**TITLE:** CodTech IT Solutions Internship - Task Documentation: “To-Do List” Using CSS, HTML, JAVASCRIPT.

**INTERN INFORMATION:**

**Name:** Nandhana Suresh Poovathingal

**ID:** ICOD6802

**INTRODUCTION**

In a world filled with endless distractions and competing priorities, staying focused and productive can be a challenge. Traditional methods of task management, such as pen and paper or basic digital notes, are often cumbersome and lack the features necessary to keep up with our busy lives.

To-Do-List fills this gap by offering a modern, user-friendly platform that empowers users to:

* Stay Organized: Keep track of all your tasks in one central location, accessible anytime, anywhere.
* Prioritize Effectively: Easily identify your most important tasks and allocate your time and energy accordingly.
* Stay Motivated: It helps you to stay motivated my accomplishing your daily tasks.

**IMPLEMENTATION**

HTML: Used for structuring the elements of the web page, including task input, task list, and task items.

CSS: Employed for styling the elements and enhancing the visual appeal of the To-Do List interface.

JavaScript: Implemented for adding dynamic functionality to the To-Do List, including task addition, task completion, and task deletion.

**CODE EXPLANATION**

1. **HTML**

<div class="container">

This acts as the main container. The heading, input box, and lists of tasks are wrapped in this div.

<h1 class="heading">

This is the heading of the container. A image is wrapped inside this <h1> tag.

<div class="todo-input">

The input box and the ‘Add Tasks’ button is enclosed within this div.

<ul id="taskList">

The to-do tasks are dynamically added using this id.

1. **CSS**

* Global settings like margin, padding is added.
* The background is gradient od 2 colours.
* The main content is set to a width occupying 70% of the webpage.
* Border-radius is applied to main container, input box and buttons for rounded borders.
* The heading is aligned at the centre.

1. **JAVASCRIPT**

Javascript is used to add tasks dynamically to the list, a checkbox and delete button is provided to each list item.

Following are the JS functions used:

1. **addTask():**

* This function is called when the user wants to add a new task.
* It retrieves the task input from the HTML element with the ID "taskInput".
* Checks if the input is not empty.
* Creates a new list item (‘li’) with a checkbox, task text, and delete button.
* Appends the new list item to the task list (taskList).
* Clears the task input field.
* Calls saveData() to save the updated task list to local storage.

1. **toggleTask(event):**

* This function is called when the checkbox associated with a task is toggled.
* It toggles the "checked" class on the parent list item to visually indicate task completion.
* Calls saveData() to save the updated task list to local storage.

1. **deleteTask(event):**

* This function is called when the delete button associated with a task is clicked.
* It prompts the user to confirm the deletion.
* If confirmed, it removes the parent list item from the task list.
* Calls saveData() to save the updated task list to local storage.

1. **saveData(taskList):**

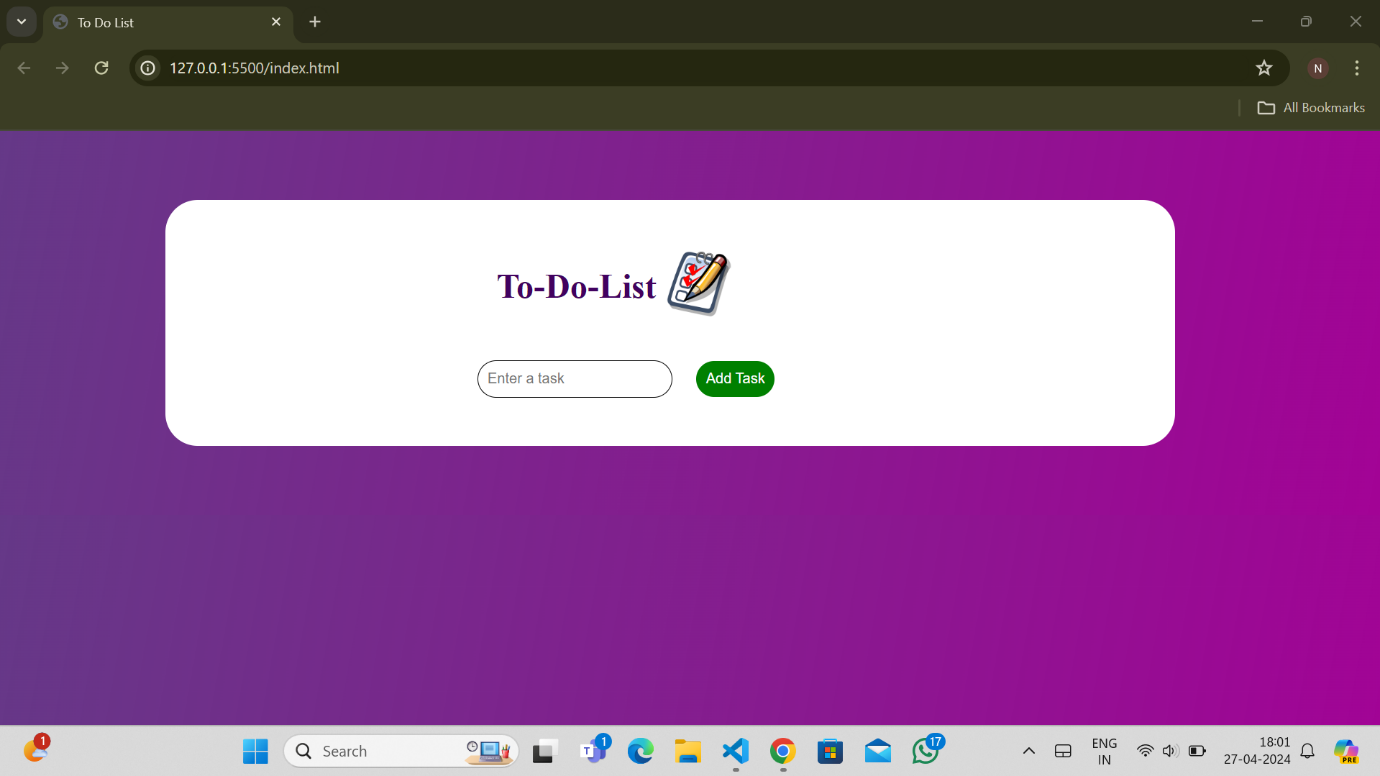
* This function is responsible for saving the task list HTML to local storage.
* It uses localStorage.setItem() to store the inner HTML of the task list.
* The task list HTML is stored under the key "data" in local storage.

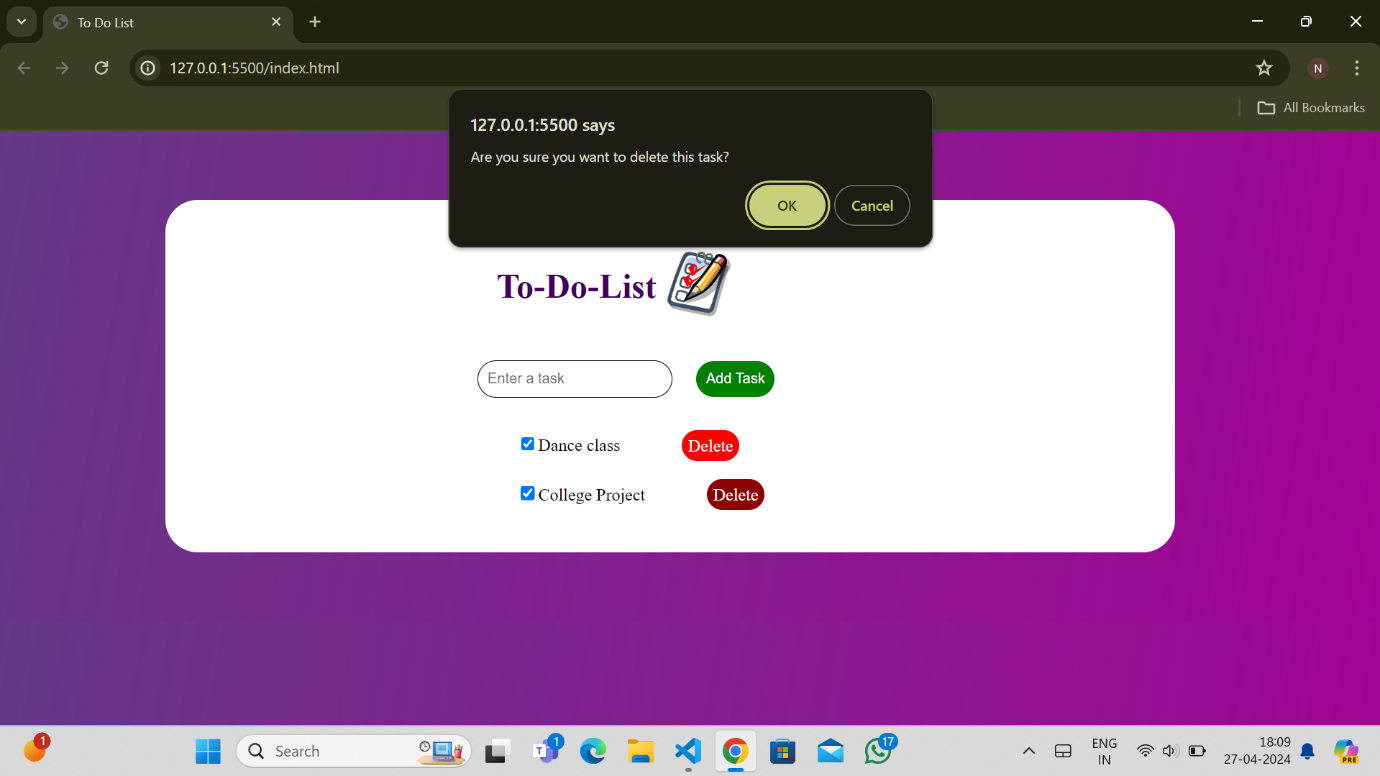
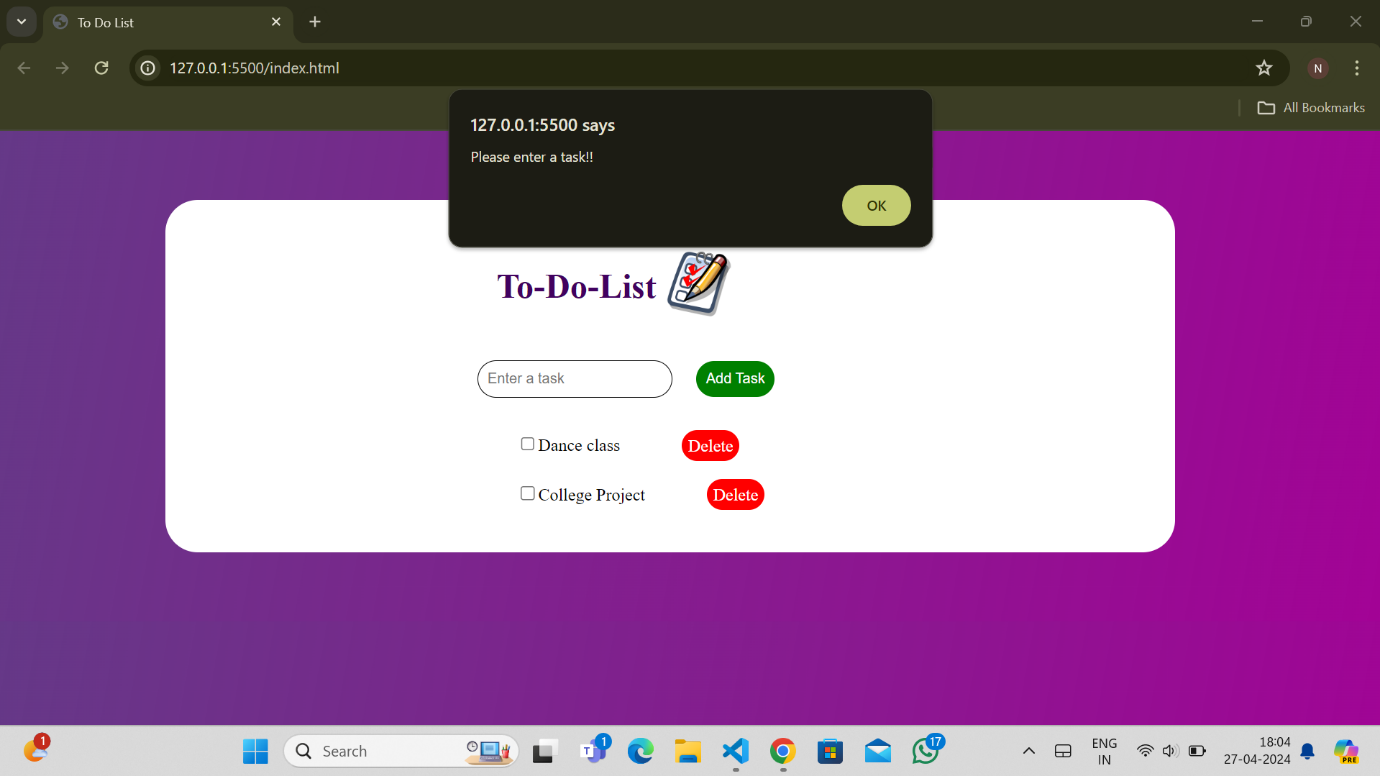
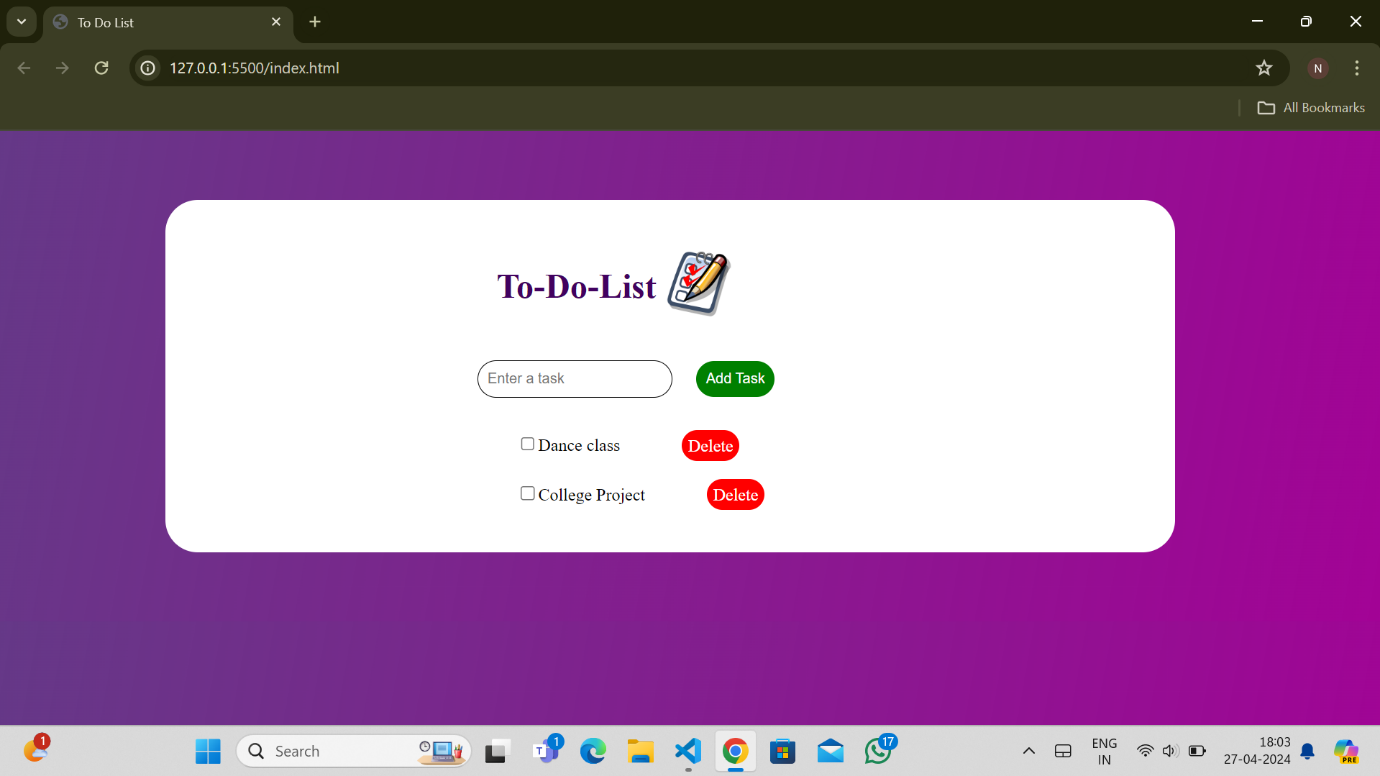
**CONCLUSION**

In conclusion, the provided code constitutes a simple to-do list project implemented using HTML, CSS, and JavaScript. The project allows users to add tasks, mark them as completed by toggling checkboxes, and delete tasks.

Key functionalities include:

* Adding Tasks: Users can input tasks using an input field and add them to the list by clicking a button. Empty task entries are not accepted.
* Completing Tasks: Tasks can be marked as completed by toggling checkboxes associated with each task. Completed tasks are visually differentiated from incomplete ones.
* Deleting Tasks: Users can delete tasks individually by clicking a delete button associated with each task. A confirmation dialog ensures deletion is intentional.
* Data Persistence: The task list is saved to the browser's local storage, allowing users to revisit the page and retain their task list.

**OUTPUT**

****