```
# Create project directory
mkdir todo-app
cd todo-app

# Initialize frontend (React)
npx create-react-app client

# Initialize backend (Express)
mkdir server
cd server
npm init -y
npm install express body-parser cors mongoose
const express = require('express');
const bodyParser = require('body-parser');
const cors = require('cors');
const mongoose = require('mongoose');

const app = express();
const PORT = process.env.PORT || 5000;

app.use(cors());
app.use(bodyParser.json());
```

```
// MongoDB Connection (replace with your
MongoDB URI)
const MONGO_URI =
"mongodb://localhost:27017/todos"; // Example
local URI
mongoose.connect(MONGO_URI, {
    useNewUrlParser: true,
    useUnifiedTopology: true
}).then(() => console.log("Connected to
MongoDB")).catch(err =>
console.error("MongoDB connection error:",
err));


// Todo Schema
const todoSchema = new mongoose.Schema({
    text: { type: String, required: true },
    completed: { type: Boolean, default: false },
});

const Todo = mongoose.model('Todo',
todoSchema);
```

```
// API Endpoints
app.get('/api/todos', async (req, res) => {
  try {
    const todos = await Todo.find();
    res.json(todos);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});

app.post('/api/todos', async (req, res) => {
  try {
    const newTodo = new Todo(req.body);
    const savedTodo = await newTodo.save();
    res.status(201).json(savedTodo);
  } catch (err) {
    res.status(400).json({ error: err.message });
  }
});

app.put('/api/todos/:id', async (req, res) => {
  try {
    const updatedTodo = await
```

```
Todo.findByIdAndUpdate(req.params.id,
req.body, { new: true });
        res.json(updatedTodo);
    } catch (err) {
        res.status(400).json({ error: err.message });
    }
});

app.delete('/api/todos/:id', async (req, res) => {
    try {
        await
Todo.findByIdAndDelete(req.params.id);
        res.status(204).send(); // No content
    } catch (err) {
        res.status(400).json({ error: err.message });
    }
});

app.listen(PORT, () => console.log(`Server
listening on port ${PORT}`));
import React, { useState, useEffect } from 'react';
import './App.css';
```

```
function App() {
  const [todos, setTodos] = useState([]);
  const [newTodo, setNewTodo] = useState('');

  useEffect(() => {
    fetch('/api/todos')
      .then(res => res.json())
      .then(data => setTodos(data));
  }, []);

  const addTodo = () => {
    fetch('/api/todos', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ text: newTodo }),
    }).then(res => res.json()).then(newTodoData
=> {
      setTodos([...todos, newTodoData]);
      setNewTodo('');
    });
  };

  const toggleTodo = (id, completed) => {
```

```
    fetch(`/api/todos/${id}`, {
      method: 'PUT',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ completed: !
completed }),
    }).then(() => {
      setTodos(todos.map(todo =>
        todo._id === id ? {...todo, completed: !
completed} : todo
      ))
    })
  };

  const deleteTodo = (id) => {
    fetch(`/api/todos/${id}`, {
      method: 'DELETE'
    }).then(() => {
      setTodos(todos.filter(todo => todo._id !==
id))
    })
  }

  return (
```

```
  return (
    <div className="App">
      <h1>Todo App</h1>
      <input type="text" value={newTodo}
onChange={e => setNewTodo(e.target.value)} />
      <button onClick={addTodo}>Add Todo</
button>
      <ul>
        {todos.map(todo => (
          <li key={todo._id}>
            <input type="checkbox"
checked={todo.completed} onChange={() =>
toggleTodo(todo._id, todo.completed)} />
            <span style={{ textDecoration:
todo.completed ? 'line-through' : 'none' }}
>{todo.text}</span>
            <button onClick={() =>
deleteTodo(todo._id)}>Delete</button>
          </li>
        ))}
      </ul>
    </div>
  );
```

```
>{todo.text}</span>
          <button onClick={() =>
deleteTodo(todo._id)}>Delete</button>
        </li>
      ))}
    </ul>
  </div>
 );
}

export default App;
# In the server directory:
cd server
node index.js

# In the client directory:
cd client
npm start
```