React Basics Exercise: Build a Task Manager

Overview

In this hands-on exercise, you'll build a Task Manager application from scratch to practice:

- Component syntax (JSX)
- Props
- State management with useState
- Event handling
- Conditional rendering
- Lists and keys

Setup Instructions

1. Create a new Next.js project

`` `npx create-next-app@latest` ``

When prompted, use the default settings

2. Start the dev server

`` `npm run dev` ``

Open http://localhost:3000/ in your browser.

3. Clean up app/page.tsx

Open app/page.tsx and delete everything. Start with this minimal structure:

```
export default function Home() {
  return (
    <main className="min-h-screen p-8 bg-gray-50">
      <div className="max-w-2xl mx-auto">
        <h1 className="text-3xl font-bold mb-6">My Task Manager</h1>
      </div>
    </main>
  );
}
```

Task 1: Create a TaskItem Component

Create a new file app/task-item.tsx (right click the app folder → New File)

Your goal: Create a component that displays a task with a title and description.

Requirements:

- The component should be named TaskItem
- It should accept props: title and description
- Display the title in a bold/larger font
- Display the description in a smaller, gray font
- Wrap everything in a white box with some padding and shadow

Task 2: Use your TaskItem component

Go back to app/page.tsx and:

1. Import your TaskItem component at the top
2. Use it 2-3 times with different task data

Task 3: Adding State with useState

Update your task-item.tsx component to:

1. Track whether the task is completed (use useState)
2. Add a button to mark the task complete/incomplete
3. When completed, make the title crossed out and the whole card slightly faded

Requirements:

- Import useState from React: import { useState } from 'react';
- Create state: const [completed, setCompleted] = useState(false);
- Add a button with an onClick handler that toggles the completed state
- Apply conditional styling:
  - If completed: add line-through to title, opacity-50 to the card
  - Button text should change: "Mark Complete" vs "Undo"

Task 4: Move Tasks to State

Instead of hardcoding tasks, let's manage them with state in app/page.tsx.

Your goals:

1. Add 'use client'; at the very top of the file (before imports)
2. Import useState
3. Create a state variable tasks that holds an array of task objects
4. Each task should have: id, title, description
5. Use .map() to render a TaskItem for each task
6. Don't forget the key prop!