

# Front fetch

Connexion du front et du back





# Préparation du backend



## Port

Si le backend démarre sur le port 3000, modifier le numéro par 4000 pour permettre au front et au back de tourner en même temps

2 applications ne peuvent pas tourner en même temps sur la même adresse IP et le même port



## Cors

Pour contacter notre backend depuis notre frontend, nous devons installer la librairie cors et l'utiliser comme middleware, comme nous l'avons fait avec express.json()

```
backend > src > TS index.ts > ...  
1  import "reflect-metadata";  
2  import express from "express";  
3  import cors from "cors";  
4  import { AppDataSource } from "./dataSource";  
5  import { Ad } from "./entity/Ad";  
6  import { Category } from "./entity/Category";  
7  import { validate } from "class-validator";  
8  
9  const app = express();  
10  
11  app.use(cors());  
12  app.use(express.json());  
13  
14  const port = 4000;
```



**axios**

Une librairie facilitant la communication  
front - back



# Installation d'axios côté frontend

**npm install axios**

```
11  "dependencies": {  
12    "@types/node": "20.6.2",  
13    "@types/react": "18.2.22",  
14    "@types/react-dom": "18.2.7",  
15    "axios": "^1.5.0",  
16    "eslint": "8.49.0",  
17    "eslint-config-next": "13.4.19",  
18    "next": "13.4.19",  
19    "react": "18.2.0",  
20    "react-dom": "18.2.0",  
21    "typescript": "5.2.2"  
22  }
```



## Affichage des annonces dans la console

Nous allons dans un premier temps afficher les annonces provenant du backend dans la console.

Modifie le fichier **RecentAds.tsx** pour afficher les annonces dans la console:



# Affichage des annonces dans la console

TS RecentAds.tsx U X

src > components > TS RecentAds.tsx > [🔍] RecentAds > [🔍] ads

```
1  import { useState, useEffect } from "react";
2  import axios from "axios";
3  import AdCard, { AdCardProps } from "../AdCard";
4
5  const RecentAds = () => {
6    const [total, setTotal] = useState(0);
7    useEffect(() => {
8      const fetchData = async () => {
9        try {
10          const result = await axios.get("http://localhost:4000/ad");
11          console.log(result);
12        } catch (err) {
13          console.log("error", err);
14        }
15      };
16      fetchData();
17    }, []);
```






# Affichage des annonces dans la console

THE GOOD CORNER

Ameublement • Électroménager • Photographie • Informatique • Téléphonie • Vélos • Véhicules • Sport • Habillement • Bébé • Outillage • S


## Annonces récentes

Prix total: 0 €




**Table** 120 €

Add price to total




**Dame-jeanne** 75 €

Add price to total




**Bougie** 4 €

Add price to total




**Porte magazine** 45 €

Add price to total



**Vaisselle** 125 €

Add price to total



**Vide-poche** 15 €

Add price to total

Elements Console Components Profiler Recorder Network

Search X Filter

Fetch/XHR JS CSS Img Media Font Doc WS Wasm Manifest Other

Console

RecentAds.tsx:11

```
{data: Array(5), status: 200, statusText: 'OK', headers: AxiosHeaders, config: {...}, ...}
  config: {transitional: {...}, adapter: 'xhr', transformRequest: Array(1), transformResponse: Array(1), timeout: ...}
  data: Array(5)
    0: {author: "car.seller@gmail.com", category: {id: 'd051d861-558d-4ebf-aaaa-8be7f72ed560', name: 'vehicles', creationDate: '2023-09-20T23:09:24.000Z', city: "Paris", creationDate: "2023-09-20T23:09:24.000Z", description: "The car is a 206", id: "046c3d94-4ac0-4da9-9cab-29e2f3e653e6", picture: "https://images.lecho.be/view?iid=dc:113129565&context=ONLINE&ratio=16/9&width=640&u=15082424!", price: 3000, title: "I'm selling a car"}, [[Prototype]]: Object
    1: {id: 'aa8a6900-5db2-4cdc-8e29-7daf717b1164', title: 'Car to sell', description: 'The car is a 207', a
    2: {id: '583a983c-7a3d-4ae2-967b-19f6905c24cf', title: '308 à vendre', description: 'Je vends ma 308, el
    3: {id: '0be7bf42-4642-436b-9bd6-e7347811c67d', title: '308 à vendre', description: 'Je vends ma 308, el
    4: {id: 'ae12de0d-3f3b-43c3-92fe-flad07f9a6eb', title: '308 à vendre', description: 'Je vends ma 308, el
      length: 5
    [[Prototype]]: Array(0)
  headers: AxiosHeaders {content-length: '2188', content-type: 'application/json; charset=utf-8'}
  request: XMLHttpRequest {onreadystatechange: null, readyState: 4, timeout: 0, withCredentials: false, uplo
    status: 200
    statusText: "OK"
  [[Prototype]]: Object
```



## Typage de result

On type manuellement les données reçues par l'api. Quelques ajustement à prévoir si les props ne sont pas nommées de la même manière que les données reçues.

On utilise ici le type générique de axios, une quête est disponible à ce sujet

```
useEffect(() => {  
  const fetchData = async () => {  
    try {  
      const result = await axios.get<AdCardProps[]>(  
        "http://localhost:4000/ad"  
      );  
    } catch (err) {  
      console.log("error", err);  
    }  
  };  
  fetchData();  
}, []);
```



## Création d'un state ads

Afin que notre page **réagisse** et que nos annonces s'affichent dès que les données du backend ont été reçues, on va supprimer le tableau de données ads et le remplacer par un **state**

On type ce **state** avec le type générique de **useState**

```
const RecentAds = () => {  
  const [total, setTotal] = useState(0);  
  const [ads, setAds] = useState<AdCardProps[]>([]);
```



## Enregistrement dans le state

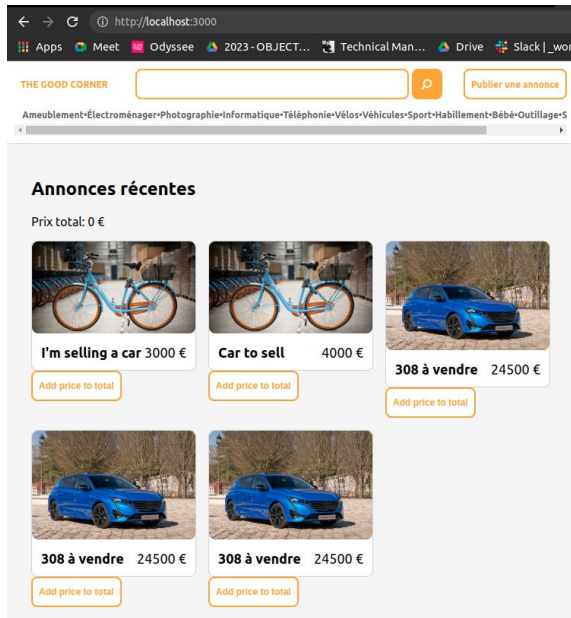
Une fois les données reçues de la part du backend, on les enregistre dans le state:

```
useEffect(() => {  
  const fetchData = async () => {  
    try {  
      const result = await axios.get<AdCardProps[]>(  
        "http://localhost:4000/ad"  
      );  
      setAds(result.data);  
    }  
  }  
});
```



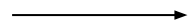
# Enregistrement dans le state

On constate que notre page affiche maintenant les annonces provenant du backend:





## A toi de jouer !



- Réalise toutes ces étapes et affiche les **annonces** provenant du backend dans ton projet frontend ainsi que les **catégories**