■ Notes App - Full Stack CRUD (Node + React)

Backend - server.js

```
const express = require("express");
const cors = require("cors");

const app = express();
const PORT = 5000;

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

let notes = [
  { id: 1, title: "First Note", content: "This is my first note" },
  { id: 2, title: "Second Note", content: "This is another note" }
];

// Get all notes
app.get("/notes", (req, res) => res.json(notes));

// Add new note
app.post("/notes", (req, res) => {
  const newNote = {
    id: notes.length + 1,
    title: req.body.title,
    content: req.body.content
  };
  notes.push(newNote);
  res.json(newNote);
});

// Update note
app.put("/notes/:id", (req, res) => {
  const id = parseInt(req.params.id);
  notes = notes.map((n) =>
    n.id === id ? { ...n, title: req.body.title, content: req.body.content } : n
  );
  res.json({ success: true });
});

// Delete note
app.delete("/notes/:id", (req, res) => {
  const id = parseInt(req.params.id);
  notes = notes.filter((n) => n.id !== id);
  res.json({ success: true });
});

app.listen(PORT, () =>
  console.log(`■ Notes API running on http://localhost:${PORT}`)
);
```

Backend - package.json

```
{
  "name": "notes-backend",
  "version": "1.0.0",
  "main": "server.js",
  "scripts": { "start": "node server.js" },
  "dependencies": {
    "cors": "^2.8.5",
    "express": "^4.18.2"
  }
}
```

Frontend - App.js

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

```
function App() {
  const [notes, setNotes] = useState([]);
  const [title, setTitle] = useState("");
  const [content, setContent] = useState("");
  const [editId, setEditId] = useState(null);

  useEffect(() => {
    fetch("http://localhost:5000/notes")
      .then((res) => res.json())
      .then(setNotes);
  }, []);

  const addNote = () => {
    if (!title || !content) return;
    fetch("http://localhost:5000/notes", {
      method: "POST",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify({ title, content }),
    })
      .then((res) => res.json())
      .then((newNote) => {
        setNotes([...notes, newNote]);
        setTitle("");
        setContent("");
      });
  };

  const updateNote = (id) => {
    fetch(`http://localhost:5000/notes/${id}`, {
      method: "PUT",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify({ title, content }),
    }).then(() => {
      setNotes(
        notes.map((n) =>
          n.id === id ? { ...n, title, content } : n
        )
      );
      setTitle("");
      setContent("");
      setEditId(null);
    });
  };

  const deleteNote = (id) => {
    fetch(`http://localhost:5000/notes/${id}`, { method: "DELETE" })
      .then(() => setNotes(notes.filter((n) => n.id !== id)));
  };

  const startEdit = (note) => {
    setEditId(note.id);
    setTitle(note.title);
    setContent(note.content);
  };

  return (
    <div className="bg-gray-100 min-h-screen flex flex-col items-center p-6">
      <h1 className="text-3xl font-bold mb-4">■ Notes App</h1>

      <div className="bg-white p-4 shadow-md rounded-lg w-full max-w-md mb-6">
        <input
          type="text"
          placeholder="Title"
          value={title}
          onChange={(e) => setTitle(e.target.value)}
          className="w-full p-2 border rounded mb-2"
        />
        <textarea
          placeholder="Content"
          value={content}
          onChange={(e) => setContent(e.target.value)}
          className="w-full p-2 border rounded mb-2"
        />
        <button
          onClick={() => (editId ? updateNote(editId) : addNote())}
          className="bg-blue-500 text-white px-4 py-2 rounded"
        >
```

```
          {editId ? "Update Note" : "Add Note"}
        </button>
      </div>

      <div className="w-full max-w-md">
        {notes.map((note) => (
          <div key={note.id} className="bg-white p-4 shadow-md rounded-lg mb-3">
            <h2 className="text-xl font-semibold">{note.title}</h2>
            <p className="text-gray-700">{note.content}</p>
            <div className="mt-2 flex gap-2">
              <button
                onClick={() => startEdit(note)}
                className="bg-yellow-400 text-white px-2 py-1 rounded"
              >
                Edit
              </button>
              <button
                onClick={() => deleteNote(note.id)}
                className="bg-red-500 text-white px-2 py-1 rounded"
              >
                Delete
              </button>
            </div>
          </div>
        ))}
      </div>
    </div>
  );
}
export default App;
```

Frontend - tailwind.config.js

```
module.exports = {
  content: ["./src/**/*.{js,jsx,ts,tsx}"],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

Frontend - package.json

```
{
  "name": "notes-frontend",
  "version": "1.0.0",
  "private": true,
  "dependencies": {
    "react": "^18.2.0",
    "react-dom": "^18.2.0",
    "react-scripts": "5.0.1"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build"
  },
  "devDependencies": {
    "tailwindcss": "^3.0.24"
  }
}
```

Instructions

1. Start backend:
   ```
   cd backend
   npm install
   npm start
   Runs on http://localhost:5000
   ```

2. Start frontend:

```
cd frontend
npm install
npm start
Runs on http://localhost:3000
```