**Assignment 4: Lifecycle Methods**

**Explore component lifecycle methods.**

**Use componentDidMount to fetch data from a fake API (e.g., JSONPlaceholder).**

In React, lifecycle methods are special methods that are called at different stages of a component's life. The componentDidMount lifecycle method is particularly useful for fetching data from an external source, such as a fake API like JSONPlaceholder. Here's a simple example:

| import React, { Component } from 'react';  class DataFetchingComponent extends Component {  constructor(props) {  super(props);  this.state = {  data: null,  loading: true,  error: null,  };  }  componentDidMount() {  // Use a fake API like JSONPlaceholder  const apiUrl = 'https://jsonplaceholder.typicode.com/todos/1';  fetch(apiUrl)  .then((response) => {  if (!response.ok) {  throw new Error(`HTTP error! Status: ${response.status}`);  }  return response.json();  })  .then((data) => {  this.setState({  data: data,  loading: false,  });  })  .catch((error) => {  this.setState({  loading: false,  error: error.message,  });  });  }  render() {  const { data, loading, error } = this.state;  if (loading) {  return <div>Loading...</div>;  }  if (error) {  return <div>Error: {error}</div>;  }  return (  <div>  <h1>Data from Fake API:</h1>  <pre>{JSON.stringify(data, null, 2)}</pre>  </div>  );  }  }  export default DataFetchingComponent; |
| --- |

In this example, the componentDidMount method is used to initiate a fetch request to the fake API (JSONPlaceholder). The data is then stored in the component's state when the request is successful, and the component is re-rendered.

Remember to replace the API URL with the appropriate endpoint from JSONPlaceholder or any other fake API you want to use.

Also, ensure that you handle loading and error states appropriately, as shown in the example, to provide a better user experience.