Documentation  
  
**App.js**  
import React, { useState, useEffect } from 'react';

import axios from 'axios';

function App() {

const [todos, setTodos] = useState([]);

const [task, setTask] = useState('');

const [message, setMessage] = useState('');

useEffect(() => {

fetchTodos();

}, []);

const fetchTodos = async () => {

const res = await axios.get('/todos');

setTodos(res.data);

};

const addTodo = async () => {

if (!task) return;

await axios.post('/todos', { task });

setTask('');

fetchTodos();

};

const deleteTodo = async (id) => {

await axios.delete(`/todos/${id}`);

fetchTodos();

};

const summarizeTodos = async () => {

try {

const res = await axios.post('/summarize');

setMessage(`Success: ${res.data.summary}`);

} catch {

setMessage('Failed to send summary.');

}

};

return (

<div style={{ maxWidth: 600, margin: 'auto', padding: 20 }}>

<h1>Todo Summary Assistant</h1>

<input

type="text"

value={task}

onChange={e => setTask(e.target.value)}

placeholder="Enter new task"

/>

<button onClick={addTodo}>Add Todo</button>

<ul>

{todos.map(todo => (

<li key={todo.id}>

{todo.task}{' '}

<button onClick={() => deleteTodo(todo.id)}>Delete</button>

</li>

))}

</ul>

<button onClick={summarizeTodos}>Summarize & Send to Slack</button>

{message && <p>{message}</p>}

</div>

);

}

export default App;  
  
  
**server.js**  
require('dotenv').config();

console.log('Loaded CO\_API\_KEY:', process.env.CO\_API\_KEY ? 'Yes' : 'No');

console.log('Loaded SLACK\_WEBHOOK\_URL:', process.env.SLACK\_WEBHOOK\_URL ? 'Yes' : 'No');

const express = require('express');

const axios = require('axios');

const { CohereClient } = require('cohere-ai');

const cohere = new CohereClient({ apiKey: process.env.CO\_API\_KEY });

const app = express();

app.use(express.json());

const PORT = process.env.PORT || 5000;

let todos = [];

// GET /todos

app.get('/todos', (req, res) => {

res.json(todos);

});

// POST /todos

app.post('/todos', (req, res) => {

const { task } = req.body;

if (!task) return res.status(400).json({ error: 'Task is required' });

const newTodo = { id: Date.now(), task, completed: false };

todos.push(newTodo);

res.status(201).json(newTodo);

});

// DELETE /todos/:id

app.delete('/todos/:id', (req, res) => {

const id = Number(req.params.id);

todos = todos.filter(todo => todo.id !== id);

res.json({ message: 'Deleted successfully' });

});

// POST /summarize

app.post('/summarize', async (req, res) => {

try {

const todoTexts = todos

.filter(todo => !todo.completed)

.map(todo => `- ${todo.task}`)

.join('\n');

if (!todoTexts) {

return res.status(400).json({ error: 'No pending todos to summarize' });

}

// Cohere generate

const response = await cohere.chat({

model: 'command-r',

message: `Summarize the following todo list in a concise way:\n${todoTexts}`,

temperature: 0.5,

});

const summary = response.text.trim();

// Send summary to Slack

await axios.post(process.env.SLACK\_WEBHOOK\_URL, {

text: `\*Todo Summary:\*\n${summary}`,

});

res.json({ message: 'Summary sent to Slack', summary });

} catch (error) {

console.error('Error in /summarize:', error.response?.data || error.message);

res.status(500).json({ error: 'Failed to generate or send summary' });

}

});

app.listen(PORT, () => {

console.log(`Server running on http://localhost:${PORT}`);

});

**.env**  
DATABASE\_URL=postgresql://postgres:Yasmeen%40187@db.atktgpdamynvqfujxrwu.supabase.co:5432/postgres

CO\_API\_KEY=klzrHwpZvTeLtBlSxAljy3uYWMGrwaPKOnDuOlBd

SLACK\_WEBHOOK\_URL=https://hooks.slack.com/services/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

PORT=5000

**Package.json (frontend waala most probably)**

{

"name": "frontend",

"version": "0.1.0",

"private": true,

"proxy": "http://localhost:5000",

"dependencies": {

"@testing-library/dom": "^10.4.0",

"@testing-library/jest-dom": "^6.6.3",

"@testing-library/react": "^16.3.0",

"@testing-library/user-event": "^13.5.0",

"axios": "^1.9.0",

"react": "^19.1.0",

"react-dom": "^19.1.0",

"react-scripts": "5.0.1",

"web-vitals": "^2.1.4"

},

"scripts": {

"start": "react-scripts start",

"build": "react-scripts build",

"test": "react-scripts test",

"eject": "react-scripts eject"

},

"eslintConfig": {

"extends": [

"react-app",

"react-app/jest"

]

},

"browserslist": {

"production": [

">0.2%",

"not dead",

"not op\_mini all"

],

"development": [

"last 1 chrome version",

"last 1 firefox version",

"last 1 safari version"

]

}

}