DAY 4

Note: The names for my HTML, CSS, and JavaScript files (like DAYTHREE.html, DAYTHREE.css, DAYTHREE.js) are **not recommended** as they are considered **bad practice** in professional projects. However, since this is a **mini project**, I prefer using these names to **distinguish them from my other files**. For **larger or professional projects**, it's better to use standard and descriptive file names like index.html, style.css, and script.js.

DAYFOUR.HTML

- **<link rel="icon" href="...">**

Adds a **favicon** (little image on the tab).

- **<h1>DAY 4</h1>**

Big heading on top that says "DAY 4".

- ** ** **with** **<svg>**

- This creates a **curved text effect** (arc).
- Uses **SVG (Scalable Vector Graphics)** to draw the arc and text.
- MY TO DO LIST appears in a **smiling curve**.

- ** **

This is the **main box** where:

- Your **tasks** will appear.
- There's a text input to **add new tasks**.
- And buttons: **Add**, **Reset All**, **About**.

- **<ul id="taskList">**

This is an **empty list** now.
When you add tasks, each one appears here as an <li> (list item) with a checkbox.

- **<input type="text" id="taskInput" />**

The **textbox** where you type your task (like "Buy groceries").

- **Buttons:**

- **Add** → adds the task to the list.

- **Reset All** → removes all tasks.

DAYFOUR.CSS

```
body {
  margin: 0;
  padding: 0;
  font-family: 'Comic Sans MS', cursive, sans-serif;
  min-height: 100vh;
  background: linear-gradient(to bottom, #ffdce7, #f9e1f0, #fbefff);
}
```

- margin: 0; padding: 0; → Removes default spacing around the page.
- font-family → Uses a fun, playful font for text.
- min-height: 100vh; → Makes sure the body takes up **full screen height**.
- background → Adds a **soft pink gradient** from top to bottom.

```
h1 {
  text-decoration: underline solid white;
  text-align: center;
  color: white;
}
```

- Adds a **white underline** to your heading.
- Centers the text.
- Makes the text **white**.

```
#curve {
  fill: transparent;
}
```

- This is used inside the SVG curved text.
- It makes the **path** (the invisible arc line) **transparent**.

```
.text {
  font-size: 3rem;
  font-weight: bold;
  fill: #ff1493;
  filter: drop-shadow(5px 5px 0 white);
}
```

- font-size: 3rem; → Big size text.
- fill: #ff1493; → Pink color text (used in SVG).
- drop-shadow → Creates a **white shadow** behind the pink text, making it look **3D or glowing**.

```
.header {
  display: flex;
  justify-content: center;
  margin-top: 30px;
}
```

- Uses **flexbox** to center the curved heading.
- Adds some space from the top.

```
.input-block {
  max-width: 400px;
  margin: 30px auto;
  background-color: #ffffff;
  padding: 30px 20px;
  border-radius: 30px;
  box-shadow: 0px 4px 20px rgba(0,0,0,0.1);
  text-align: center;
}
```

This is the **white task box** in the center:

- max-width = limits width.
- margin: auto = centers it horizontally.
- border-radius: 30px = rounded corners.
- box-shadow = soft shadow to give a "floating card" feel.

```
.task-container {
  list-style: none;
  padding: 0;
  margin-bottom: 20px;
}
```

- Removes the default **bullet points**.
- Adds space below the task list.

```
.task-container li {
  background: #fff0f5;
  border-radius: 40px;
  margin: 10px 0;
  padding: 10px 20px;
  display: flex;
  align-items: center;
  font-size: 18px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.05);
}
```

Each task (li):

- Has a soft pink background.
- Is rounded and spaced out.
- Uses flex to place **checkbox** and **text side by side**.

Custom Checkbox Style

```
.task-container input[type="checkbox"] {
  appearance: none;
  width: 20px;
  height: 20px;
```

```
  border: 2px solid #ff1493;
  border-radius: 50%;
  margin-right: 15px;
  cursor: pointer;
  position: relative;
}
```

- Hides default checkbox style (appearance: none).
- Makes it **circular** (border-radius: 50%).
- Pink border.
- Adds space between checkbox and task text.

When Checked:

```
.task-container input[type="checkbox"]:checked::after {
  content: '✓';
  position: absolute;
  top: -2px;
  left: 4px;
  color: #ff1493;
  font-weight: bold;
  font-size: 16px;
}
```

- Shows a pink **tick mark (✓)** when checked.

```
.input-block input {
  padding: 10px;
  border-radius: 20px;
  border: 2px solid #ff1493;
  outline: none;
  width: 80%;
  margin-bottom: 15px;
  font-size: 16px;
}
```

- Styles the **task input box** (rounded border, pink outline, large text).

```
.button-group button {
  margin: 5px;
  padding: 10px 15px;
  border: none;
  border-radius: 20px;
  background-color: #ff1493;
  color: white;
  cursor: pointer;
  font-weight: bold;
}
```

Each button:

- Is pink with white text.
- Rounded like a pill.

- No border.
- Pointer cursor on hover.

On Hover:

```
.button-group button:hover {
  background-color: #e0118b;
}
```

- Slightly darker pink when you hover the mouse.

DAYFOUR.JS

1. Get the Input and Task List Elements

```
const input = document.getElementById('taskInput');
const taskList = document.getElementById('taskList');
```

- Grabs the text input box (where you type tasks) using its ID: taskInput
- Grabs the list where tasks will appear: taskList

2. Function to Add a Task

```
function addTask() {
  const taskText = input.value.trim();  // Get and clean the input text
  if (taskText === '') return;         // If it's empty, stop here

  const li = document.createElement('li');      // Create a new list item
  li.innerHTML = `<input type="checkbox" /> <span>${taskText}</span>`;  // Add checkbox + task
  taskList.appendChild(li);            // Add it to the list
  input.value = '';                    // Clear the input box
}
```

What it does:

- Takes the text you typed.
- If it's **not empty**, it:
  - Creates a new list item (<li>).
  - Adds a **circular checkbox** and your task text.
  - Adds this <li> to the main task list.
  - Clears the input box for the next task.

3. Function to Clear All Tasks

```
function resetTasks() {
  taskList.innerHTML = '';   // Clears all tasks
  input.value = '';          // Clears the input box too
}
```

What it does:

- Wipes out everything inside the task list (<ul>).
- Empties the input box.

4. Allow “Enter” Key to Add a Task

```
input.addEventListener('keydown', function (e) {
  if (e.key === 'Enter') {
    addTask();
  }
});
```

What it does:

- Listens for any key press **while typing in the input box**.
- If you press the **Enter key**, it automatically calls the addTask() function — just like clicking the Add button.