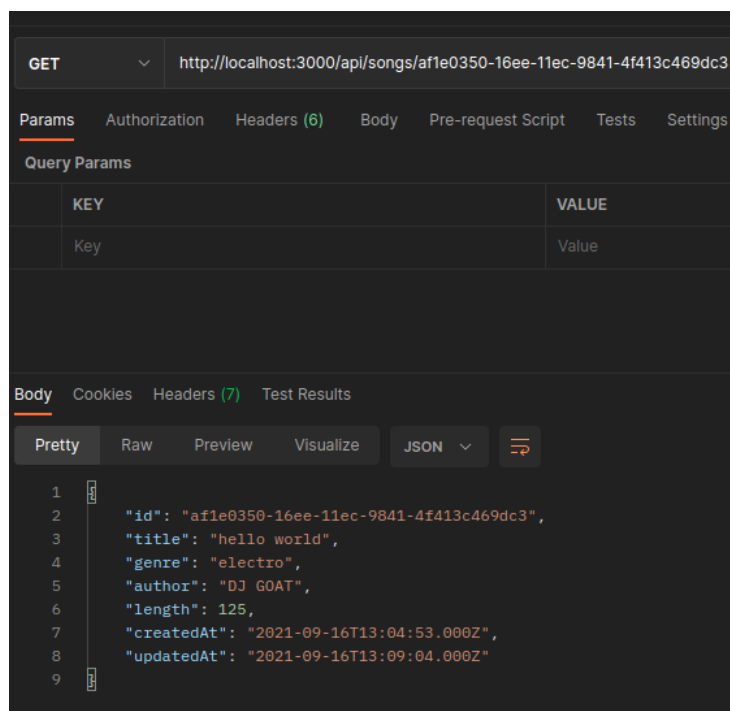
```
1 {
2   "title": "hello world",
3   "genre": "electro",
4   "author": "DJ GOAT",
5   "length": 125
6 }
```

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "song updated successfully"
3 }
```

GET A SONG (a check if update is ok)



GET http://localhost:3000/api/songs/af1e0350-16ee-11ec-9841-4f413c469dc3

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "af1e0350-16ee-11ec-9841-4f413c469dc3",
3   "title": "hello world",
4   "genre": "electro",
5   "author": "DJ GOAT",
6   "length": 125,
7   "createdAt": "2021-09-16T13:04:53.000Z",
8   "updatedAt": "2021-09-16T13:09:04.000Z"
9 }
```

GET ALL SONGS OF AN ARTIST

GET ▼ <http://localhost:3000/api/songs/artists?q=DJ GOAT>

Params ● Authorization Headers (6) Body Pre-request Script Tests

Query Params

KEY	VALUE
<input checked="" type="checkbox"/> q	DJ GOAT
Key	Value

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON ▼ ≡

```

1  {
2    "id": "af1e0350-16ee-11ec-9841-4f413c469dc3",
3    "title": "hello world",
4    "genre": "electro",
5    "author": "DJ GOAT",
6    "length": 125,
7    "createdAt": "2021-09-16T13:04:53.000Z",
8    "updatedAt": "2021-09-16T13:09:04.000Z"
9  },
10 {
11   "id": "b08ad9c0-16ee-11ec-9841-4f413c469dc3",
12   "title": "ninja",
13   "genre": "electro",
14   "author": "DJ GOAT",
15   "length": 128,
16   "createdAt": "2021-09-16T13:04:56.000Z",
17   "updatedAt": "2021-09-16T13:04:56.000Z"
18 }
19
20
```

POST A SONG WITH WRONG FORMAT

POST ▼ <http://localhost:3000/api/songs/>

Params Authorization Headers (8) **Body** ● Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL ▼

```

1  {
2    "title": "bad",
3    "genre": "songBad",
4    "author": "a very long string to check if validator works",
5    "length": 126
6  }

```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize HTML ▼ ≡

```

1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5    <meta charset="utf-8">
6    <title>Error</title>
7  </head>
8
9  <body>
10   <pre>ValidationError: Validation Failed</pre>
11 </body>
12
13 </html>

```

DELETE A SONG

DELETE ▼ <http://localhost:3000/api/songs/b08ad9c0-16ee-11ec-9841-4f413c469dc3>

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON ▼ ≡

```

1  {
2    "message": "song deleted successfully"
3  }

```

DELETE ALL SONGS

DELETE ▼ <http://localhost:3000/api/songs/>

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON ▼ ≡

```

1  {
2    "message": "songs deleted successfully"
3  }

```