**Accordion Project Documentation**

**Introduction:**

The React FAQ Accordion project is an interactive web application that dynamically displays frequently asked questions (FAQs). Users can click on each question to expand and view the answer. The accordion allows only one question to be open at a time, ensuring a clean and structured UI.

**Tools & Frameworks Used:**

React JS - A JavaScript library for building user interfaces- React Hooks (useState, useEffect) - For managing state and handling API requests- CSS - For styling the FAQ accordion component- JSON Server - A simple mock API to provide FAQ data- VS Code - Code Editor- Node.js & npm - JavaScript runtime and package manager

**Project Structure:**

A screenshot of a computer

AI-generated content may be incorrect.

**Project Implementation Process:**

**1. Initialize a React project**A screenshot of a computer

AI-generated content may be incorrect.

**2. Install JSON Server:**

-Installing the json server Globally in the localsystem to access it where we want.

**Commad to install json-server:**

A screen shot of a computer

AI-generated content may be incorrect.

**-import the useState and useEffect from react by using import**

**Statement**

A black rectangular object with white text

AI-generated content may be incorrect.

**3.Defining useState:**

The useState hook is a built-in React hook that allows functional components to have state.In class components, we usedthis.state, but in functional components, we use useState.



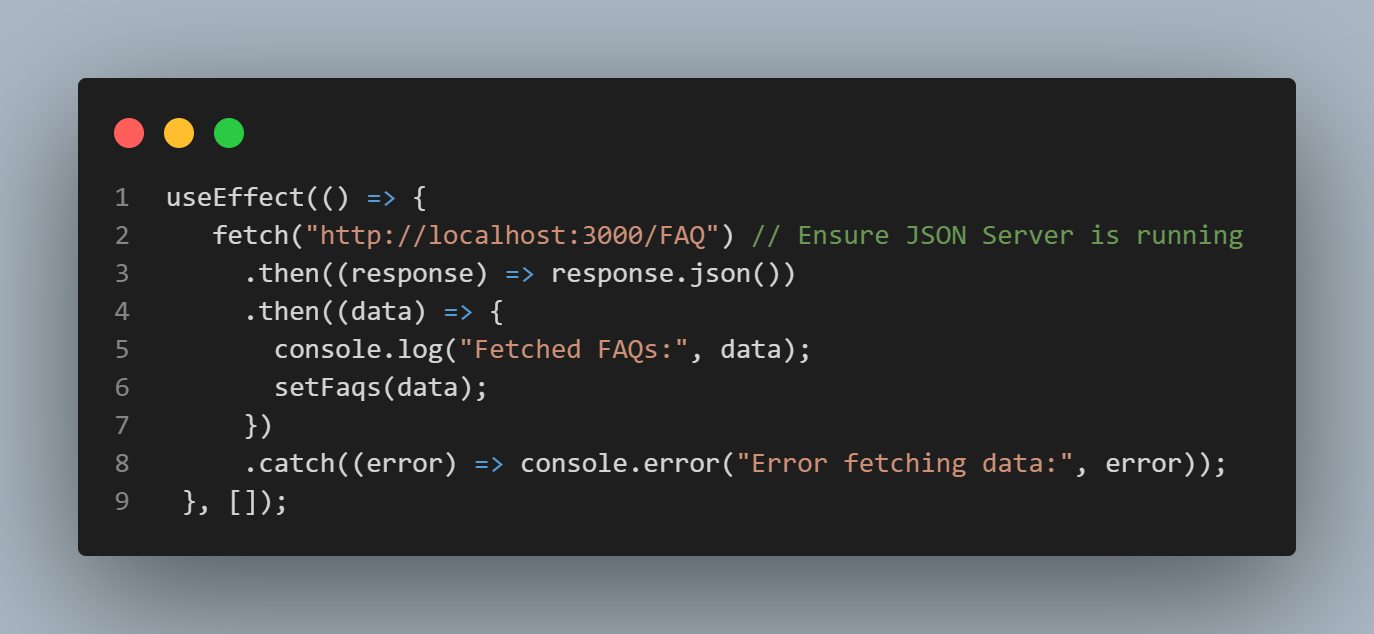
faqs (FAQ List State):

* Holds an array of FAQ items (initially empty []).
* setFaqs(data) updates this list, typically after fetching data.

openId (Currently Open FAQ ID):

* Tracks which FAQ is currently expanded (null means none is open).
* setOpenId(id) sets the open FAQ when clicked.
* Helps in toggling FAQ visibility in an accordion-style UI

**4.** **Fetch the data From Mock Api:**



This useEffect hook is responsible for fetching FAQ data from a local JSON server when the component mounts

**Code Flow Explanation:**

**useEffect(() => { ... }, [])**

The useEffect hook runs the provided function when the component is first rendered.

The empty dependency array ([]) ensures that this effect runs only once, similar to componentDidMount in class components.

**Fetch Request:**

fetch("http://localhost:3000/FAQ") makes an HTTP GET request to http://localhost:3000/FAQ.

The assumption here is that a JSON server is running locally on port 3000 and has an endpoint /FAQ that returns FAQ data in JSON format.

**Handling the Response:**

.then((response) => response.json()): Once the request is successful, the response is converted to JSON format.

.then((data) => { console.log("Fetched FAQs:", data); setFaqs(data); })

Logs the fetched FAQ data to the console for debugging.

Calls setFaqs(data), which updates the component state with the fetched FAQs.

**Error Handling:**

.catch((error) => console.error("Error fetching data:", error));

If any error occurs during fetching (e.g., the server is down or there's a network issue), it is logged in the console.

**Execution Flow:**

When the component mounts:

useEffect runs.

A fetch request is sent to http://localhost:3000/FAQ.

The response is converted into JSON.

The data is logged in the console and stored in the state via setFaqs.

If an error occurs, it is logged.

**5.Implementing Tooglelogic:**

This function is used to toggle the visibility of an FAQ answer when the user clicks on a question. It ensures that only one FAQ is expanded at a time.

A computer screen shot of a black rectangle with white text

AI-generated content may be incorrect.

**Function is Called with an id**

When a user clicks an FAQ question, toggleFAQ(id) is executed with that FAQ's id.

**Using setOpenId to Update State**

setOpenId updates the openId state, which tracks which FAQ is currently open.

**Arrow Function Inside setOpenId (Functional Update)**

setOpenId((prevId) => {...})

The function receives prevId, which holds the current value of openId.

**Conditional Toggle Logic**

A black rectangular with yellow green and blue text

AI-generated content may be incorrect.

If the clicked FAQ is already open (prevId === id), set openId to null (close it).

If a different FAQ is clicked (prevId !== id), update openId to id (open it).

**6.Rendaring the data:**

A computer screen shot of a computer code

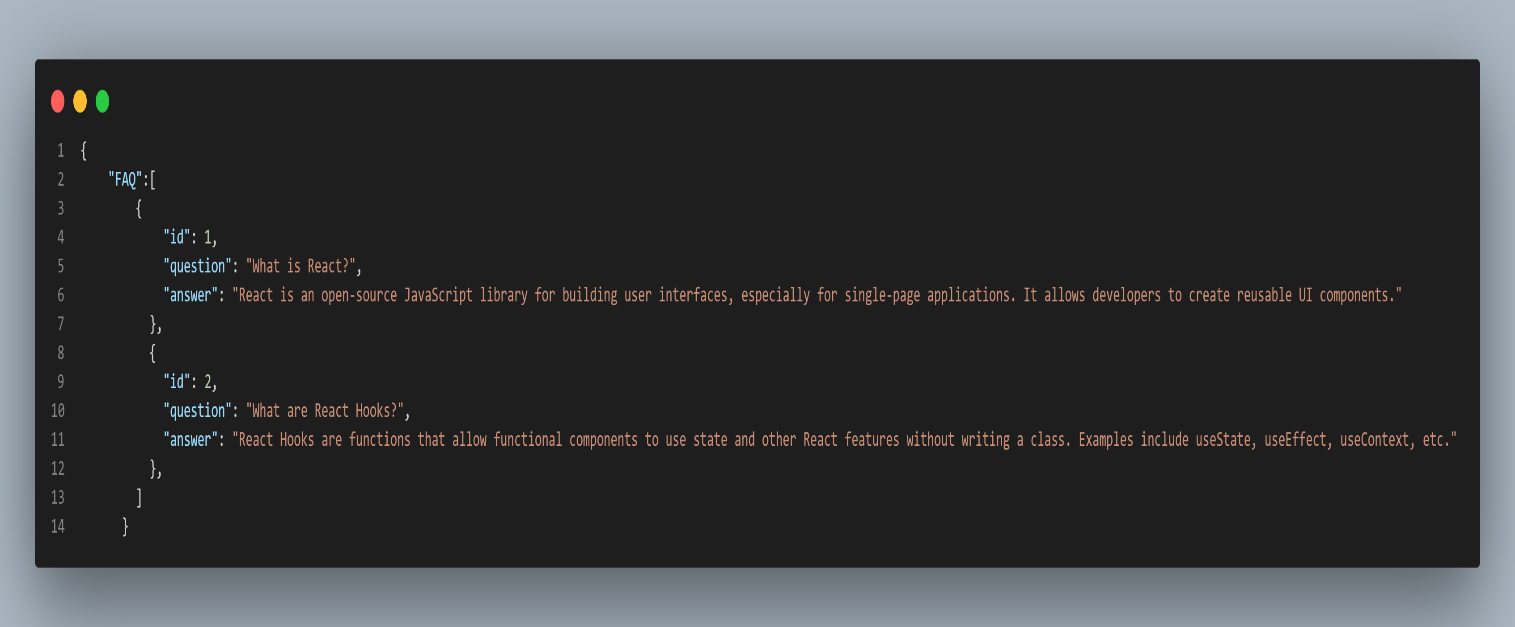
AI-generated content may be incorrect.

**Mapping through FAQs:**

- The faqs array (which is likely stored in a JSON file like Faq.json) contains objects with id, question, and answer.

-The .map() method loops through the faqs array and creates a new <div> for each FAQ item.

**- Structure of faqs:**



**FAQ Item Wrapper:**

A black rectangular object with text

AI-generated content may be incorrect.

The **key={id}** ensures React efficiently updates the list when it changes.

The **className={faq-item ${openId === id ? "open" : ""}}:**

* Adds "open" to the class when the FAQ is expanded (helps in styling animations).

**FAQ Question Section:**

**<div className="faq-question" onClick={() => toggleFAQ(id)}>**

* Displays the question.
* When clicked, **toggleFAQ(id)** is called (likely a function that sets **openId** to the clicked question’s id to toggle visibility).

**Expand/Collapse Button:**

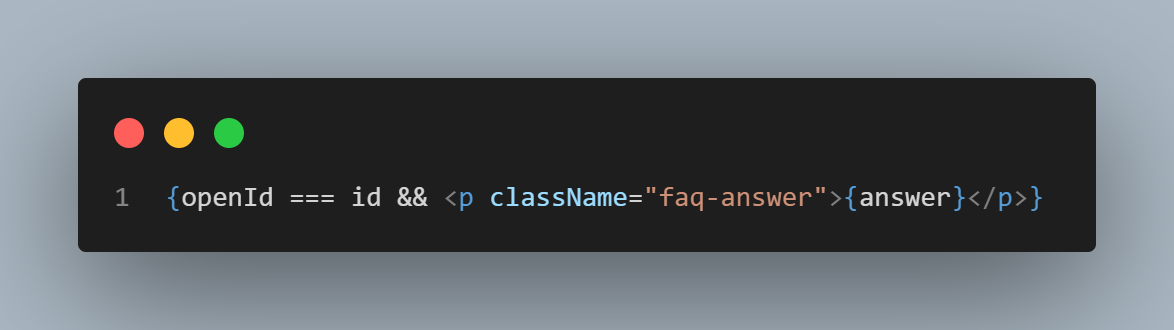
A black rectangular object with white text

AI-generated content may be incorrect.

- Shows **"+"** when the FAQ is **collapsed**.

- Shows **"-"** when the FAQ is **expanded**.

**FAQ Answer Section:**



- If **openId === id**, meaning the FAQ is open, the answer is displayed.

- Otherwise, the answer remains hidden

**Expedted Result:**

**A screenshot of a computer

AI-generated content may be incorrect.**

-when we clicked on the Question it checks the if any Question is open or not when Another Question is in open it gets collopesed and it checks the clicked question and {id} and displayes code ..

**Conclusion:**

-implement the Accordion with react.js and json-server that can fetch the data for localsystem api and render the Question’s in browser and it display the answer while on the Question according to id.