**Project Report: Todo List Web App**

**1. Introduction**

The Todo List Web App is designed to help users manage their daily tasks efficiently. It allows users to add, edit, mark as completed, and delete tasks. The application stores tasks in the local storage, ensuring data persistence across sessions. This project utilizes HTML, CSS, and JavaScript to provide a simple yet effective user interface.

**2. Project Objective**

The primary objective of this project is to develop a web application that helps users manage their tasks efficiently. The application should be user-friendly, visually appealing, and should ensure data persistence using local storage.

**3. Technologies Used**

* **HTML**: To structure the content of the web app.
* **CSS**: For styling the web app to make it visually appealing.
* **JavaScript**: To add interactivity and handle the logic for managing tasks.

**4. Features**

**4.1 Input Section**

* **Task Input**: Users can enter a new task.
* **Add Button**: Allows users to add the task to the list.

**4.2 Task List**

* **Task Display**: Displays a list of all recorded tasks.
* **Edit Functionality**: Users can edit existing tasks.
* **Toggle Functionality**: Users can mark tasks as completed or uncompleted.
* **Delete Functionality**: Users can delete individual tasks or all tasks at once.
* **Task Count**: Displays the total number of tasks.

**5. Implementation Details**

**5.1 HTML Structure**

The HTML structure consists of:

* An input section for adding tasks.
* A list to display tasks.
* Buttons for adding and deleting tasks.
* A counter to display the total number of tasks.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<link rel="stylesheet" href="./style.css" />

<script src="./script.js" defer></script>

<title>Todo</title>

</head>

<body>

<section class="todo">

<h2>To-do list</h2>

<div class="input">

<input

type="text"

class="input-field"

id="todoInput"

placeholder="Add a new todo"

/>

<button class="btn">Add</button>

</div>

<ul class="scroll">

<li id="todoList"></li>

</ul>

<div>

<hr class="counter" />

<div class="counter-container">

<p><span id="todoCount">0</span> items total</p>

<button id="deleteButton">Delete All</button>

</div>

</div>

</section>

<footer>

<div class="footer">

<p class="made-by">Made By</p>

<p class="author">John Doe</p>

</div>

</footer>

</body>

</html>

**5.2 CSS Styling**

The CSS provides styling for the web app, making it visually appealing and user-friendly.

:root {

--gradient: linear-gradient(

180deg,

rgba(45, 112, 1253, 0.73) 0%,

#163e92 100%

);

--dark: #001747;

--grey: #b1bacb;

--grey-border: rgba(210, 210, 210, 0.75);

--grey-light: #eeeeee;

--grey-dark: #405175;

--blue: #2d70fd;

--green: #00d8a7;

--white: #ffffff;

}

body {

background: var(--gradient);

margin: 0;

height: 100vh;

display: flex;

flex-direction: column;

justify-content: space-evenly;

align-items: center;

}

.btn {

color: var(--white);

font-size: 1.1rem;

padding: 0.7rem 1.5rem;

border-radius: 0.3rem;

background-color: var(--blue);

border: none;

position: absolute;

right: 0.5rem;

bottom: 0.5rem;

}

h1,

h2,

h3,

h4,

h5,

h6,

p {

margin: 0;

}

/\* Todo Container \*/

.todo {

display: flex;

flex-direction: column;

justify-content: space-around;

border-radius: 2rem;

background: var(--white);

padding: 3rem;

height: 50%;

width: 60%;

box-shadow: 0 1rem 3rem 1rem rgba(0, 23, 71, 0.15);

max-width: 30rem;

}

h2 {

text-transform: uppercase;

height: 3rem;

color: var(--dark);

text-align: center;

}

.input {

position: relative;

display: flex;

}

.input-field {

width: 100%;

border: 0.06rem solid #d2d2d2bf;

border-radius: 0.5rem;

padding: 1.25rem;

font-size: 1rem;

}

input[type="text"]::placeholder {

color: var(--grey);

}

.todo-container {

display: flex;

gap: 1rem;

}

ul {

padding: 0;

margin: 0;

overflow-y: scroll;

}

li {

display: flex;

flex-direction: column;

gap: 1.2rem;

padding: 1.3rem;

}

#todoList p {

display: flex;

gap: 1rem;

color: var(--dark);

align-items: center;

}

#todoList .disabled {

color: #8f98a8;

}

.disabled {

display: flex;

text-decoration: line-through;

}

input[type="checkbox"] {

appearance: none;

-webkit-appearance: none;

-moz-appearance: none;

cursor: pointer;

}

input[type="checkbox"]::before {

content: "\2713";

display: inline-block;

width: 2rem;

height: 2rem;

font-size: 1.7rem;

text-align: center;

border: 0.06rem solid var(--grey-border);

border-radius: 50%;

color: transparent;

}

input[type="checkbox"]:checked::before {

color: var(--white);

background-color: var(--green);

border: 0.06rem solid var(--green);

border-radius: 50%;

}

.counter {

border: 0.06rem solid var(--grey-light);

}

.counter-container {

height: 2rem;

display: flex;

justify-content: space-between;

color: var(--grey);

}

.counter-container p {

align-self: center;

}

.counter-container button {

border: none;

background-color: transparent;

color: var(--grey);

font-size: 1rem;

}

.footer {

display: flex;

gap: 1.8rem;

background-color: var(--white);

padding: 1.2rem;

border-radius: 0.5rem;

}

.made-by,

.author {

font-size: 0.9rem;

}

.made-by {

color: var(--grey-dark);

}

.author {

color: var(--blue);

font-weight: bold;

}

.scroll {

height: 15rem;

scrollbar-width: thin;

}

.scroll::-webkit-scrollbar {

width: 0.6rem;

}

.scroll::-webkit-scrollbar-thumb {

background-color: var(--blue);

border-radius: 0.5rem;

}

.scroll::-webkit-scrollbar-track {

display: none;

}

**5.3 JavaScript Functionality**

The JavaScript adds interactivity and handles the logic for managing tasks.

// Retrieve todo from local storage or initialize an empty array

let todo = JSON.parse(localStorage.getItem("todo")) || [];

const todoInput = document.getElementById("todoInput");

const todoList = document.getElementById("todoList");

const todoCount = document.getElementById("todoCount");

const addButton = document.querySelector(".btn");

const deleteButton = document.getElementById("deleteButton");

// Initialize

document.addEventListener("DOMContentLoaded", function () {

addButton.addEventListener("click", addTask);

todoInput.addEventListener("keydown", function (event) {

if (event.key === "Enter") {

event.preventDefault(); // Prevents default Enter key behavior

addTask();

}

});

deleteButton.addEventListener("click", deleteAllTasks);

displayTasks();

});

function addTask() {

const newTask = todoInput.value.trim();

if (newTask !== "") {

todo.push({ text: newTask, disabled: false });

saveToLocalStorage();

todoInput.value = "";

displayTasks();

}

}

function displayTasks() {

todoList.innerHTML = "";

todo.forEach((item, index) => {

const p = document.createElement("p");

p.innerHTML = `

<div class="todo-container">

<input type="checkbox" class="todo-checkbox" id="input-${index}" ${

item.disabled ? "checked" : ""

}>

<p id="todo-${index}" class="${

item.disabled ? "disabled" : ""

}" onclick="editTask(${index})">${item.text}</p>

</div>

`;

p.querySelector(".todo-checkbox").addEventListener("change", () =>

toggleTask(index)

);

todoList.appendChild(p);

});

todoCount.textContent = todo.length;

}

function editTask(index) {

const todoItem = document.getElementById(`todo-${index}`);

const existingText = todo[index].text;

const inputElement = document.createElement("input");

inputElement.value = existingText;

todoItem.replaceWith(inputElement);

inputElement.focus();

inputElement.addEventListener("blur", function () {

const updatedText = inputElement.value.trim();

if (updatedText) {

todo[index].text = updatedText;

saveToLocalStorage();

}

displayTasks();

});

}

function toggleTask(index) {

todo[index].disabled = !todo[index].disabled;

saveToLocalStorage();

displayTasks();

}

function deleteAllTasks() {

todo = [];

saveToLocalStorage();

displayTasks();

}

function saveToLocalStorage() {

localStorage.setItem("todo", JSON.stringify(todo));

}

**6. Conclusion**

The Todo List Web App is a simple and efficient tool for managing tasks. By utilizing HTML, CSS, and JavaScript, the application provides a user-friendly interface with essential functionalities like adding, editing, toggling, and deleting tasks. The use of local storage ensures that user data is preserved across sessions, enhancing the overall user experience.