Adapt the quiz app to use Axios instead of `fetch` to get the quiz data from your JSON Server. Stater files are `fetchquiz` and solutions are `axiosquiz`.

1. Install Axios

If you haven't installed Axios yet, you can do so with:

```
npm install axios
```

2. Modify `useFetch` Hook

```
import axios from 'axios';

function useFetch(url) {
  const [data, setData] = useState(null);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState(null);

  useEffect(() => {
    axios.get(url)
      .then(response => {
        setData(response.data);
      })
      .catch(err => {
        setError(err);
      })
      .finally(() => {
        setLoading(false);
      });
  }, [url]);

  return { data, loading, error };
}
```

- We import `axios` at the top.
- Inside `useEffect`, we replace the `fetch` call with `axios.get(url)`.
- Axios handles parsing the JSON response, so we can directly access the data using `response.data`.

3. Update `Quiz` Component

Ensure you are still using your JSON Server endpoint:

```
const { data: questions, loading, error } =
useFetch('http://localhost:3000/questions');
```

Complete `Quiz` Component Differences

```
import React, { useState, useEffect } from 'react';
import './index.css';
import axios from 'axios';

// ... (Question component remains the same)

function useFetch(url) { /* ... (as shown above) ... */ }

function Quiz() {
  // ... (rest of the Quiz component remains the same)
}

export default Quiz;
```

Key Changes

- **Axios:** We use Axios to make the HTTP request to your JSON Server. Axios simplifies error handling and directly parses JSON data.
- **Error Handling:** Axios provides a concise way to handle errors with the `.catch()` block.

Why use Axios?

While both `fetch` and Axios can fetch data from APIs, Axios offers some advantages:

- **Automatic JSON Parsing:** Axios automatically parses JSON responses, whereas you need to manually parse them with `fetch`.
- **Simpler Error Handling:** Axios error handling is more intuitive than `fetch`, which throws errors only for network issues, not HTTP errors (like 404).
- **Interceptors:** Axios provides interceptors, which allow you to modify requests and responses globally.
- **Widely Used:** Axios is a popular library in the React ecosystem.

- **All code together:**

```
import React, { useState, useEffect } from 'react';
import './index.css';
```

```
import axios from 'axios';

// Question Component (unchanged)
function Question({ questionText, options, correctAnswer,
selectedOption, onSelectOption }) {
  return (
    <div>
      <p>{questionText}</p>
      {options.map((option) => (
        <label key={option}>
          <input
            type="radio"
            name={questionText}
            value={option}
            checked={selectedOption === option}
            onChange={() => onSelectOption(option)}
          />
          {option}
        </label>
      ))}
    </div>
  );
}

// Custom useFetch hook with Axios
function useFetch(url) {
  const [data, setData] = useState(null);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState(null);

  useEffect(() => {
    axios.get(url)
      .then(response => {
        setData(response.data); // Assuming your API
returns data directly in the response
      })
      .catch(err => {
        setError(err);
      })
      .finally(() => {
```

```
        setLoading(false);
      });
  }, [url]);

  return { data, loading, error };
}

// Quiz Component
function Quiz() {
  const [score, setScore] = useState(0);
  const [currentQuestionIndex, setCurrentQuestionIndex] =
useState(0);
  const [selectedOption, setSelectedOption] =
useState(null);

  // Fetch questions from JSON Server
  const { data: questions, loading, error } =
useFetch('http://localhost:3000/questions');

  const handleNextQuestion = () => {
    if (questions[currentQuestionIndex].correctAnswer ===
selectedOption) {
      setScore(score + 1);
    }
    setCurrentQuestionIndex(currentQuestionIndex + 1);
    setSelectedOption(null);
  };

  if (loading) return <p>Loading questions...</p>;
  if (error) return <p>Error: {error.message}</p>;

  return (
    <div>
      {currentQuestionIndex < questions.length ? (
        <>
          <Question

questionText={questions[currentQuestionIndex].questionText}

options={questions[currentQuestionIndex].options}
```

```
correctAnswer={questions[currentQuestionIndex].correctAnswe
r}
            selectedOption={selectedOption}
            onSelectOption={setSelectedOption}
          />
          <button onClick={handleNextQuestion}>Next</
button>
        </>
      ) : (
        <p>You scored {score} out of {questions.length}</p>
      )}
    </div>
  );
}

export default Quiz;
```

Tips:

- **Axios:** Use `axios.get()` to fetch the questions from the JSON Server.
- **Error Handling:** The `useFetch` hook handles potential errors during the fetch process.
- **Loading State:** The `Quiz` component displays a "Loading questions..." message while the data is being fetched.
- **JSON Server:** Make sure your JSON Server is running (with `json-server --watch db.json`) and that the endpoint (`http://localhost:3000/questions`) matches your server setup.