Assignment-3

React.js

1. Build a Small Task Management Website in React.js

Create a small website for managing tasks. The website should have the following features:

1. Class Components: Implement a class component called Task Manager that will serve as the main container for managing tasks. This component should have a state to store the list of tasks.

2. Functional Components: Create a functional component called Task List responsible for displaying a list of tasks. It should receive the list of tasks as props from the Task Manager component.

3. Nested Components: Inside the Task List component, create a child component called Task Item that represents an individual task.

4. Conditional and Looping Constructs: Implement conditional rendering in the Task List component. If there are no tasks, display a message like "No tasks to display."; Use a loop (e.g., map) to render each Task Item.

5. State: In the Task Manager component, manage the list of tasks using state. Initially, you can provide some sample tasks within the state.

6. Props: Pass the list of tasks as props from the Task Manager component to the Task List component. Each Task Item should receive a task object as a prop.

7. Form (Controlled Component): Create a form component called Task Form within the Task Manager component. This form should allow users to add new tasks. Use controlled components to manage form input values.

The user should be able to:

● View a list of tasks (if any).

● Add new tasks using the form.

● Mark tasks as completed (implement a button or checkbox to toggle task completion status).

● Delete tasks (implement a button to remove a task).

Ensure that the website is visually appealing and easy to use. You can style it using CSS or a UI library of your choice. This assignment will cover various React.js concepts, making it a comprehensive exercise for your students.

Answer:

TaskManager.jsx

```
import React, { Component } from 'react';
import TaskList from './TaskList';
import TaskForm from './TaskForm';
class TaskManager extends Component {
  state = {
    tasks: [
      { id: 1, title: 'writing', completed: false },
      { id: 2, title: 'Reading', completed: false }
    ]
  };
  addTask = (title) => {
    const newTask = {
      id: Date.now(),
      title,
      completed: false
    };
    this.setState({ tasks: [...this.state.tasks, newTask] });
  };
  toggleTaskCompletion = (id) => {
    this.setState({
      tasks: this.state.tasks.map(task =>
        task.id === id ? { ...task, completed: !task.completed } : task
      )
    });
  };
  deleteTask = (id) => {
    this.setState({
      tasks: this.state.tasks.filter(task => task.id !== id)
    });
  };
  render() {
    return (
      <div>
        <TaskForm addTask={this.addTask} />
        <TaskList
          tasks={this.state.tasks}
          toggleTaskCompletion={this.toggleTaskCompletion}
          deleteTask={this.deleteTask}
        />
      </div>
    );
  }
}
export default TaskManager;
```

TaskForm.jsx

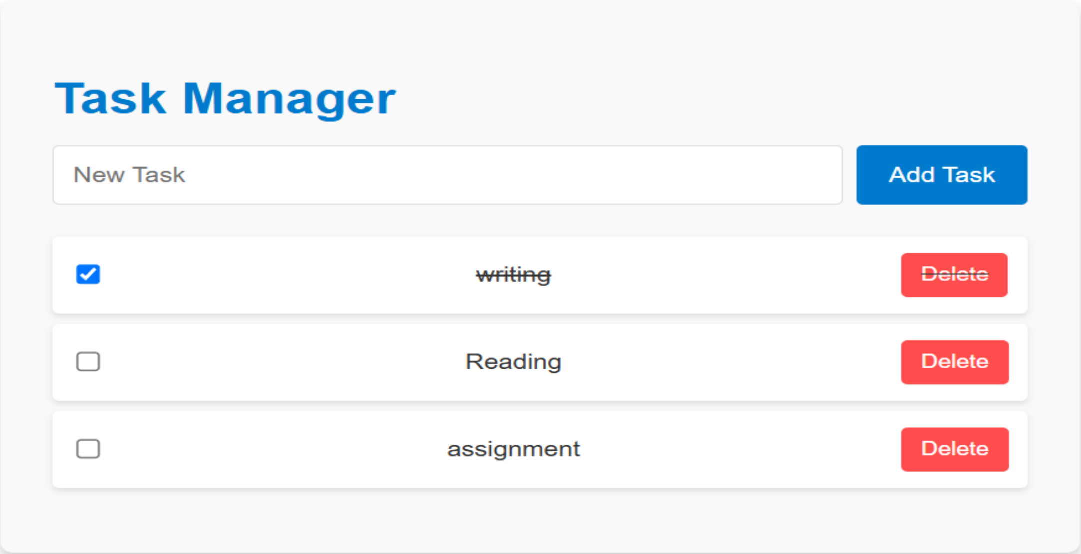
```
import React, { useState } from 'react';
function TaskForm({ addTask }) {
  const [title, setTitle] = useState('');
  const handleSubmit = (e) => {
    e.preventDefault();
    if (title.trim()) {
      addTask(title);
      setTitle('');
    }
  };
  return (
    <form onSubmit={handleSubmit}>
      <input
        type="text"
        placeholder="New Task"
        value={title}
        onChange={(e) => setTitle(e.target.value)}
      />
      <button type="submit">Add Task</button>
    </form>
  );
}
export default TaskForm;
```

TaskList.jsx

```
import React from 'react';
import TaskItem from './TaskItem';
function TaskList({ tasks, toggleTaskCompletion, deleteTask }) {
  if (tasks.length === 0) {
    return <p>No tasks to display.</p>;
  }
  return (
    <ul>
      {tasks.map(task => (
        <TaskItem
          key={task.id}
          task={task}
          toggleTaskCompletion={toggleTaskCompletion}
          deleteTask={deleteTask}
        />
      ))}
    </ul>
  );
}
export default TaskList;
```

TaskItem.jsx

```
import React from 'react';
function TaskItem({ task, toggleTaskCompletion, deleteTask }) {
  return (
    <li style={{ textDecoration: task.completed ? 'line-through' : 'none' }}>
      <input
        type="checkbox"
        checked={task.completed}
        onChange={() => toggleTaskCompletion(task.id)}
      />
      {task.title}
      <button onClick={() => deleteTask(task.id)}>Delete</button>
    </li>
  );
}export default TaskItem;
```


Task Manager
New Task
Add Task
☑ ~~writing~~ ~~Delete~~
□ Reading Delete
□ assignment Delete