

## Compte-rendu Test technique

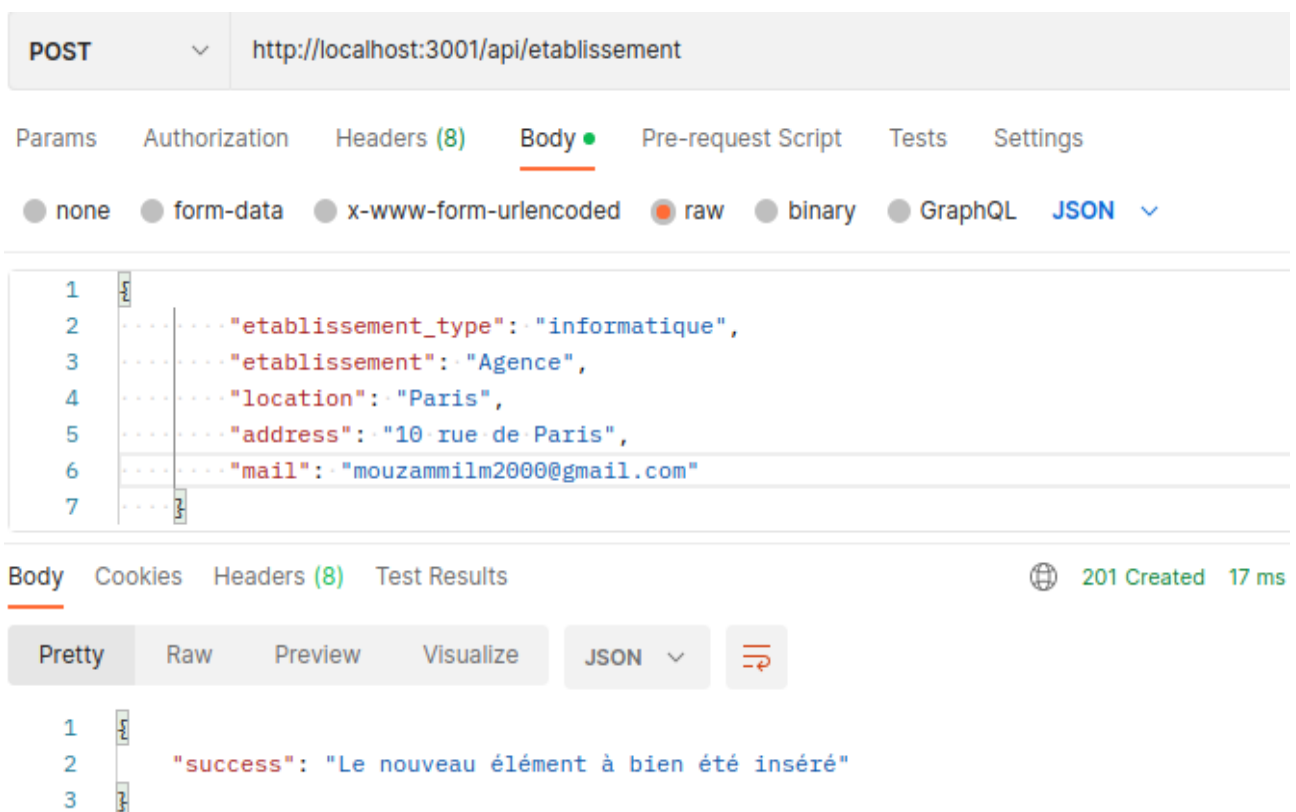
Dans la partie Front-end, j'utilise une bibliothèque JavaScript (ReactJS) et j'utilise axios pour récupérer des données API backend et je les affiche dans une table Chakra UI.

Dans la partie Back-end, j'utilise le framework ExpressJS pour traiter les requêtes HTTP et j'utilise express.json() pour envoyer des réponses. J'utilise une application Postman pour tester des API.

Exercice n°1 :

1) Création d'un élément avec vérification de la validité des valeurs :

J'utilise la méthode push() pour ajouter des valeurs à un tableau json. Nous pouvons donc vérifier avec la requête POST, que les valeurs ont bien été créées.



Front-End :

Nous pouvons donc vérifier ci-dessous que ces valeurs ont bien été récupérés avec axios et affichés sur une table Chakra UI.

1001

informatique

Agence

Paris

10 rue de Paris

mouzammim2000@gmail.com

Supprimer

## 2) Suppression d'un élément à l'aide de son ID :

J'utilise une méthode splice() pour modifier le contenu d'un tableau en retirant des éléments et en ajoutant de nouveaux éléments.

DELETE

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

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (8) Test Results ☒ 200 OK 93 ms 316 B

Pretty Raw Preview Visualize JSON

```
1  
2  "success": "L'élément a bien été supprimé."  
3
```

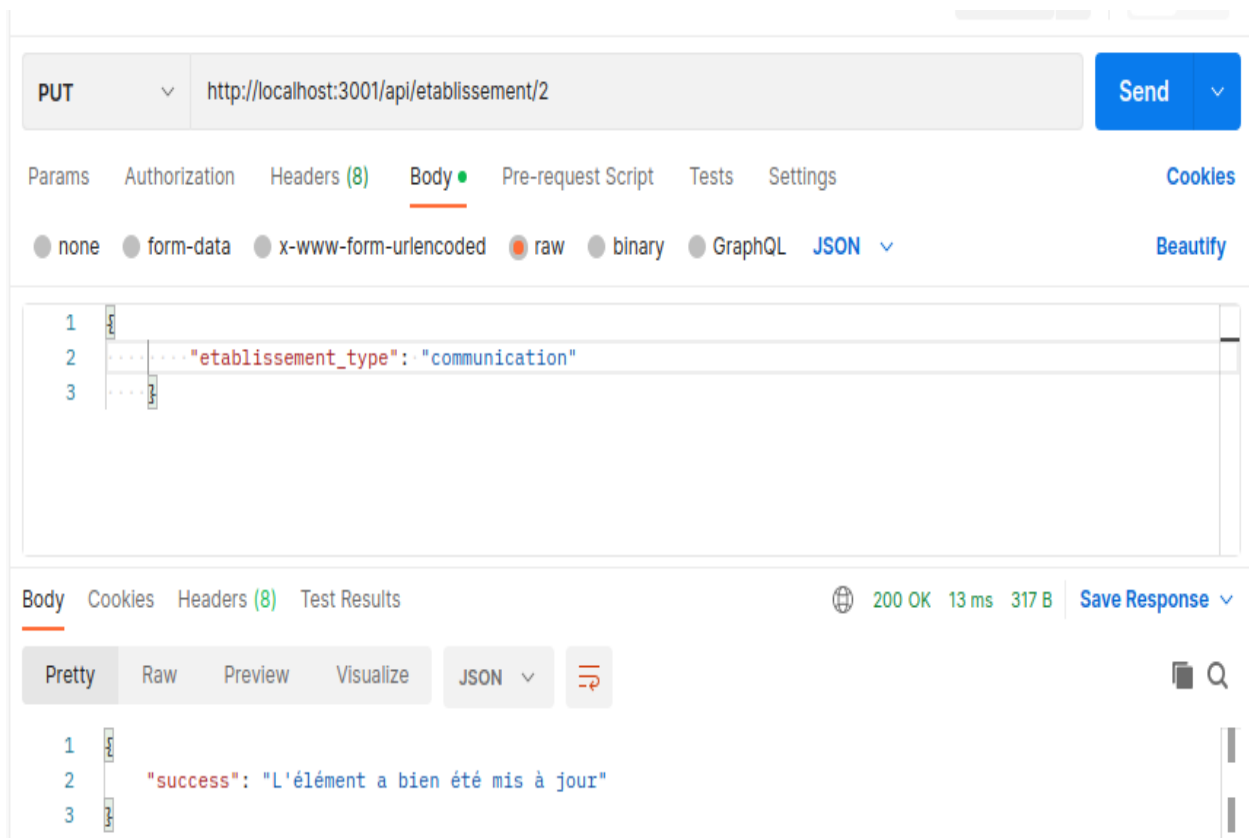
Front-End :

Nous pouvons donc vérifier que l'élément a bien été supprimés.

ID	ETABLISSEMENT_TYPE	ETABLISSEMENT	LOCATION	ADDRESS	MAIL	
2	Kids	Novadaq Technologies Inc	Alor Setar	4 Ilene Drive	cgowthorpe1@yahoo.com	Supprimer

## 3) Modification d'un élément à l'aide de son ID :

Je récupère les nouveaux valeurs saisies puis je mets à jour ces nouveaux valeurs.



Front-End :

Nous pouvons donc vérifier que l'élément a bien été modifié.

ID	ETABLISSEMENT_TYPE	ETABLISSEMENT	LOCATION	ADDRESS	MAIL	
2	communication	Novadaq Technologies Inc	Alor Setar	4 Ilene Drive	cgowthorpe1@yahoo.com	Supprimer

4) Obtention d'un élément à l'aide de son nom :

Dans l'URL avec la requête GET, lorsque je tape le nom d'établissement, nous pouvons voir les données json correspondant par le nom d'établissement.

The screenshot shows a web browser's developer tools interface. At the top, the URL bar displays `http://localhost:3001/api/utillsateur/Amplify Online Retail ETF`. Below the URL bar, the 'Send' button is visible. The 'Body' tab is selected, showing a list of request body items. The first item is expanded, showing a JSON response. The response is a JSON object with the following properties: `"id": 3`, `"etablissement_type": "Home"`, `"etablissement": "Amplify Online Retail ETF"`, `"location": "Luzon"`, `"address": "73 Village Green Pass"`, and `"mail": "mculp2@wired.com"`. The status bar at the bottom indicates a 200 OK status, a response time of 47 ms, and a size of 427 B. The 'Save Response' button is also visible.

```
{
  "id": 3,
  "etablissement_type": "Home",
  "etablissement": "Amplify Online Retail ETF",
  "location": "Luzon",
  "address": "73 Village Green Pass",
  "mail": "mculp2@wired.com"
}
```

## Exercice n°2 :

1) Obtention de la somme de tous les commerces d'un secteur d'activité donnée :

J'utilise une méthode `reduce()` qui traite chaque valeur d'une liste (de la gauche vers la droite) afin de la réduire à une seule valeur). Nous pouvons donc vérifier la somme de tous les secteurs d'activité.

```
GET http://localhost:3001/api/secteur Send
Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies
Body Cookies Headers (8) Test Results Status: 200 OK Time: 261 ms Size: 543 B Save Response
Pretty Raw Preview Visualize JSON
1 2 "Shoes": 49,
3 "Kids": 43,
4 "Home": 51,
5 "Baby": 46,
6 "Music": 45,
7 "Toys": 62,
8 "Tools": 51,
9 "Garden": 44,
10 "Health": 50,
11 "Computers": 49,
12 "Sports": 44,
13 "Books": 45,
14 "Grocery": 38,
15 "Movies": 43,
16 "Jewelry": 42,
17 "Beauty": 37,
18 "Automotive": 43,
19 "Outdoors": 44,
20 "Clothing": 52,
21 "Industrial": 46,
22 "Games": 36,
23 "Electronics": 40
24
```

2) Obtention de la somme de tous les commerces d'une ville donnée :

```
GET http://localhost:3001/api/location Send
Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies
Body Cookies Headers (8) Test Results Status: 200 OK Time: 150 ms Size: 13.27 KB Save Response
Pretty Raw Preview Visualize JSON
1 2 "Mojokerto": 1,
3 "Alor Setar": 1,
4 "Luzon": 1,
5 "L'Assomption": 1,
6 "Masis": 1,
7 "Karangmulyo": 1,
8 "Ongjin": 1,
9 "Panalingaan": 1,
10 "Beylagan": 1,
11 "Petrovsk": 1,
12 "Shancheng": 1,
13 "Dongjin": 1,
14 "Nārāyanganj": 1,
15 "Guinoaliuan": 1,
16 "Buštěhrad": 1,
17 "Ra-ngae": 1,
18 "Pantalowice": 1,
19 "Yanshang": 1,
20 "Selkirk": 1,
21 "Nimes": 1,
22 "Maunatlala": 1,
23 "Erba": 1,
24 "Kampong Chhnang": 1,
```

```

25     "Chahe": 1,
26     "Xianyuan": 1,
27     "Alicia": 1,
28     "Songping": 1,
29     "Osasco": 1,
30     "Bassano": 1,
31     "Putrajaya": 2,

```

3) Obtention de la somme de tous les commerces d'un secteur d'activité dans une ville donnée :

J'utilise la méthode `map()` pour créer un nouveau tableau avec les résultats de l'appel d'une fonction fournie, la méthode `filter()` pour remplir une condition déterminée par la fonction callback, et la méthode `reduce()` pour obtenir la somme.

The screenshot shows a REST client interface. At the top, a GET request is configured to `http://localhost:3001/api/secteur_ville`. Below the request bar, tabs for Params, Authorization, Headers (6), Body, Pre-request Script, Tests, and Settings are visible. The 'Body' tab is selected, and the content type is set to JSON. The response body is displayed in a text area with the number '1'. Below the response area, a detailed view of the response is shown. It includes a status bar indicating 'Status: 200 OK', 'Time: 127 ms', and 'Size: 56.77 KB'. The response body is a JSON array of three objects, each representing a location and its associated establishment types. The 'Pretty' tab is selected for the response view.

```

1  [
2    {
3      "location": "Mojokerto",
4      "etablissement_type": {
5        "Shoes": 1
6      }
7    },
8    {
9      "location": "Alor Setar",
10     "etablissement_type": {
11       "Kids": 1
12     }
13   },
14   {
15     "location": "Luzon",
16     "etablissement_type": {
17       "Home": 1
18     }
19   }
20 ]

```

```

545 |
546 |     "location": "Awilega",
547 |     "etablissement_type": {
548 |       "Movies": 1
549 |     }
550 |   },
551 |   {
552 |     "location": "San Miguel Dueñas",
553 |     "etablissement_type": {
554 |       "Automotive": 2
555 |     }
556 |   },
557 |   {
558 |     "location": "Huayang",
559 |     "etablissement_type": {
560 |       "Clothing": 1
561 |     }
562 |   },

```

## Exercice n°3 :

### 1) Obtention de tous les établissements en fonction d'une ville donnée :

Nous pouvons donc vérifier qu'on a obtenu tous les établissements en fonction d'une ville.

GET
http://localhost:3001/api/etablissement\_Ville
Send

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

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

1

Body
Cookies
Headers (8)
Test Results
Status: 200 OK
Time: 175 ms
Size: 68.19 KB
Save Response

Pretty
Raw
Preview
Visualize
JSON

```

1 | {
2 |   {
3 |     "location": "Mojokerto",
4 |     "etablissement": [
5 |       "Seattle Genetics, Inc."
6 |     ]
7 |   },
8 |   {
9 |     "location": "Alor Setar",
10 |    "etablissement": [
11 |      "Novadaq Technologies Inc"
12 |    ]
13 |   },
14 |   {
15 |     "location": "Luzon",
16 |     "etablissement": [
17 |       "Amplify Online Retail ETF"
18 |     ]
19 |   },

```

```

176     {
177         "location": "Putrajaya",
178         "etablissement": [
179             "Walker & Dunlop, Inc.",
180             "Choice Hotels International, Inc."
181         ]
182     },
183     {
184         "location": "Baochang",
185         "etablissement": [
186             "Wesco Aircraft Holdings, Inc."
187         ]
188     },
189     {
190         "location": "Dan Khun Thot",
191         "etablissement": [
192             "Wendy's Company (The)"
193         ]
194     },

```

2) Obtention de tous les établissements en fonction du secteur et de la ville :

GET
http://localhost:3001/api/etablissement\_secteur\_ville
Send

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

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

1

Body
Cookies
Headers (8)
Test Results
Status: 200 OK
Time: 321 ms
Size: 99.24 KB
Save Response

Pretty
Raw
Preview
Visualize
JSON

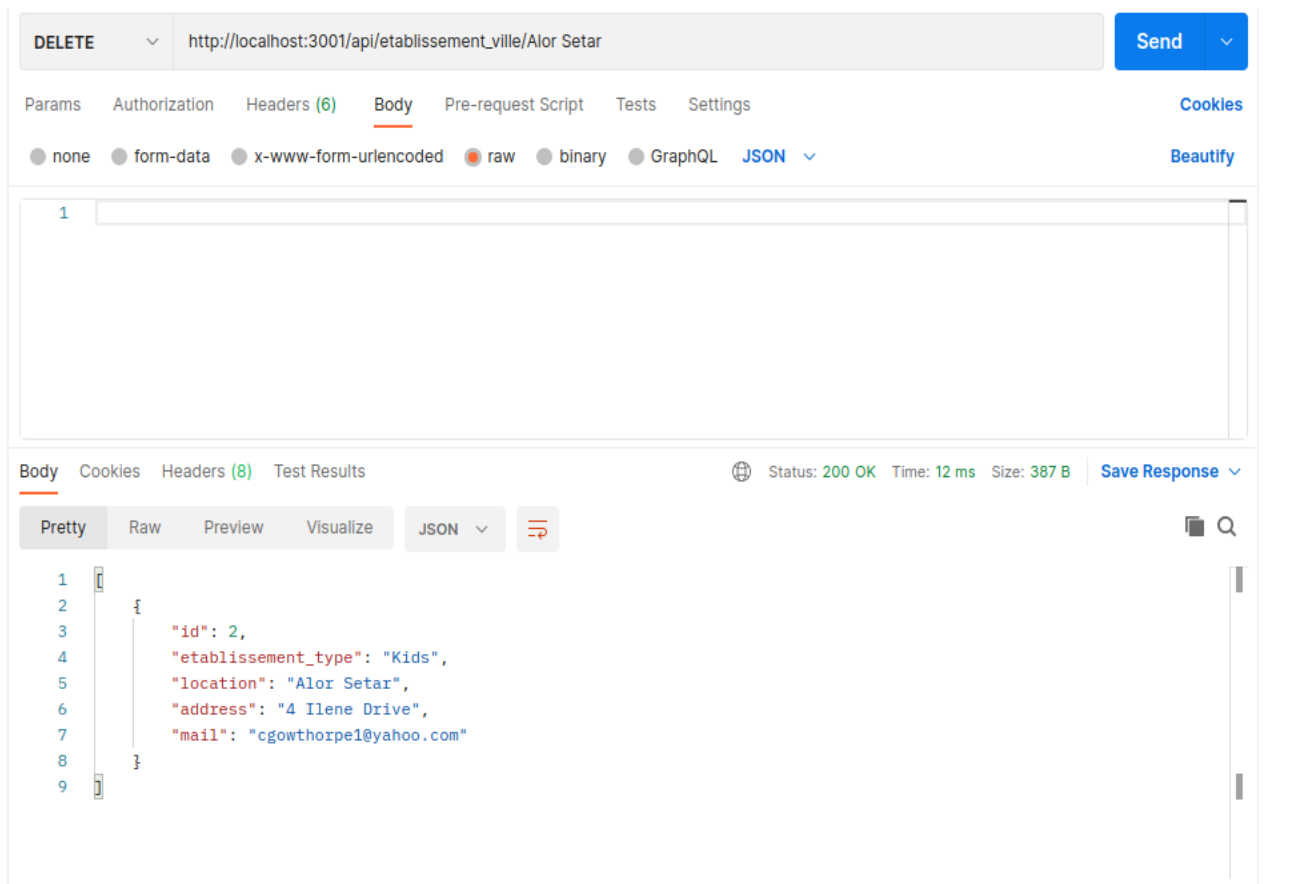
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19

{
 "location": "Mojokerto",
 "etablissement": [
 "Seattle Genetics, Inc."
 ],
 "etablissement\_type": [
 "Shoes"
 ]
},
{
 "location": "Alor Setar",
 "etablissement": [
 "Novadaq Technologies Inc"
 ],
 "etablissement\_type": [
 "Kids"
 ]
},



3) Suppression de tous les établissements d'une ville :

J'utilise une opérateur delete pour retirer une propriété d'un objet.

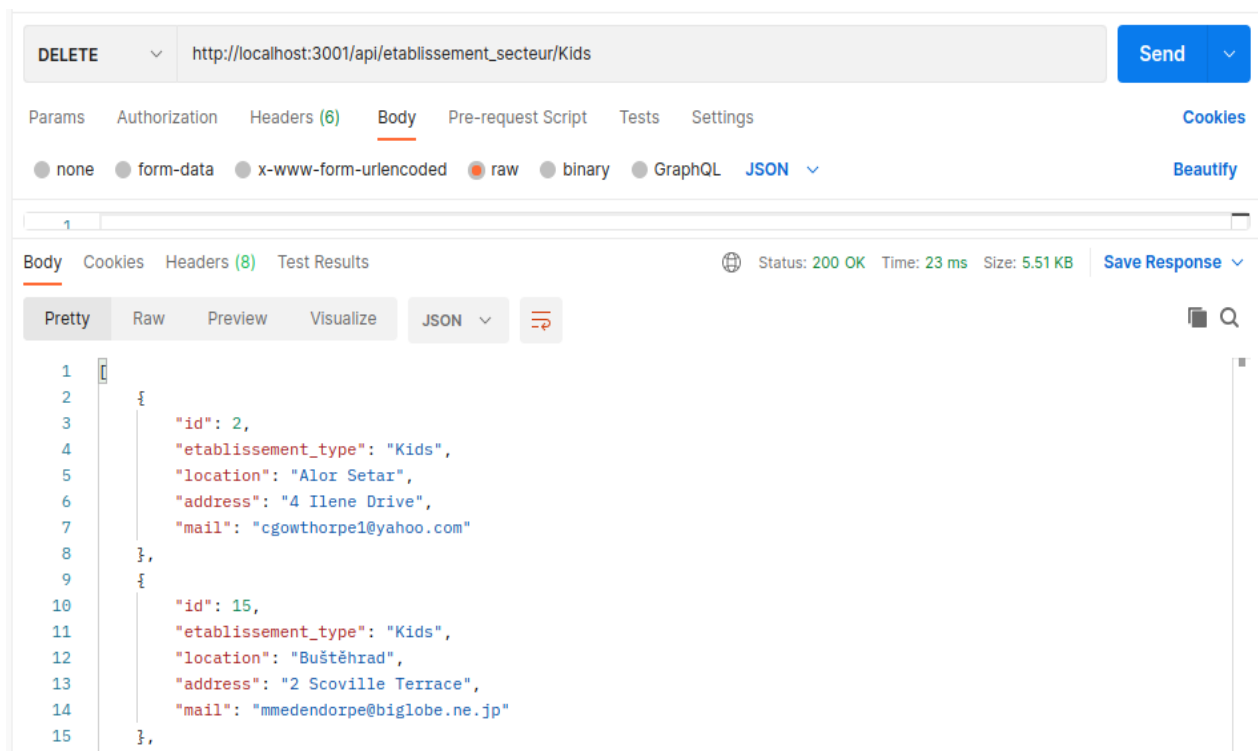


Front-End :

Nous pouvons voir que la valeur ETABLISSEMENT a bien été supprimé.

ID	ETABLISSEMENT_TYPE	ETABLISSEMENT	LOCATION	ADDRESS	MAIL	
2	Kids		Alor Setar	4 Ilene Drive	cgowthorpe1@yahoo.com	Supprimer

#### 4) Suppression de tous les établissements d'un secteur d'activité :



#### Front-End :

Nous pouvons vérifier que dans l'ID 1 et 15, les valeurs de l'établissement ont été supprimés.

ID	ETABLISSEMENT_TYPE	ETABLISSEMENT	LOCATION	ADDRESS	MAIL	
2	Kids		Alor Setar	4 Ilene Drive	cgowthorpe1@yahoo.com	Supprimer
15	Kids		Buštěhrad	2 Scoville Terrace	mmedendorpe@biglobe.ne.jp	Supprimer

#### Exercices Front-End :

##### - Récupération des données API avec axios :

Je récupère les données d'API avec la requête GET, puis j'affiche sur une table Chakra UI. De plus, j'ai créé un bouton Supprimer avec un fond de couleur pour supprimer les données de Back-End. Nous pouvons vérifier ci-dessous :

ID	ETABLISSEMENT_TYPE	ETABLISSEMENT	LOCATION	ADDRESS	MAIL	
2	Kids		Alor Setar	4 Ilene Drive	cgowthorpe1@yahoo.com	Supprimer
3	Home	Amplify Online Retail ETF	Luzon	73 Village Green Pass	mculp2@wired.com	Supprimer
4	Baby	FBL Financial Group, Inc.	L'Assomption	87 Westerfield Way	swalenta3@ucsd.edu	Supprimer
5	Music	First Financial Corporation Indiana	Masis	7 Longview Junction	nwickwar4@tiny.cc	Supprimer
6	Toys	Apogee Enterprises, Inc.	Karangmulyo	8 Talisman Place	whillett5@vimeo.com	Supprimer

- Suppression des données Back-End avec un bouton :

Pour supprimer des données Back-End, j'utilise axios avec la requête DELETE et je fais lier le bouton à l'API pour supprimer une ligne purement graphique. Nous pouvons voir ci-dessous que lorsqu'on clique sur le bouton, on a bien supprimé l'ID 2 et 3.

ID	ETABLISSEMENT_TYPE	ETABLISSEMENT	LOCATION	ADDRESS	MAIL	
4	Baby	FBL Financial Group, Inc.	L'Assomption	87 Westerfield Way	swalenta3@ucsd.edu	Supprimer
5	Music	First Financial Corporation Indiana	Masis	7 Longview Junction	nwickwar4@tiny.cc	Supprimer
6	Toys	Apogee Enterprises, Inc.	Karangmulyo	8 Talisman Place	whillett5@vimeo.com	Supprimer
7	Tools	Village Super Market, Inc.	Ongjin	215 Norway Maple Way	sdownie6@sogou.com	Supprimer
8	Home	Internet Initiative Japan, Inc.	Panalingaan	08475 Alpine Alley	egritsaev7@cocolog-nifty.com	Supprimer

## Conclusion :

Ce test était vraiment très intéressant pour moi, j'ai appris encore beaucoup de choses en React, Express. Dans la partie Back-End, j'ai appris à manipuler des données json avec des requêtes différentes (GET, POST, PUT, DELETE). Dans la partie Front-End, j'ai réussi à récupérer des données API avec axios méthode GET et à afficher sur une table Chakra UI, de plus, à supprimer des données Back-End avec un bouton.