

Rapport ShareLoc

Objectif

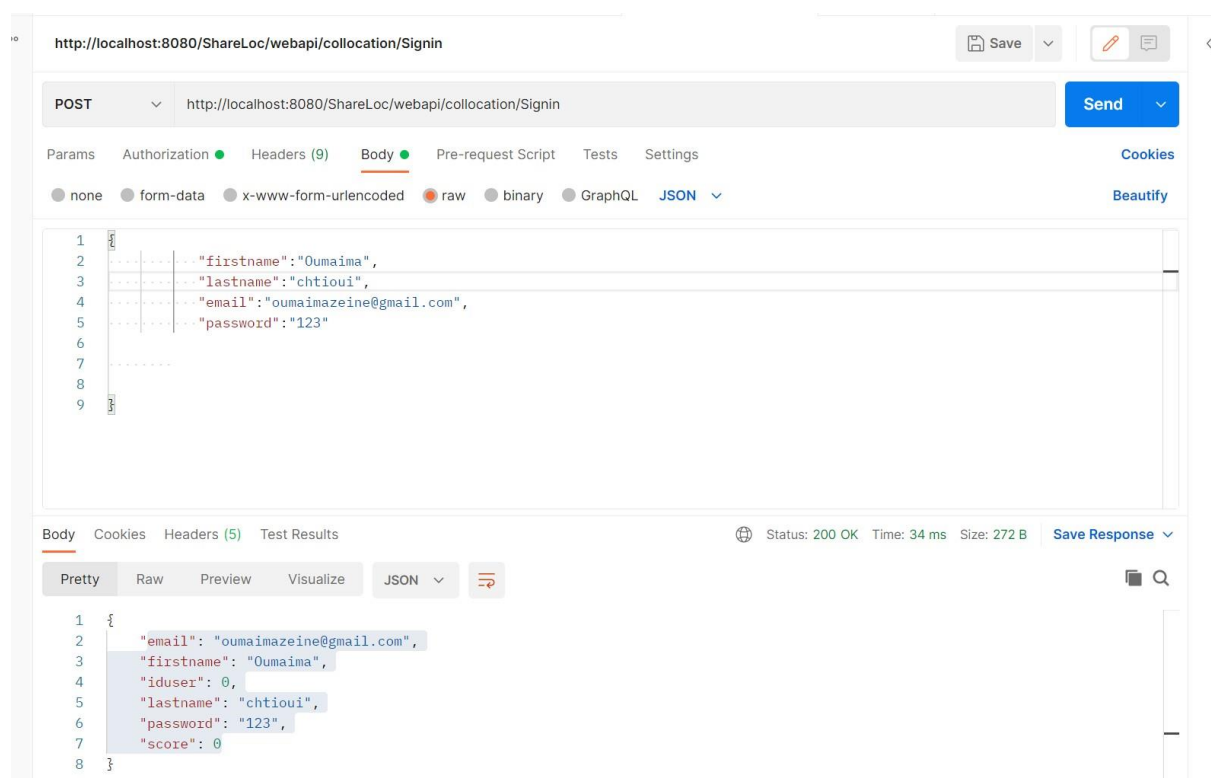
L'objectif de ce projet est de développer une plateforme permettant de s'échanger des services au sein d'une colocation ou d'une famille. Chaque colocation dispose de son propre espace au sein duquel les membres sont attachés.

Fonctionnalités

Inscription d'un utilisateur :

Pour s'inscrire, l'utilisateur passe en paramètres son prénom (firstname), nom (lastname), email et mot de passe (password).

Lors de l'inscription, le score de l'utilisateur est initialisé à zéro.



Authentification d'un utilisateur :

Pour s'authentifier, l'utilisateur passe en paramètres son email et son mot de passe avec la méthode GET.

Il recevra alors un **token** qui lui permettra de s'authentifier à chaque fois qu'il utilisera l'application. Ce token est généré aléatoirement en utilisant le nom de l'utilisateur et le timestamp avec un hachage aléatoire.

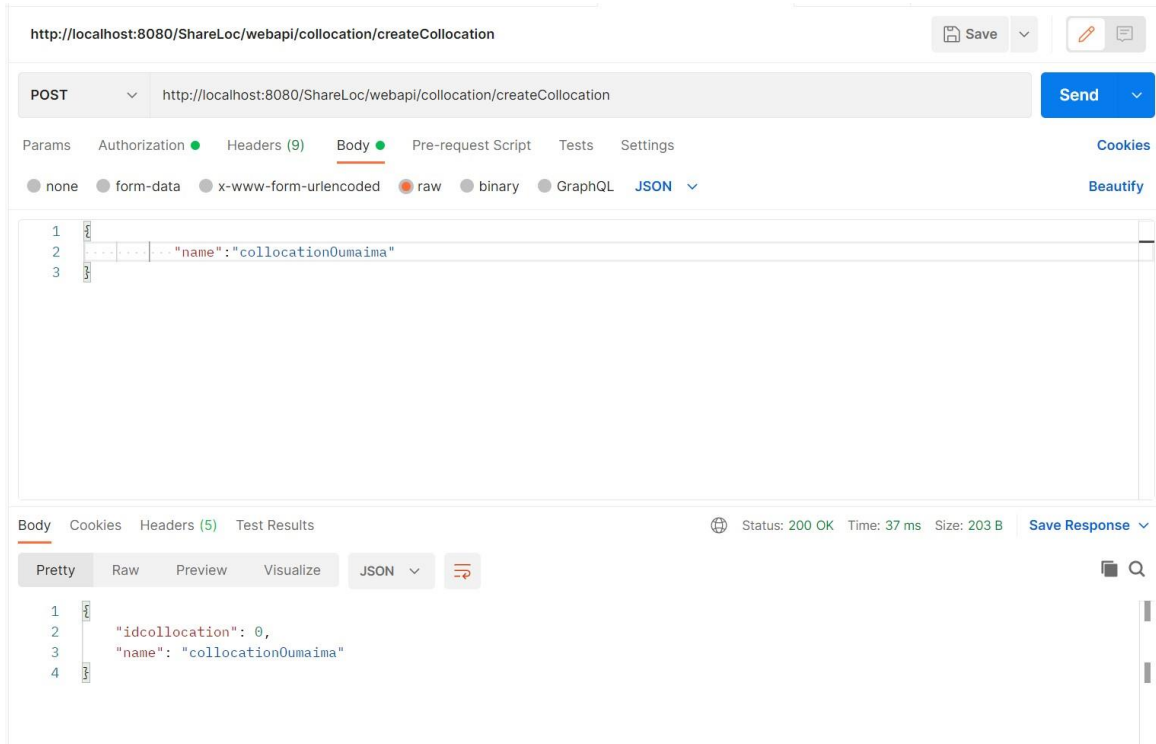
The screenshot displays a REST client interface with the following components:

- URL Bar:** `http://localhost:8080/ShareLoc/webapi/collocation/connect/oumaimazeine@gmail.com/123`
- Method:** `GET`
- Send Button:** A blue button labeled "Send".
- Request Body:** A message stating "This request does not have a body".
- Response Section:**
 - Status:** 200 OK
 - Time:** 250 ms
 - Size:** 191 B
 - Save Response:** A button to save the response.
- Response Body:**
 - Tab:** "Text" is selected.
 - Content:** A single line of text: `1 Token : U1cGbhsGZb9Eya9ncZG0Iyq7EXmyiKet`

Création d'un espace colocation :

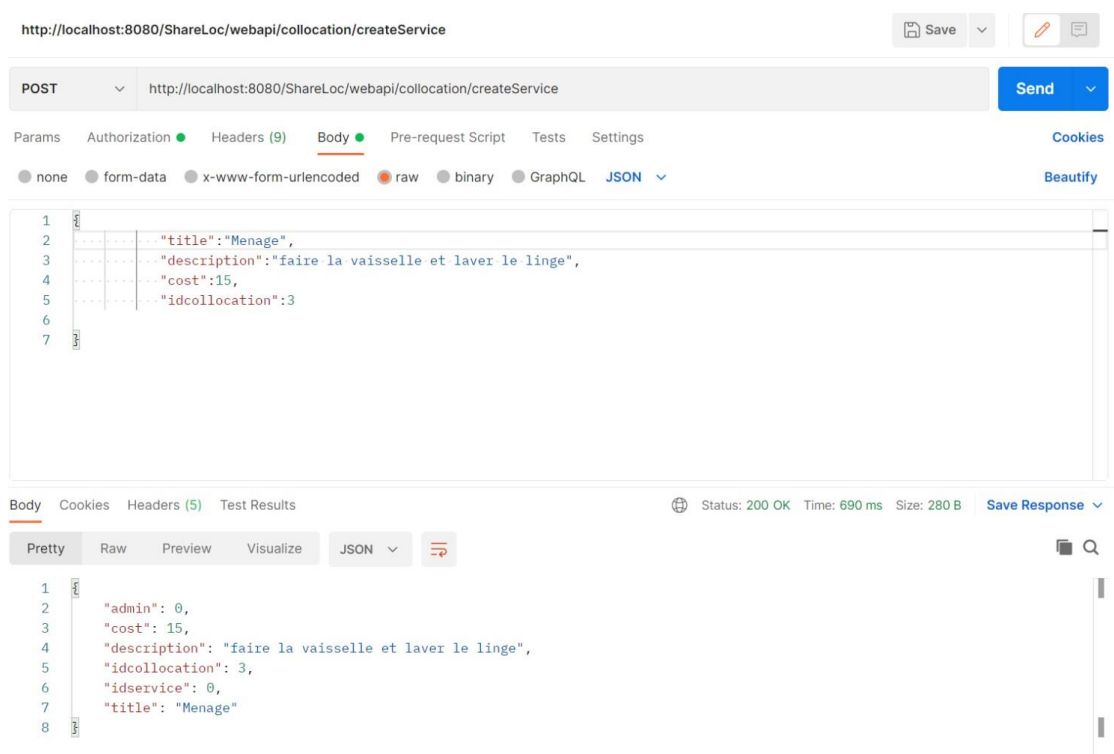
Pour créer une colocation, l'utilisateur lui passe en paramètres un nom avec la méthode POST.

Ce chemin est sécurisé de façon à ne pas permettre à une personne en dehors de l'application de la manipuler.



Création d'un service :

Pour créer un service, l'utilisateur lui passe en paramètres un titre, une description et un coût avec la méthode POST.



Suppression d'un utilisateur :

Pour supprimer une personne, l'utilisateur lui passe en paramètres son ID avec la méthode DELETE.

Untitled Request

DELETE http://localhost:8080/restapi/webapi/collocation/deleteUser

Send Save

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Code

TYPE: Bearer Token

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Token: wUCCAboc-Bor-iWKWOCTw4doGwffOAAP

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. [Learn more about variables](#)

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 567 ms Size: 185 B Save Response

Pretty Raw Preview Visualize Text

```
1 delete from user where iduser = '11'
```

On remarque après dans la base que cette ligne a bien été supprimée :

	iduser	email	password	firstname	lastname	score	token
<input type="checkbox"/> Edit Copy Delete	1	oumaima.chtioui@uha.fr	oumaima	chtioui	oumaima	0000000000	0
<input type="checkbox"/> Edit Copy Delete	2	Zeineb@uha.fr	oumaima	chtioui	oumaima	0000000000	0
<input type="checkbox"/> Edit Copy Delete	3	Zeineb@uha.fr	oumaima	chtioui	oumaima	0000000000	0
<input type="checkbox"/> Edit Copy Delete	4	Zeineb@uha.fr	oumaima	chtioui	oumaima	0000000000	0
<input type="checkbox"/> Edit Copy Delete	5	o	o	o	o	0000000000	0
<input type="checkbox"/> Edit Copy Delete	6	zdhpi	dzpihzd	zidh	ouzdg	0000000000	0
<input type="checkbox"/> Edit Copy Delete	7	zid	zdni	kzdzj	oumaimaa	0000000000	0
<input type="checkbox"/> Edit Copy Delete	8	jj	jj	jj	jj	0000000000	0
<input type="checkbox"/> Edit Copy Delete	9	eosj	ed,oj	zpdj	azrour	0000000000	0
<input type="checkbox"/> Edit Copy Delete	10	ZDCSZ	KZC.M	KZCDC	ZSMKZ	0000000000	0
<input type="checkbox"/> Edit Copy Delete	12	oumaimazeine@gmail.com	123	Oumaima	chtioui	0000000000	U1cGbhsGZb9Eya9ncZGOlyq7fXmyiKet
<input type="checkbox"/> Edit Copy Delete	13	hello@	zeinebzelleg	zeineb	zelleg	0000000000	NULL

Suppression d'un service :

Pour supprimer un service, l'utilisateur lui passe en paramètres son titre avec la méthode DELETE.

Seul l'utilisateur créateur de ce service a le droit de le supprimer.

The screenshot shows a REST client interface with the following details:

- Method:** DELETE
- URL:** `http://localhost:8080/restapi/webapi/collocation/deleteService/Menage`
- Body:** A JSON object: `{ "name": "Collocation de Zeineb" }`
- Response:** Status: 200 OK, Time: 824 ms, Size: 206 B. The response body contains the SQL query: `delete from service where title='Menage' and admin ='11'`

Suppression d'une colocation :

Pour supprimer une colocation, l'utilisateur lui passe en paramètres son nom avec la méthode DELETE.

Seul l'utilisateur créateur de cette colocation a le droit de la supprimer.

The screenshot shows a REST client interface with the following details:

- Method:** DELETE
- URL:** `http://localhost:8080/restapi/webapi/collocation/deleteCollocation/Collocation de Zeineb`
- Body:** A JSON object: `{ "name": "Collocation de Zeineb" }`
- Response:** Status: 200 OK, Time: 786 ms, Size: 225 B. The response body contains the SQL query: `delete from collocation where name='Collocation de Zeineb' and idUser ='11'`

Modification d'un utilisateur :

L'utilisateur peut modifier son email avec la méthode POST.

The screenshot shows a REST client interface with the following details:

- Request:**
 - Method: POST
 - URL: `http://localhost:8080/restapi/webapi/collocation/updateUser`
 - Body (JSON):

```
{  "email": "zeineb.zelleg@uha.fr"}
```
- Response:**
 - Status: 200 OK
 - Time: 553 ms
 - Size: 209 B
 - Body (JSON):

```
{  "email": "zeineb.zelleg@uha.fr",  "iduser": 0,  "score": 0}
```

Modification d'une collocation :

L'utilisateur peut modifier le nom d'une collocation avec la méthode POST.

Seul l'utilisateur créateur de cette collocation a le droit de la modifier.

The screenshot shows a REST client interface with the following details:

- Request:**
 - Method: POST
 - URL: `http://localhost:8080/restapi/webapi/collocation/updateCollocation`
 - Body (JSON):

```
{  "name": "Collocation de Zeineb"}
```
- Response:**
 - Status: 200 OK
 - Time: 1245 ms
 - Size: 206 B
 - Body (JSON):

```
{  "idcollocation": 0,  "name": "Collocation de Zeineb"}
```

On remarque après dans la base que le nom de la colocation a bien été modifié :

Options

				idCollocation	name	idUser
<input type="checkbox"/>	Edit	Copy	Delete	1	skiski	1
<input type="checkbox"/>	Edit	Copy	Delete	2	Collocation de Zeineb	11

↑ ☐ Check all With selected: Edit Copy Delete Export

☐ Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Query results operations

Print Copy to clipboard Export Display chart Create view

Modification d'un service :

L'utilisateur peut modifier un service en modifiant son titre, sa description et son coût avec la méthode POST.

Seul l'utilisateur créateur de ce service a le droit de le modifier.

Untitled Request

POST http://localhost:8080/restapi/webapi/collocation/updateService Send Save

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Code

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

```
1 {
2   "title": "Menage",
3   "description": "test",
4   "cost": 9
5 }
```

Body Cookies Headers (5) Test Results Status: 200 OK Time: 563 ms Size: 236 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "cost": 9,
3   "description": "test",
4   "idcollocation": 0,
5   "idservice": 0,
6   "title": "Menage"
7 }
```

On remarque après dans la base que le titre, la description et le coût du service ont bien été modifiés :

+ Options

↩

→

▼

idservice

title

description

cost

idcollocation

admin

☐

Edit

Copy

Delete

2

Menage test

9

1

11

↶

☐ Check all

With selected:

Edit

Copy

Delete

Export

☐ Show all

|

Number of rows:

25

▼

Filter rows:

Search this table

Query results operations

Print

Copy to clipboard

Export

Display chart

Create view

Ajout d'un utilisateur à une colocation :

Pour ajouter une personne à une colocation, l'utilisateur passe en paramètres le nom de la colocation et son mot de passe avec la méthode GET.

Pour ce faire, il faut :

1. Disposer du nom de la colocation.
2. Disposer du mail de la personne qu'on veut inviter.
3. Être le créateur de la colocation.

La liaison colocation - personne est faite dans une table CollocationUser où on stocke le nom de la colocation et le mail de la personne.