

# Node JS

## Développement d'applications Web

**Intervenant : Jean-Frédéric VINCENT**





Module 8

# Communiquer avec une API avec Axios et Fetch

# Communiquer avec une API avec Axios et Fetch

Mise en place d'Axios

Ajouter avec POST

Modifier avec PUT

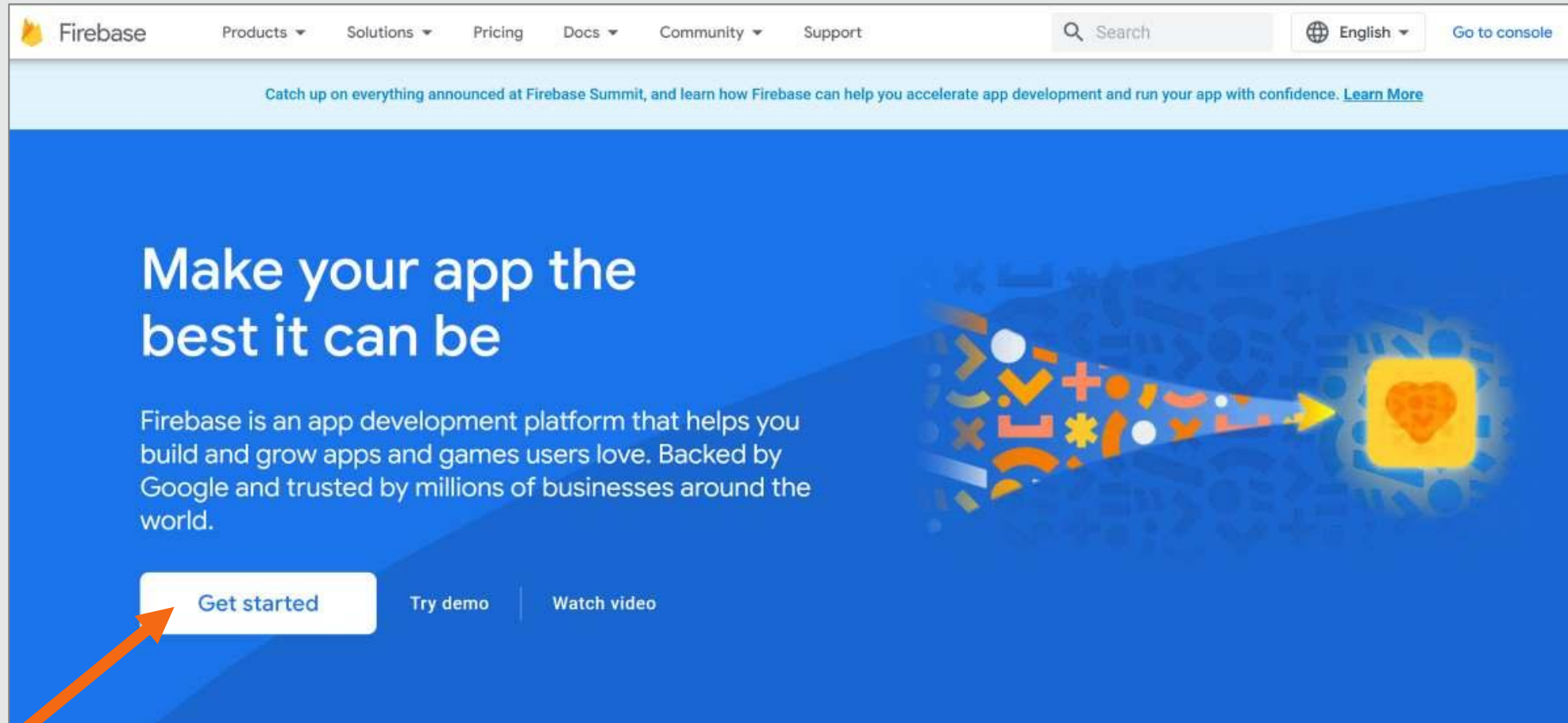
Modifier avec PATCH

Enlever avec DELETE

Mise en place avec fetch POST / PUT / PATCH et DELETE

# Communiquer avec une API avec Axios et Fetch

## Firestore



# Communiquer avec une API avec Axios et Fetch

## Firestore pricing

pricing

Gratuit



Realtime Database			
Simultaneous connections ?	100		200k/database
GB stored	1 GB		\$5/GB
GB downloaded	10 GB/month		\$1/GB
Multiple databases per project	×		✓

# Communiquer avec une API avec Axios et Fetch

## Firestore

### Créer un projet



× Créer un projet(Étape 1 sur 3)

**Commençons par donner un nom à votre projet ?**

**Saisissez le nom de votre projet**

☐ J'accepte les [Conditions d'utilisation de Firebase](#)

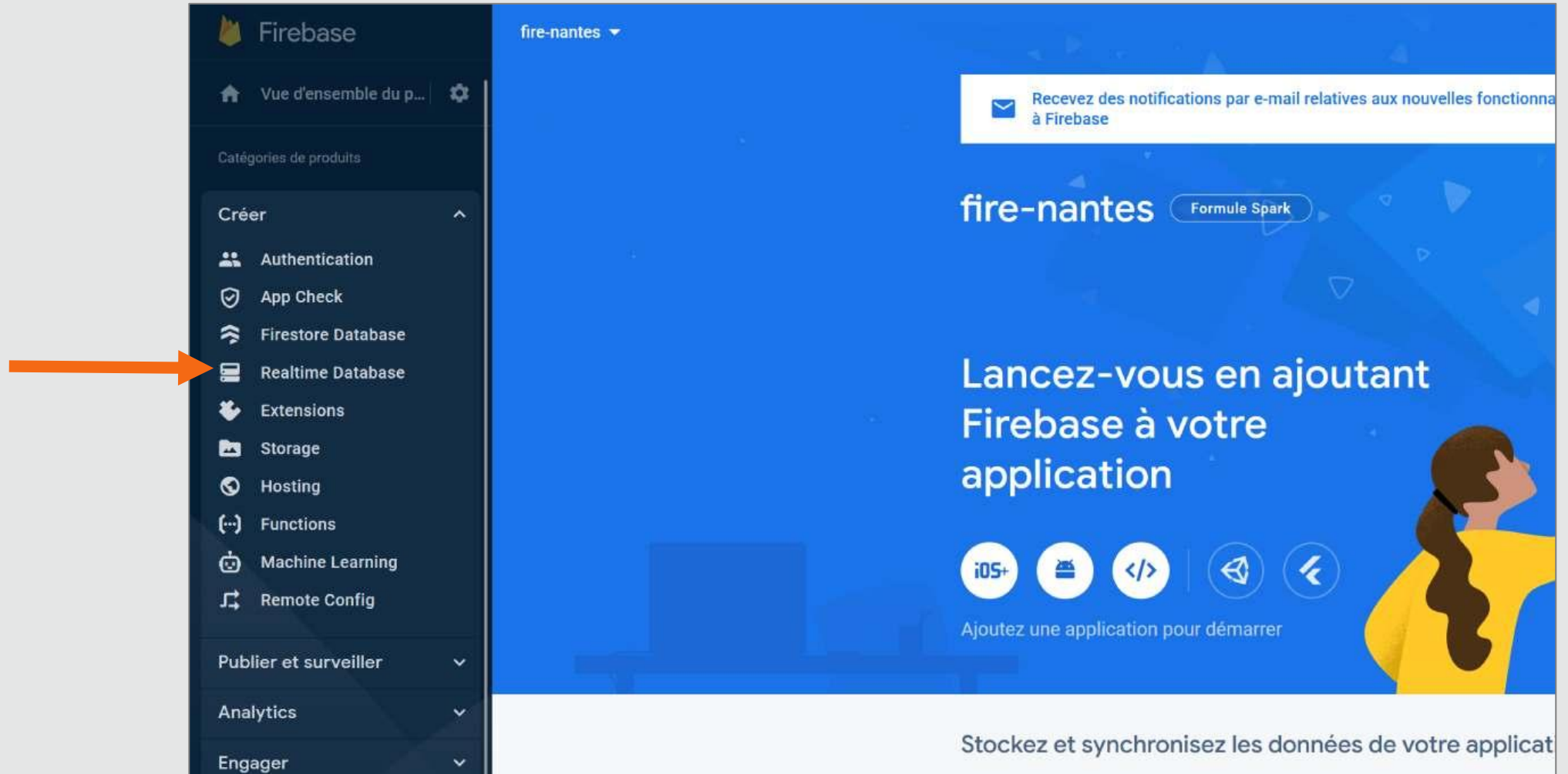
☐ Je confirme que je n'utiliserai Firebase que pour mes activités commerciales, mon entreprise, mes créations ou ma profession.

Continuer



Communiquer avec une API avec Axios et Fetch

# Créer un service : Realtime Database



Communiquer avec une API avec Axios et Fetch

# Créer un service : Realtime Database

On Enlève l'authentification

Attention tout le monde peut lire, écrire , effacer ....

Penser à remettre **false** à la fin du TP



### Configurer une base de données

1 Options de la base de données

2 Règles de sécurité

Après avoir défini la structure de vos données, vous devrez spécifier des règles pour les sécuriser.  
[En savoir plus](#)

☐ Commencer en **mode verrouillé**  
Par défaut, vos données sont privées. L'accès client en lecture/écriture ne sera autorisé qu'en fonction de vos règles de sécurité.

☒ Démarrer en **mode test**  
Par défaut, vos données sont publiques pour permettre une configuration rapide. Toutefois, vous devez modifier vos règles de sécurité dans les 30 jours pour autoriser l'accès client en lecture/écriture sur le long terme.

```
{
  "rules": {
    ".read": "now < 1671922800000", // 2022-12-25
    ".write": "now < 1671922800000", // 2022-12-25
  }
}
```

! Par défaut, les règles de sécurité en mode test autorisent tout utilisateur disposant de la référence de votre base de données à afficher, modifier et supprimer toutes les données qu'elle contient pendant les 30 prochains jours

Annuler

Activer

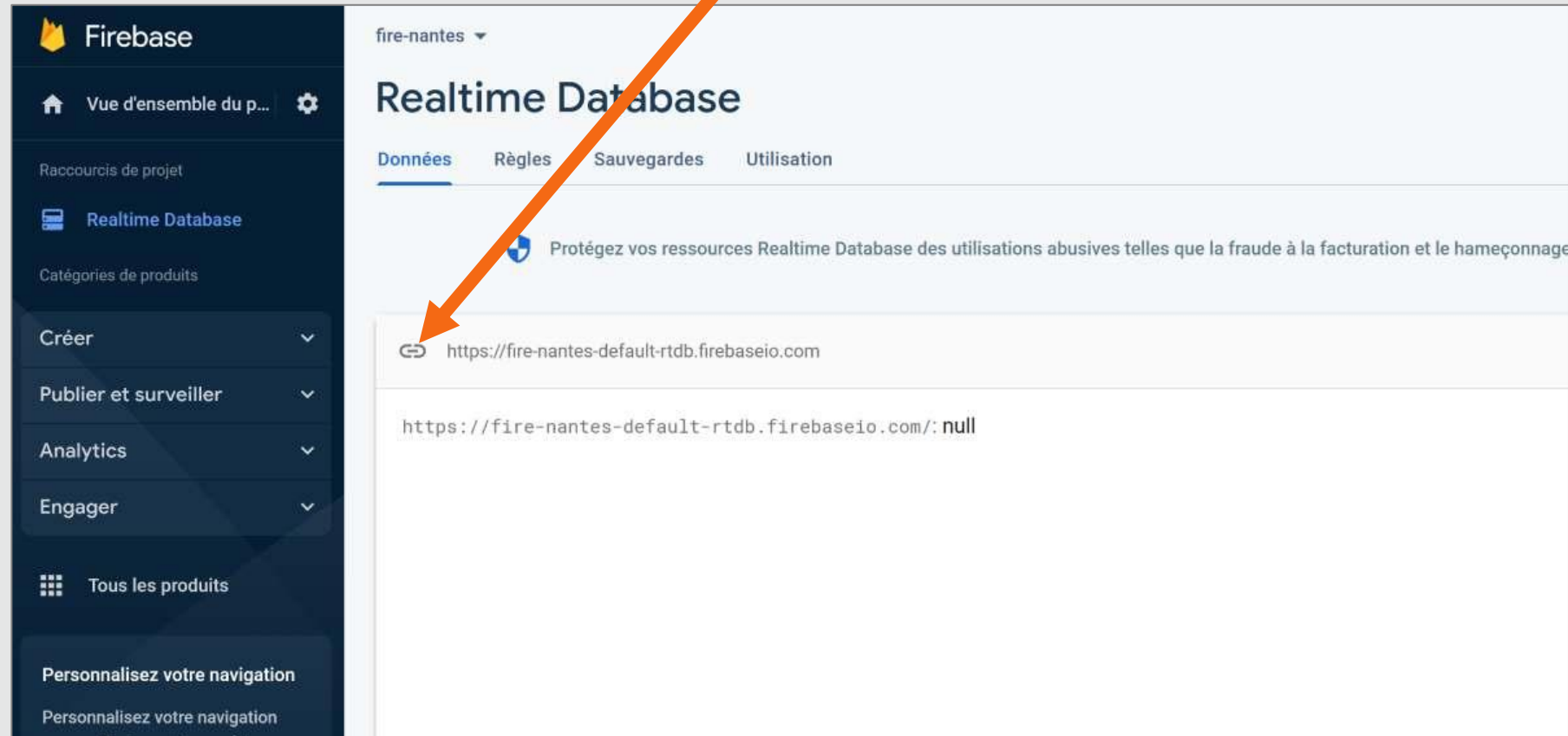


Communiquer avec une API

# Créer un service : Realtime Database

Copier l'URL et c'est fini !

Backend = NO Backend



The screenshot shows the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo and navigation options like 'Vue d'ensemble du p...', 'Realtime Database', and 'Créer'. The main panel on the right is titled 'fire-nantes Realtime Database' and has tabs for 'Données', 'Règles', 'Sauvegardes', and 'Utilisation'. An orange arrow points from the top right towards the URL 'https://fire-nantes-default-rtdb.firebaseio.com' displayed in the 'Données' tab. Below this URL, a text input field contains the string 'https://fire-nantes-default-rtdb.firebaseio.com/: null'.

# Communiquer avec une API avec Axios et Fetch

## Axios

- Pour utiliser Axios, il faut charger une librairie
  - Nous allons donc utiliser le CDN d'Axios
- Axios est plus simple à utiliser que Fetch pour les méthodes :
  - POST
  - PUT
  - PATCH
  - DELETE

The Axios logo is displayed in a bold, purple, sans-serif font. It consists of the letters 'A', 'X', 'I', 'O', and 'S' in a spaced-out arrangement. The letter 'I' is stylized with a small downward-pointing arrow integrated into its vertical stroke.

# Communiquer avec une API avec Axios et Fetch

## Axios GET

- La méthode **GET** Lecture
  - On économise une ligne par rapport à Fetch

```
axios.get(url)
```

```
const dbFire = 'https://alpha-javascript-default-rtdb.europe-west1.firebaseio.com/';  
const noeud = 'films';
```

```
const url = `${dbFire}${noeud}.json`;  
const response = await axios.get(url);  
console.log(response.data);
```

**A X I O S**

# Communiquer avec une API avec Axios et Fetch

## Axios POST

- La méthode **POST** ajouter un nouvel enregistrement
  - Pas besoin de sérialiser notre objet envoyé

```
axios.post(url, objet)
```

```
const dbFire = 'https://alpha-javascript-default-rtdb.europe-west1.firebaseio.com/';  
const noeud = 'films';
```

```
const url = `${dbFire}${noeud}.json`;  
const film = { name: "The Matrix", year: "1999" };  
const response = await axios.post(url, film);  
console.log(response.data);
```

AXIOS

# Communiquer avec une API avec Axios et Fetch

## Axios PATCH

- La méthode **PATCH** modifier un enregistrement
  - Le PATCH permet de mettre à jour que certains attributs

```
axios.patch(url, objet)
```

```
const dbFire = 'https://alpha-javascript-default-rtdb.europe-west1.firebaseio.com/';  
const noeud = 'films';
```

```
const id = "-OB5neA070M-9UiT_0bZ";  
const url = `${dbFire}${noeud}/${id}.json`;  
const film = { year: "1901" };  
const response = await axios.patch(url, film);  
console.log(response.data);
```

AXIOS

# Communiquer avec une API avec Axios et Fetch

## Axios PUT

- La méthode **PUT** modifier un enregistrement
  - Le PUT permet de mettre à jour tout l'objet ou le nœud

```
axios.put(url, objet)
```

```
const dbFire = 'https://alpha-javascript-default-rtdb.europe-west1.firebaseio.com/';  
const noeud = 'films';
```

```
const id = "-OB5neA070M-9UiT_0bZ";  
const url = `${dbFire}${noeud}/${id}.json`;  
const film = { name: "STAR WARS", year: "1977" };  
const response = await axios.put(url, film);  
console.log(response.data);
```

AXIOS



Communiquer avec une API avec Axios et Fetch

# Axios DELETE

- La méthode **DELETE** modifier un enregistrement
  - Le DELETE permet d'enlever un nœud ou un enregistrement

```
axios.delete(url)
```

```
const dbFire = 'https://alpha-javascript-default-rtdb.europe-west1.firebaseio.com/';  
const noeud = 'films';
```

```
const id = "-OB5neA070M-9UiT_0bZ";  
const url = `${dbFire}${noeud}/${id}.json`;  
const film = { year: "1901" };  
const response = await axios.delete(url, film);  
console.log(response.data);
```

**A X I O S**

# Communiquer avec une API avec Axios et Fetch

## **Fetch**

Ajouter avec POST

Modifier avec PUT

Modifier avec PATCH

Enlever avec DELETE

# Communiquer avec une API avec Axios et Fetch

## Fetch POST

- La méthode **POST** ajouter un enregistrement
  - Il faut sérialiser l'objet et définir un header

```
fetch(url, params)
```

```
const dbFire = 'https://alpha-javascript-default-rtdb.europe-west1.firebaseio.com/';  
const noeud = 'films';
```

```
const url = `${dbFire}${noeud}.json`;  
const film = { name: "The Matrix", year: "1999" };  
const response = await fetch(url, {  
  method: "POST",  
  headers: {  
    Accept: "application/json",  
    "Content-Type": "application/json",  
  },  
  body: JSON.stringify(film),  
});  
const info = await response.json();  
console.log(info);
```

# Communiquer avec une API avec Axios et Fetch

## Fetch PATCH

- La méthode **PATCH** modifier un enregistrement
  - Le PATCH permet de mettre à jour que certains attributs

```
fetch(url, params)
```

```
const dbFire = 'https://alpha-javascript-default-rtdb.europe-west1.firebaseio.com/';  
const noeud = 'films';
```

```
const id = "-OB61hz-EJ8b3Ig7KrdG";  
const url = `${dbFire}${noeud}/${id}.json`;  
const film = { year: "1901" };  
const response = await fetch(url, {  
  method: "PATCH",  
  headers: {  
    Accept: "application/json",  
    "Content-Type": "application/json",  
  },  
  body: JSON.stringify(film),  
});  
const info = await response.json();  
console.log(info);
```

# Communiquer avec une API avec Axios et Fetch

## Fetch PUT

- La méthode **PUT** modifier un enregistrement
  - Le PUT permet de mettre à jour tout l'objet ou le nœud

```
fetch(url, params)
```

```
const dbFire = 'https://alpha-javascript-default-rtdb.europe-west1.firebaseio.com/';  
const noeud = 'films';
```

```
const id = "-OB61hz-EJ8b3Ig7KrdG";  
const url = `${dbFire}${noeud}/${id}.json`;  
const film = { name: "STAR WARS", year: "1977" };  
const response = await fetch(url, {  
  method: "PUT",  
  headers: {  
    Accept: "application/json",  
    "Content-Type": "application/json",  
  },  
  body: JSON.stringify(film),  
});  
const info = await response.json();  
console.log(info);
```

# Communiquer avec une API avec Axios et Fetch

## Fetch DELETE

- La méthode **DELETE** modifier un enregistrement
  - Le DELETE permet d'enlever un nœud ou un enregistrement

```
fetch(url, params)
```

```
const dbFire = 'https://alpha-javascript-default-rtdb.europe-west1.firebaseio.com/';  
const noeud = 'films';
```

```
const id = "-OB61hz-EJ8b3Ig7KrdG";  
const url = `${dbFire}${noeud}/${id}.json`;  
const film = { year: "1901" };  
const response = await fetch(url, {method: "DELETE"});  
const info = await response.json();  
console.log(info);
```