

PRAKTIKUM JAVASCRIPT & MANIPULASI DOM

Minggu ke-6

Mata Kuliah:

Pemrograman Web

Oleh:

Abrizal Raka Firdiyanto

NIM. 24091397037

Kelas 2024 B



PROGRAM STUDI D4 MANAJEMEN INFORMATIKA

FAKULTAS VOKASI

UNIVERSITAS NEGERI SURABAYA

TAHUN 2025

DAFTAR ISI

DAFTAR ISI.....	II
SOURCE CODE.....	1
OUTPUT.....	4
PENJELASAN CODE.....	7
A. Penjelasan Code HTML.....	7
B. Penjelasan Kode JavaScript	8
C. Penjelasan Code CSS.....	11

SOURCE CODE



```
● ● ●  
1 <label class="switch">  
2   <input type="checkbox" id="theme-toggle">  
3   <span class="slider"></span>  
4   <span class="mode-text" id="mode-text">Light</span>  
5 </label>
```

Gambar 1. 1 kode html untuk switch mode



```
● ● ●  
1 <button id="toggle-photo">Sembunyikan Foto</button>
```

Gambar 1. 2 code html untuk button



```
● ● ●  
1 <script src="kode.js"></script>
```

Gambar 1. 3 code html untuk menghubungkan ke file script file lain



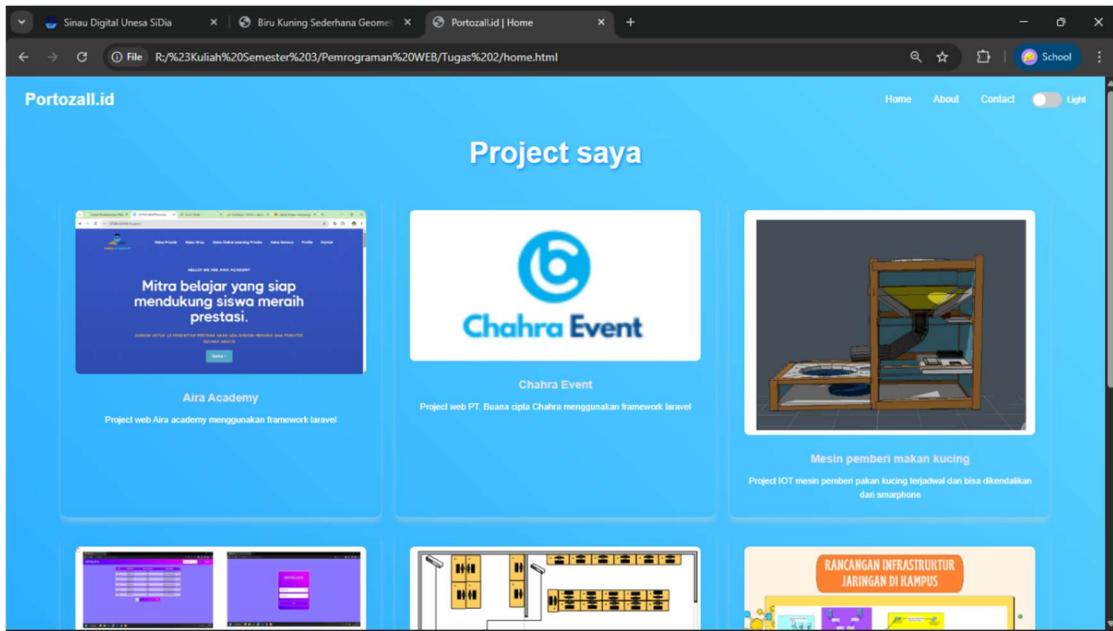
```
1 // Mengatur navbar ketika di scroll
2 window.addEventListener("scroll", () => {
3   const nav = document.querySelector("nav");
4   if (nav) {
5     if (window.scrollY > 50) {
6       nav.classList.add("scrolled");
7     } else {
8       nav.classList.remove("scrolled");
9     }
10  }
11 });
12
13 // Mengatur mode light/dark
14 const toggle = document.getElementById("theme-toggle");
15 const modeText = document.getElementById("mode-text");
16 const body = document.body;
17
18 if (localStorage.getItem("theme") === "dark") {
19   body.classList.add("dark-mode");
20   toggle.checked = true;
21   modeText.textContent = "Dark";
22 }
23
24 toggle.addEventListener("change", () => {
25   if (toggle.checked) {
26     body.classList.add("dark-mode");
27     localStorage.setItem("theme", "dark");
28     modeText.textContent = "Dark";
29   } else {
30     body.classList.remove("dark-mode");
31     localStorage.setItem("theme", "light");
32     modeText.textContent = "Light";
33   }
34 });
35
36 // Notifikasi contact
37 document.addEventListener("DOMContentLoaded", () => {
38   const form = document.querySelector(".contact-form");
39   const messageBox = Object.assign(document.createElement("div"), { className: "notification" });
40   form.appendChild(messageBox);
41
42   form.addEventListener("submit", e => {
43     e.preventDefault();
44     const name = form.querySelector("#name").value.trim();
45     const email = form.querySelector("#email").value.trim();
46     const pesan = form.querySelector("#message").value.trim();
47
48     const valid = name && email && pesan;
49     messageBox.textContent = valid ? "✓ Pesan berhasil dikirim." : "✗ Semua field harus diisi!";
50     messageBox.className = `notification ${valid ? "success" : "error"} show`;
51
52     if (valid) {
53       form.reset();
54       setTimeout(() => messageBox.classList.remove("show"), 3000);
55     }
56   });
57 });
58
59 // Tampilkan / sembunyikan foto profil
60 const photo = document.getElementById("profile-pic");
61 const button = document.getElementById("toggle-photo");
62
63 button.addEventListener("click", () => {
64   photo.classList.toggle("hide");
65
66   if (photo.classList.contains("hide")) {
67     button.textContent = "Tampilkan Foto";
68   } else {
69     button.textContent = "Sembunyikan Foto";
70   }
71 });
72
```

Gambar 1. 4 Source code javascript

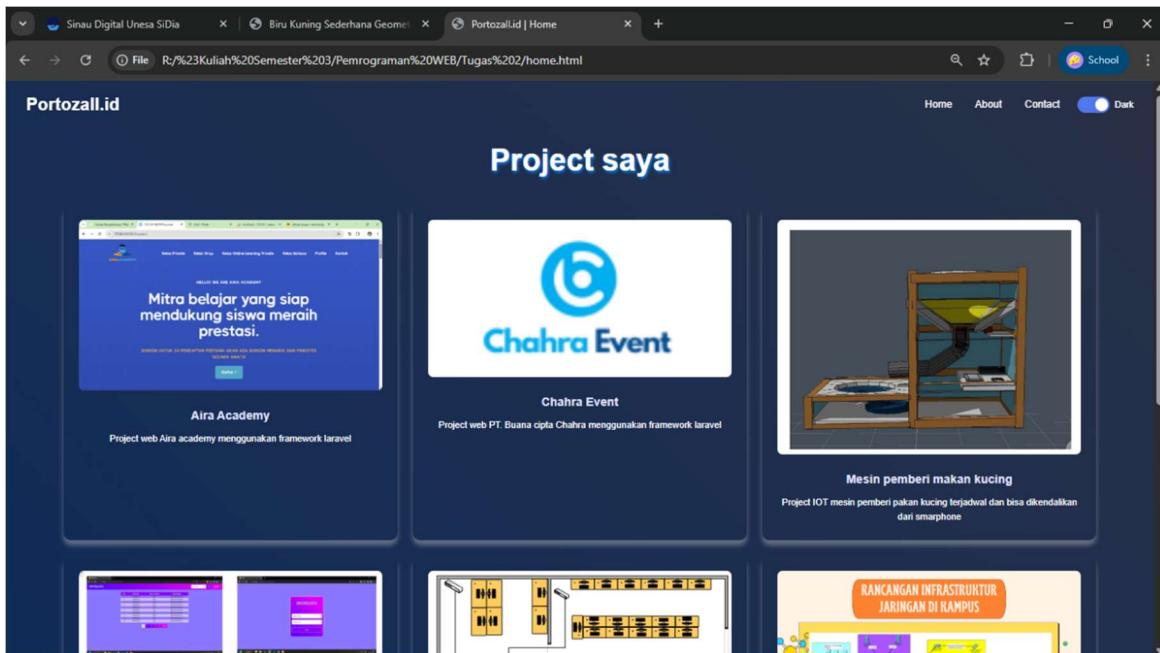
```
● ○ ●
1  /* mengatur mode */
2  .switch {
3    display: flex;
4    align-items: center;
5    gap: 8px;
6    cursor: pointer;
7  }
8
9  .switch input {
10   display: none;
11 }
12
13 .slider {
14   position: relative;
15   width: 50px;
16   height: 26px;
17   background: #ccc;
18   border-radius: 30px;
19   transition: 0.3s;
20 }
21
22 .slider::before {
23   content: "";
24   position: absolute;
25   inset: 3px;
26   width: 20px;
27   height: 20px;
28   background: #fff;
29   border-radius: 50%;
30   transition: 0.3s;
31 }
32
33 input:checked + .slider {
34   background: var(--biru-color);
35 }
36
37 input:checked + .slider::before {
38   transform: translateX(24px);
39 }
40
41 .mode-text {
42   font-size: 14px;
43   color: var(--text-color);
44 }
45
46 /* notifikasi */
47 .notification {
48   margin-top: 10px;
49   padding: 8px 12px;
50   border-radius: 6px;
51   font-weight: 600;
52   opacity: 0;
53   transition: opacity 0.5s;
54 }
55
56 .notification.show { opacity: 1; }
57
58 .success { background: var(--text-color); color: var(--succes-color); }
59 .error  { background: var(--text-color); color: var(--error-color); }
60
61 /* display dark mode */
62 body.dark-mode {
63   background: var(--gradasi-kedua);
64   background-size: 400% 400%;
65   background-attachment: fixed;
66   animation: gradientBG 10s ease infinite;
67   font-family: var(--font-main);
68 }
69
```

Gambar 1. 5 Source Code CSS

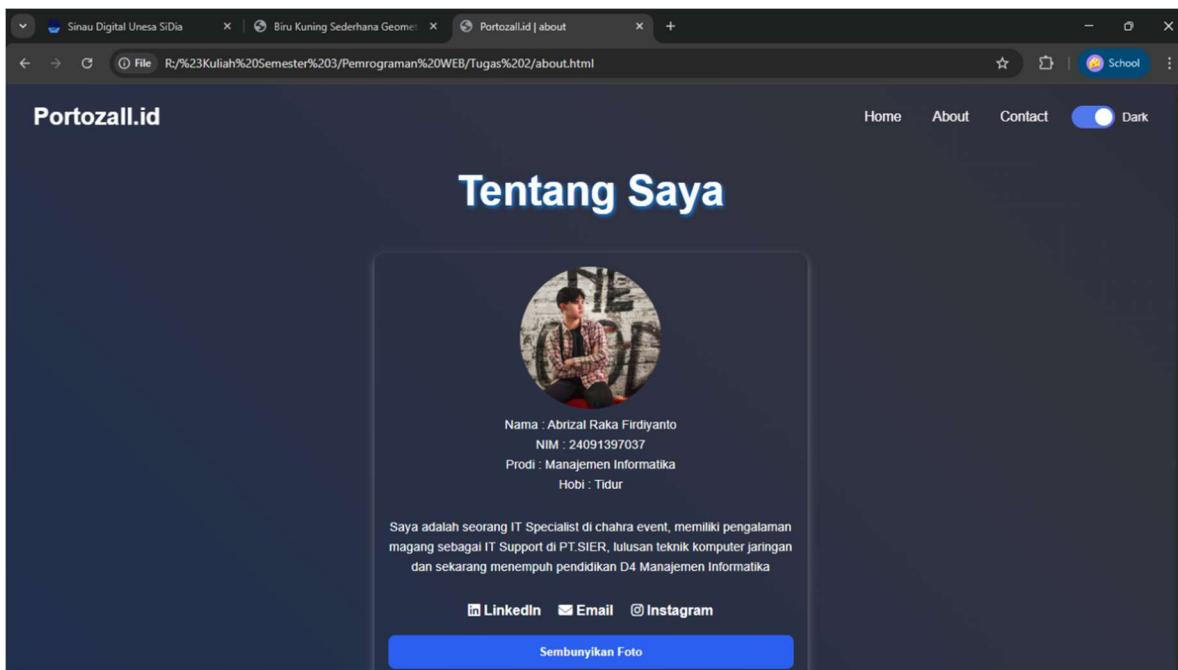
OUTPUT



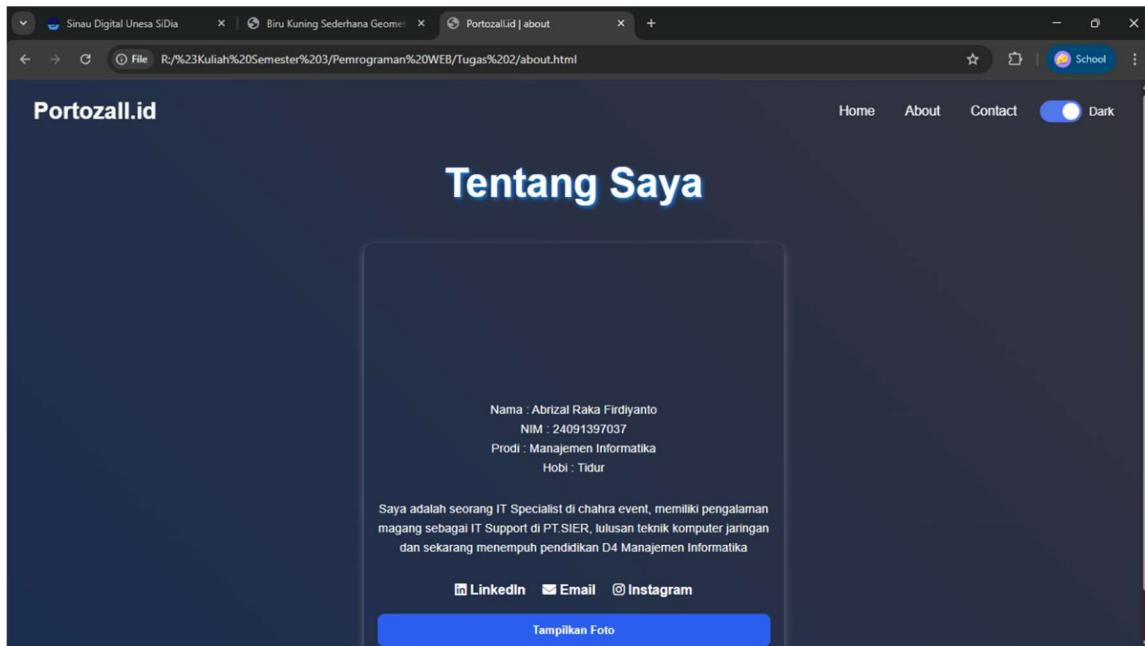
Gambar 1. 6 Ketika mode terang



Gambar 1. 7 Ketika mode gelap



Gambar 1. 8 sebelum klik button



Gambar 1. 9 Setelah klik button

Screenshot of a contact form titled "Hubungi saya" on the website Portozall.id. The form includes fields for Name, Email, and Pesan. The Name and Email fields are populated with "ABRIZAL RAKA FIRDIYANTO" and "abrizalraka@gmail.com" respectively. The Pesan field is empty. A blue button labeled "Kirim" is at the bottom. A red error message box displays the text "Please fill out this field." over the Pesan field.

Gambar 1. 10 Input form tidak lengkap

Screenshot of a contact form titled "Hubungi saya" on the website Portozall.id. The form fields are identical to the previous screenshot: Name (ABRIZAL RAKA FIRDIYANTO), Email (abrizalraka@gmail.com), and Pesan (empty). The "Kirim" button has been clicked, and a green success message box at the bottom states "Pesan berhasil dikirim."

Gambar 1. 11 Input terkirim

PENJELASAN CODE

A. Penjelasan Code HTML



```
1 <label class="switch">
2   <input type="checkbox" id="theme-toggle">
3   <span class="slider"></span>
4   <span class="mode-text" id="mode-text">Light</span>
5 </label>
```

Tag `<label class="switch">` membungkus seluruh elemen toggle. `<input type="checkbox" id="theme-toggle">` adalah checkbox yang akan digunakan untuk mengubah tema. `` berfungsi menampilkan tampilan slider sebagai pengganti checkbox default melalui CSS. `Light` menampilkan teks status tema yang bisa diubah lewat javascript.



```
1 <button id="toggle-photo">Sembunyikan Foto</button>
```

`<script src="kode.js"></script>` digunakan untuk memanggil file javascript eksternal bernama kode.js



```
1 <script src="kode.js"></script>
```

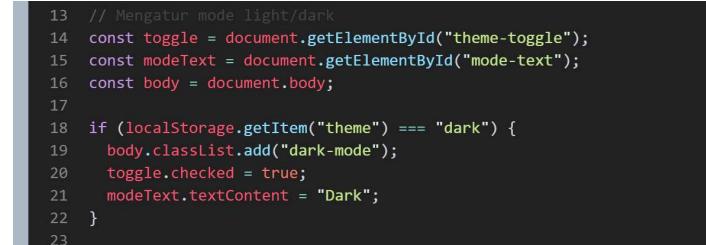
`<button id="toggle-photo">Sembunyikan Foto</button>` digunakan untuk membuat sebuah tombol. Elemen `<button>` berfungsi sebagai tombol interaktif, sedangkan atribut `id="toggle-photo"` memberi identitas unik sehingga tombol ini bisa diakses dan dikontrol menggunakan CSS atau javascript.

B. Penjelasan Kode JavaScript



```
1 // Mengatur navbar ketika di scroll
2 window.addEventListener("scroll", () => {
3   const nav = document.querySelector("nav");
4   if (nav) {
5     if (window.scrollY > 50) {
6       nav.classList.add("scrolled");
7     } else {
8       nav.classList.remove("scrolled");
9     }
10  }
11 });
12 
```

Menambahkan event listener pada window untuk mendeteksi ketika halaman di-scroll. Fungsi panah () => { ... } dijalankan setiap kali terjadi pergeseran halaman. Di dalamnya, document.querySelector("nav") digunakan untuk mengambil elemen <nav>. Kondisi if (window.scrollY > 50) mengecek apakah posisi scroll vertikal lebih dari 50 piksel. Jika benar, maka nav.classList.add("scrolled") menambahkan kelas CSS scrolled pada elemen <nav>. Jika tidak, nav.classList.remove("scrolled") menghapus kelas tersebut.



```
13 // Mengatur mode light/dark
14 const toggle = document.getElementById("theme-toggle");
15 const modeText = document.getElementById("mode-text");
16 const body = document.body;
17
18 if (localStorage.getItem("theme") === "dark") {
19   body.classList.add("dark-mode");
20   toggle.checked = true;
21   modeText.textContent = "Dark";
22 }
23 
```

Untuk mengatur tema halaman berdasarkan data yang tersimpan di localstorage. Pertama, elemen checkbox dengan id theme-toggle, elemen teks dengan id mode-text, dan elemen <body> disimpan dalam variabel. Selanjutnya, program mengecek apakah di localstorage terdapat nilai theme yang disimpan sebagai "dark". Jika iya, maka kelas dark-mode ditambahkan ke <body> agar tampilan berubah menjadi mode gelap. Pada saat yang sama, checkbox diatur menjadi aktif dengan properti checked = true, dan teks mode diubah menjadi "Dark" untuk menyesuaikan tampilan.

```
24 toggle.addEventListener("change", () => {
25   if (toggle.checked) {
26     body.classList.add("dark-mode");
27     localStorage.setItem("theme", "dark");
28     modeText.textContent = "Dark";
29   } else {
30     body.classList.remove("dark-mode");
31     localStorage.setItem("theme", "light");
32     modeText.textContent = "Light";
33   }
34});
```

Untuk menangani perubahan pada elemen toggle (checkbox). Event listener "change" akan dijalankan setiap kali pengguna mengaktifkan atau menonaktifkan toggle. Jika toggle dalam kondisi aktif (checked = true), maka kelas dark-mode ditambahkan ke <body>, nilai tema "dark" disimpan ke localstorage, dan teks status mode diubah menjadi "Dark". Sebaliknya, jika toggle tidak aktif, kelas dark-mode dihapus dari <body>, nilai tema "light" disimpan ke localstorage, dan teks status mode diganti menjadi "Light". Dengan cara ini, tema halaman dapat berubah secara langsung sesuai interaksi pengguna, sekaligus disimpan agar tetap sama saat halaman dimuat kembali.

```
35
36 // Notifikasi contact
37 document.addEventListener("DOMContentLoaded", () => {
38   const form = document.querySelector(".contact-form");
39   const messageBox = Object.assign(document.createElement("div"), { className: "notification" });
40   form.appendChild(messageBox);
41
42   form.addEventListener("submit", e => {
43     e.preventDefault();
44     const name = form.querySelector("#name").value.trim();
45     const email = form.querySelector("#email").value.trim();
46     const pesan = form.querySelector("#message").value.trim();
47   });
48});
```

Kode ini dipakai untuk membuat interaksi pada form kontak ketika halaman sudah selesai dimuat. Event listener domcontentloaded memastikan seluruh elemen HTML siap digunakan sebelum kode dijalankan. Variabel form mengambil elemen dengan class contact-form. Lalu dibuat sebuah elemen baru berupa <div> dengan class notification menggunakan Object.assign(document.createElement("div"), { classname: "notification" }), kemudian elemen ini ditambahkan ke dalam form dengan form.appendChild(messagebox) untuk menampilkan notifikasi.

Selanjutnya, ditambahkan event listener submit pada form. Fungsi e.preventDefault() digunakan agar form tidak melakukan aksi default (refresh atau kirim data ke server). Setelah itu, nilai dari input name, email, dan message diambil menggunakan

form.querySelector(...).value.trim(). Method .trim() dipakai untuk menghapus spasi di awal dan akhir input agar data lebih bersih sebelum diproses

```
47
48     const valid = name && email && pesan;
49     messageBox.textContent = valid ? "✓ Pesan berhasil dikirim." : "✗ Semua field harus diisi!";
50     messageBox.className = `notification ${valid ? "success" : "error"} show`;
51
52     if (valid) {
53         form.reset();
54         setTimeout(() => messageBox.classList.remove("show"), 3000);
55     }
56 });
57 });
58 //
```

Ode ini berfungsi untuk melakukan validasi sederhana pada form sebelum menampilkan notifikasi. Variabel valid akan bernilai true jika semua input (name, email, dan pesan) terisi, dan false jika ada yang kosong. Setelah itu, isi teks pada messagebox diubah dengan textcontent: jika valid maka ditampilkan pesan sukses "✓ Pesan berhasil dikirim.", jika tidak valid ditampilkan pesan error "✗ Semua field harus diisi!".

Class pada messagebox juga diperbarui menjadi notification success show ketika valid, atau notification error show ketika tidak valid. Jika data valid, form dikosongkan kembali dengan form.reset(). Terakhir, fungsi setTimeout dipakai untuk menghapus class show setelah 3 detik, sehingga notifikasi akan otomatis hilang setelah ditampilkan. Dengan begitu, pengguna mendapat umpan balik langsung apakah input form sudah benar atau belum.

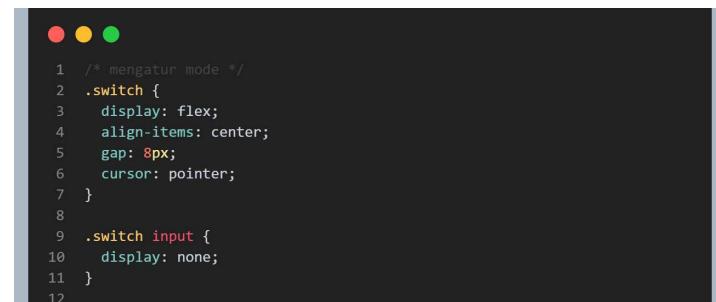
```
59 // Tampilkan / sembunyikan foto profil
60 const photo = document.getElementById("profile-pic");
61 const button = document.getElementById("toggle-photo");
62
63 button.addEventListener("click", () => {
64     photo.classList.toggle("hide");
65
66     if (photo.classList.contains("hide")) {
67         button.textContent = "Tampilkan Foto";
68     } else {
69         button.textContent = "Sembunyikan Foto";
70     }
71 });
72
```

Kode ini digunakan untuk membuat tombol yang bisa menyembunyikan dan menampilkan foto secara bergantian. Variabel photo mengambil elemen gambar dengan id profile-pic,

sedangkan button mengambil tombol dengan id toggle-photo. Event listener click dipasang pada tombol agar kode dijalankan setiap kali tombol ditekan.

Di dalamnya, photo.classList.toggle("hide") akan menambahkan atau menghapus kelas hide pada foto. Jika foto sedang memiliki kelas hide, maka teks tombol diubah menjadi "Tampilkan Foto", artinya gambar sedang tersembunyi. Sebaliknya, jika foto tidak memiliki kelas hide, teks tombol berubah menjadi "Sembunyikan Foto", artinya gambar terlihat. Dengan cara ini, satu tombol bisa berfungsi ganda sebagai penampil sekaligus menyembunyi foto.

C. Penjelasan Code CSS



```
1 /* mengatur mode */
2 .switch {
3   display: flex;
4   align-items: center;
5   gap: 8px;
6   cursor: pointer;
7 }
8
9 .switch input {
10   display: none;
11 }
12
```

Mengatur elemen dengan class .switch agar ditampilkan menggunakan flexbox, sehingga kontennya sejajar secara horizontal dan rata tengah dengan align-items: center. Properti gap: 8px memberi jarak antar elemen, dan cursor: pointer membuat kursor berubah menjadi tangan saat diarahkan ke elemen. Pada bagian .switch input, properti display: none menyembunyikan elemen input (checkbox) agar tidak terlihat, sehingga hanya elemen gaya kustom seperti slider yang tampil.



```
12
13 .slider {
14   position: relative;
15   width: 50px;
16   height: 26px;
17   background: #ccc;
18   border-radius: 30px;
19   transition: 0.3s;
20 }
21
```

Membuat class .slider berbentuk tombol geser (toggle). Properti position: relative dipakai agar elemen di dalamnya bisa diatur dengan posisi absolut. Lebar dan tinggi ditentukan dengan width: 50px dan height: 26px, lalu diberi warna dasar abu-abu dengan background:

#ccc. Bentuknya dibuat membulat dengan border-radius: 30px, dan efek transisi transition: 0.3s ditambahkan agar perubahan warna atau posisi saat toggle lebih halus.

```
21 .slider::before {  
22   content: "";  
23   position: absolute;  
24   inset: 3px;  
25   width: 20px;  
26   height: 20px;  
27   background: #fff;  
28   border-radius: 50%;  
29   transition: 0.3s;  
30 }  
31 }
```

Membuat elemen semu .slider::before yang berfungsi sebagai bulatan tombol pada switch. Properti content: "" wajib ditulis agar pseudo-element muncul. Position: absolute dipakai supaya bulatan bisa diposisikan bebas di dalam .slider. Dengan inset: 3px, bulatan ditempatkan 3px dari setiap sisi, menghasilkan ukuran pas. Lebar dan tinggi ditetapkan 20px, diberi warna putih (background: #fff) dan dibulatkan penuh dengan border-radius: 50% sehingga berbentuk lingkaran. Efek transition: 0.3s membuat pergerakan bulatan saat switch berubah lebih halus.

```
32  
33 input:checked + .slider {  
34   background: var(--biru-color);  
35 }  
36  
37 input:checked + .slider::before {  
38   transform: translateX(24px);  
39 }  
40
```

Mengatur tampilan saat toggle aktif. Selector input:checked + .slider mengubah warna latar belakang slider menjadi var(--biru-color) ketika checkbox dicentang. Pada input:checked + .slider::before, bulatan slider digeser ke kanan sejauh 24px dengan transform: translateX(24px), sehingga terlihat seperti tombol berpindah posisi. Sedangkan class .mode-text mengatur teks status mode dengan ukuran huruf 14px dan warna yang diambil dari variabel --text-color.

```
41 .mode-text {  
42   font-size: 14px;  
43   color: var(--text-color);  
44 }  
45  
46 /* notifikasi */  
47 .notification {  
48   margin-top: 10px;  
49   padding: 8px 12px;  
50   border-radius: 6px;  
51   font-weight: 600;  
52   opacity: 0;  
53   transition: opacity 0.5s;  
54 }  
55  
56 .notification.show { opacity: 1; }  
57  
58 .success { background: var(--text-color); color: var(--succes-color); }  
59 .error { background: var(--text-color); color: var(--error-color); }
```

Mengatur tampilan notifikasi pada form. Class .notification memberi jarak atas (margin-top: 10px), ruang dalam (padding: 8px 12px), sudut membulat (border-radius: 6px), serta membuat teks tebal (font-weight: 600). Properti opacity: 0 menyembunyikan notifikasi secara default, lalu transition: opacity 0.5s membuat efek munculnya lebih halus. Saat class tambahan .show ditambahkan, opacity berubah menjadi 1 sehingga notifikasi terlihat. Class .success memberi latar sesuai variabel --text-color dan teks dengan warna --succes-color, sedangkan .error menggunakan warna teks --error-color untuk membedakan pesan kesalahan.

```
60  
61 /* display dark mode */  
62 body.dark-mode {  
63   background: var(--gradasi-kedua);  
64   background-size: 400% 400%;  
65   background-attachment: fixed;  
66   animation: gradientBG 10s ease infinite;  
67   font-family: var(--font-main);  
68 }  
69
```

Kode CSS ini mengatur gaya halaman saat mode gelap aktif dengan menambahkan class dark-mode pada elemen <body>. Latar belakang diatur menggunakan background: var(--gradasi-kedua) yang biasanya berupa warna gradasi. Properti background-size: 400% 400% membuat gradasi lebih besar agar animasi terlihat halus, sedangkan background-attachment: fixed menjaga latar tetap diam saat halaman digulir. Efek animasi diberikan dengan animation: gradientbg 10s ease infinite sehingga latar bergerak perlahan tanpa henti. Terakhir, teks diatur menggunakan font-family: var(--font-main) agar konsisten dengan font utama yang sudah ditentukan.