

# **LAPORAN PRAKTIKUM**

## **“Web Storage ”**

Dalam rangka memenuhi tugas mata kuliah Praktikum Desain Website

Dosen pengampu: Vearen Dika Sofirudin, S.Pd., M.Ed.



Disusun oleh:

Nama : Khoirul Bagus Wicaksono

NIM : K3524077

**PENDIDIKAN TEKNIK INFORMATIKA DAN KOMPUTER**

**FAKULTAS KEGURUAN DAN ILMU PENDIDIKAN**

**UNIVERSITAS SEBELAS MARET**

**SURAKARTA**

**2025**

## 1. LATIHAN 1

### a. Kode Program:

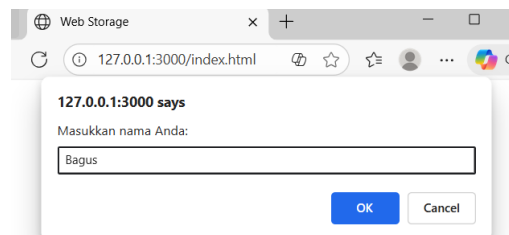
```
index.html
D:\Web Design SMT 3\November 2025\Web Storage\Latihan 1\index.html
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>Web Storage</title>
7 </head>
8 <body>
9   <center>
10    <h1>Selamat Datang, Member!</h1>
11    <button onclick="gantiNama(event)">Masukkan Nama</button>
12  </center>
13
14  <script type="text/javascript">
15    const namaMember = document.querySelector('h1');
16
17    // Mengecek apakah sudah ada nama tersimpan di localStorage
18    const namaTersimpan = localStorage.getItem('nama');
19    if (namaTersimpan) {
20      namaMember.innerHTML = 'Halo ' + namaTersimpan + '!';
21    }
22
23    function gantiNama(e) {
24      let nama = prompt('Masukkan nama Anda:');
25      if (nama && nama.trim() !== '') {
26        localStorage.setItem('nama', nama);
27        namaMember.innerHTML = 'Halo ' + nama + '!';
28      } else {
29        alert('Nama tidak boleh kosong!');
30      }
31      e.preventDefault();
32    }
33  </script>
34
35 </body>
36 </html>
```

### b. Jalankan, lalu masukkan isian sesuai perintah. Apa yang terjadi?

← → ↻ http://127.0.0.1:3000/index.html?serverWindowId=4c98fd9e-5929-4109-91d7-ab61b3773144

Selamat Datang, Member!

Masukkan Nama



Halo Bagus!

Masukkan Nama

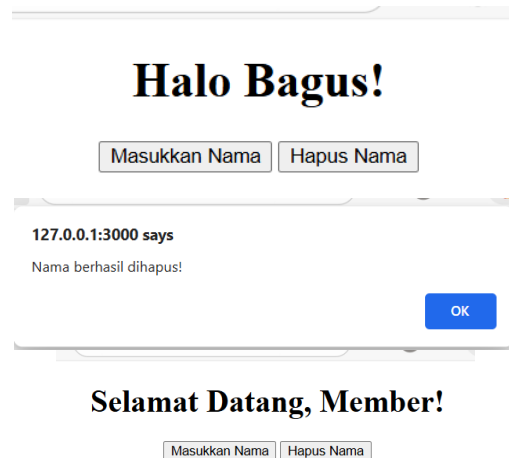
Penjelasan : Ketika kode tersebut dijalankan di browser dan pengguna menekan tombol “Masukkan Nama”, sebuah jendela prompt akan muncul untuk meminta pengguna mengisi nama. Jika pengguna memasukkan nama yang valid, nama tersebut langsung disimpan ke dalam localStorage dan teks pada elemen <h1> berubah menjadi “Halo !”. Perubahan ini bersifat permanen selama data localStorage tidak dihapus, sehingga ketika halaman direfresh atau dibuka kembali, browser otomatis menampilkan sapaan dengan nama yang sudah tersimpan tanpa perlu mengklik tombol lagi. Namun, jika pengguna menekan OK tanpa mengisi nama atau hanya mengetik spasi, maka sistem akan menolak input tersebut dan menampilkan peringatan “Nama tidak boleh kosong!”, sementara teks pada halaman tidak berubah. Dengan demikian, kode ini memberikan pengalaman personalisasi yang sederhana dengan memanfaatkan penyimpanan lokal di browser.

### C. Tambahkan sebuah tombol yang dapat menghapus nama dari member!

Kode program:

```
index.html > ...
1 </DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>Web Storage</title>
7 </head>
8 <body>
9   <center>
10    <h1 id="judul">Selamat Datang, Member!</h1>
11    <button onclick="gantiNama(event)">Masukkan Nama</button>
12    <button onclick="hapusNama(event)">Hapus Nama</button>
13  </center>
14
15  <script src="script.js"></script>
16
17 </body>
18 </html>
19
```

Hasil :



Penjelasan : Setelah kode diubah, kini terdapat sebuah tombol tambahan bernama “**Hapus Nama**” yang berfungsi untuk menghilangkan nama yang sebelumnya disimpan di localStorage. Ketika tombol ini ditekan, fungsi hapusNama() akan dijalankan untuk menghapus data dengan key 'nama' dari localStorage menggunakan localStorage.removeItem('nama'). Setelah data

dihapus, teks pada elemen `<h1>` dikembalikan ke tampilan awal yaitu “Selamat Datang, Member!”, dan sebuah notifikasi akan muncul untuk memberi tahu bahwa nama telah berhasil dihapus. Dengan penambahan tombol ini, pengguna bisa mengganti atau menghapus identitasnya kapan saja tanpa harus menghapus data browser secara manual.

#### D. Modifikasi javascript sehingga data nama member tetap akan ada selama belum dihapus!

Kode HTML :

```
index.html > ...
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>Web Storage</title>
7 </head>
8 <body>
9   <center>
10    <h1 id="judul">Selamat Datang, Member!</h1>
11    <button onclick="gantiNama(event)">Masukkan Nama</button>
12    <button onclick="hapusNama(event)">Hapus Nama</button>
13  </center>
14
15  <script src="script.js"></script>
16
17 </body>
18 </html>
19 |
```

Kode Javascript:

```
JS script.js > ...
1 const namaMember = document.getElementById('judul');
2
3 const namaTersimpan = localStorage.getItem('nama');
4 if (namaTersimpan) {
5   namaMember.innerHTML = 'Halo ' + namaTersimpan + '!';
6 }
7
8 function gantiNama(e) {
9   let nama = prompt('Masukkan nama Anda:');
10  if (nama && nama.trim() !== '') {
11    localStorage.setItem('nama', nama);
12    namaMember.innerHTML = 'Halo ' + nama + '!';
13  } else {
14    alert('Nama tidak boleh kosong!');
15  }
16
17  e.preventDefault();
18 }
19
20 function hapusNama(e) {
21   localStorage.removeItem('nama');
22   namaMember.innerHTML = 'Selamat Datang, Member!';
23   alert('Nama berhasil dihapus!');
24
25   e.preventDefault();
26 }
27
```

Penjelasan : Modifikasi ini memisahkan kode JavaScript dari HTML sehingga lebih rapi dan sesuai standar pengembangan web. Nama member tetap tersimpan karena data disimpan dengan `localStorage`, yang tidak akan hilang meskipun halaman direfresh atau browser ditutup, selama tidak dihapus dengan tombol "Hapus Nama" atau secara manual dari browser.

**a. skrip :**

**a. skrip :**

[illegible]

**b. Jalankan, maka akan muncul tampilan seperti berikut**



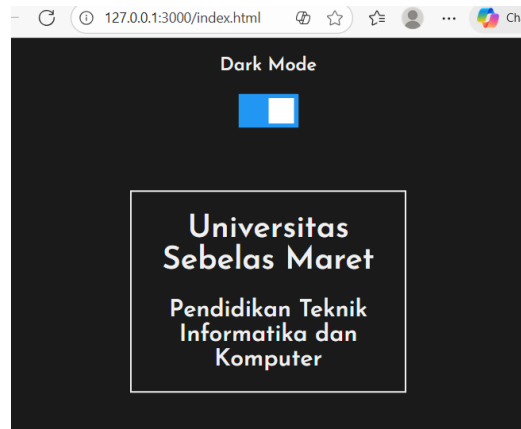
Universitas Sebelas Maret  
Pendidikan Teknik Informatika dan Komputer

**C. Modifikasi file tersebut dengan menambahkan css dan javascript yang tepat dengan memanfaatkan local storage sehingga fungsi dark mode dapat berjalan ketika toggle dark mode dijalankan.**

Script :



Hasil :



Penjelasan :

Berikut adalah penjelasan singkat mengenai cara kerja ketiga file tersebut secara bersama-sama untuk membuat fitur **Dark Mode** yang tersimpan (tidak hilang saat di-refresh):

### 1. HTML (index.html) sebagai Kerangka

File ini adalah struktur utamanya.

- **Penghubung:** Menggunakan `<link>` untuk memanggil file CSS dan `<script>` untuk memanggil file JS.
- **Pemicu (Trigger):** Pada elemen `<input type="checkbox">`, terdapat atribut `onchange="ubahTema(event)"`. Artinya, setiap kali tombol digeser, ia akan langsung memanggil fungsi JavaScript bernama `ubahTema`.
- **Identitas:** Kita memberi `id="darkModeToggle"` agar JavaScript bisa menemukan tombol tersebut saat halaman baru dimuat (untuk mengatur posisi on/off).

### 2. CSS (style.css) sebagai Tampilan

File ini mengatur "wajah" dari website.

- **Kelas Spesial:** Kita membuat aturan khusus bernama `.dark-mode`.
  - Secara default, website menggunakan warna terang.
  - Namun, jika tag `<body>` memiliki kelas `.dark-mode` (contoh: `<body class="dark-mode">`), maka aturan CSS di blok `body.dark-mode` akan aktif (latar jadi hitam, tulisan jadi putih).
- **Transisi:** `transition: 0.3s` ditambahkan agar perubahan warna tidak kaget, melainkan berubah perlahan (halus).

### 3. JavaScript (script.js) sebagai Otak & Memori

Di sinilah logika **Web Storage** bekerja. Ada dua tugas utama:

- **Tugas 1: Mengingat Pilihan (Saat halaman dimuat)**
  - Kode `window.onload` berjalan otomatis saat Anda membuka web.

- Ia bertanya pada browser: *"Halo Local Storage, apakah ada simpanan tema bernama 'theme'?"*
- Jika jawabannya **'dark'**, maka JS akan otomatis menambahkan kelas .dark-mode ke HTML dan mencentang tombol switch agar sesuai.
- **Tugas 2: Menyimpan Perubahan (Saat tombol diklik)**
  - Fungsi ubahTema berjalan saat Anda klik tombol.
  - **Jika dicentang (ON):** Tambahkan kelas .dark-mode ke body **DAN** simpan kata 'dark' ke dalam memori browser (localStorage.setItem).
  - **Jika tidak dicentang (OFF):** Hapus kelas .dark-mode **DAN** simpan kata 'light' ke memori browser.