TP : Mini-App E-Commerce avec Fetch API

API utilisée : Fake Store API

https://fakestoreapi.com/

Partie 1 – Afficher la liste des produits

👉 Crée une page `index.html` avec un bouton **"Charger les produits"** et une `<div id="produits"></div>`.

- Au clic, fais un `fetch` sur `https://fakestoreapi.com/products`.
- Affiche pour chaque produit :
  - image
  - titre
  - prix

Partie 2 – Détails d'un produit

👉 Ajoute un champ **"ID produit"** + un bouton **"Rechercher"**.

- Au clic, fais un `fetch` sur `https://fakestoreapi.com/products/{id}`.
- Affiche le titre, la description, le prix et l'image du produit.
- Si l'ID n'existe pas → affiche un message d'erreur.

Partie 3 – Ajouter un produit (POST)

👉 Crée un petit formulaire :

- titre, prix, description, catégorie.
- Au clic sur "Ajouter", envoie la requête `POST`.
- Affiche la réponse (produit créé).

Partie 4 – Modifier un produit (PUT)

👉 Ajoute un formulaire de modification :

- ID + nouveau titre + nouveau prix.
- Envoie la requête `PUT` sur `https://fakestoreapi.com/products/{id}`.
- Affiche la réponse.

Partie 5 – Supprimer un produit (DELETE)

👉 Ajoute un champ **ID** + bouton **"Supprimer"**.

- Envoie une requête `DELETE`.
- Affiche un message de confirmation.

Partie 6 – Bonus : Version `async/await`

Réécris la partie 1 (liste des produits) avec `async/await` pour montrer une syntaxe plus moderne :

```
async function chargerProduits() {
  try {
    let res = await fetch("https://fakestoreapi.com/products");
    if (!res.ok) throw new Error("Erreur HTTP : " + res.status);
    let produits = await res.json();
    console.log(produits);
  } catch (err) {
    console.error("Erreur :", err.message);
  }
}
```

🎓 Projet final

Crée une mini-application de gestion de produits avec :

1. Liste des produits.
2. Recherche par ID.
3. Ajout d'un produit.

4. Modification d’un produit.
5. Suppression d’un produit.