Chapter 11: Handling API Calls

1 Introduction

Is chapter mein humne seekha kaise React mein **API calls** handle karte hain. API se data fetch karke state mein store karte hain aur UI mein render karte hain. Humne `fetch` API aur `axios` library ka use dekha, with `useEffect`, loading states, aur error handling.

2 JavaScript Concepts for API Calls

API calls ke liye JavaScript ke yeh concepts zaroori hain:

2.1 What is an API?

API ek bridge hai jo client aur server ke beech data exchange karta hai, usually JSON format mein.

2.2 Asynchronous JavaScript

API calls async hote hain, toh hum **Promises** aur **async/await** use karte hain.

- **Promise**: Future value, states: Pending, Fulfilled, Rejected.

```
const promise = new Promise((resolve, reject) => {
  setTimeout(() => resolve("Data mil gaya!"), 1000);
});
promise.then((data) => console.log(data));
```

- **Async/Await**: Cleaner Promise syntax.

```
async function getData() {
  const result = await Promise.resolve("Data mil gaya!");
  console.log(result);
}
getData();
```

2.3 Fetch API

JavaScript ka built-in `fetch` HTTP requests ke liye hai, Promise return karta hai.

```
fetch('https://api.example.com/data')
  .then((response) => response.json())
  .then((data) => console.log(data))
  .catch((error) => console.log("Error:", error));
```

Async/Await:

```
async function fetchData() {
  try {
    const response = await fetch('https://api.example.com/data');
    const data = await response.json();
```

```
    console.log(data);
  } catch (error) {
    console.log("Error:", error);
  }
}
fetchData();
```

2.4 Axios

`axios` ek library hai jo `fetch` se zyada features deti hai: automatic JSON parsing, better error handling.

```
<script
   src="https://cdn.jsdelivr.net/npm/axios@1.4.0/dist/axios.min.js"></script
```

```
async function fetchData() {
  try {
    const response = await
       axios.get('https://api.example.com/data');
    console.log(response.data);
  } catch (error) {
    console.log("Error:", error);
  }
}
fetchData();
```

3 11.1 Why API Calls in React?

APIs se dynamic data fetch hota hai, jaise menu items ya user details. `useEffect` side effects ke liye use hota hai.

3.1 Real-World Example

Food delivery app mein, menu items server se API ke through fetch hote hain aur UI mein render hote hain.

4 11.2 Using Fetch API

`fetch` ke saath `useEffect` mein API call karte hain.

4.1 Code Example: Fetching Menu Items

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Fetch API</title>
  <script
     src="https://cdn.jsdelivr.net/npm/react@18.2.0/umd/react.development.js
```

```
  <script
     src="https://cdn.jsdelivr.net/npm/react-dom@18.2.0/umd/react-dom.develo
  <script
     src="https://cdn.jsdelivr.net/npm/@babel/standalone@7.20.6/babel.min.js
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body>
  <div id="root"></div>
  <script type="text/babel">
    function Menu() {
      const [items, setItems] = React.useState([]);
      const [loading, setLoading] = React.useState(true);
      const [error, setError] = React.useState(null);

      React.useEffect(() => {
        fetch('https://jsonplaceholder.typicode.com/posts?_limit=3')
          .then((response) => response.json())
          .then((data) => {
            setItems(data);
            setLoading(false);
          })
          .catch((err) => {
            setError('Failed to fetch data');
            setLoading(false);
          });
      }, []);

      if (loading) return <div className="text-center
         p-4">Loading...</div>;
      if (error) return <div className="text-center p-4
         text-red-500">{error}</div>;

      return (
        <div className="text-center p-4">
          <h1 className="text-3xl font-bold
             text-blue-600">Menu</h1>
          <ul className="list-disc list-inside">
            {items.map((item) => (
              <li key={item.id} className="text-lg">
                {item.title}
              </li>
            ))}
          </ul>
        </div>
      );
    }

    const root =
       ReactDOM.createRoot(document.getElementById('root'));
    root.render(<Menu />);
  </script>
```

```
</body>
</html>
```

4.2 Output

Browser mein yeh dikhega:

- **Initially**: "Loading..." (center-aligned).
- **After Fetch**:
  - Heading: "Menu" (large, bold, blue).
  - Bulleted list: 3 dummy post titles (e.g., "sunt aut facere repellat...", etc.).
- **Error Case**: Red text "Failed to fetch data".

4.3 Explanation

- `useState`: `items`, `loading`, `error` store karta hai.
- `useEffect`: Component mount pe API call (`[]`).
- `fetch`: JSONPlaceholder se 3 posts.
- Conditional rendering for loading/error.

5 11.3 Using Axios

`axios` simpler syntax aur better error handling deta hai.

5.1 Code Example: Fetching with Axios

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Axios API</title>
  <script
     src="https://cdn.jsdelivr.net/npm/react@18.2.0/umd/react.development.js
  <script
     src="https://cdn.jsdelivr.net/npm/react-dom@18.2.0/umd/react-dom.develo
  <script
     src="https://cdn.jsdelivr.net/npm/@babel/standalone@7.20.6/babel.min.js
  <script
     src="https://cdn.jsdelivr.net/npm/axios@1.4.0/dist/axios.min.js"></scri
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body>
  <div id="root"></div>
  <script type="text/babel">
    function Menu() {
      const [items, setItems] = React.useState([]);
```

```
      const [loading, setLoading] = React.useState(true);
      const [error, setError] = React.useState(null);

      React.useEffect(() => {
        axios.get('https://jsonplaceholder.typicode.com/posts?_limit=3')
          .then((response) => {
            setItems(response.data);
            setLoading(false);
          })
          .catch((err) => {
            setError('Failed to fetch data');
            setLoading(false);
          });
      }, []);

      if (loading) return <div className="text-center
         p-4">Loading...</div>;
      if (error) return <div className="text-center p-4
         text-red-500">{error}</div>;

      return (
        <div className="text-center p-4">
          <h1 className="text-3xl font-bold text-blue-600">Menu
             with Axios</h1>
          <ul className="list-disc list-inside">
            {items.map((item) => (
              <li key={item.id} className="text-lg">
                {item.title}
              </li>
            ))}
          </ul>
        </div>
      );
    }

    const root =
       ReactDOM.createRoot(document.getElementById('root'));
    root.render(<Menu />);
  </script>
</body>
</html>
```

5.2 Output

Browser mein yeh dikhega:

- **Initially**: "Loading..." (center-aligned).
- **After Fetch**:
  - Heading: "Menu with Axios" (large, bold, blue).

  - Bulleted list: 3 dummy post titles.
- **Error Case**: Red text "Failed to fetch data".

5.3 Explanation

- `axios.get`: Simple syntax, direct JSON data.
- `response.data`: JSON automatically parsed.
- Rest: Same as `fetch` example.

6 11.4 Combining with Routing and State

API calls ko React Router aur state ke saath combine karte hain.

6.1 Code Example: API with Routing

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>API with Routing</title>
  <script
     src="https://cdn.jsdelivr.net/npm/react@18.2.0/umd/react.development.js
  <script
     src="https://cdn.jsdelivr.net/npm/react-dom@18.2.0/umd/react-dom.develo
  <script
     src="https://cdn.jsdelivr.net/npm/@babel/standalone@7.20.6/babel.min.js
  <script
     src="https://cdn.jsdelivr.net/npm/react-router-dom@6.3.0/dist/umd/react
  <script
     src="https://cdn.jsdelivr.net/npm/axios@1.4.0/dist/axios.min.js"></scri
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body>
  <div id="root"></div>
  <script type="text/babel">
    const { BrowserRouter, Routes, Route, Link, useParams } =
       ReactRouterDOM;

    // Menu Component
    function Menu() {
      const [items, setItems] = React.useState([]);
      const [loading, setLoading] = React.useState(true);
      const [error, setError] = React.useState(null);

      React.useEffect(() => {
        axios.get('https://jsonplaceholder.typicode.com/posts?_limit=3')
          .then((response) => {
            setItems(response.data);
            setLoading(false);
```

```
29            })
30            .catch((err) => {
31              setError('Failed to fetch data');
32              setLoading(false);
33            });
34        }, []);
35
36        if (loading) return <div className="text-center
              p-4">Loading...</div>;
37        if (error) return <div className="text-center p-4
              text-red-500">{error}</div>;
38
39        return (
40          <div className="text-center p-4">
41            <h1 className="text-3xl font-bold
                  text-blue-600">Menu</h1>
42            <ul className="list-disc list-inside">
43              {items.map((item) => (
44                <li key={item.id}>
45                  <Link to={`/item/${item.id}`}
                        className="text-blue-500 hover:underline">
46                    {item.title}
47                  </Link>
48                </li>
49              ))}
50            </ul>
51          </div>
52        );
53      }
54
55      // Item Detail Component
56      function ItemDetail() {
57        const { id } = useParams();
58        const [item, setItem] = React.useState(null);
59        const [loading, setLoading] = React.useState(true);
60        const [error, setError] = React.useState(null);
61
62        React.useEffect(() => {
63          axios.get(`https://jsonplaceholder.typicode.com/posts/${id}`)
64            .then((response) => {
65              setItem(response.data);
66              setLoading(false);
67            })
68            .catch((err) => {
69              setError('Failed to fetch item');
70              setLoading(false);
71            });
72        }, [id]);
73
74        if (loading) return <div className="text-center
              p-4">Loading...</div>;
```

```
75        if (error) return <div className="text-center p-4
              text-red-500">{error}</div>;
76
77        return (
78          <div className="text-center p-4">
79            <h1 className="text-3xl font-bold
                  text-blue-600">{item.title}</h1>
80            <p className="text-lg">Details: {item.body}</p>
81            <Link to="/menu" className="text-blue-500
                  hover:underline">Back to Menu</Link>
82          </div>
83        );
84      }
85
86      // Main App Component
87      function App() {
88        return (
89          <BrowserRouter>
90            <div className="p-4">
91              <nav className="flex justify-center space-x-4">
92                <Link to="/" className="text-blue-500
                        hover:underline">Home</Link>
93                <Link to="/menu" className="text-blue-500
                        hover:underline">Menu</Link>
94              </nav>
95              <Routes>
96                <Route path="/" element={<h1
                        className="text-center text-3xl font-bold
                        text-blue-600 p-4">Home</h1>} />
97                <Route path="/menu" element={<Menu />} />
98                <Route path="/item/:id" element={<ItemDetail />} />
99              </Routes>
100           </div>
101         </BrowserRouter>
102       );
103     }
104
105     const root =
            ReactDOM.createRoot(document.getElementById('root'));
106     root.render(<App />);
107   </script>
108 </body>
109 </html>
```

6.2 Output

Browser mein yeh dikhega:

- **Top Navigation Bar**: "Home", "Menu" links (blue, hover pe underline).
- **URL:** /: Heading "Home" (large, bold, blue).

- **URL: /menu**:
  - Initially: "Loading...".
  - After Fetch: Heading "Menu", bulleted list of 3 post titles (clickable).
  - Error: Red text "Failed to fetch data".
- **URL: /item/1**:
  - Initially: "Loading...".
  - After Fetch: Heading (post title), Details (post body), "Back to Menu" link.
  - Error: Red text "Failed to fetch item".

6.3 Explanation

- `Menu`: API se items fetch aur list mein `Link`.
- `ItemDetail`: Specific item ka data fetch using `useParams`.
- `Routes`: Navigation aur dynamic routing.

7 Common Mistakes

- **Missing `useEffect` Dependency Array**: Infinite API calls.
- **No Loading/Error States**: User feedback zaroori hai.
- **Missing Error Handling**: `try/catch` ya `.catch` use karo.

8 Interview Tips

- **React mein API calls kaise handle karte hain?**
  - `useEffect` ke saath `fetch` ya `axios`, state mein data store.
- **Loading aur error states kaise manage karte hain?**
  - `loading`, `error` states aur conditional rendering.
- **`fetch` vs `axios`?**
  - `fetch`: Built-in, manual JSON parsing.
  - `axios`: Better syntax, auto JSON, error handling.

9 Assignment: Practice Time

9.1 Task 1: Simple API Fetch

- `MenuApp` banao jo JSONPlaceholder se posts fetch kare ($?_limit = 5$).$Bulletedlistmeinrender$.
- Loading aur error states handle karo.
- Tailwind se style karo.

9.2 Task 2: API with Dynamic Details

- `FoodApp` banao jisme `/menu` pe API se items.
- `Link` se `/item/:id` pe jaye.
- `ItemDetail` mein item ka data fetch aur render.
- Tailwind se style karo.

9.3 Task 3: Searchable API Data

- `SearchApp` banao jo API se posts fetch kare aur search input ho.
- Search input se client-side filtering.
- Loading aur error states handle karo.
- Tailwind se style karo.