PRACTICAL 10

Designing RESTful API and building front-end application that consumes it

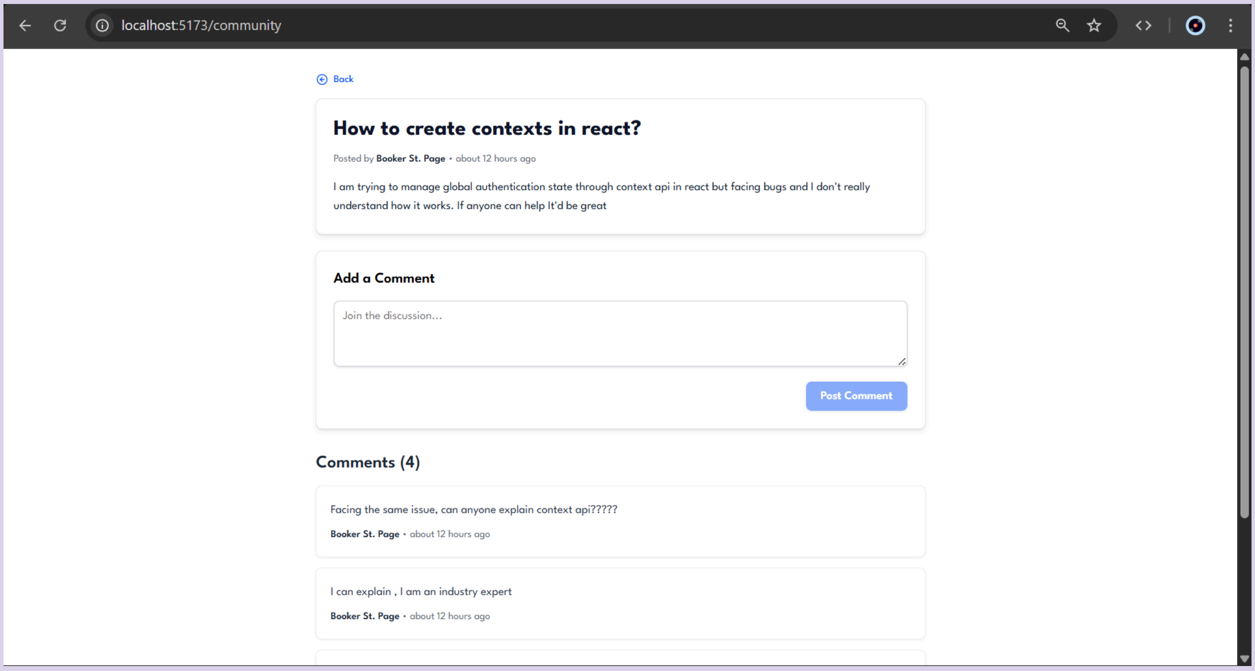
RESTful API for thread-like service

```
const express = require('express');
const cors = require('cors');
const app = express();
const PORT = 5001;

app.use(cors());
app.use(express.json());

let db = {
  communities: [
    { id: 1, name: 'General', description: 'A place for general discussion.'
},
    { id: 2, name: 'React Developers', description: 'All about React.js.' },
  ],
  posts: [
    { id: 101, communityId: 1, title: 'Welcome!', content: 'Welcome to the
general forum.' },
    { id: 102, communityId: 2, title: 'What are React Hooks?', content: 'Can
someone explain useEffect?' },
  ],
};

let nextCommunityId = 3;
let nextPostId = 103;

// Route: GET /api/communities
app.get('/api/communities', (req, res) => {
  console.log('GET /api/communities');
  res.json(db.communities);
});

// Route: POST /api/communities
app.post('/api/communities', (req, res) => {
  const { name, description } = req.body;
  if (!name || !description) {
    return res.status(400).json({ message: 'Name and description are
required.' });
  }
  const newCommunity = {
    id: nextCommunityId++,
    name,
    description,
  };
  db.communities.push(newCommunity);
  console.log('POST /api/communities -> Created:', newCommunity);
  res.status(201).json(newCommunity);
});

// Route: GET /api/communities/:id/posts
app.get('/api/communities/:id/posts', (req, res) => {
  const communityId = parseInt(req.params.id, 10);
  const communityPosts = db.posts.filter(post => post.communityId ===
communityId);
  console.log(`GET /api/communities/${communityId}/posts -> Found
${communityPosts.length} posts`);
  res.json(communityPosts);
});

// Route: POST /api/communities/:id/posts
app.post('/api/communities/:id/posts', (req, res) => {
  const communityId = parseInt(req.params.id, 10);
  const { title, content } = req.body;
  if (!title || !content) {
    return res.status(400).json({ message: 'Title and content are required.'
});
  }
  const newPost = {
    id: nextPostId++,
    communityId,
    title,
    content,
  };
  db.posts.push(newPost);
  console.log(`POST /api/communities/${communityId}/posts -> Created:`,
newPost);
  res.status(201).json(newPost);
});

app.listen(PORT, () => {
  console.log(`Simple API server running on http://localhost:${PORT}`);
  console.log('--- Mock Database Initial State ---');
  console.log(db);
  console.log('-----------------------------------');
});
```

Frontend.jsx → React Component

```
import React, { useState, useEffect } from 'react';

const API_URL = 'http://localhost:5001/api';

export default function CommunityApp() {
  const [communities, setCommunities] = useState([]);
  const [posts, setPosts] = useState([]);
  const [view, setView] = useState('communities'); // 'communities' or 'posts'
  const [selectedCommunity, setSelectedCommunity] = useState(null);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState(null);
  const [newCommName, setNewCommName] = useState('');
  const [newCommDesc, setNewCommDesc] = useState('');
  const [newPostTitle, setNewPostTitle] = useState('');
  const [newPostContent, setNewPostContent] = useState('');

  useEffect(() => {
    fetchCommunities();
  }, []);
  const fetchCommunities = async () => {
    setLoading(true);
    try {
      const response = await fetch(`${API_URL}/communities`);
      const data = await response.json();
      setCommunities(data);
    } catch (err) {
      setError('Failed to fetch communities.');
    }
    setLoading(false);
  };

  const fetchPosts = async (communityId) => {
    setLoading(true);
    try {
      const response = await
fetch(`${API_URL}/communities/${communityId}/posts`);
      const data = await response.json();
      setPosts(data);
    } catch (err) {
      setError('Failed to fetch posts.');
    }
    setLoading(false);
  };

  const handleCreateCommunity = async (e) => {
    e.preventDefault();
    try {
      const response = await fetch(`${API_URL}/communities`, {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ name: newCommName, description: newCommDesc }),
      });
      const newCommunity = await response.json();
      setCommunities([...communities, newCommunity]);
      setNewCommName('');
      setNewCommDesc('');
    } catch (err) {
      setError('Failed to create community.');
    }
  };

  const handleCreatePost = async (e) => {
    e.preventDefault();
    try {
      const response = await
fetch(`${API_URL}/communities/${selectedCommunity.id}/posts`, {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ title: newPostTitle, content: newPostContent
}),
      });
      const newPost = await response.json();
      setPosts([...posts, newPost]);
      setNewPostTitle('');
      setNewPostContent('');
    } catch (err) {
      setError('Failed to create post.');
    }
  };

  const handleSelectCommunity = (community) => {
    setSelectedCommunity(community);
    setView('posts');
    fetchPosts(community.id);
  };

  const handleGoBack = () => {
    setView('communities');
    setSelectedCommunity(null);
    setPosts([]);
  };

  if (loading) return <div className="p-8">Loading...</div>;
  if (error) return <div className="p-8 text-red-600">Error: {error}</div>;

  return (
    <div className="max-w-4xl mx-auto p-8 font-sans">
      {view === 'communities' && (
        <div>
          <h1 className="text-3xl font-bold mb-6">Communities</h1>

          <form onSubmit={handleCreateCommunity} className="bg-white p-6
rounded-lg shadow-md mb-8">
            <h2 className="text-xl font-semibold mb-4">Create New
Community</h2>

            <div className="mb-4">
              <label className="block text-sm font-medium
text-gray-700">Name</label>
              <input
                type="text"
                value={newCommName}
                onChange={(e) => setNewCommName(e.target.value)}
                className="mt-1 block w-full px-3 py-2 border border-gray-300
rounded-md shadow-sm"
                placeholder="React Developers"
              />
            </div>

            <div className="mb-4">
              <label className="block text-sm font-medium
text-gray-700">Description</label>
              <input
                type="text"
                value={newCommDesc}
                onChange={(e) => setNewCommDesc(e.target.value)}
                className="mt-1 block w-full px-3 py-2 border border-gray-300
rounded-md shadow-sm"
                placeholder="A place to discuss React"
              />
            </div>
            <button
              type="submit"
              className="py-2 px-4 bg-blue-600 text-white rounded-lg
font-semibold hover:bg-blue-700"
            >
              Create
            </button>
          </form>

          <div className="space-y-4">
            {communities.map((community) => (
              <div
                key={community.id}
                className="bg-white p-6 rounded-lg shadow-md cursor-pointer
hover:shadow-lg transition-shadow"
                onClick={() => handleSelectCommunity(community)}
              >
                <h3 className="text-xl font-semibold
text-blue-700">{community.name}</h3>
                <p className="text-gray-600">{community.description}</p>
              </div>
            ))}
          </div>
        </div>
      )}

      {view === 'posts' && selectedCommunity && (
        <div>
          <button onClick={handleGoBack} className="mb-4 text-blue-600
hover:underline">
            &larr; Back to Communities
          </button>

          <h1 className="text-3xl font-bold mb-6">
            Posts in {selectedCommunity.name}
          </h1>

          <form onSubmit={handleCreatePost} className="bg-white p-6 rounded-lg
shadow-md mb-8">
            <h2 className="text-xl font-semibold mb-4">Create New Post</h2>

            <div className="mb-4">
              <label className="block text-sm font-medium
text-gray-700">Title</label>
              <input
                type="text"
                value={newPostTitle}
                onChange={(e) => setNewPostTitle(e.target.value)}
                className="mt-1 block w-full px-3 py-2 border border-gray-300
rounded-md shadow-sm"
                placeholder="My first post"
              />
            </div>

            <div className="mb-4">
              <label className="block text-sm font-medium
text-gray-700">Content</label>
```

```
              <textarea
                value={newPostContent}
                onChange={(e) => setNewPostContent(e.target.value)}
                rows="3"
                className="mt-1 block w-full px-3 py-2 border border-gray-300
rounded-md shadow-sm"
                placeholder="What's on your mind?"
              />
            </div>

            <button
              type="submit"
              className="py-2 px-4 bg-blue-600 text-white rounded-lg
font-semibold hover:bg-blue-700"
            >
              Post
            </button>
          </form>

          {/* Posts List */}
          <div className="space-y-4">
            {posts.length === 0 ? (
              <p className="text-gray-500">No posts in this community yet.</p>
            ) : (
              posts.map((post) => (
                <div key={post.id} className="bg-white p-6 rounded-lg
shadow-md">
                  <h3 className="text-xl font-semibold">{post.title}</h3>
                  <p className="text-gray-600">{post.content}</p>
                </div>
              ))
            )}
          </div>
        </div>
      )}
    </div>
  );
}
```



How to create contexts in react?

Posted by Booker St. Page • about 12 hours ago

I am trying to manage global authentication state through context api in react but facing bugs and I don't really understand how it works. If anyone can help It'd be great

Add a Comment

Comments (4)

Facing the same issue, can anyone explain context api?????

Booker St. Page • about 12 hours ago

I can explain , I am an industry expert

Booker St. Page • about 12 hours ago