SOURCE CODE

Backend (Express.js)

App.j

Const express = require(‘express’);

Const bodyParser = require(‘body-parser’);

Const cors = require(‘cors’);

Const app = express();

App.use(bodyParser.json());

App.use(cors());

Let tasks = [];

Let completedTasks = [];

// Get tasks

App.get(‘/tasks’, (req, res) => {

Res.json({ tasks, completedTasks });

});

// Add a new task

App.post(‘/tasks’, (req, res) => {

Const task = req.body.task;

If (task) {

Tasks.push(task);

Res.status(201).json({ message: ‘Task added successfully!’ });

} else {

Res.status(400).json({ message: ‘Task cannot be empty!’ });

}

});

// Mark a task as complete

App.post(‘/tasks/complete’, (req, res) => {

Const index = req.body.index;

If (index !== undefined && tasks[index]) {

completedTasks.push(tasks[index]);

tasks.splice(index, 1);

res.json({ message: ‘Task marked as complete!’ });

} else {

Res.status(400).json({ message: ‘Invalid task index!’ });

}

});

// Remove a completed task

App.delete(‘/tasks/completed’, (req, res) => {

Const index = req.body.index;

If (index !== undefined && completedTasks[index]) {

completedTasks.splice(index, 1);

res.json({ message: ‘Task removed successfully!’ });

} else {

Res.status(400).json({ message: ‘Invalid completed task index!’ });

}

});

App.listen(3001, () => {

Console.log(‘Server running on <http://localhost:3001>’);

});

Frontend (React)

App.js

Import React, { useState, useEffect } from ‘react’;

Function App() {

Const [tasks, setTasks] = useState([]);

Const [completedTasks, setCompletedTasks] = useState([]);

Const [newTask, setNewTask] = useState(‘’);

// Fetch tasks from the backend

useEffect(() => {

fetch(‘http://localhost:3001/tasks’)

.then((response) => response.json())

.then((data) => {

setTasks(data.tasks); setCompletedTasks(data.completedTasks);

});

}, []);

// Add a new task

Const addTask = () => {

If (newTask.trim()) { Fetch(‘http://localhost:3001/tasks’, {

Method: ‘POST’,

Headers: { ‘Content-Type’: ‘application/json’ },

Body: JSON.stringify({ task: newTask }), })

.then(() => {

setTasks([…tasks, newTask]);

setNewTask(‘’);

});

}

};

// Mark a task as complete

Const completeTask = (index) => { Fetch(‘http://localhost:3001/tasks/complete’, {

Method: ‘POST’,

Headers: { ‘Content-Type’: ‘application/json’ },

Body: JSON.stringify({ index }),

}).then(() => {

Const completed = tasks[index];

setTasks(tasks.filter((\_, i) => I !== index)); setCompletedTasks([…completedTasks, completed]);

});

};

// Remove a completed task

Const removeTask = (index) => { Fetch(‘http://localhost:3001/tasks/completed’, {

Method: ‘DELETE’,

Headers: { ‘Content-Type’: ‘application/json’ },

Body: JSON.stringify({ index }),

}).then(() => { setCompletedTasks(completedTasks.filter((\_, i) => I !== index));

});

};

Return (

<div style={styles.container}>

<h1 style={styles.header}>Todo App</h1>

<div style={styles.inputContainer}>

<input

Type=”text”

Value={newTask}

onChange={€ => setNewTask(e.target.value)}

placeholder=”Add new task”

style={styles.input}

/>

<button onClick={addTask} style={styles.addButton}>Add Task</button>

</div>

<h2 style={styles.subHeader}>Tasks</h2>

<ul style={styles.taskList}>

{tasks.map((task, index) => (

<li key={index} style={styles.taskItem}>

{task}

<button onClick={() => completeTask(index)} style={styles.completeButton}>

Complete

</button>

</li>

))}

</ul>

<h2 style={styles.subHeader}>Completed Tasks</h2>

<ul style={styles.taskList}>

{completedTasks.map((task, index) => (

<li key={index} style={styles.completedItem}>

{task}

<button onClick={() => removeTask(index)} style={styles.removeButton}>

Remove

</button>

</li>

))}

</ul>

</div>

);

}

Const styles = {

Container: {

textAlign: ‘center’,

margin: ‘30px auto’,

maxWidth: ‘600px’,

fontFamily: ‘Arial, Helvetica, sans-serif’,

backgroundColor: ‘#f9f3db’,

borderRadius: ‘12px’,

padding: ‘25px’,

boxShadow: ‘0 8px 16px rgba(0, 0, 0, 0.15)’,

},

Header: {

Color: ‘#ff6347’,

fontSize: ‘42px’,

marginBottom: ‘25px’,

fontWeight: ‘700’,

},

subHeader: {

color: ‘#ff4500’,

fontSize: ‘28px’,

margin: ‘20px 0’,

fontWeight: ‘500’,

},

inputContainer: {

display: ‘flex’,

justifyContent: ‘center’,

marginBottom: ‘20px’,

paddingBottom: ‘10px’,

borderBottom: ‘2px solid #ffa500’,

},

Input: {

Padding: ‘15px’,

fontSize: ‘18px’,

borderRadius: ‘8px’,

border: ‘2px solid #ffa500’,

width: ‘70%’,

marginRight: ‘15px’,

transition: ‘0.3s’,

},

addButton: {

padding: ‘15px 30px’,

fontSize: ‘18px’,

backgroundColor: ‘#ff6347’,

color: ‘white’,

border: ‘none’,

borderRadius: ‘8px’,

cursor: ‘pointer’,

transition: ‘background-color 0.3s’,

},

addButtonHover: {

backgroundColor: ‘#e53e36’,

},

taskList: {

listStyleType: ‘none’,

padding: 0,

margin: 0,

},

taskItem: {

display: ‘flex’,

justifyContent: ‘space-between’,

alignItems: ‘center’,

backgroundColor: ‘#fff’,

padding: ‘15px’,

margin: ‘10px 0’,

borderRadius: ‘8px’,

boxShadow: ‘0 4px 8px rgba(0, 0, 0, 0.1)’,

color: ‘#333’,

},

completeButton: {

padding: ‘8px 16px’,

fontSize: ‘16px’,

backgroundColor: ‘#32cd32’,

color: ‘white’,

border: ‘none’,

borderRadius: ‘8px’,

cursor: ‘pointer’,

transition: ‘background-color 0.3s’,

},

completedItem: {

display: ‘flex’,

justifyContent: ‘space-between’,

alignItems: ‘center’,

backgroundColor: ‘#d1e7dd’,

padding: ‘15px’,

margin: ‘10px 0’,

borderRadius: ‘8px’,

textDecoration: ‘line-through’,

color: ‘#6c757d’,

},

removeButton: {

padding: ‘8px 16px’,

fontSize: ‘16px’,

backgroundColor: ‘#dc3545’,

color: ‘white’,

border: ‘none’

borderRadius: ‘8px’,

cursor: ‘pointer’,

transition: ‘background-color 0.3s’,

},

};

Export default App;

Index.js

Import React from ‘react’;

Import ReactDOM from ‘react-dom/client’;

Import App from ‘./App’;

Const root = ReactDOM.createRoot(document.getElementById(‘root’));

Root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);