**Voting Game**

A fun web application where users can vote on daily jokes using emojis! Think of it as "Daily news" but with emoji reactions.

**🎯 Project Overview**

This project consists of two main parts:

1. A React frontend where users interact with jokes and vote
2. A Node.js backend API that handles data and voting logic

**🎨 Frontend (React)**

**1. Joke Display**

* Show random jokes in a auestion & answer format
* Display emoji reactions with vote counts
* Include a "Next Joke" button

**2. Voting System**

* Users can vote using emoji reactions
* Vote counts update instantly via API interaction
* 🌟 Bonus: Each joke maintains its own vote history

**3. User Interface**

* Modern, clean design using your choice of:
  + [Tailwind CSS](https://tailwindcss.com/)
  + [Mantine](https://mantine.dev/)
  + you can freely use plain CSS as you wish
* 🌟 Bonus: Make it mobile-responsive!

**4. State Management**

* Handle jokes and votes using React state
* Communicate with backend using fetch API
* 🌟 Bonus Features:
  + Implement React Context for session management
  + Add [React Query](https://tanstack.com/query/latest) for data fetching

**⚙️ Backend (API)**

**Setup Requirements**

1. Install [Node.js](https://nodejs.org/en)
2. Explore [Free Public APIs](https://www.freepublicapis.com/)
3. Use the [TeeHee Joke API](https://www.freepublicapis.com/teehee-joke-api)

**1. Data Storage**

* Use [MongoDB](https://www.mongodb.com/) for joke storage
* Store jokes and their vote counts
* 🌟 Bonus: Add [Express](https://expressjs.com/) & CORS configuration

**2. API Endpoints**

GET /api/joke → Fetch a random joke

POST /api/joke/:id → Submit a vote

* 🌟 Bonus: the rest of CRUD operations

DELETE /api/joke/:id → Delete the joke

UPDATE /api/joke/:id → Update the content of certain joke

**3. Data Structure**

{

"id": "unique\_joke\_id",

"question": "Why did the developer go broke?",

"answer": "Because he used up all his cache!",

"votes": [

{ "value": 10, "label": "😂" },

{ "value": 5, "label": "👍" },

{ "value": 3, "label": "❤️" }

],

"availableVotes": ["😂", "👍", "❤️"]

}

**4. Key Features**

* Track votes per joke
* Support multiple emoji reactions
* Validate incoming votes

**🎉 Success Criteria**

Your project should deliver:

1. App that show a joke with ability to vote via emojis
2. Smooth navigation between jokes via "Next button"
3. Clean, intuitive interface

**💡 Tips for Success**

1. Start with the basic features before attempting bonus items
2. Test your API endpoints thoroughly
3. Pay attention to user experience
4. Keep your code organized and well-commented by separeing each feature with single commit message
5. Use Git & GitHub for version control

Happy coding! 🚀