

LAPORAN PRAKTIKUM PEMROGRAMAN WEB D



SOFTWARE ENGINEERING
LABORATORY

NAMA : Firja Adyatma Sitepu

NIM : 202431104

KELAS : D

DOSEN : Muhammad Fadli Prathama, S.SI., M.MSI

NO.PC : 21

ASISTEN : 1. Sakura Amastasya Salsabila

2. Setiyanto

3. Muhammad Farhan Fahrezy

4. Kashrina Masyid Azka

**INSTITUT TEKNOLOGI PLN
TEKNIK INFORMATIKA**

2025

INDEX.HTML

Laporan_praktikum2 > <> index.html > html

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Pencemaran Lingkungan</title>
7  </head>
8  <body>
9
10 </body>
11 </html>
```

202431104_Firja Adyatma Sitepu

Ini adalah codingan awal, yang belum ada apa apa karena baru memakai shortcut yaitu "!". Yang mana shortcut tersebut akan otomatis membuat codingan seperti yang ada pada gambar.

```
1  <!DOCTYPE html>
2  <html lang="id">
3  <head>
4      <meta charset="utf-8" />
5      <meta name="viewport" content="width=device-width,initial-scale=1" />
6      <title>Pencemaran Lingkungan</title>
7      <link rel="stylesheet" href="style.css" />
8  </head>
9  <body id="body" class="tema-hijau rapi">
10     <!-- HEADER / IDENTITAS -->
11     <header class="header">
12         <div class="identitas">
13             <div><strong>Nama:</strong> Firja Adyatma Sitepu</div>
14             <div><strong>NIM:</strong> 202431104 </div>
15         </div>
16     </header>
```

202431104_Firja Adyatma Sitepu

Dokumen dimulai dengan deklarasi `<!DOCTYPE html>`. Bagian ini digunakan untuk memberi tahu browser bahwa file tersebut adalah dokumen HTML5. Setelah itu, tag `<html lang="id">` digunakan untuk membuka struktur utama HTML dan memberi tahu bahwa bahasa utama halaman ini adalah bahasa Indonesia.

Di dalam tag `<head>`, terdapat beberapa elemen penting. Pertama, `<meta charset="utf-8" />` berfungsi untuk menentukan jenis karakter yang digunakan, yaitu UTF-8, sehingga semua huruf, simbol, dan tanda baca dapat ditampilkan dengan benar. Lalu ada `<meta name="viewport" content="width=device-width,initial-scale=1" />` yang membuat tampilan web menjadi responsif di layar ponsel, tablet, maupun komputer. Tag `<title>` berisi judul halaman yang akan tampil pada tab browser, yaitu "Praktikum - Pencemaran Lingkungan". Setelah itu, terdapat tag `<link rel="stylesheet" href="style.css" />` yang menghubungkan file HTML dengan file CSS eksternal bernama style.css. File CSS tersebut berfungsi untuk mengatur tampilan, warna, dan gaya dari halaman web.

Masuk ke bagian `<body id="body" class="tema-hijau rapi">`. Tag `<body>` adalah wadah utama yang berisi seluruh konten web. Atribut `id="body"` memberi identitas khusus yang dapat dipakai di

CSS atau JavaScript. Atribut `class="tema-hijau rapi"` berarti elemen ini memiliki dua kelas, yaitu "tema-hijau" dan "rapi", biasanya digunakan untuk menerapkan gaya visual tertentu.

Selanjutnya terdapat bagian komentar `<!-- HEADER / IDENTITAS -->` yang hanya sebagai catatan pada kode dan tidak akan tampil di halaman web. Setelah itu, terdapat tag `<header class="header">` sebagai bagian kepala halaman (header). Kelas "header" umumnya mengatur tampilan area ini di file CSS.

Di dalam header ada `<div class="identitas">` yang berfungsi sebagai kontainer untuk menampilkan informasi pengguna. Di dalam div tersebut terdapat dua baris informasi: yang pertama menampilkan nama dengan format `Nama: Firja Adyatma Sitepu`, di mana tag `` digunakan untuk menebalkan teks "Nama:". Baris kedua menampilkan NIM dengan format `NIM: 202431104`. Kedua informasi ini ditempatkan dalam tag `<div>` agar tetap rapi dan mudah diatur secara visual melalui CSS.

```
18 <!-- KONTEN UTAMA -->
19 <main class="container">
20 <!-- PENJELASAN UMUM -->
21 <section class="umum">
22 <h2>Penjelasan Umum tentang Pencemaran Lingkungan</h2>
23 <p>
24 Pencemaran lingkungan adalah masuknya zat, energi, atau makhluk hidup ke lingkungan sehingga
25 kualitas lingkungan menurun dan dapat membahayakan kehidupan manusia, hewan, dan tumbuhan.
26 Sumber utamanya antara lain limbah industri, emisi kendaraan, pembakaran sampah, dan limbah rumah tangga.
27 </p>
28
29 <div class="tombol-row">
30 <button id="btnShowGallery" onclick="location.href='gambar.html'">Tampilkan Gambar (Galeri)</button>
31 </div>
32 </section>
33 <hr />
```

202431104_Firja Adyatma Sitepu

Bagian ini diawali dengan komentar `<!-- KONTEN UTAMA -->` yang hanya berfungsi sebagai penanda dalam kode, tidak akan terlihat di halaman web. Setelah komentar tersebut, terdapat tag `<main class="container">`. Tag `<main>` digunakan untuk menandai bagian utama dari isi halaman, yaitu area yang memuat konten terpenting. Atribut `class="container"` biasanya digunakan untuk mengatur tata letak agar konten memiliki jarak yang rapi dengan tepi halaman, seperti memberikan margin atau lebar tertentu melalui CSS.

Di dalam main terdapat sebuah `<section class="umum">`. Tag `<section>` digunakan untuk mengelompokkan konten yang memiliki tema atau topik tertentu—inilah bagian khusus yang menjelaskan penjelasan umum tentang pencemaran lingkungan. Kelas "umum" kemungkinan mengatur gaya tampilan untuk bagian ini, seperti warna latar, ukuran teks, atau spasi.

Pada bagian awal section terdapat tag `<h2>Penjelasan Umum tentang Pencemaran Lingkungan</h2>`. Tag `<h2>` menampilkan judul bagian dengan ukuran heading kedua terbesar setelah `<h1>`. Judul ini memberi tahu pengguna bahwa bagian tersebut berisi penjelasan umum mengenai pencemaran lingkungan.

Di bawah judul terdapat tag `<p> ... </p>` yang berisi paragraf penjelasan. Paragraf tersebut menerangkan bahwa pencemaran lingkungan terjadi ketika zat, energi, atau organisme tertentu masuk ke lingkungan hingga menyebabkan kualitasnya menurun dan membahayakan kehidupan makhluk hidup. Dalam paragraf juga disebutkan beberapa sumber pencemaran, seperti limbah

industri, emisi kendaraan, pembakaran sampah, dan limbah rumah tangga. Tag <p> sendiri berfungsi untuk menampilkan teks dalam bentuk paragraf dengan jarak dan format yang rapi.

Setelah paragraf, terdapat sebuah <div class="tombol-row">. Ini adalah kontainer yang digunakan untuk menempatkan elemen tombol agar posisinya bisa diatur secara horizontal atau memiliki tata letak tertentu sesuai CSS. Di dalamnya terdapat sebuah tombol <button>, yaitu:

```
<button id="btnShowGallery" onclick="location.href='gambar.html'">
```

Tampilkan Gambar (Galeri)

```
</button>
```

Tombol ini memiliki id="btnShowGallery" untuk identifikasi, dan memiliki atribut onclick yang berisi perintah JavaScript sederhana. Saat tombol ditekan, browser akan dijalankan menuju file gambar.html. Dengan kata lain, tombol ini berfungsi sebagai tautan (link) menuju halaman galeri gambar.

Setelah section selesai, terdapat tag <hr />. Tag <hr> adalah garis horizontal yang digunakan untuk memisahkan bagian konten secara visual. Garis ini menunjukkan bahwa setelah area penjelasan umum, akan ada bagian konten lain yang berbeda.

```
35 <!-- PETUNJUK -->
36 <section class="topik">
37   <h3 class="petunjuk"> Klik judul untuk menampilkan / menyembunyikan penjelasan </h3>
38
39   <!-- 7 JUDUL DENGAN INTERAKSI -->
40   <article>
41     <h4 onclick="togglePenjelasan('p1')">1. Pencemaran Udara</h4>
42     <p id="p1" class="penjelasan"> Pencemaran udara terjadi karena emisi yang dihasilkan oleh kendaraan bermotor maupun
    aktivitas industri yang terus meningkat setiap tahunnya. Kendaraan mengeluarkan gas buang berupa karbon monoksida,
    nitrogen oksida, dan hidrokarbon yang dapat membahayakan kesehatan manusia serta merusak lingkungan. Sementara itu,
    pabrik dan proses pembakaran di industri menghasilkan partikel halus (PM2.5 dan PM10) serta berbagai gas beracun yang
    dapat mencemari atmosfer dalam jangka panjang. Jika tidak dikendalikan, kedua sumber emisi ini dapat memperburuk kualitas
    udara, menyebabkan kabut asap, serta meningkatkan risiko penyakit pernapasan pada masyarakat. peningkatan jumlah
    kendaraan dan pertumbuhan industri yang tidak diimbangi dengan pengawasan ketat membuat pencemaran udara semakin sulit
    dikendalikan. </p>
43   </article>
44
45   <article>
46     <h4 onclick="togglePenjelasan('p2')">2. Pencemaran Air</h4>
47     <p id="p2" class="penjelasan"> Limbah cair industri dan pembuangan sampah ke sungai maupun laut menjadi penyebab utama
    pencemaran air yang semakin mengkhawatirkan. Banyak industri membuang limbah tanpa melalui proses pengolahan yang
    memadai, sehingga zat kimia berbahaya seperti logam berat, deterjen, dan bahan beracun lainnya langsung masuk ke
    perairan. Hal ini tidak hanya merusak ekosistem air, tetapi juga mengancam makhluk hidup yang bergantung pada sumber air
    tersebut, termasuk manusia. Air yang tercemar dapat menyebabkan gangguan kesehatan, menurunnya kualitas air bersih, serta
    kerusakan jangka panjang pada lingkungan. Selain itu, kebiasaan masyarakat
    proses pencemaran semakin parah dan sulit untuk dipulihkan. </p>
48   </article>
```

202431104_Firja Adyatma Sitepu

Bagian ini dimulai dengan komentar <!-- PETUNJUK --> sebagai penanda dalam kode bahwa bagian yang muncul setelahnya berisi petunjuk atau instruksi bagi pengguna. Komentar seperti ini tidak ditampilkan di halaman web, tetapi membantu pembaca kode memahami struktur dokumen.

Setelah komentar, terdapat elemen <section class="topik">. Tag <section> berfungsi untuk mengelompokkan konten berdasarkan tema tertentu. Dalam hal ini, bagian tersebut

mengelompokkan beberapa topik pencemaran lingkungan. Atribut `class="topik"` memungkinkan area ini diberi gaya tampilan khusus melalui CSS, seperti margin, padding, atau warna latar tertentu.

Di dalam section terdapat elemen `<h3 class="petunjuk">` Klik judul untuk menampilkan / menyembunyikan penjelasan `</h3>`. Ini adalah judul tingkat tiga yang digunakan sebagai petunjuk bagi pengguna. Kelas "petunjuk" kemungkinan dipakai untuk mengatur tampilan teks, misalnya ukuran font atau warnanya. Teks ini memberi informasi bahwa pengguna dapat mengklik setiap judul untuk membuka atau menutup penjelasan yang tersembunyi.

Bagian berikutnya terdapat komentar `<!-- 7 JUDUL DENGAN INTERAKSI -->`, yang menjelaskan bahwa akan ada tujuh judul yang bisa diklik. Kode yang ditunjukkan merupakan salah satu dari tujuh judul tersebut.

Kemudian terdapat tag `<article>`. Elemen `<article>` digunakan untuk menampilkan sebuah konten atau topik yang berdiri sendiri. Setiap topik pencemaran dipisahkan dengan tag ini agar strukturnya lebih rapi dan mudah dipahami.

Di dalam article terdapat:

1. Judul interaktif

```
<h4 onclick="togglePenjelasan('p1')">1. Pencemaran Udara</h4>
```

Tag `<h4>` digunakan sebagai judul sub-topik dengan tingkat heading keempat. Atribut `onclick="togglePenjelasan('p1')"` memberikan fungsi interaktif ketika judul ini diklik. Fungsi `togglePenjelasan('p1')` (yang kemungkinan ada di file JavaScript atau bagian `<script>`) digunakan untuk menampilkan atau menyembunyikan paragraf penjelasan dengan id `p1`. Ini artinya paragraf penjelasan dapat di-"toggle", yaitu bisa diperlihatkan atau disembunyikan sesuai tindakan pengguna.

2. Paragraf penjelasan

```
<p id="p1" class="penjelasan"> ... </p>
```

Paragraf ini memiliki `id="p1"`, yaitu identitas unik yang digunakan oleh fungsi JavaScript agar bisa diakses dan diubah tampilannya. Kelas CSS `penjelasan` biasanya digunakan untuk mengatur apakah paragraf ini terlihat atau tersembunyi, misalnya dengan `display: none` atau gaya tulisan tertentu.

Isi paragraf menjelaskan secara detail tentang pencemaran udara: penyebabnya, sumber emisinya, jenis polutan seperti CO, NOx, hidrokarbon, PM2.5, PM10, serta dampaknya terhadap kesehatan dan lingkungan. Kalimat terakhir menegaskan bahwa pertumbuhan kendaraan dan industri yang tidak diawasi menyebabkan pencemaran udara semakin sulit dikendalikan.

```

49
50 <article>
51 <h4 onclick="togglePenjelasan('p3')">3. Pencemaran Tanah</h4>
52 <p id="p3" class="penjelasan"> Pestisida, limbah padat, dan tumpahan bahan kimia menjadi faktor utama yang menurunkan
kualitas tanah dan mengganggu keseimbangan ekosistem darat. Penggunaan pestisida secara berlebihan dalam kegiatan
pertanian dapat membunuh organisme tanah yang sebenarnya berperan penting dalam menjaga kesuburan. Selain itu, limbah
padat seperti plastik, logam, dan sisa industri yang dibuang sembarangan dapat menghambat proses penyerapan air oleh
tanah serta mencemari lapisan tanah dengan zat berbahaya. Tumpahan bahan kimia dari kegiatan industri juga dapat meresap
ke dalam tanah dan merusak struktur serta kandungan mineral alami di dalamnya. Akibatnya, tanah menjadi kurang subur dan
tidak lagi mampu mendukung tanaman dengan baik. </p>
53 </article>
54
55 <article>
56 <h4 onclick="togglePenjelasan('p4')">4. Kebisingan</h4>
57 <p id="p4" class="penjelasan"> Kebisingan yang berasal dari kendaraan dan mesin memberikan dampak negatif yang signifikan
terhadap kesehatan dan kenyamanan manusia. Suara bising dari lalu lintas yang padat, terutama di kawasan perkotaan, dapat
meningkatkan tingkat stres, mengganggu konsentrasi, serta memicu gangguan tidur bagi masyarakat yang tinggal di
sekitarnya. Di sisi lain, penggunaan mesin industri yang bekerja dengan intensitas tinggi juga menghasilkan tingkat
kebisingan yang cukup besar, sehingga berpotensi merusak pendengaran pekerja maupun warga sekitar jika terjadi secara
terus-menerus. Kondisi ini menjadikan kebisingan sebagai salah satu bentuk polusi yang sering diabaikan, padahal memiliki
dampak jangka panjang yang serius. Selain itu, tingkat kebisingan yang tidak terkontrol dapat menurunkan kualitas hidup
dan membuat lingkungan terasa tidak nyaman untuk ditinggali. </p>
58 </article>

```

202431104_Firja Adyatma Sitepu

```

59
60 <article>
61 <h4 onclick="togglePenjelasan('p5')">5. Sampah Plastik</h4>
62 <p id="p5" class="penjelasan"> Plastik merupakan salah satu jenis limbah yang paling sulit terurai, sehingga
keberadaannya dapat mencemari lingkungan darat maupun laut dalam jangka waktu yang sangat lama. Ketika dibuang
sembarangan, plastik akan menumpuk di tanah dan menghambat proses penyerapan air serta pertumbuhan tanaman. Di perairan,
sampah plastik dapat terbawa arus hingga mencapai laut dan terpecah menjadi mikroplastik yang sulit dideteksi. Keberadaan
mikroplastik ini dapat masuk ke rantai makanan dan membahayakan organisme yang hidup di ekosistem tersebut. Fauna laut
seperti ikan, penyu, dan burung sering kali menelan plastik karena mengiranya sebagai makanan, yang pada akhirnya
menyebabkan luka internal, keracunan, bahkan kematian. Peningkatan jumlah sampah plastik yang tidak terkelola dengan baik
menjadikan pencemaran semakin sulit diatasi. </p>
63 </article>
64
65 <article>
66 <h4 onclick="togglePenjelasan('p6')">6. Limbah Industri</h4>
67 <p id="p6" class="penjelasan"> Limbah industri yang tidak diolah dengan baik dapat memasukkan berbagai zat beracun ke
lingkungan dan menimbulkan dampak yang luas bagi ekosistem. Banyak pabrik yang menghasilkan limbah berupa bahan kimia
berbahaya seperti logam berat, senyawa organik, dan zat korosif yang dapat mencemari tanah, air, maupun udara. Ketika
limbah tersebut dibuang tanpa melalui proses pengolahan yang memadai, zat-zat beracun tersebut dapat meresap ke dalam
tanah, mencemari sumber air, serta mengganggu keseimbangan ekologi. Dampaknya tidak hanya dirasakan oleh tumbuhan dan
hewan, tetapi juga oleh manusia yang menggunakan air atau mengonsumsi produk dari lingkungan yang telah tercemar. Selain
itu, pencemaran dari limbah industri yang tidak dikendalikan dapat menyebabkan kerusakan lingkungan jangka panjang yang
sulit dipulihkan. </p>
68 </article>

```

202431104_Firja Adyatma Sitepu

```

69
70 <article>
71 <h4 onclick="togglePenjelasan('p7')">7. Emisi Kendaraan Bermotor</h4>
72 <p id="p7" class="penjelasan"> Emisi kendaraan bermotor menghasilkan berbagai polutan berbahaya seperti karbon monoksida
(CO), nitrogen oksida (NOx), dan partikel halus yang dapat mencemari udara secara signifikan. Karbon monoksida dapat
mengganggu kemampuan darah dalam mengangkut oksigen, sedangkan nitrogen oksida berkontribusi terhadap pembentukan hujan
asam serta iritasi pada saluran pernapasan. Partikel halus seperti PM2.5 dan PM10 juga sangat berbahaya karena ukurannya
yang kecil memungkinkan mereka masuk hingga ke paru-paru dan aliran darah. Kondisi ini membuat daerah dengan lalu lintas
padat sering mengalami penurunan kualitas udara yang berdampak pada kesehatan masyarakat, terutama bagi anak-anak,
lansia, dan penderita penyakit pernapasan. Dan juga, meningkatnya jumlah kendaraan setiap tahun menyebabkan konsentrasi
polutan di udara semakin sulit dikendalikan. </p>
73 </article>
74 </section>

```

202431104_Firja Adyatma Sitepu

Ketiga gambar tersebut sama saja codingan nya, hanya isi nya saja yang saya bedakan

```
75 |  
76 | <!-- MODE TAMPILAN -->  
77 | <section class="modes">  
78 |   <h3>Mode Tampilan</h3>  
79 |   <div class="tombol-row">  
80 |     <button id="btnRapi" onclick="modeRapi()"> Mode Rapi </button>  
81 |     <button id="btnBerantakan" onclick="modeBerantakan()">Mode Berantakan (hati hati kak/bang)</button>  
82 |   </div>  
83 | </section>  
84 |  
85 | </main>  
86 |  
87 | <footer class="footer">  
88 |   <button onclick="goBack()">Kembali</button>  
89 | </footer>  
90 |  
91 | <script src="script.js"></script>  
92 | </body>  
93 | </html>  
94 |
```

202431104_Firja Adyatma Sitepu

komentar `<!-- MODE TAMPILAN -->`. Komentar ini hanya sebagai penanda dalam kode untuk menunjukkan bahwa bagian berikutnya mengatur mode tampilan halaman. Komentar tidak akan muncul di halaman web dan hanya dapat dilihat oleh pembaca kode.

Selanjutnya terdapat sebuah `<section class="modes">`. Tag `<section>` digunakan untuk mengelompokkan konten berdasarkan fungsi tertentu, dan di sini bagian tersebut berfungsi sebagai area untuk mengganti mode tampilan website. Kelas “modes” memungkinkan style khusus diterapkan pada bagian ini melalui CSS.

Di dalam section terdapat `<h3>Mode Tampilan</h3>`. Ini adalah judul tingkat tiga (heading 3) yang memberi tahu pengguna bahwa bagian ini digunakan untuk mengubah mode tampilan web.

Setelah itu ada `<div class="tombol-row">`, yaitu kontainer yang menampung dua tombol dalam satu baris. Kelas ini sangat mungkin mengatur tata letak tombol, misalnya agar mereka sejajar secara horizontal.

Dua Tombol Mode Tampilan

Di dalam div terdapat dua tombol:

1. Tombol Mode Rapi

```
<button id="btnRapi" onclick="modeRapi()"> Mode Rapi </button>
```

- Tombol ini memiliki `id="btnRapi"` untuk identifikasi unik.
- Atribut `onclick="modeRapi()"` berarti ketika tombol ditekan, akan menjalankan fungsi **modeRapi()** yang didefinisikan di file `script.js`.
- Fungsi ini kemungkinan mengganti tampilan situs menjadi rapi, misalnya menata posisi elemen menjadi lurus, teratur, dan terstruktur.

2. Tombol Mode Berantakan

```
<button id="btnBerantakan" onclick="modeBerantakan()">Mode Berantakan (hati hati kak/bang)</button>
```

- Tombol kedua memiliki id="btnBerantakan".
- Ketika diklik, tombol memanggil fungsi **modeBerantakan()**, juga dari file script.js.
- Mode ini biasanya mengacak posisi elemen, memutar objek, membuat tampilan kacau atau “berantakan” sering digunakan sebagai variasi lucu atau percobaan efek CSS/JavaScript.
- Catatan “(hati hati kak/bang)” menunjukkan bahwa mode ini sengaja dibuat untuk membuat tampilan benar-benar rusak atau kacau.

Bagian <section> ditutup, lalu tag </main> menandai akhir dari area utama halaman.

Bagian Footer

```
<footer class="footer">
```

```
<button onclick="goBack()">Kembali</button>
```

```
</footer>
```

- Tag <footer> berfungsi sebagai bagian kaki halaman.
- Kelas “footer” biasanya memberikan style seperti warna latar, posisi di bawah, atau penataan teks.
- Di dalam footer terdapat tombol “Kembali” yang menjalankan fungsi **goBack()** saat diklik, kemungkinan untuk kembali ke halaman sebelumnya menggunakan `history.back()` atau `redirect` ke halaman utama.

Memanggil File JavaScript

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

- Tag <script> ini menghubungkan halaman HTML dengan file eksternal JavaScript bernama **script.js**.
- File tersebut berisi fungsi-fungsi seperti:
 - `togglePenjelasan()`
 - `modeRapi()`
 - `modeBerantakan()`
 - `goBack()`
- Semua interaksi, animasi, perubahan mode tampilan, dan penampikan-sembunyikan paragraf dikendalikan melalui file JavaScript ini.

halaman ditutup dengan </body> dan </html>, masing-masing menandai penutupan isi halaman dan keseluruhan dokumen HTML.

SCRIPT.JS

```
1 function togglePenjelasan(id) {  
2     var el = document.getElementById(id);  
3     if (!el) return;  
4  
5     // jika sekarang tersembunyi, tanya untuk menampilkan  
6     var shown = !(el.style.display === 'none' || getComputedStyle(el).display === 'none');  
7     if (shown) {  
8         var t = confirm("Apakah Anda ingin menyembunyikan penjelasan ini?");  
9         if (t) el.style.display = 'none';  
10    } else {  
11        var t = confirm("Apakah Anda ingin menampilkan penjelasan ini?");  
12        if (t) el.style.display = 'block';  
13    }  
14 }
```

202431104_Firja Adyatma Sitepu

Fungsi `togglePenjelasan(id)` adalah sebuah fungsi JavaScript yang digunakan untuk menampilkan atau menyembunyikan paragraf penjelasan berdasarkan ID elemen yang diberikan. Pada baris pertama, fungsi menerima parameter `id`, lalu JavaScript mencari elemen HTML dengan ID tersebut menggunakan `document.getElementById(id)` dan menyimpannya dalam variabel `el`. Jika elemen tidak ditemukan, fungsi langsung dihentikan untuk mencegah error. Setelah itu, fungsi memeriksa apakah elemen tersebut sedang tampil atau tersembunyi dengan mengecek nilai `display` pada style elemen, baik dari style langsung maupun dari hasil perhitungan CSS global. Jika nilai `display`-nya bukan `'none'`, berarti elemen sedang terlihat; sebaliknya jika `'none'`, berarti elemen sedang tersembunyi.

Ketika elemen sedang terlihat, fungsi menampilkan kotak konfirmasi yang bertanya apakah pengguna ingin menyembunyikan penjelasan tersebut. Jika pengguna memilih "OK", maka elemen disembunyikan dengan mengubah `el.style.display` menjadi `'none'`. Jika elemen sedang tersembunyi, fungsi juga memberikan kotak konfirmasi tetapi isinya menanyakan apakah pengguna ingin menampilkan penjelasan tersebut. Bila pengguna menyetujuinya, elemen ditampilkan kembali dengan mengubah nilai `display` menjadi `'block'`. Dengan demikian, fungsi ini memberikan fitur interaktif yang memungkinkan pengguna mengatur sendiri bagian penjelasan mana yang ingin dibuka atau ditutup, lengkap dengan konfirmasi untuk menghindari perubahan yang tidak disengaja.

```
16 // Mode Rapi  
17 function modeRapi() {  
18     var b = document.getElementById('body');  
19     b.classList.remove('berantakan');  
20     b.classList.add('rapi');
```

202431104_Firja Adyatma Sitepu

`modeRapi()` adalah sebuah fungsi JavaScript yang digunakan untuk mengubah tampilan halaman menjadi mode yang lebih rapi dan teratur. Pada baris pertama, fungsi membuat variabel `b` yang mengambil elemen HTML dengan ID `'body'`, yaitu elemen `<body>` pada halaman web. Setelah elemen tersebut berhasil diambil, fungsi mengakses daftar kelas (`classList`) yang dimiliki elemen tersebut. Langkah berikutnya adalah menghapus kelas `'berantakan'` menggunakan `b.classList.remove('berantakan')`, karena kelas ini biasanya berisi aturan CSS yang membuat tampilan halaman menjadi acak, berantakan, atau tidak teratur. Setelah kelas tersebut dihapus, fungsi kemudian menambahkan kelas `'rapi'` melalui `b.classList.add('rapi')`. Kelas `'rapi'` inilah yang

berisi pengaturan tampilan yang lebih bersih, teratur, dan normal sesuai desain halaman. Dengan demikian, fungsi ini bekerja dengan cara menghilangkan efek tampilan berantakan dan menggantinya dengan tampilan rapi hanya dengan satu kali klik.

```
22 // Hapus style inline yang dibuat saat berantakan
23 cleanupChaos();
24 }
25 // Mode Berantakan (EXTREME)
26 function modeBerantakan() {
27     var b = document.getElementById('body');
28     b.classList.remove('rapi');
29     b.classList.add('berantakan');
30
31     // Aktifkan chaos: acak posisi, rotasi, ukuran font, warna
32     unleashChaos();
33 }
```

202431104_Firja Adyatma Sitepu

Potongan kode tersebut berfungsi untuk mengatur tampilan halaman web agar dapat berubah antara mode “rapi” dan mode “berantakan”. Pada bagian pertama, fungsi modeRapi() mengambil elemen <body> melalui document.getElementById('body'), kemudian menghapus kelas berantakan dan menambahkan kelas rapi untuk mengubah tampilan kembali menjadi teratur. Setelah itu, fungsi ini juga memanggil cleanupChaos() yang bertugas menghapus seluruh gaya inline yang sebelumnya ditambahkan pada mode berantakan sehingga tampilan benar-benar kembali normal.

Sementara itu, fungsi modeBerantakan() bekerja dengan cara mengambil elemen <body> yang sama, lalu menghapus kelas rapi dan menambahkan kelas berantakan agar tampilan menjadi kacau. Setelah mengubah kelas, fungsi ini memanggil unleashChaos() yang bertugas mengacak posisi elemen, mengubah rotasi, memperbesar atau memperkecil ukuran font, serta mengubah warna secara acak untuk menciptakan efek tampilan yang benar-benar berantakan dan ekstrem.

```
35 // membersihkan semua efek chaos (kembali normal)
36 function cleanupChaos(){
37     // kembalikan semua elemen ke keadaan semula
38     var all = document.querySelectorAll('.card, h4, .identitas, p, h2, h3, .tombol-row, button');
39     all.forEach(function(el){
40         el.style.position = '';
41         el.style.left = '';
42         el.style.top = '';
43         el.style.transform = '';
44         el.style.fontSize = '';
45         el.style.background = '';
46         el.style.color = '';
47         el.style.zIndex = '';
48         el.style.animation = '';
49     });
50 }
```

202431104_Firja Adyatma Sitepu

Fungsi cleanupChaos() dibuat untuk menghapus seluruh efek kekacauan yang sebelumnya diterapkan oleh mode berantakan, sehingga semua elemen di halaman dapat kembali ke kondisi normal seperti semula. Di dalam fungsi ini, terlebih dahulu dilakukan pencarian terhadap semua elemen yang mungkin terkena efek chaos, yaitu elemen dengan kelas .card, seluruh elemen h4, elemen dengan kelas .identitas, semua paragraf p, judul h2 dan h3, baris tombol .tombol-row, serta

tombol button. Pencarian ini dilakukan menggunakan `document.querySelectorAll()` yang menghasilkan kumpulan elemen dalam bentuk `NodeList`.

Setelah semua elemen tersebut terkumpul, fungsi menjalankan `forEach` untuk memproses masing-masing elemen satu per satu. Pada setiap elemen, seluruh properti `style inline` yang sebelumnya diubah secara acak pada mode berantakan akan dikembalikan menjadi kosong. Properti yang dikosongkan mencakup `position`, `left`, `top`, `transform`, `fontSize`, `background`, `color`, `zIndex`, dan `animation`. Dengan mengosongkan semua nilai ini, setiap elemen tidak lagi memiliki gaya `inline` tambahan sehingga tampilannya kembali mengikuti aturan CSS normal yang digunakan pada mode rapi. Secara keseluruhan, fungsi ini memastikan bahwa halaman dapat pulih sepenuhnya dari efek visual ekstrem yang diterapkan sebelumnya.

```
56 // buat gaya acak untuk membuat layout benar-benar berantakan
57 function unleashChaos(){
58   // 1) ubah container gallery jadi relatif agar kita bisa posisikan anaknya absolut
59   var gallery = document.getElementById('gallery');
60   if (gallery) {
61     gallery.style.position = 'relative';
62     // untuk supaya kartu bisa keluar dari grid, kita juga set width relatif
63     gallery.style.minHeight = '500px';
64   }
```

202431104_Firja Adyatma Sitepu

Fungsi `unleashChaos()` dibuat untuk menciptakan tampilan yang benar-benar berantakan dengan memberikan gaya acak pada elemen-elemen halaman. Pada bagian awal fungsi ini, langkah pertama yang dilakukan adalah mengubah gaya elemen dengan ID `gallery` agar memiliki posisi `relative`. Pengubahan posisi container ke `relative` sangat penting karena memungkinkan elemen-elemen di dalamnya (seperti kartu-kartu gambar atau konten) untuk diposisikan secara absolut ketika kekacauan diterapkan. Selain itu, fungsi ini juga memberikan nilai `minHeight` sebesar `500px` pada `gallery` untuk memastikan ruang tampilan cukup besar sehingga elemen-elemen yang diposisikan secara acak dapat bergerak atau keluar dari susunan grid aslinya tanpa terpotong. Dengan pengaturan ini, mode berantakan dapat bekerja secara ekstrem dengan memberikan posisi, rotasi, atau ukuran acak pada setiap anak elemen di dalam `gallery`.

```

66 // 2) ambil semua kartu, judul, paragraf, tombol dan acak posisinya
67 var cards = document.querySelectorAll('.card');
68 var titles = document.querySelectorAll('h4');
69 var paragraphs = document.querySelectorAll('p');
70 var buttons = document.querySelectorAll('button');
71
72 // fungsi bantu: random integer
73 function rnd(min, max){ return Math.floor(Math.random()*(max-min+1))+min; }
74
75 // acak kartu
76 cards.forEach(function(card, idx){
77     // buat ukuran acak
78     var w = rnd(120, 380);
79     card.style.width = w + 'px';
80
81     // posisikan absolut di dalam gallery (random top/left)
82     card.style.position = 'absolute';
83     // top di area 0..(galleryHeight - 100)
84     var gRect = gallery ? gallery.getBoundingClientRect() : {width:800, height:600};
85     var maxLeft = Math.max(0, (gallery ? gallery.clientWidth : 800) - w);
86     var left = rnd(-50, Math.min(maxLeft + 100, 700));
87     var top = rnd(-60, Math.max(100, (idx * 40)));
88     card.style.left = left + 'px';
89     card.style.top = top + 'px';
90     card.style.transform = 'rotate(' + rnd(-30,30) + 'deg)';
91     card.style.zIndex = 100 + rnd(0,200);
92     card.style.background = 'hsl(' + rnd(0,360) + ', 90%, 75%)';
93     card.style.border = (rnd(2,8) + 'px dashed hsl(' + rnd(0,360) + ',70%,40%)');
94     card.style.animation = 'blink ' + (1 + Math.random()*2).toFixed(2) + 's';
95 });

```

202431104_Firja Adyatma Sitepu

Pada bagian kedua fungsi `unleashChaos()`, kode ini mulai mengambil berbagai elemen di halaman, yaitu semua elemen dengan kelas `.card`, semua judul `<h4>`, semua paragraf `<p>`, dan seluruh tombol `<button>` menggunakan `querySelectorAll`. Setelah elemen-elemen tersebut dikumpulkan, sebuah fungsi pembantu bernama `rnd(min, max)` dibuat untuk menghasilkan angka acak dalam rentang tertentu, yang nantinya dipakai untuk mengacak ukuran, posisi, dan tampilan elemen-elemen tersebut.

Langkah pertama yang diacak adalah elemen-elemen kartu (`.card`). Untuk setiap kartu, kode memberikan ukuran lebar acak antara 120 hingga 380 piksel, sehingga setiap kartu memiliki ukuran yang tidak seragam. Selanjutnya, kartu tersebut diberi `position: absolute` agar bisa digerakkan bebas di dalam kontainer `gallery`. Untuk menentukan batas pergerakan, kode mengambil ukuran kontainer dengan `getBoundingClientRect()`, kemudian menghitung posisi acak untuk nilai `left` dan `top`, sehingga kartu dapat berpindah ke lokasi-lokasi yang tidak wajar, termasuk sedikit keluar dari area normal tampilan.

Selain posisinya, kartu juga diberi rotasi acak menggunakan `transform: rotate()`, sehingga beberapa kartu tampak miring ke kiri atau ke kanan. Nilai `z-index` juga diacak untuk membuat kartu saling tumpang secara acak. Kode kemudian mengubah tampilan visual kartu dengan memberikan warna latar belakang acak menggunakan format HSL serta border garis putus-putus (`dashed`) yang juga memiliki ketebalan dan warna acak. Untuk memberikan efek yang lebih kacau, setiap kartu diberi animasi berkedip menggunakan keyframe `blink` dengan durasi acak antara satu hingga sekitar tiga detik. Semua pengaturan ini ditujukan untuk menciptakan mode berantakan yang ekstrem, sehingga tata letak kartu benar-benar kacau, tidak rata, dan tampak seperti "meledak" ke berbagai arah secara spontan.

```

97 // acak judul (h4)
98 titles.forEach(function(t){
99   t.style.position = 'relative';
100   t.style.left = (rnd(-40,40)) + 'px';
101   t.style.transform = 'rotate(' + rnd(-20,20) + 'deg)';
102   t.style.fontSize = (rnd(14,30)) + 'px';
103   t.style.color = 'hsl(' + rnd(0,360) + ', 80%, 30%)';
104   t.style.background = 'hsl(' + rnd(0,360) + ', 80%, 85%)';
105   t.style.zIndex = 500;
106   t.style.animation = 'blink ' + (1 + Math.random()*1.5).toFixed(2) + 's infinite';
107 });
108
109 // acak paragraf
110 paragraphs.forEach(function(p){
111   p.style.color = 'hsl(' + rnd(0,360) + ', 80%, 20%)';
112   p.style.fontSize = (rnd(12,20)) + 'px';
113   p.style.left = (rnd(-20,20)) + 'px';
114   p.style.position = 'relative';
115   p.style.zIndex = 200;
116   p.style.animation = 'blink ' + (1.5 + Math.random()*2).toFixed(2) + 's infinite';
117 });
118
119 // acak tombol (warna & sedikit gerak)
120 buttons.forEach(function(btn){
121   btn.style.background = 'hsl(' + rnd(0,360) + ', 70%, 40%)';
122   btn.style.color = 'hsl(' + rnd(0,360) + ', 90%, 95%)';
123   btn.style.transform = 'rotate(' + rnd(-12,12) + 'deg)';
124   btn.style.zIndex = 600;
125   btn.style.animation = 'blink ' + (0.6 + Math.random()*1.4).toFixed(2) + 's infinite';
126 });

```

202431104_Firja Adyatma Sitepu

Potongan kode ini merupakan lanjutan dari fungsi `unleashChaos()`, yaitu bagian yang mengacak tampilan elemen judul, paragraf, dan tombol agar seluruh halaman benar-benar terlihat kacau saat mode berantakan diaktifkan. Untuk setiap elemen judul `<h4>`, kode memberikan posisi relatif sehingga elemen masih mengikuti alur dokumen tetapi dapat digeser sedikit menggunakan properti `left`. Nilai `left` diacak antara -40 hingga 40 piksel, membuat judul tampak sedikit bergeser ke kiri atau ke kanan secara acak. Selain itu, judul juga diberi rotasi acak antara -20 hingga 20 derajat, sehingga tampak miring dan tidak teratur. Ukuran teksnya pun diacak antara 14 sampai 30 piksel, sehingga ada judul yang tampak besar dan ada yang tampak lebih kecil. Kode kemudian memberikan warna teks dan warna latar belakang secara acak menggunakan format warna HSL, sehingga setiap judul memiliki kombinasi warna yang tidak seragam. Untuk memberikan efek visual yang lebih hidup, setiap judul diberi animasi berkedip dengan durasi acak dan ditempatkan pada lapisan lebih atas melalui `zIndex` 500.

Pada bagian pengacakan paragraf, fungsi melakukan hal serupa namun lebih ringan. Setiap paragraf diberi warna teks acak dalam format HSL dan ukuran font yang juga berubah-ubah antara 12 hingga 20 piksel, sehingga teks paragraf tampak berbeda satu sama lain. Paragraf juga digeser secara acak ke kiri atau kanan menggunakan properti `left` dengan rentang -20 hingga 20 piksel. Seperti judul, paragraf diberi `position: relative` agar dapat digeser tanpa meninggalkan alur dokumen. Paragraf ditempatkan pada lapisan `z-index` 200 dan diberi animasi berkedip dengan durasi acak antara 1.5 hingga beberapa detik, membuat teks terlihat bergetar atau menghilang secara berkala.

Sementara itu, elemen tombol juga ikut diacak untuk memperkuat kesan berantakan. Setiap tombol diberi latar belakang acak dalam format HSL dan warna teks acak yang sangat cerah agar tetap terbaca. Tombol-tombol tersebut juga diputar sedikit dengan rotasi acak antara -12 hingga 12 derajat, membuatnya tampak seperti ditempel sembarangan. Setiap tombol ditempatkan pada

lapisan paling depan (zIndex 600) sehingga tidak tertutupi elemen lain. Selain itu, tombol diberi animasi berkedip dengan kecepatan yang lebih cepat, karena durasinya berada di antara 0.6 hingga sekitar 2 detik, sehingga mereka bergerak atau muncul–hilang secara dinamis.

Secara keseluruhan, seluruh pengaturan ini bertujuan menciptakan efek "kacau total", di mana judul, paragraf, dan tombol tampil dalam posisi, ukuran, warna, rotasi, serta animasi yang berbeda-beda, membuat halaman tampak benar-benar berantakan dan tidak beraturan sebagaimana diminta untuk mode ekstrem.

```
128 // Scroll ke atas supaya efek berantakan terlihat langsung
129 window.scrollTo({top:0, behavior:'smooth'});
130 }
131
132 // Tombol kembali (footer)
133 function goBack(){
134     if (document.referrer) {
135         window.history.back();
136     } else {
137         window.location.href = 'index.html';
138     }
139 }
140
141 // Inisialisasi: tampilkan semua penjelasan awalnya
142 window.addEventListener('DOMContentLoaded', function(){
143     for (var i=1;i<=7;i++){
144         var el = document.getElementById('p'+i);
145         if (el) el.style.display = 'block';
146     }
147 });
148
```

202431104_Firja Adyatma Sitepu

Potongan kode pertama berfungsi untuk menggulirkan halaman secara otomatis ke bagian paling atas ketika mode berantakan diaktifkan. Perintah `window.scrollTo({top:0, behavior:'smooth'})`; menginstruksikan browser agar memindahkan posisi layar ke koordinat vertikal nol, yaitu puncak halaman, dengan efek animasi yang halus. Hal ini dilakukan agar pengguna langsung dapat melihat perubahan drastis yang terjadi saat mode berantakan mulai diterapkan, sehingga efek visual kacau yang dihasilkan terlihat secara langsung tanpa perlu menggulir manual.

Selanjutnya terdapat fungsi `goBack()` yang digunakan untuk tombol “Kembali” pada bagian footer. Fungsi ini mengecek apakah halaman sebelumnya dapat diakses melalui `document.referrer`, yaitu informasi tentang dari mana pengguna datang sebelum membuka halaman saat ini. Jika nilai tersebut ada, maka kode akan menjalankan `window.history.back()`, yaitu perintah untuk kembali satu langkah ke halaman sebelumnya. Namun, apabila tidak ada data `referrer`—misalnya halaman dibuka secara langsung tanpa berasal dari halaman lain—maka fungsi ini akan mengarahkan pengguna ke halaman `'index.html'` sebagai alternatif. Dengan cara ini, tombol kembali tetap berfungsi dalam dua kondisi berbeda.

Bagian terakhir adalah proses inisialisasi yang dijalankan ketika halaman selesai dimuat. Pada event DOMContentLoaded, kode mengeksekusi sebuah fungsi yang melakukan loop dari angka 1 hingga 7. Pada setiap iterasi, ia mencari elemen paragraf dengan ID seperti p1, p2, p3, hingga p7 menggunakan document.getElementById('p'+i). Jika elemen tersebut ditemukan, maka elemen akan diatur agar tampil dengan mengubah propertinya menjadi el.style.display = 'block'. Tujuan dari mekanisme ini adalah agar semua penjelasan dalam daftar topik langsung terlihat sejak awal ketika halaman dibuka, tanpa harus diklik atau dibuka secara manual satu per satu. Dengan demikian, pengguna dapat langsung membaca seluruh isi materi tanpa interaksi tambahan, dan halaman juga berada dalam kondisi siap sebelum fitur sembunyi–tampilkan digunakan.

STYLE.CSS

```

1  /* WARNA TEMA HIJAU (laki-laki NIM genap) */
2  :root{
3      --hijau-muda: #e9f8ee;
4      --hijau-tua: #146914;
5      --kontras-bad: #ffff00;
6  }
7
8  *{box-sizing:border-box;}
9  body{
10     margin:0;
11     font-family: Arial, sans-serif;
12     background: var(--hijau-muda);
13     color: var(--hijau-tua);
14     line-height:1.5;
15 }
16
17 /* HEADER */
18 .header{ padding:12px; background:#dfff5d; border-bottom:1px solid #bfe6c1; }
19 .identitas{ text-align:center; font-weight:600; }
20
21 /* CONTAINER */
22 .container{ max-width:1000px; margin:18px auto; padding:0 16px; }
23
24 /* TYPO */
25 h2{ margin-top:0; }
26 .petunjuk{ font-weight:400; color:#444; font-size:0.95rem; }
27
28 /* Penjelasan */
29 .penjelasan{ display:block; margin:8px 0 18px 0; }
30

```

202431104_Firja Adyatma Sitepu

Bagian kode ini mengatur gaya tampilan halaman web dengan tema warna hijau sesuai ketentuan, yaitu untuk pengguna berjenis kelamin laki-laki dengan NIM genap. Pada bagian paling atas terdapat deklarasi :root yang berfungsi sebagai tempat mendefinisikan variabel CSS global. Tiga variabel warna didefinisikan, yaitu --hijau-muda untuk warna hijau muda yang lembut, --hijau-tua untuk warna hijau tua yang lebih tegas dan dipakai sebagai warna teks utama, serta --kontras-bad yang merupakan warna kuning terang sebagai warna kontras khusus jika dibutuhkan. Dengan menggunakan variabel ini, warna–warna tertentu dapat dipanggil kembali di berbagai bagian stylesheet dengan mudah dan konsisten.

Seluruh elemen dalam halaman diatur menggunakan aturan `*{ box-sizing: border-box; }` yang memastikan bahwa perhitungan lebar dan tinggi elemen mencakup padding dan border, sehingga pengaturan layout menjadi lebih stabil dan mudah dikendalikan. Pada bagian pengaturan untuk body, margin default dihapus agar tampilan halaman dimulai dari sisi paling pinggir layar. Font utama ditetapkan menggunakan Arial atau font sans-serif lainnya jika Arial tidak tersedia. Latar belakang halaman menggunakan warna hijau muda dari variabel `--hijau-muda`, sedangkan warna teks utama menggunakan hijau tua dari variabel `--hijau-tua`. Selain itu, jarak antarbaris ditingkatkan dengan `line-height: 1.5` agar teks lebih mudah dibaca dan terlihat lebih rapi.

Bagian berikutnya mengatur tampilan header, yang diberi padding agar tidak terlalu menempel pada tepian, sementara warna latar belakangnya menggunakan hijau muda yang sedikit lebih pekat (`#dff5dd`). Di bawahnya terdapat garis pembatas berwarna hijau terang (`#bfe6c1`) untuk memberi batas visual antara header dan konten utama. Elemen `.identitas` di dalam header dibuat rata tengah dan diberi gaya huruf tebal untuk menonjolkan informasi identitas seperti nama dan NIM.

Selanjutnya, kelas `.container` digunakan untuk membatasi lebar area konten agar tidak terlalu melebar pada layar besar. Lebarnya dibatasi maksimum 1000 piksel dan ditempatkan di tengah halaman dengan margin otomatis. Di bagian kiri dan kanan kontainer ditambahkan padding agar konten tidak terlalu menempel pada tepi layar. Pada aturan tipografi, tag `<h2>` diatur agar tidak memiliki jarak atas tambahan sehingga judul section tampak lebih melekat pada bagian atas kontainer. Kelas `.petunjuk` dipakai untuk teks petunjuk yang tampil di bawah judul topik; tampilannya dibuat lebih ringan dengan bobot huruf lebih kecil, warna abu-abu gelap, serta ukuran font sedikit lebih kecil dari teks normal untuk membedakannya dari teks utama.

Kelas `.penjelasan` digunakan untuk paragraf penjelasan dalam setiap topik pencemaran. Paragraf-paragraf ini secara default ditampilkan (`display: block`) dan diberi jarak atas dan bawah yang cukup agar tidak terlalu berhimpitan dengan elemen lain. Tujuannya adalah membuat setiap penjelasan tampak rapi, jelas, dan enak dibaca dalam struktur halaman. Dengan keseluruhan pengaturan ini, halaman web terlihat bersih, konsisten dengan tema warna hijau, dan nyaman bagi pengguna.


```
31  /* Judul-topik */
32  h4{
33      margin:10px 0 6px;
34      padding:8px;
35      background: #f3fff3;
36      border-radius:4px;
37      cursor:pointer;
38      transition: transform .2s ease;
39  }
40
41  /* Tombol */
42  button{
43      padding:8px 12px;
44      border-radius:6px;
45      border:1px solid var(--hijau-tua);
46      background: var(--hijau-tua);
47      color:white;
48      cursor:pointer;
49  }
50  button:hover{ opacity:0.95; }
51
52  .tombol-row{ margin:10px 0; }
53
54  /* GALLERY CARDS */
55  .card-container{
56      display:grid;
57      gap:12px;
58      margin-top:12px;
59  }
60
```

202431104_Firja Adyatma Sitepu

Bagian kode ini berfungsi untuk mempercantik tampilan komponen antarmuka pada halaman web, terutama judul topik, tombol, serta area galeri kartu. Pada aturan pertama, elemen h4 yang digunakan sebagai judul topik diberikan margin atas dan bawah agar memiliki jarak yang cukup dari elemen lainnya, serta padding agar teks di dalamnya terasa lebih lega. Latar belakangnya menggunakan warna hijau sangat muda (#f3fff3), sehingga tampilannya selaras dengan tema hijau keseluruhan halaman. Sudut elemen dibuat sedikit melengkung menggunakan border-radius: 4px sehingga tidak terlihat kaku. Selain itu, pointer diubah menjadi “tangan” (cursor: pointer) untuk memberikan tanda bahwa judul tersebut dapat diklik. Efek transisi juga ditambahkan pada properti transformasi, sehingga ketika judul mengalami perubahan bentuk, efek tersebut muncul secara halus dan tidak tiba-tiba.

Bagian berikutnya mengatur tampilan tombol dengan desain yang sederhana namun serasi. Tombol diberi padding yang cukup agar terasa nyaman dilihat dan mudah diklik, dengan sudut membulat yang lebih besar menggunakan border-radius: 6px. Garis pinggir tombol menggunakan warna hijau tua dari variabel tema, dan latar belakang tombol juga dibuat serupa agar menghasilkan tampilan yang konsisten. Warna teks pada tombol dibuat putih agar kontras dan mudah terbaca. Sama seperti judul, tombol juga menggunakan cursor: pointer untuk menunjukkan bahwa tombol dapat berinteraksi. Ketika pointer diarahkan ke tombol, efek hover membuat tombol sedikit memudar (opacity: 0.95), memberikan kesan responsif saat disentuh. Kelas .tombol-row hanya berfungsi memberikan jarak vertikal agar tombol tidak menempel pada elemen lain di sekitarnya.

Untuk bagian galeri, kelas `.card-container` digunakan sebagai wadah kartu-kartu gambar atau informasi. Wadah ini diatur menggunakan sistem grid (`display: grid`) agar setiap kartu dapat ditampilkan dalam susunan kotak yang rapi dan teratur. Ruang antar kartu diatur menggunakan `gap: 12px`, sehingga setiap elemen dalam grid memiliki jarak yang konsisten dan tidak terlalu rapat. Margin atas ditambahkan agar area galeri tidak langsung menempel pada elemen di atasnya. Secara keseluruhan, kombinasi aturan-aturan ini membuat tampilan judul, tombol, dan galeri terlihat modern, bersih, serta mudah digunakan dalam halaman web bertema pencemaran lingkungan.

```

61  /* CARD */
62  .card{
63      background: #fff;
64      border:1px solid #ddd;
65      padding:6px;
66      border-radius:6px;
67      text-align:center;
68      transition: transform .25s ease;
69  }
70  .card img{ width:100%; height:auto; display:block; border-radius:4px; }
71
72  /* FOOTER */
73  .footer{ text-align:center; padding:18px; }
74
75  /* ----- MODE: RAPI (default) ----- */
76  .rapi h4{ box-shadow:none; }
77  .rapi .card{ transform:none; position:static; left:auto; top:auto; }
78
79  /* ----- MODE: BERANTAKAN (GILA) -----
80  Selain CSS ini, JS akan menambahkan inline styles acak (rotate, top/left, font-size, warna).
81  Di CSS kita set beberapa gaya dasar agar terlihat 'berantakan' juga.
82  */
83  .berantakan{
84      background: var(--kontras-bad) !important;
85      color: #ff0000 !important;
86  }

```

202431104_Firja Adyatma Sitepu

Potongan kode ini mengatur tampilan kartu, footer, serta dua mode tampilan utama pada halaman, yaitu mode rapi dan mode berantakan. Kelas `.card` digunakan untuk mengatur tampilan setiap kartu pada galeri. Setiap kartu diberi latar belakang putih agar gambar dan teks di dalamnya terlihat jelas. Batas luar dibuat menggunakan garis abu-abu tipis (`border: 1px solid #ddd`), sementara padding ditambahkan agar konten di dalamnya tidak terlalu menempel pada tepi kartu. Sudut kartu dibuat membulat dengan `border-radius: 6px`, menjadikan tampilannya lebih modern dan lembut. Teks dalam kartu diatur agar berada di tengah dengan `text-align: center`. Transisi pada properti transformasi ditambahkan untuk memberikan efek halus saat kartu mengalami perubahan posisi atau rotasi—efek yang dimanfaatkan ketika mode berantakan aktif. Pada bagian gambar, aturan `.card img` memastikan gambar memenuhi lebar kartu secara penuh dengan menjaga proporsi tinggi secara otomatis. Gambar juga ditampilkan sebagai blok dan diberikan sudut melengkung agar selaras dengan bentuk kartu.

Selanjutnya, kelas `.footer` berfungsi mengatur tampilan area footer. Teks pada footer ditempatkan di tengah halaman, dan diberi padding yang cukup agar tombol kembali tidak terlihat terlalu dekat dengan pinggir bawah halaman. Bagian ini sederhana, namun penting untuk menjaga kenyamanan navigasi pengguna.

Setelah itu terdapat pengaturan khusus untuk mode tampilan rapi. Kelas `.rapi h4` menghilangkan bayangan pada judul-judul topik, sehingga tampilannya lebih bersih dan teratur. Pada kartu, `.rapi`

.card memastikan kartu tetap berada dalam posisi normal tanpa transformasi apa pun, yaitu tidak berpindah, tidak miring, dan tidak bergeser. Properti position: static memastikan kartu mengikuti alur layout normal grid tanpa manipulasi posisi absolut, sementara left dan top dikembalikan ke nilai default agar tidak ada pergeseran akibat mode berantakan.

Bagian terakhir adalah konfigurasi untuk mode berantakan. Dalam mode ini, selain JavaScript yang menambahkan gaya acak seperti rotasi, perubahan warna, pergeseran posisi, dan ukuran teks, CSS dasar juga ikut mendukung tampilan kacau tersebut. Kelas .berantakan memaksa warna latar belakang berubah menjadi kuning mencolok menggunakan variabel --kontras-bad, sementara warna teks diubah menjadi merah terang. Keduanya menggunakan !important untuk memastikan perubahan ini tetap berlaku meskipun ada gaya lain yang berpotensi menyimpannya. Dengan kombinasi ini, mode berantakan tidak hanya acak secara posisi dan transformasi, tetapi juga sangat mencolok secara visual, sehingga memberikan kontras yang ekstrem dari mode rapi.

```

87
88 /* beberapa perubahan visual 'berantakan' */
89 .berantakan h4{
90     background: #fff200;
91     transform: rotate(-2deg);
92     box-shadow: 6px 6px 0 #000;
93 }
94 .berantakan .card{
95     background: #ffecb3;
96     border: 3px dashed #ff00aa;
97     transform: rotate(3deg);
98 }
99
100 /* ---- RWD: 1 / 2 / 3 kolom ---- */
101 @media (max-width:600px){
102     .card-container{ grid-template-columns: 1fr; }
103 }
104 @media (min-width:601px) and (max-width:992px){
105     .card-container{ grid-template-columns: 1fr 1fr; }
106 }
107 @media (min-width:993px){
108     .card-container{ grid-template-columns: 1fr 1fr 1fr; }
109 }
110
111 /* tambahan: animasi berkedip & kontras untuk mode berantakan */
112 @keyframes blink {
113     0%{opacity:1}
114     50%{opacity:0.2}
115     100%{opacity:1}
116 }
117 .berantakan *{
118     animation: blink 2s infinite;
119 }
120

```

202431104_Firja Adyatma Sitepu

Kode ini berfungsi untuk mengatur efek visual tambahan ketika mode berantakan diaktifkan serta mengatur responsivitas jumlah kolom pada galeri kartu. Pada bagian pertama, aturan .berantakan h4 memberikan tampilan yang lebih acak pada setiap judul topik dengan mengubah warna latarnya menjadi kuning mencolok, memiringkan sedikit judul tersebut menggunakan transform: rotate(-2deg), dan menambahkan bayangan besar untuk menegaskan kesan tidak rapi. Sementara itu, aturan .berantakan .card mengubah penampilan setiap kartu di dalam galeri. Kartu tersebut diberi latar belakang berwarna krem terang, garis batas diganti menjadi garis putus-putus berwarna magenta

dengan ketebalan besar, serta diputar sedikit sehingga tampak miring. Semua perubahan ini membantu memperkuat nuansa kacau yang diinginkan pada mode berantakan.

Selanjutnya, bagian aturan media atau *responsive web design* (RWD) mengatur berapa banyak kolom yang muncul pada kontainer kartu bergantung pada ukuran layar. Pada layar kecil dengan lebar maksimum 600 piksel, kontainer kartu hanya memiliki satu kolom sehingga setiap kartu muncul bertumpuk dari atas ke bawah dan mudah dibaca pada perangkat ponsel. Ketika ukuran layar berada di antara 601 hingga 992 piksel, kontainer akan menampilkan dua kolom agar tata letak lebih optimal untuk tablet atau layar menengah. Jika lebar layar lebih dari 993 piksel, kontainer menampilkan tiga kolom, menjadikannya cocok untuk tampilan desktop yang luas. Pengaturan ini memastikan tata letak galeri tetap proporsional dan nyaman digunakan pada berbagai perangkat.

Bagian terakhir adalah penambahan animasi berkedip dan peningkatan kontras untuk memperkuat suasana berantakan. Animasi `@keyframes blink` mengatur perubahan tingkat opasitas elemen dari terlihat penuh, kemudian memudar hingga sangat transparan, lalu kembali muncul. Siklus ini berlangsung terus-menerus sehingga menciptakan efek berkedip. Aturan `.berantakan *` kemudian menerapkan animasi ini ke semua elemen ketika mode berantakan aktif. Hal ini menyebabkan setiap elemen pada halaman berkedip secara berulang, yang menambah kesan ekstrem dan tidak stabil pada mode berantakan. Dengan kombinasi warna mencolok, rotasi acak, dan animasi berkedip, seluruh tampilan halaman menjadi benar-benar kacau sesuai tujuan fitur tersebut.

GAMBAR.HTML

```
1 <!DOCTYPE html>
2 <html lang="id">
3 <head>
4   <meta charset="utf-8" />
5   <meta name="viewport" content="width=device-width,initial-scale=1" />
6   <title>Galeri - Contoh Pencemaran</title>
7   <link rel="stylesheet" href="style.css" />
8 </head>
9 <body id="body" class="tema-hijau rapi">
10  <header class="header">
11    <div class="identitas">
12      <div><strong>Nama:</strong> Firja Adyatma Sitepu</div>
13      <div><strong>NIM:</strong> 202431104</div>
14    </div>
15  </header>
```

Bagian `<!DOCTYPE html>` digunakan untuk memberi tahu browser bahwa dokumen ini memakai standar HTML5. Tanpa deklarasi ini, browser bisa membaca halaman dengan mode lama sehingga tampilan atau fitur tertentu dapat bermasalah.

Tag `<html lang="id">` adalah tag paling luar yang membungkus seluruh isi website, dan atribut `lang="id"` menunjukkan bahwa bahasa utama halaman ini adalah Bahasa Indonesia. Ini membantu mesin pencari, pembaca layar, dan beberapa fitur browser lain.

Di dalam tag `<head>` terdapat informasi yang tidak tampil langsung di layar, tetapi sangat penting untuk konfigurasi halaman. `meta charset="utf-8"` memastikan semua karakter, termasuk huruf

dengan simbol atau aksara non-Latin, dapat ditampilkan dengan benar. Lalu meta name="viewport" content="width=device-width,initial-scale=1" berfungsi supaya website tampil responsif di HP. Tanpa ini, tampilan bisa mengecil atau tidak proporsional saat dibuka di layar kecil.

Tag <title> memberi judul pada tab browser. Judul ini muncul di bagian atas jendela atau tab, bukan di dalam halaman. Di sini judulnya adalah "Galeri - Contoh Pencemaran".

Setelah itu, link rel="stylesheet" href="style.css" digunakan untuk menghubungkan file HTML dengan file CSS eksternal. CSS inilah yang berfungsi mengatur warna, layout, ukuran teks, dan desain visual halaman. Jadi HTML memberi struktur, CSS memberi gaya.

Bagian <body id="body" class="tema-hijau rapi"> adalah tempat seluruh konten yang terlihat oleh pengguna. Atribut id="body" digunakan untuk membantu JavaScript menarget elemen body. Sedangkan class="tema-hijau rapi" memberikan dua kelas default pada body: satu untuk tema warna hijau, dan satu lagi untuk mode tampilan rapi. Nantinya kelas ini bisa diubah menggunakan tombol mode gila/berantakan.

Lalu <header class="header"> menandai bagian atas halaman yang biasanya berisi judul, logo, atau identitas. Di dalamnya terdapat <div class="identitas"> yang memuat data pribadi pengguna, yaitu nama dan NIM. Dua <div> di dalamnya berisi tulisan "Nama: Firja Adyatma Sitepu" dan "NIM: 202431104". Penulisan digunakan agar teks "Nama" dan "NIM" tampil lebih tebal sebagai penekanan.

```
17 <main class="container">
18   <h2>Galeri - Contoh Pencemaran (7 gambar)</h2>
19   <p>Pada halaman ini setiap gambar mewakili salah satu masalah pencemaran. Gambar bisa diunduh jika diperlukan.</p>
20
21   <div class="card-container" id="gallery">
22     <!-- Gambar: beberapa contoh menggunakan URL online, dan satu contoh memakai file unggahan lokal -->
23     <div class="card"><p>Pencemaran Udara</p></div>
24     <div class="card"><p>Pencemaran Air</p></div>
25     <div class="card"><p>Pencemaran Tanah</p></div>
26     <div class="card"><p>Kebisingan (ilustrasi)</p></div>
27     <div class="card"><p>Sampah Plastik</p></div>
28     <div class="card"><p>Limbah Industri</p></div>
29     <div class="card"><p>Emisi Kendaraan</p></div>
30   </div>
31
32   <div class="tombol-row">
33     <button onclick="location.href='index.html'">Kembali ke Halaman Utama</button>
34   </div>
35 </main>
36
37 <script src="script.js"></script>
38 </body>
39 </html>
40
```

Bagian <main class="container"> adalah area utama halaman yang berisi isi konten utama yang ingin ditampilkan kepada pengguna. Kelas container membuat lebar konten menjadi lebih rapi dan tidak terlalu menempel ke tepi layar.

Judul `<h2>Galeri - Contoh Pencemaran (7 gambar)</h2>` menjelaskan bahwa halaman tersebut adalah galeri yang memuat tujuh gambar yang masing-masing menggambarkan jenis pencemaran. Di bawahnya ada paragraf `<p>` yang memberi penjelasan bahwa setiap gambar adalah contoh masalah pencemaran dan bisa diunduh jika diperlukan.

Elemen `<div class="card-container" id="gallery">` adalah wadah untuk menampung semua kartu gambar. CSS sebelumnya sudah mengatur bahwa `card-container` ini akan menggunakan grid layout, sehingga gambar-gambar tertata otomatis dalam satu, dua, atau tiga kolom tergantung lebar layar.

Di dalamnya terdapat tujuh `<div class="card">`. Setiap card merepresentasikan satu contoh pencemaran. Di dalam setiap card terdapat tag `` untuk menampilkan gambar dan tag `<p>` untuk menampilkan judul atau label gambar.

Semua gambar diambil dari folder lokal `./assets/`, misalnya:

```

```

Atribut `src` menentukan lokasi file, sedangkan `alt` memberi teks alternatif apabila gambar gagal dimuat. Teks dalam tag `<p>` menjelaskan jenis pencemaran yang ditampilkan.

Setelah galeri gambar, ada `<div class="tombol-row">` yang berisi satu tombol. Tombol tersebut memakai tag `<button>` dan ketika diklik akan menjalankan perintah JavaScript sederhana melalui `onclick`. Perintah tersebut mengarahkan pengguna kembali ke `index.html`, yaitu halaman utama aplikasi.

Akhir halaman ditutup dengan `<script src="script.js"></script>` yang digunakan untuk memuat file JavaScript eksternal. File ini biasanya berisi fungsi-fungsi tambahan seperti mengubah mode tampilan antara rapi dan berantakan, atau efek lain yang bekerja saat halaman berjalan.

Terakhir, tag `</body>` dan `</html>` menutup struktur halaman. Bagian ini menandakan bahwa tidak ada lagi konten setelahnya.