**Praktikum JavaScript - Aplikasi To-Do List**

****

**Nama : Andi Ode Larios**

**NIM : 4523210015**

**Mata Kuliah : Praktikum Desain Web**

**Tanggal Praktikum : 1 Oktober 2024**

**Program Studi Teknik Informatika**

**Fakultas Teknik**

**Universitas Pancasila**

**2024**

**PENDAHULUAN**

Pada praktikum ini adalah untuk mengembangkan aplikasi web menggunakan bahasa pemrograman JavaScript, dalam laporan ini saya akan membuat aplikasi To-Do List sederhana yang akan membantu pengguna mengatur dan mengelola tugas sehari hari. Aplikasi sederhana ini juga berfungsi sebagai sarana untuk menerapkan berbagai konsep dasar pemrograman. Aplikasi sederhana ini memungkinkan pengguna untuk menambah,mengedit,dan menghapus tugas. Teori yang dipakai untuk membuat aplikasi sederhana ini adalah yang pertama memanipulasi DOM yaitu bagaimana cara menggunakan JavaScript untuk berinteraksi dengan elemen HTML yang kedua adalah Class dan Object yaitu agar saya dapat menyusun kode dan mempermudah saya dalam memahami dan mengembangkan aplikasi sederhana ini. Dengan mempelajari beberapa konsep sederhana ini diharapkan pembaca dapat mendapatkan keterampilan baru sehingga mampu memahami bagaimana membangun aplikasi sederhana seperti To-Do List.

Langkah pengerjaan

1. Membuat file index.html

saya membuat halaman index.html yang berisikan struktur dasar aplikasi To-Do List. saya menggunakan <style> dalam html, lalu beberapa elemen yang saya gunakan adalah input field untuk memasukkan tugas baru, tombol Add Task untuk menambahkan tugas ke dalam daftar, dan daftar tugas yang ditampilkan ke dalam elemen <ul>

| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>To-Do List App</title>  <style>  body {  font-family: Arial, sans-serif;  margin: 0 auto;  padding: 20px;  max-width: 600px;  }  h1 {  text-align: center;  }  #task-input {  width: 80%;  padding: 10px;  font-size: 16px;  }  #add-task-btn {  padding: 10px;  font-size: 16px;  }  ul {  list-style-type: none;  padding: 0;  }  li {  display: flex;  justify-content: space-between;  padding: 10px;  border-bottom: 1px solid #ccc;  }  li.completed {  text-decoration: line-through;  }  button {  margin-left: 10px;  padding: 5px;  }  </style>  </head>  <body>  <h1>To-Do List</h1>  <input type="text" id="task-input" placeholder="Enter a new task">  <button id="add-task-btn">Add Task</button>  <h3>Filter Tasks</h3>  <button onclick="filterTasks('all')">All</button>  <button onclick="filterTasks('active')">Active</button>  <button onclick="filterTasks('completed')">Completed</button>  <ul id="task-list"></ul>  <script src="app.js"></script>  </body>  </html> |
| --- |

1. Membuat Class dan Object Class

membuat kelas task di file app.js untuk mempresentasikan setiap tugas yang ditambahkan. Kelas ini memiliki dua atribut yaitu name dan completed.

| class Task {  constructor(name) {  this.name = name;  this.completed = false;  }  toggleCompleted() {  this.completed = !this.completed;  }  } |
| --- |

1. Menambahkan Tugas

membuat fungsi addTask() digunakan untuk menambahkan tugas baru ke dalam daftar.

| let tasks = [];  function addTask() {  const taskInput = document.getElementById('task-input');  const taskName = taskInput.value.trim();  if (taskName === "") {  alert("Task cannot be empty!");  return;  }  const task = new Task(taskName);  tasks.push(task);  displayTasks();  taskInput.value = ''; *// Kosongkan input setelah menambahkan*  }  document.getElementById('add-task-btn').addEventListener('click', addTask); |
| --- |

1. Menambahkan Tugas di DOM

fungsi displayTask() memiliki fungsi menampilkan setiap tugas yang ada di array task pada elemen <ul>. Setiap tugas ditampilkan dengan checkbox dan tombol “Delete”

| function displayTasks() {  const taskList = document.getElementById('task-list');  taskList.innerHTML = '';  tasks.forEach((task, index) => {  const taskItem = document.createElement('li');  taskItem.className = task.completed ? 'completed' : '';  const taskText = document.createElement('span');  taskText.textContent = task.name;  const taskCheckbox = document.createElement('input');  taskCheckbox.type = 'checkbox';  taskCheckbox.checked = task.completed;  taskCheckbox.addEventListener('click', () => toggleTask(index));  const deleteButton = document.createElement('button');  deleteButton.textContent = 'Delete';  deleteButton.addEventListener('click', () => deleteTask(index));  taskItem.appendChild(taskCheckbox);  taskItem.appendChild(taskText);  taskItem.appendChild(deleteButton);  taskList.appendChild(taskItem);  });  } |
| --- |

1. Menambahkan fungsi toggleTask() untuk menandai tugas sebagai selesai dan deleteTask() untuk menghapus tugas dari daftar

| function toggleTask(index) {  tasks[index].toggleCompleted();  displayTasks();  }  function deleteTask(index) {  tasks.splice(index, 1);  displayTasks();  } |
| --- |

**Struktur Penjelasan**

1. Class Task : mendefinisikan setiap tugas dengan properti name dan completed. Method toggleComplated() mengubah status tugas dari selesai menjadi belum selesai atau sebaliknya

name : nama tugas yang akan diisi oleh pengguna

completed : menyimpan status apakah tugas sudah selesai atau belum

1. Fungsi addTask()

Menambahkan tugas baru ke daftar setelah memvalidasi input

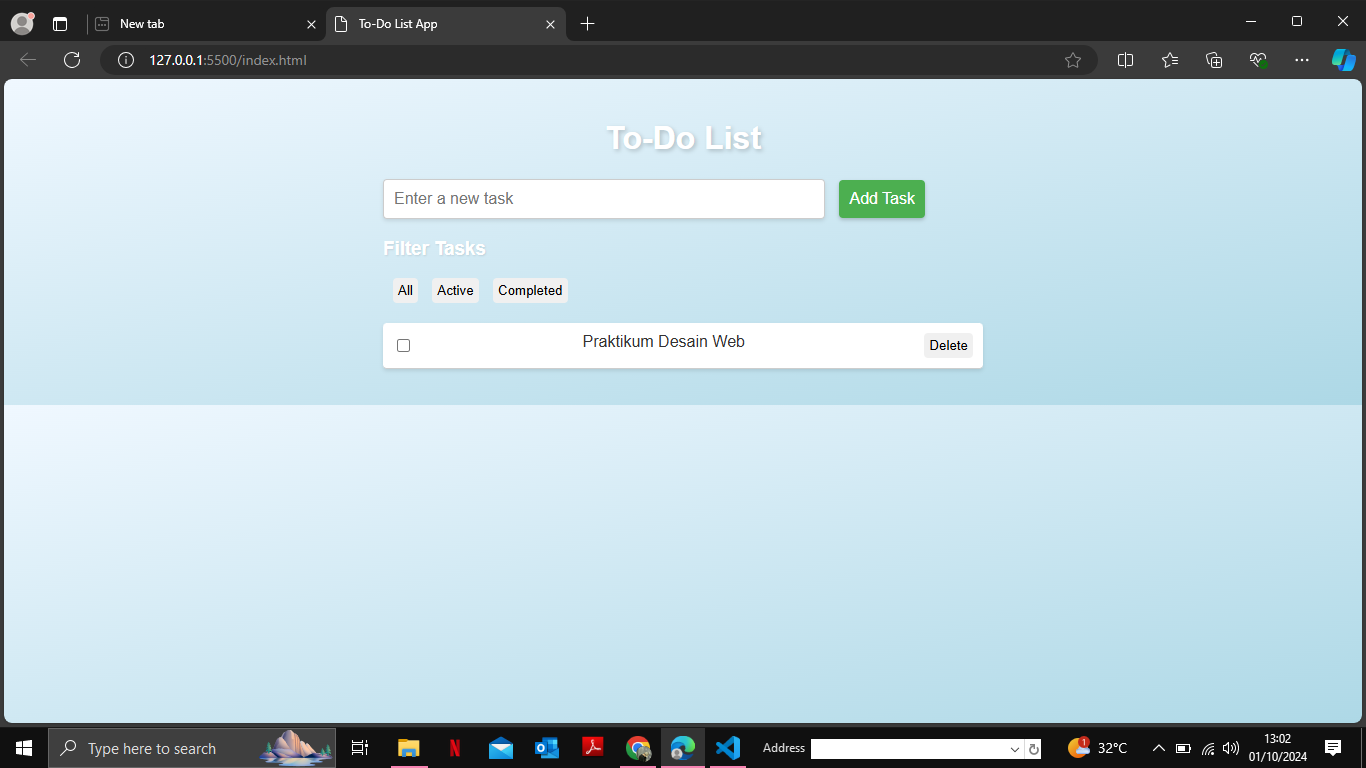
1. Fungsi displayTask()

Manipulasi DOM dengan menampilkan setiap tugas pada elemen <ul> sebagai item daftar

1. Event Handling

Menggunakan tombol Add Task checkbox untuk menandai tugas selesai dan tmbol delete untuk menghapus tugas.

**Hasil Uji Coba**



**Penjelasan :**

Hasil uji coba dari fitur To-Do List yang telah diimplementasikan menunjukkan bahwa setiap komponen berfungsi sesuai dengan fungsinya. Pada bagian input, pengguna dapat menambahkan tugas baru dengan memasukkan teks dan menekan tombol "Add Task". Setelah tugas ditambahkan, tugas tersebut muncul di bawahnya dengan opsi untuk menghapus atau menandai sebagai tugas aktif atau selesai. Fitur filter yang tersedia yaitu "All", "Active", dan "Completed" juga bekerja dengan baik, di mana tombol-tombol tersebut memfilter tugas berdasarkan statusnya. Selain itu, fungsi delete berhasil menghapus tugas yang dipilih dari daftar.

**Kesimpulan :**

Secara keseluruhan, aplikasi To-Do List telah berhasil diuji dengan baik. Setiap komponen utama, mulai dari penambahan, penghapusan, hingga pengelompokan tugas, berfungsi sesuai harapan. Kesimpulannya, aplikasi ini siap digunakan untuk mengelola daftar tugas pengguna dengan cara yang sederhana namun efektif. Implementasi fungsi filter dan manajemen tugas memberikan pengalaman pengguna yang fleksibel dan mudah dimengerti.

Link github :