To-Do List Application Assignment

Objective

Build a **To-Do List web application** that allows users to manage their daily tasks efficiently. The application must support adding, updating, deleting tasks, tracking task completion progress, and persisting data using **Local Storage**.

UI Requirements

The application should include:

- A main heading titled **"My Task Manager"**
- A task input field
- An **"Add Task"** button
- A section to display task progress
- A task list displaying all added tasks

The user interface should be **modern, visually appealing, and responsive**, using high-quality styling (colors, spacing, typography, hover effects, and layout).

Functional Requirements

1. Add Task

- Users can enter a task description and add it to the list
- The input field should clear after adding a task
- Empty tasks should not be allowed

---

2. Display Tasks

Each task should display:

- Task text
- Current status (Pending / Completed)
- Action buttons:
    - Complete
    - Edit
    - Delete

---

3. Update Task

- Users must be able to edit an existing task
- Updated task text should reflect immediately in the UI
- Changes must persist after page refresh

---

4. Complete Task

- Users can mark tasks as completed or pending
- Completed tasks should be visually distinct (e.g., strike-through text, color change)
- Completion status should be toggleable

---

5. Delete Task

- Users can remove tasks from the list
- Deleted tasks must be permanently removed from Local Storage

---

Task Progress Tracking

Display the following information:

- Total number of tasks
- Number of completed tasks
- Task progress must update automatically when tasks are added, updated, completed, or deleted

---

Local Storage Requirements

- All tasks must be saved in Local Storage
- Tasks should persist after:
    - Page refresh
    - Browser restart
- Stored tasks must be automatically loaded and displayed when the application opens

---

Bonus Challenge: API Integration

Integrate the following public API into the application:

API Endpoint:
https://api.chucknorris.io/jokes/random

Bonus Requirements

- Fetch a random joke from the API when:
  - The application loads
  - The page is refreshed
- Display the joke text prominently **at the top of the application**
- Only the joke content (value field from the API response) should be shown
- Handle API errors gracefully (e.g., display a fallback message if the API is unavailable)

Design Expectations

- Clean and professional layout
- Well-styled buttons and inputs
- Proper spacing and alignment
- Smooth user interaction
- Fully responsive design

Evaluation Criteria

- Completion of all core features
- Proper use of Local Storage
- Accurate task progress tracking
- Successful API integration (bonus)
- Overall UI quality and usability
