
 Se former autrement HONORIS UNITED UNIVERSITIES	Année Universitaire : 2023 - 2024 
Atelier n° 3/4: <h2 style="text-align: center;">Utilisation d'Axios avec React</h2>	

Objectif:

- Consommer des services web en utilisant json-server et le client HTTP Axios.

Prérequis

- Connaissance de base de React.
- Node.js installé sur votre ordinateur.

Partie 1: Introduction à Axios

1. Définition

Axios est une bibliothèque JavaScript populaire utilisée pour envoyer des requêtes HTTP à partir d'un navigateur ou un serveur Node.js. Elle est souvent utilisée dans le développement d'applications web pour récupérer des données à partir de serveurs distants, envoyer des données vers un serveur ou communiquer avec des API RESTful.

Axios est une alternative à la méthode native « *fetch* » de JavaScript pour effectuer des requêtes HTTP. Elle offre une interface plus simple et plus conviviale pour gérer les requêtes HTTP et les réponses.

2. Comment installer Axios avec React ?

Pour utiliser Axios dans un projet React, nous allons suivre les étapes suivante :

Etape1 : Installer Axios dans votre projet à l'aide de npm ou yarn :

npm install axios ou *yarn add axios*

Etape2 : Dans le fichier où vous souhaitez utiliser Axios, importez-le en haut du fichier comme suit :

import axios from 'axios';

Etape3 : Pour effectuer des requêtes HTTP GET, POST, PUT, DELETE, etc., vers des serveurs distants, nous allons utiliser Axios en se basant sur la syntaxe suivante :

```
axios.get('https://api.example.com/data').    then(response  
=> {  
  
    // Gérer la réponse ici  
  
}).catch(error => {  
  
    // Gérer les erreurs ici  
  
});
```

Avec :

axios.get(url) : permet de retourner une promesse de la réponse de la requête auprès de l'url donné.

Partie 2: Travail demandé

Dans le but de terminer la gestion des événements, nous allons implémenter la récupération des appels d'API en utilisant le serveur JSON.

- 1) Installer json-server en utilisant la commande :

npm install -g json-server ou yarn add -g json-server

- 2) Mettre le fichier **db.json** dans un dossier appelé **api** sous **src**.
- 3) Démarrer le serveur JSON en exécutant la commande suivante sous le répertoire **src/api**:

json-server --watch db.json --port 3001

```
\{^_^}/ hi!
```

```
Loading db.json  
Done
```

Resources

```
http://localhost:3001/events
```

Home

```
http://localhost:3001
```

```
Type s + enter at any time to create a snapshot of the database  
Watching...
```

- 4) Créer un nouveau fichier appelé « **api.js** » sous un nouveau dossier « **service** » sous « **src** ».

Dans le fichier « **api.js** », vous allez mettre ce contenu :

```
import axios from "axios";  
  
const url = "http://localhost:3001/events";  
  
export const getAllEvents = async (id) => {  
  id = id || "";  
  return await axios.get(`${url}/${id}`);  
};  
  
export const addEvent = async (event) => {  
  return await axios.post(url, event);  
};  
  
export const editEvent = async (id, event) => {  
  return await axios.put(`${url}/${id}`, event);  
};  
  
export const deleteEvent = async (id) => {  
  return await axios.delete(`${url}/${id}`);  
};
```

Remarque : Une fois qu’Axios a été importé dans la librairie, nous allons nous rendre dans le fichier où nous souhaitons faire notre requête API. Nous allons ensuite créer quatre méthodes Asynchrones :

- **getallEvents()** : permet de récupérer la liste de tous les évènements depuis une API.
 - **addEvent(event)** : permet d'ajouter un nouvel évènement à l'API. Elle prendra les détails de l'évènement en tant que paramètre et les enverra à l'API pour les ajouter.
 - **editEvent(id,event):** permet de modifier un événement existant dans une API en utilisant son identifiant (ID) et les nouvelles données du l'évènement.
 - **deleteEvent(id):** permet de supprimer un événement spécifié de l'API en utilisant son identifiant (ID)
- 5) Faire les changements nécessaires dans le fichier « **Events.jsx** » pour faire l’appel de la liste des évènements.
- 6) Faire les changements nécessaires dans le fichier « **Event.jsx** » pour faire l’appel d’un événement selon son identifiant.

Remarque : Si l'événement n'existe pas , un message “**Event does not exist**” va être affiché.

- 7) Créer un nouveau composant appelé « **AddEvent.jsx** » dans le dossier **Components** où vous allez implémenter un formulaire d’ajout d’un événement avec l'invocation du service web d’ajout et ajouter la route nécessaire.

Remarque :

- En cliquant sur le bouton **Add an Event**, vous allez rediriger l'utilisateur vers la page de la liste des évènements (**utiliser le hook useNavigate**).

Add a new Event to your Event List

Name

Description

Price

Number of Tickets

Image

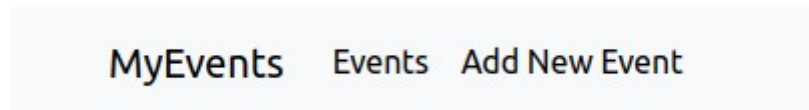
Browse...

No file selected.

Add an Event

Cancel

- 8) Ajouter un bouton **Add new Event** dans le composant « **NavigationBar** » pour accéder à l'interface d'ajout.



- 9) Créer un nouveau composant appelé « **UpdateEvent.jsx** » dans le dossier **Components** où vous allez implémenter un formulaire de mise à jour d'un événement bien déterminé avec l'invocation du service web de la modification.

Modify Festival de la médina de Tunis

Name

Festival de la médina de Tunis

Description

Le lorem ipsum est, en imprimerie, une suite de mots sans signification utilisée à titre provisoire pour calibrer une mise en page

Price

15

Number of Tickets

4

Image

Browse... No file selected.

Update Cancel

- 10) Ajouter un bouton « **Update** » dans le fichier « **Event.jsx** » pour accéder à l'interface de modification et faire les changements nécessaires dans le fichier « **App.jsx** » pour ajouter la route du composant de la modification d'un événement.



11) Ajouter un bouton **Delete** dans le fichier « **Event.jsx** ».



Festival international de Carthage

Price : 30

Number of tickets : 10

Number of participants : 10

[Like](#) [Book an event](#)

[Update](#) [Delete](#)



Festival de la médina de Tunis

Price : 15

Number of tickets : 4

Number of participants : 30

[Like](#) [Book an event](#)

[Update](#) [Delete](#)

12) Implémenter la fonction de la suppression d'un événement en invoquant le service web de suppression.