**Advanced Web Technologies**

**Submitted By: Fatima Arshad**

**Student ID: 21007105036**

**Department: BS-IT**

**Batch -07**

**Submitted To: Sir Saffiullah**

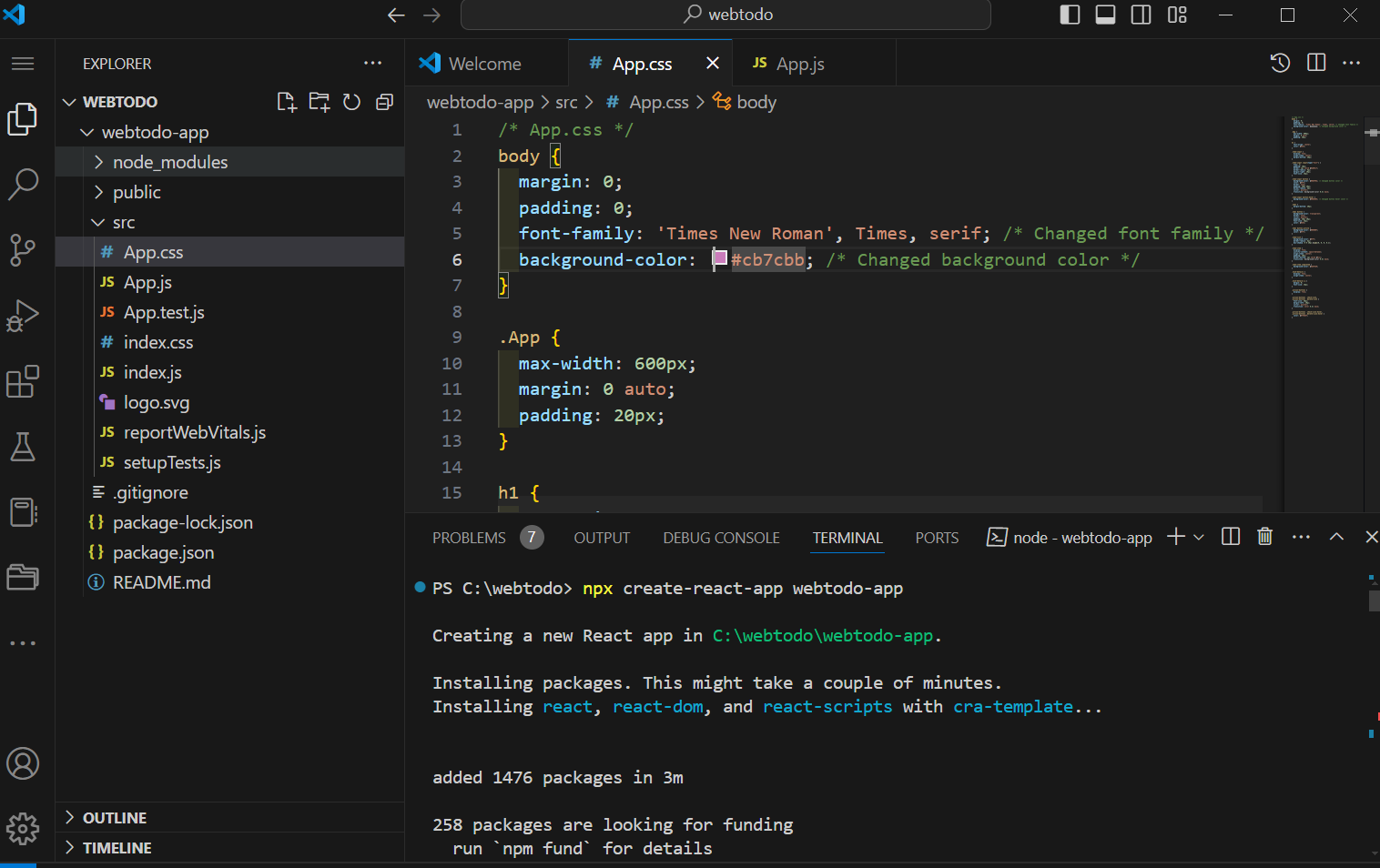
**

**University Of Management And Technology**

**Sialkot**

# Q: Build a single-page to-do list application using React.js

1. Created a folder and run **npx create-react-app dua-app** command in the terminal.



1. Wrote **js** and **css** code.

* **App.js Code:**

// App.js

import React, { useState, useEffect } from 'react';

import './App.css';

import { AiOutlineDelete, AiOutlineEdit } from 'react-icons/ai';

import { BsCheckCircle } from 'react-icons/bs';

function App() {

  const [isCompleteScreen, setIsCompleteScreen] = useState(false);

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

  const [newTask, setNewTask] = useState('');

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

  useEffect(() => {

    const savedTodos = JSON.parse(localStorage.getItem('todos'));

    const savedCompletedTasks = JSON.parse(localStorage.getItem('completedTasks'));

    if (savedTodos) {

      setTodos(savedTodos);

    }

    if (savedCompletedTasks) {

      setCompletedTasks(savedCompletedTasks);

    }

  }, []);

  useEffect(() => {

    localStorage.setItem('todos', JSON.stringify(allTodos));

  }, [allTodos]);

  useEffect(() => {

    localStorage.setItem('completedTasks', JSON.stringify(completedTasks));

  }, [completedTasks]);

  const handleAddTodo = () => {

    if (newTask.trim() === '') return;

    const newTodoItem = {

      id: Date.now(),

      task: newTask,

      completed: false,

    };

    setTodos([...allTodos, newTodoItem]);

    setNewTask('');

  };

  const handleDeleteTodo = (id) => {

    const updatedTodos = allTodos.filter(todo => todo.id !== id);

    setTodos(updatedTodos);

  };

  const handleToggleComplete = (id) => {

    const updatedTodos = allTodos.map(todo =>

      todo.id === id ? { ...todo, completed: !todo.completed } : todo

    );

    setTodos(updatedTodos);

    const completedTask = allTodos.find(todo => todo.id === id);

    if (completedTask) {

      setCompletedTasks([...completedTasks, completedTask]);

    }

  };

  const handleDeleteCompletedTask = (id) => {

    const updatedCompletedTasks = completedTasks.filter(task => task.id !== id);

    setCompletedTasks(updatedCompletedTasks);

  };

  return (

    <div className="App">

      <h1>Task Management</h1>

      <h2>To-do-List</h2>

      <div className="todo-input">

        <input

          type="text"

          placeholder="Enter your task"

          value={newTask}

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

        />

        <button onClick={handleAddTodo}>Add Task</button>

      </div>

      <div className="tabs">

        <button className={!isCompleteScreen ? 'active' : ''} onClick={() => setIsCompleteScreen(false)}>Todo</button>

        <button className={isCompleteScreen ? 'active' : ''} onClick={() => setIsCompleteScreen(true)}>Completed</button>

      </div>

      {isCompleteScreen ? (

        <div className="todo-list">

          {completedTasks.map(task => (

            <div className="todo-item" key={task.id}>

              <div className="task-details">

                <p>{task.task}</p>

              </div>

              <div className="action-buttons">

                <AiOutlineDelete

                  className="delete-icon"

                  onClick={() => handleDeleteCompletedTask(task.id)}

                  title="Delete Task"

                />

              </div>

            </div>

          ))}

        </div>

      ) : (

        <div className="todo-list">

          {allTodos.map(todo => (

            <div className={`todo-item ${todo.completed ? 'completed' : ''}`} key={todo.id}>

              <div className="task-details">

                <input

                  type="checkbox"

                  checked={todo.completed}

                  onChange={() => handleToggleComplete(todo.id)}

                />

                <p>{todo.task}</p>

              </div>

              <div className="action-buttons">

                <BsCheckCircle

                  className="check-icon"

                  onClick={() => handleToggleComplete(todo.id)}

                  title="Toggle Complete"

                />

                <AiOutlineDelete

                  className="delete-icon"

                  onClick={() => handleDeleteTodo(todo.id)}

                  title="Delete Task"

                />

              </div>

            </div>

          ))}

        </div>

      )}

    </div>

  );

}

export default App;

* **App.css code:**

/\* App.css \*/

body {

  margin: 0;

  padding: 0;

  font-family: 'Times New Roman', Times, serif; /\* Changed font family \*/

  background-color: #cb7cbb; /\* Changed background color \*/

}

.App {

  max-width: 600px;

  margin: 0 auto;

  padding: 20px;

}

h1 {

  text-align: center;

  color: #333;

}

.todo-input {

  display: flex;

  align-items: center;

  margin-bottom: 20px;

}

.todo-input input[type="text"] {

  flex: 1;

  padding: 10px;

  border: 1px solid #ed92cf;

  border-radius: 5px;

  margin-right: 10px;

  font-size: 16px;

}

.todo-input button {

  background-color: #02070e; /\* Changed button color \*/

  color: #fff;

  border: none;

  padding: 10px 20px;

  border-radius: 5px;

  cursor: pointer;

  transition: background-color 0.3s ease;

}

.todo-input button:hover {

  background-color: #020203; /\* Changed button hover color \*/

}

.tabs {

  margin-bottom: 20px;

}

.tabs button {

  background-color: transparent;

  border: none;

  cursor: pointer;

  padding: 10px 20px;

  font-size: 16px;

  color: #333;

}

.tabs button.active {

  background-color: #040607;

  color: #fff;

}

.todo-list {

  background-color: #fff;

  border-radius: 5px;

  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

.todo-item {

  display: flex;

  justify-content: space-between;

  align-items: center;

  padding: 10px;

  border-bottom: 1px solid #ccc;

  transition: background-color 0.3s ease;

}

.todo-item.completed {

  background-color: #e8f5e9;

}

.task-details {

  display: flex;

  align-items: center;

}

.task-details p {

  margin: 0;

  font-size: 16px;

}

.action-buttons {

  display: flex;

}

.action-buttons .check-icon,

.action-buttons .delete-icon {

  font-size: 20px;

  margin-left: 10px;

  cursor: pointer;

  transition: color 0.3s ease;

}

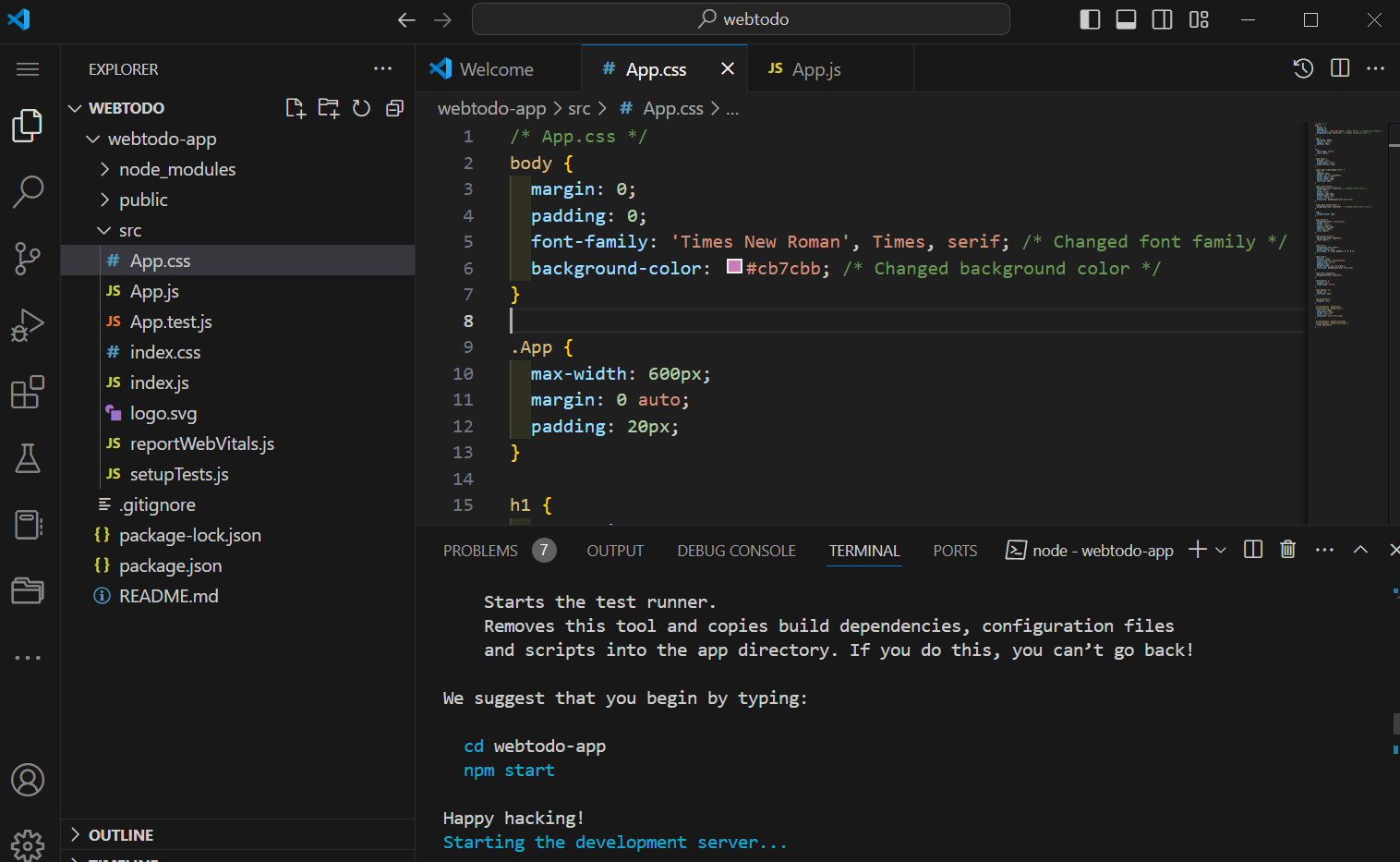
.action-buttons .check-icon:hover,

.action-buttons .delete-icon:hover {

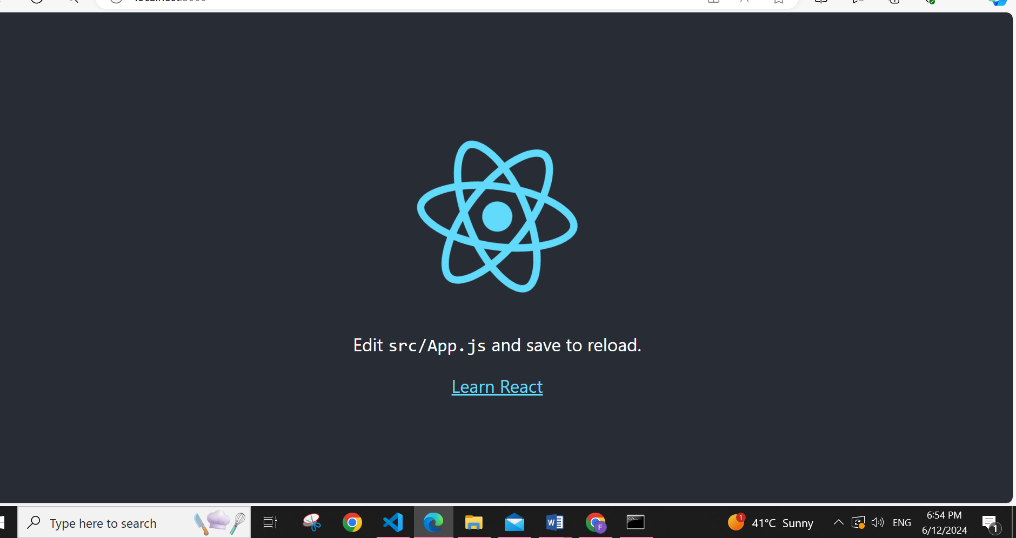
  color: #ff6347;

}

1. After creating a new **React application** using command **npx create-react-app webtodo-app.**
2. Run this command **cd webtodo-app**



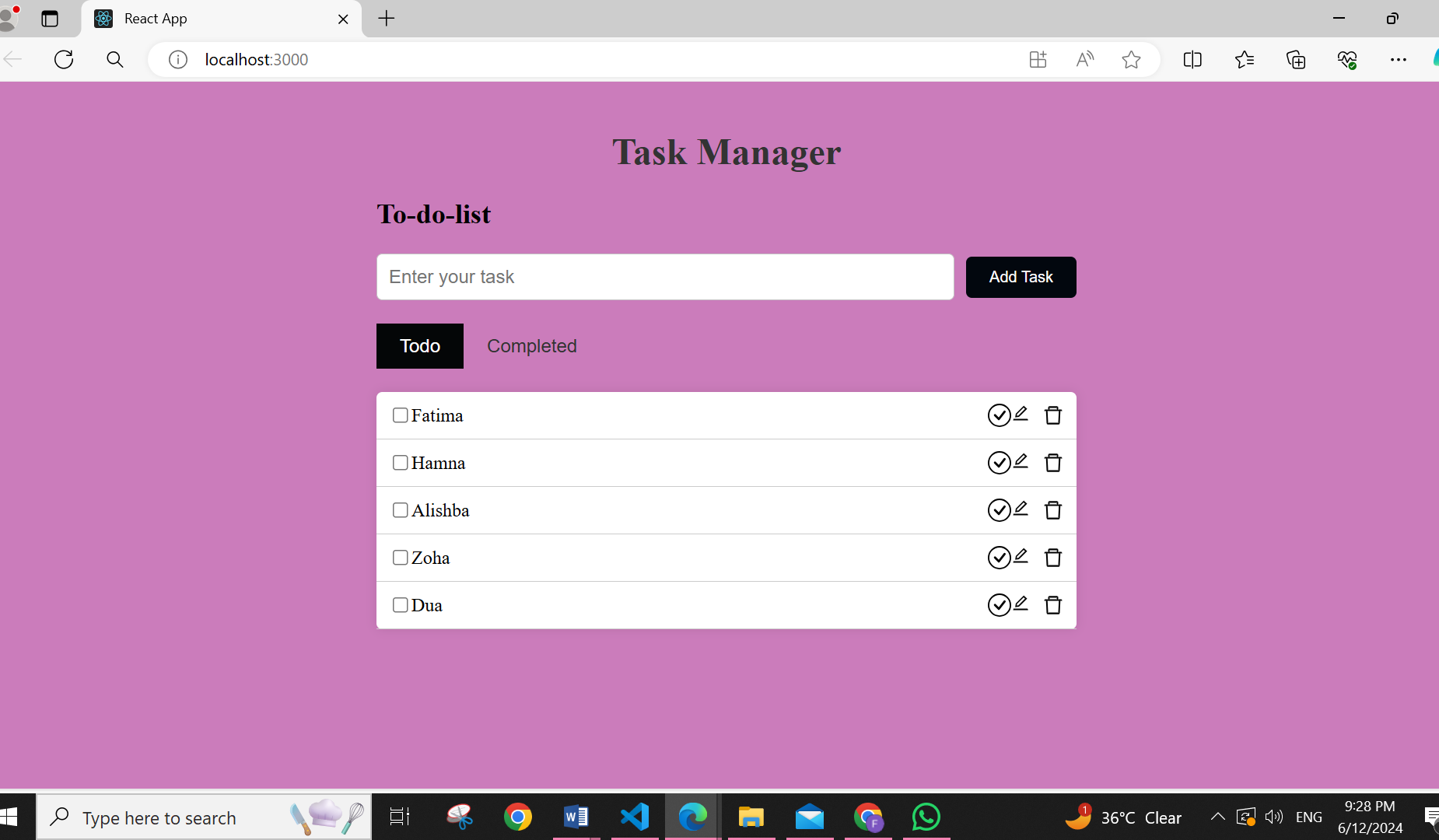
As, it is compiled successfully, it will launch your new React application in a web browser, typically available at **http://localhost:3000.**



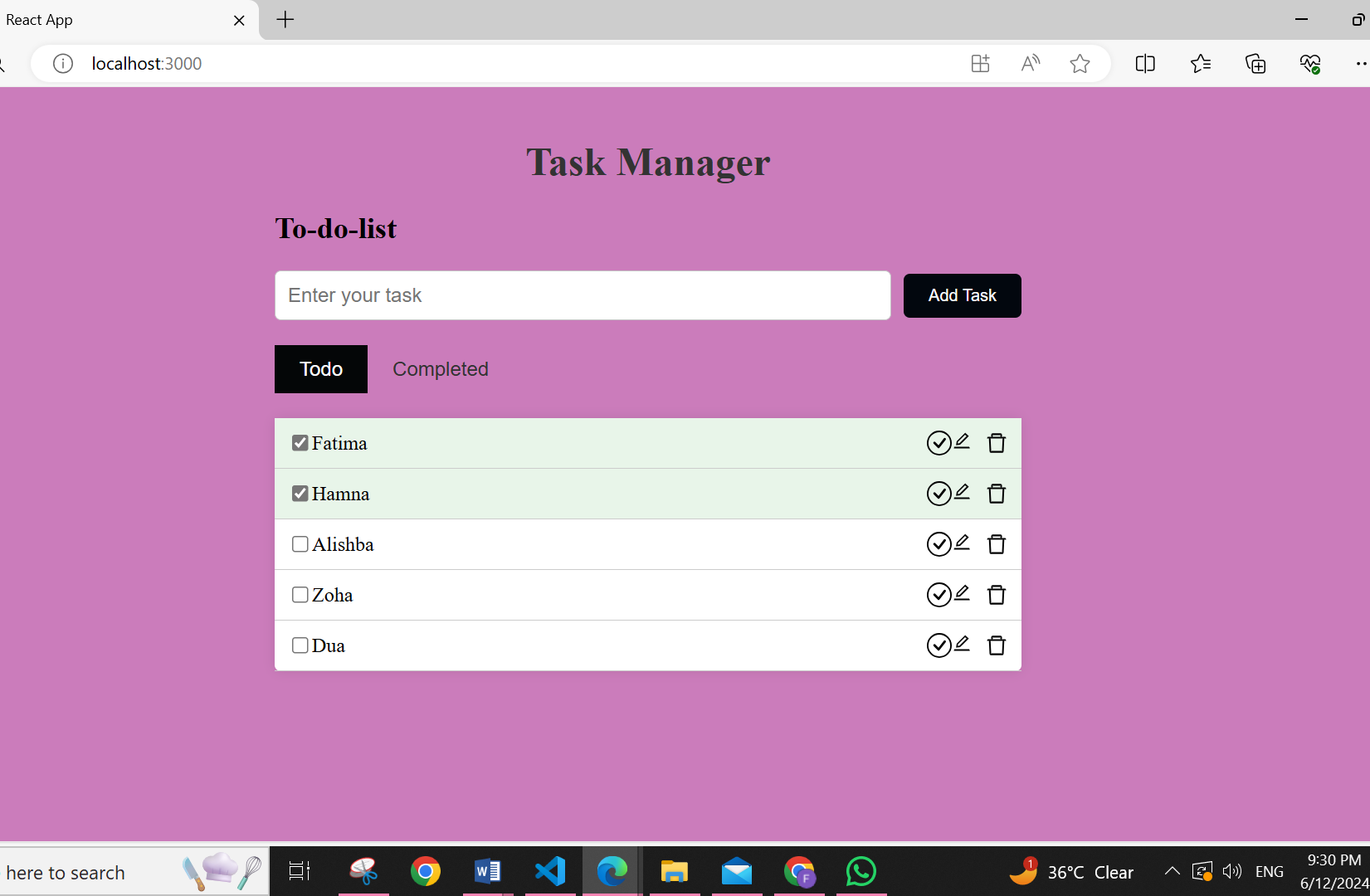
1. **User interface:**

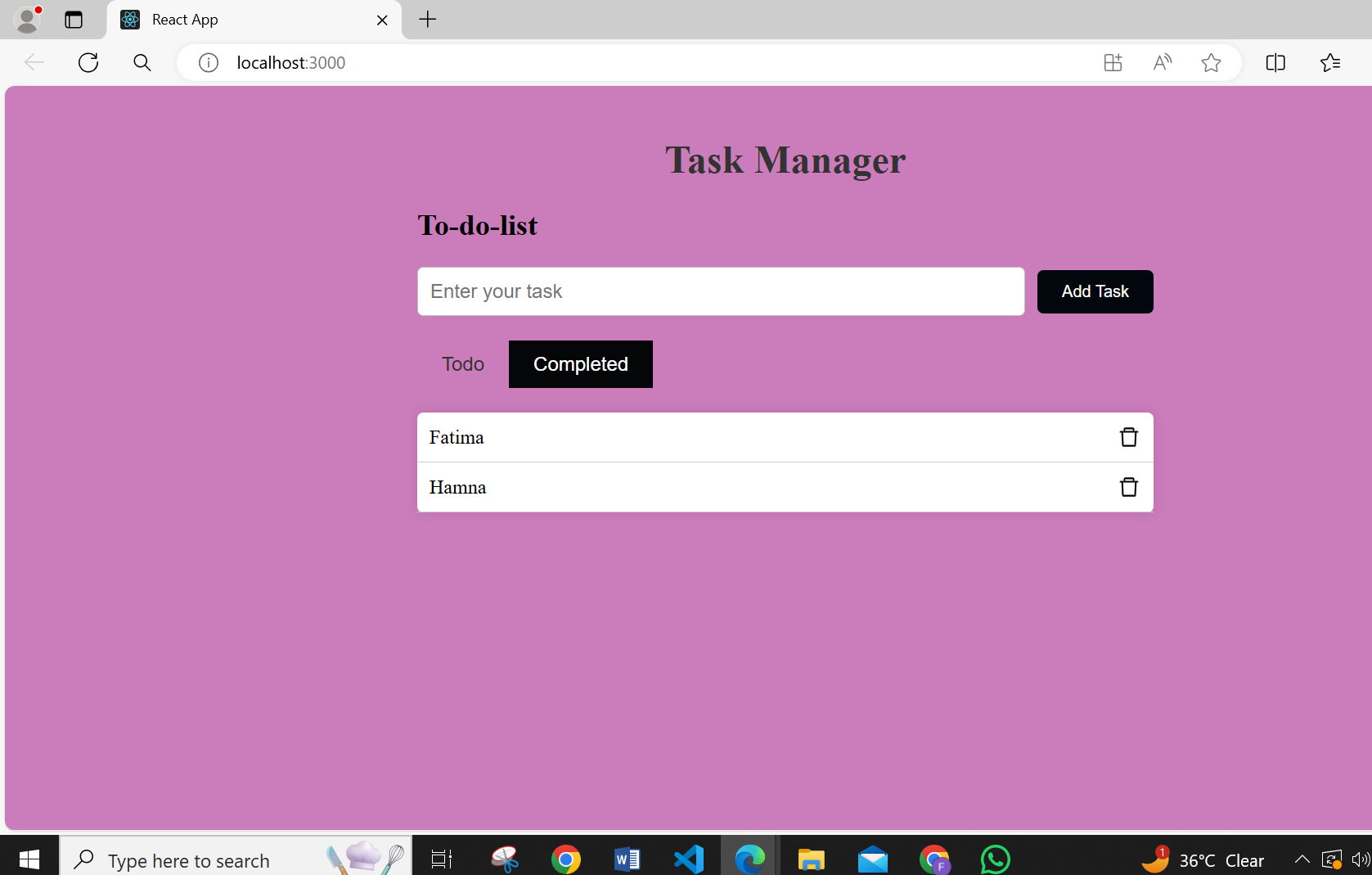
You can add, update and delete your todos list.

**1.Add:**

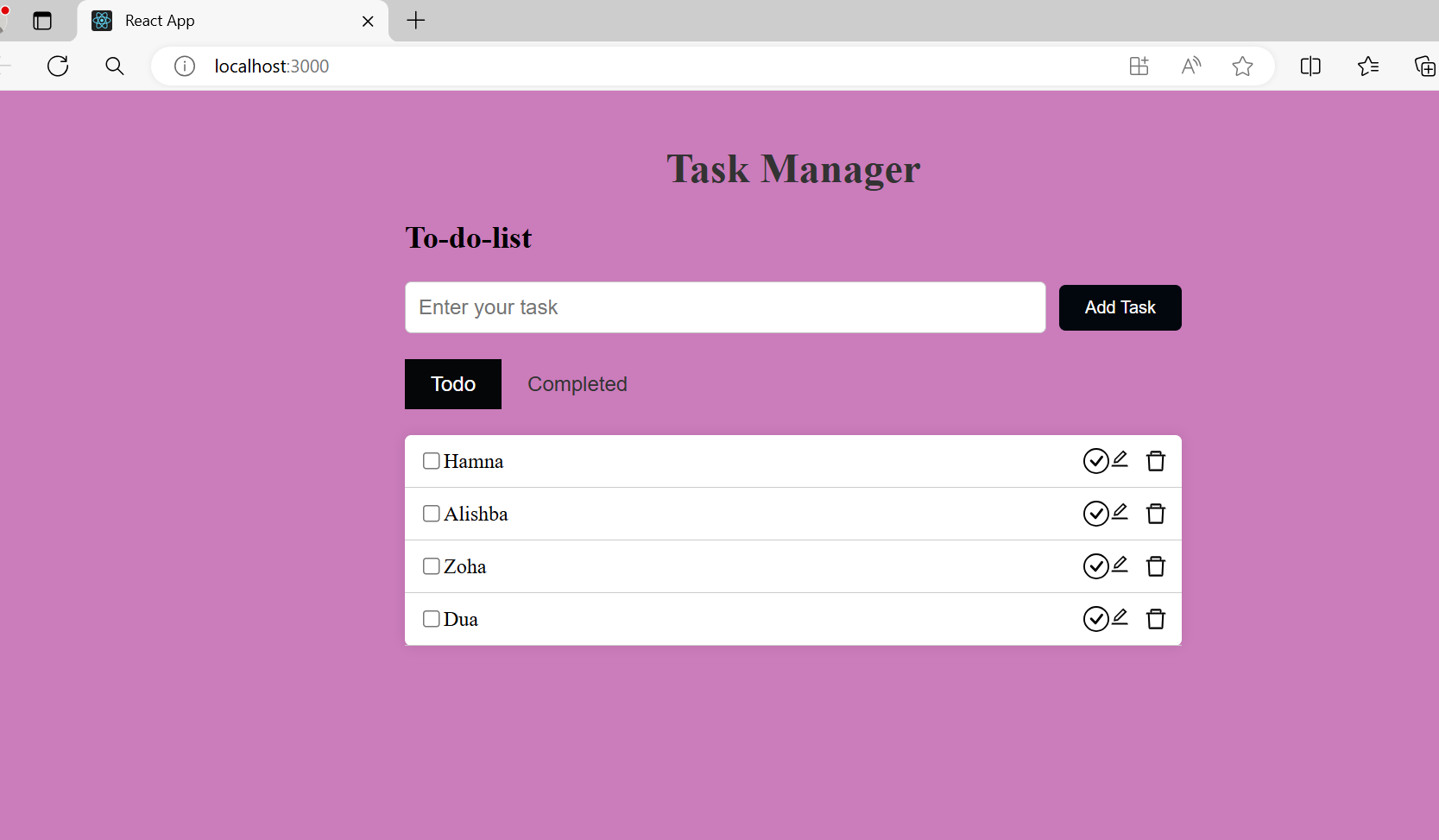


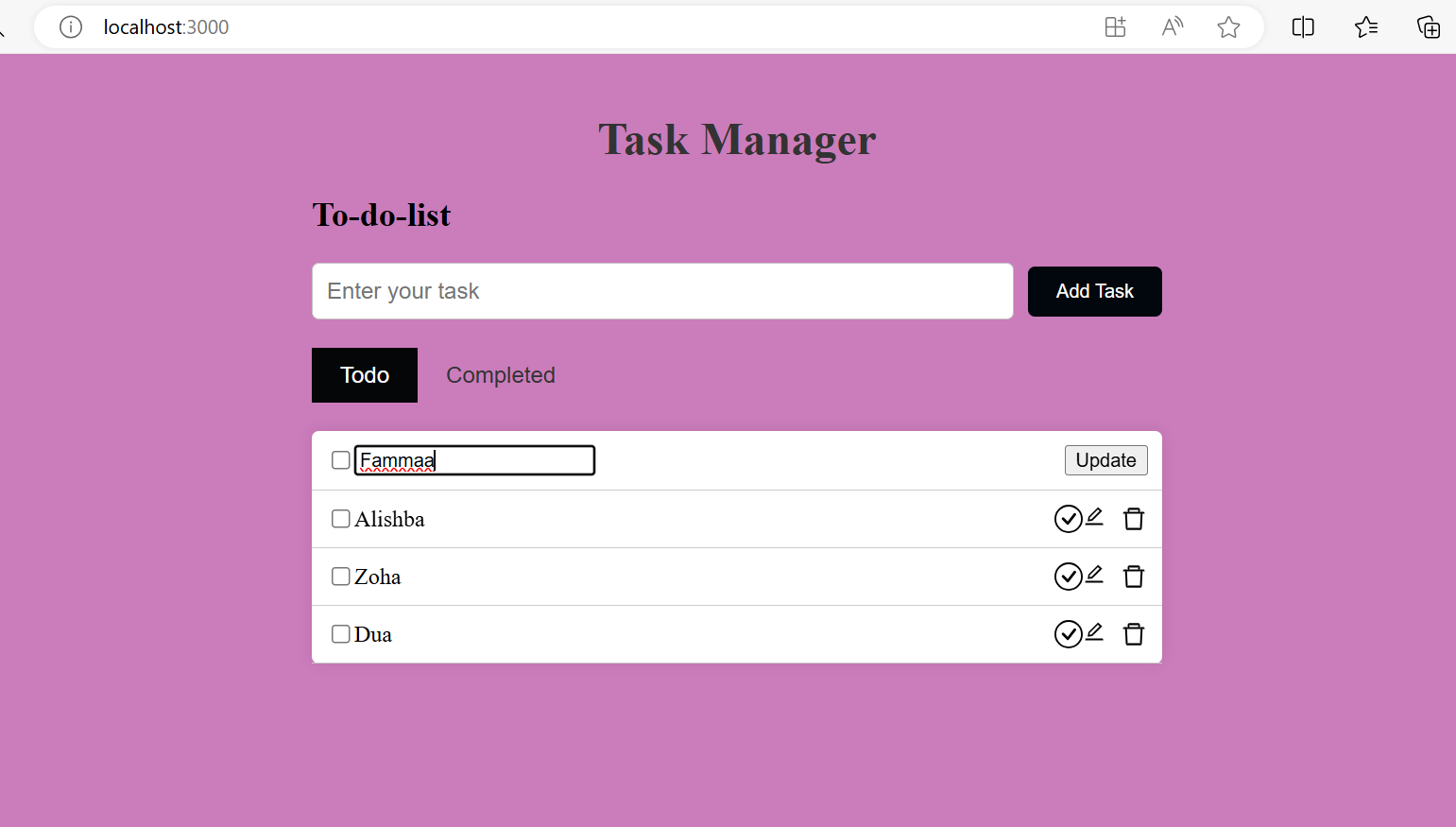
**2. Completed:**

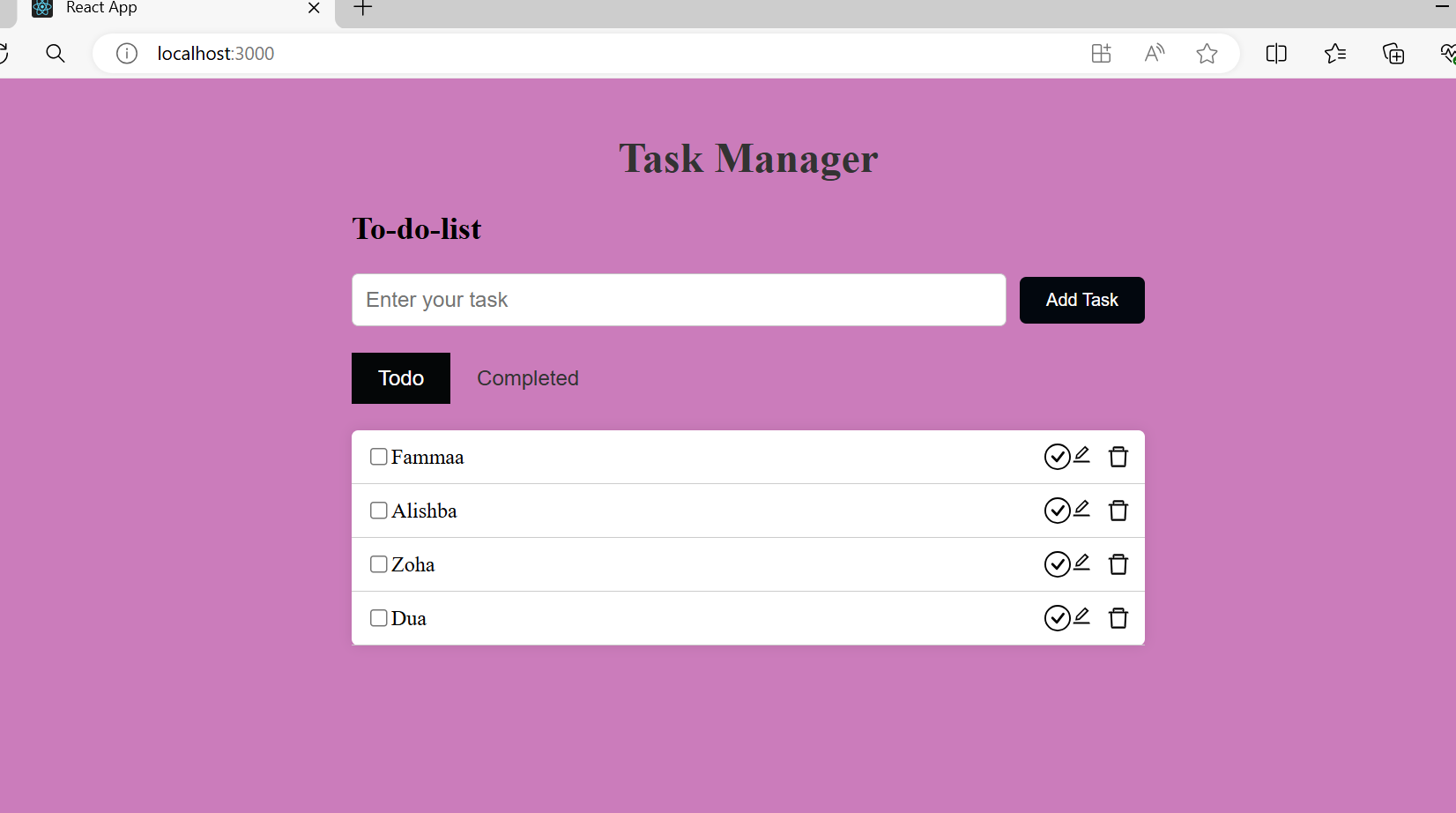




**3. Update:**







**4. Delete :**

