📌 Détail du montage de l'application et interaction avec `index.html`

Ton application React fonctionne grâce à une **intégration entre `index.html` et `main.jsx`**, où `index.html` sert de point d'ancrage pour React. Voyons cela étape par étape !

🚀 1. Structure du `index.html`

Ton `index.html` est un fichier statique qui contient :

✅ Un conteneur principal :

```
<div id="root"></div>
```

👉 C'est dans ce `<div>` que React injecte l'ensemble de l'application.

✅ Un script qui lance l'application :

```
<script type="module" src="/src/main.jsx"></script>
```

👉 Ce script charge `main.jsx`, qui va monter l'application React dans `#root`.

🛠️ 2. Exécution de `main.jsx`

Quand `index.html` est chargé par le navigateur, il exécute le script `/src/main.jsx`. Ce fichier :

1️⃣ Importe React et ReactDOM :

```
import React, { StrictMode } from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'
```

👉 React est nécessaire pour créer des composants, et ReactDOM permet de manipuler le DOM virtuel.

2️⃣ Récupère l'élément `#root` du `index.html`

```
ReactDOM.createRoot(document.getElementById('root'))
```

👉 `document.getElementById('root')` sélectionne le `<div id="root"></div>` présent dans `index.html`.

3️⃣ Monte l'application React dans `#root`

```
.render(
  <StrictMode>
    <App />
  </StrictMode>
)
```

👉 Ici, on affiche le composant `<App />` dans `#root`.
👉 **React remplace le contenu de `#root`** par ce qu'affiche `App.jsx` (l'application React).

🔄 3. Comment l'application React interagit avec `index.html` ?

✅ **Avant le montage React**
Quand le navigateur charge `index.html`, il voit :

```
<div id="root"></div>
```

👉 Ce `div` est **vide** tant que React ne l'a pas rempli.

✅ **Après le montage React**
React injecte le contenu rendu par `<App />` à l'intérieur de `#root`.
Si `App.jsx` contient :

```
export default function App() {
  return <h1>Hello, React!</h1>;
}
```

Alors `index.html` devient dynamiquement :

```
<div id="root">
  <h1>Hello, React!</h1>
</div>
```

👉 **React contrôle maintenant entièrement le contenu de `#root`**.

🔥 4. Pourquoi React fonctionne ainsi ?

💡 **Séparation des responsabilités**

- `index.html` reste **minimal** : c'est juste un point d'entrée.
- `main.jsx` et `App.jsx` prennent en charge **l'affichage et la logique**.

💡 **React manipule un DOM virtuel**

- React utilise un **Virtual DOM** pour optimiser les mises à jour.
- Seules les parties du DOM qui changent sont mises à jour.

💡 **Facilité de développement**

- Pas besoin de manipuler directement le `DOM` avec `document.createElement()` ou `innerHTML`.
- L'application est **composée de composants réutilisables** (`App.jsx`, `components/`, etc.).

✅ Résumé

Étape	Ce qui se passe
1	Le navigateur charge `index.html` et trouve `<div id="root"></div>`.
2	Il exécute le script `main.jsx`, qui initialise React.
3	`ReactDOM.createRoot().render(<App />)` remplit `#root` avec l'application React.
4	React prend le contrôle de l'affichage, et met à jour dynamiquement `#root`.

👉 **Au final, `index.html` ne sert que de squelette initial. Tout le rendu est géré par React ! 🎯**