Laporan Pembuatan Program Perancangan

Nama: Anthony Tan

Nim: 241130802

Matkul: Perancangan Web

Program HTML dan JavaScript di atas merupakan halaman dashboard utama untuk aplikasi e-commerce bernama Vea Commerce. Halaman ini dirancang sebagai pusat kendali pengguna untuk melihat daftar produk berdasarkan kategori, mengakses pesanan, dan mengelola keranjang belanja. Tampilan dashboard dibuat interaktif dan responsif, dengan sidebar di sebelah kiri, navigasi antarseksi, fitur keranjang belanja di kanan atas, dan sistem login/logout di pojok kanan atas.

Penjelasan Fungsional dan Bukti Kodenya

1. Struktur dan Navigasi Dashboard

Dashboard ini memiliki sidebar statis yang berisi navigasi ke berbagai bagian seperti Dashboard, Kategori Produk, dan Pesanan. Setiap bagian ditampilkan menggunakan sistem tab dinamis berbasis JavaScript.

```
<div class="nav-link active" onclick="navigate('dashboard')">📊 Dashboard</div>
<div id="dashboard" class="section active">...</div>
<div id="pesanan" class="section">...</div>
```

```
function navigate(id) {
  document.querySelectorAll('.section').forEach(section => section.classList.remove('active'));
  document.getElementById(id).classList.add('active');
}
```

Penjelasan: Fungsi navigate() menyembunyikan semua konten dan hanya menampilkan bagian yang dipilih. Kelas active digunakan untuk mengontrol tampilan aktif antar halaman seperti "Dashboard" atau "Pesanan".

2. Sistem Kategori Produk (Baju, Sepatu, Tas, Kosmetik, Jam)

Produk dikelompokkan berdasarkan kategori dan ditampilkan menggunakan elemen .card, lengkap dengan gambar, nama, harga, dan stok. Setiap produk diorganisasi dalam blok category-block.

```
<h3 class="category">👕 Baju</h3>
<div class="card">
  <div class="product-image">
    <img src="./KATEGORI BAJU/PRIA/1.jpg" alt="Kemeja Pria">
  </div>
  <h3>Kemeja Pria</h3>
  <p>Rp 400.000</p>
  <p>Stok: 35</p>
</div>
```

Penjelasan: Setiap kategori memiliki kumpulan kartu produk yang ditampilkan secara fleksibel dalam layout .card-container. CSS memberikan efek hover dan transisi untuk meningkatkan tampilan visual.

3. Fitur Keranjang Belanja (Cart)

Keranjang belanja ditampilkan sebagai ikon tetap (sticky) di kanan atas. Jumlah item ditampilkan menggunakan .cart-count, dan isinya disimpan di localStorage untuk persistensi.

```
<div class="cart-icon" onclick="navigate('pesanan')">🛒
  <span class="cart-count" id="cart-count">0</span>
</div>
```

```
let cart = JSON.parse(localStorage.getItem('cart')) || [];
function updateCartCount() {
  const total = cart.reduce((sum, item) => sum + item.quantity, 0);
  document.getElementById('cart-count').textContent = total;
}
```

Penjelasan: Saat halaman dimuat, data keranjang dibaca dari localStorage, lalu ditampilkan jumlah itemnya. Setiap perubahan (tambah/kurang/hapus) otomatis memperbarui tampilan.

4. Fitur Checkout dan Penyimpanan Pesanan

Saat checkout, sistem akan menyimpan isi keranjang sebagai "pesanan" dan mengalihkan ke halaman Pesanan.html.

```
function checkout() {
  if (cart.length === 0) {
      alert('Keranjang belanja kosong!');
      return;
  }
  localStorage.setItem('orderItems', JSON.stringify(cart));
  window.location.href = 'Pesanan.html';
  cart = [];
  saveCart();
  updateCartCount();
}
```

Penjelasan: Fitur checkout akan menyimpan data pesanan ke localStorage sebelum keranjang dikosongkan dan pengguna diarahkan ke halaman baru.

5. Fungsi Render dan Kelola Item dalam Keranjang

Fitur ini memungkinkan pengguna menambah, mengurangi, dan menghapus item dari keranjang langsung di tampilan.

```
function renderCartItems() {
  ...
  <button onclick="updateQuantity(${index}, -1)">-</button>
  <span>${item.quantity}</span>
  <button onclick="updateQuantity(${index}, 1)">+</button>
  <button onclick="removeFromCart(${index})">Hapus</button>
  ...
}
```

Penjelasan: Tombol interaktif ini memanggil fungsi updateQuantity() dan removeFromCart() untuk memanipulasi kuantitas item di keranjang.

6. User Display dan Tombol Login/Logout

Di pojok kanan atas terdapat tampilan username dan tombol login/logout yang bisa dimanfaatkan untuk interaksi pengguna.

```
<div class="user-controls">
  <span id="username-display"></span>
  <button id="login-logout-btn">Login</button>
</div>
```

Penjelasan: Area ini siap untuk ditambahkan logika login pengguna dan bisa dihubungkan dengan halaman login sebelumnya.

Program ini membuktikan struktur dashboard e-commerce interaktif dan modular. Fitur utama seperti navigasi dinamis, keranjang belanja persistensi, kategori produk, serta proses checkout semuanya dikelola di sisi klien menggunakan JavaScript murni dan localStorage. Meski belum menggunakan backend atau database sungguhan, struktur program ini sudah sangat kuat sebagai fondasi aplikasi e-commerce sederhana.

Laporan Pembuatan Program Perancangan

Nama: Calvin

Nim: 241130091

Matkul: Perancangan Web

STRUKTUR HTML

Program di atas merupakan halaman web katalog sepatu pria dan wanita yang menampilkan produk secara rapi dan interaktif. Setiap produk ditampilkan dalam bentuk kartu (card) yang berisi gambar sepatu, nama produk, harga, dan tombol “Tambah ke Keranjang”. Saat pengguna mengklik tombol tersebut, produk akan ditambahkan ke keranjang belanja yang tersimpan di localStorage browser, memungkinkan data tetap tersimpan walaupun halaman dimuat ulang. Selain itu, tombol akan berubah menjadi "Ditambahkan ✓" selama dua detik sebagai umpan balik visual kepada pengguna.

Desain antarmuka halaman ini responsif, menggunakan grid layout CSS agar dapat menyesuaikan dengan ukuran layar, termasuk mobile. Warna utama halaman adalah oranye (#ee4d2d), yang digunakan untuk elemen seperti tombol dan judul. Halaman ini juga memiliki navigasi ke dashboard utama dan footer sederhana di bagian bawah. JavaScript pada halaman digunakan untuk menangani penambahan produk ke keranjang serta perubahan visual pada tombol pembelian.

Bukti-bukti dari Kode

1. Tampilan Produk Katalog

Produk ditampilkan dalam elemen .product-card, berisi gambar, nama produk, dan harga.

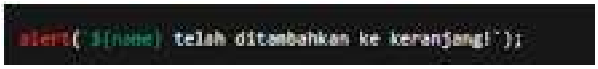
2. Tombol Tambah ke Keranjang Interaktif

Tombol ini memanggil fungsi addToCart() dengan data produk yang relevan.

3. Fungsi Menambahkan Produk ke Keranjang

Produk disimpan di localStorage sebagai keranjang belanja. Jika sudah ada, kuantitasnya ditambah.

4. Tampilan Notifikasi Setelah Tambah



Menampilkan pesan pop-up bahwa produk berhasil ditambahkan.

5. Efek Visual pada Tombol Setelah Klik

Umpan balik visual membuat tombol berubah untuk sementara selama 2 detik.

6. Automatisasi Tombol dari Elemen HTML

JavaScript ini secara otomatis mengambil informasi dari elemen HTML produk dan mengatur tombol agar memanggil fungsi addToCart() sesuai datanya.

7. Desain Responsif dan Rapi (Grid Layout)

Produk ditampilkan dalam grid layout responsif, otomatis menyesuaikan jumlah kolom sesuai ukuran layar.

8. Tampilan Mobile-Friendly

Pada layar kecil, lebar produk disesuaikan untuk tampil tetap rapi dan mudah dilihat.

9. Warna dan Branding Konsisten

Warna oranye (#ee4d2d) digunakan konsisten untuk identitas merek dan tombol.

Kombinasi HTML, CSS, dan JavaScript dalam halaman ini menunjukkan bahwa katalog Vea Sepatu bukan hanya menampilkan produk dengan menarik, tetapi juga memiliki fitur e-commerce sederhana melalui penyimpanan produk dalam keranjang belanja yang dikelola oleh localStorage. Program ini dapat menjadi bagian dari sistem belanja online yang lebih besar, dan sudah siap dikembangkan lebih lanjut, misalnya dengan halaman checkout atau login pengguna.

Laporan Pembuatan Program Perancangan

Nama: Daniel

Nim: 241130915

Matkul: Perancangan Web

Struktur HTML

Halaman ini merupakan halaman detail pesanan dari toko online Vea Commerce, yang menampilkan informasi lengkap terkait satu transaksi pembelian oleh pelanggan. Tampilan halaman dibagi ke dalam beberapa bagian penting: informasi pemesanan, alamat pengiriman, metode pembayaran, produk yang dipesan, serta ringkasan biaya pesanan. Informasi-informasi tersebut dimuat secara dinamis menggunakan JavaScript dengan mengambil data dari localStorage — terutama dari item yang telah disimpan sebelumnya (seperti orderItems) oleh pengguna saat memilih produk.

Desain halaman ini modern dan responsif, menggunakan kombinasi flex dan grid layout, serta warna dominan oranye (--primary: #ee4d2d) yang mencerminkan branding Vea Commerce. Jika tidak ada produk yang dipesan, sistem akan menampilkan notifikasi bahwa tidak ada item dalam pesanan. Selain itu, terdapat dua tombol tindakan: satu untuk mencetak pesanan dan satu untuk kembali ke halaman utama (Dashboard). Fungsi utama seperti penentuan subtotal, total, dan ongkos kirim dilakukan oleh JavaScript dan ditampilkan secara otomatis setelah halaman dimuat.

Bukti-Bukti Kode

1. Struktur Halaman Pesanan

```
<div class="order-container">
    <div class="order-header">
        <h1>Detail Pesanan</h1>
        <p>No. Pesanan: <strong id="order-number">...</strong></p>
        <p>Tanggal: <strong id="order-date">...</strong></p>
        <span class="order-status">Selesai</span>
    </div>
```

Ini adalah bagian informasi utama pesanan, seperti nomor pesanan, tanggal transaksi, dan status.

2. Menampilkan Informasi Pelanggan dan Pembayaran

```
<p><strong id="customer-name">Nama Pelanggan</strong></p>
<p id="customer-phone">No. Telepon: 081234567890</p>
<p id="customer-address">Alamat: Jl. Contoh No. 123, Kota Contoh, 12345</p>
<p id="payment-method">Transfer Bank (BCA)</p>
<p id="payment-status">Status: <strong>Lunas</strong></p>
```

Informasi ini menunjukkan data pembeli dan metode pembayaran, yang nantinya dapat diisi dinamis via JavaScript.

3. Menampilkan Daftar Produk Dipesan

```
<tbody id="order-items-list">
    <!-- Items will be populated by JavaScript -->
</tbody>
```

Ini tempat daftar produk ditampilkan. Elemen ini diisi oleh JavaScript berdasarkan data dari localStorage.

4. Mengambil dan Menampilkan Data Pesanan Secara Otomatis

```
const orderItems = JSON.parse(localStorage.getItem('orderItems')) || [];
orderItems.forEach(item => {
    const itemTotal = item.price * item.quantity;
    subtotal += itemTotal;
    ...
});
```

JavaScript memproses daftar produk (orderItems) yang sebelumnya disimpan dari halaman lain (misal halaman produk).

5. Perhitungan Otomatis Subtotal, Ongkos Kirim, dan Total

```
const shippingCost = 15000;
const total = subtotal + shippingCost;

document.getElementById('subtotal').textContent = `Rp ${subtotal.toLocaleString('id-ID')}`;
document.getElementById('shipping-cost').textContent = `Rp ${shippingCost.toLocaleString('id-ID')}`;
document.getElementById('total-amount').textContent = `Rp ${total.toLocaleString('id-ID')}`;
```

Kalkulasi biaya dilakukan otomatis. subtotal ditambah dengan ongkir menghasilkan total yang ditampilkan.

6. Nomor Pesanan dan Tanggal Dibuat Dinamis

```
const now = new Date();
const orderNumber = `VC-${now.getFullYear()}${String(now.getMonth()+1).padStart(2,'0')}${String(no
document.getElementById('order-number').textContent = orderNumber;

document.getElementById('order-date').textContent = now.toLocaleDateString('id-ID', options);
```

Ini menunjukkan nomor pesanan dibuat otomatis berdasarkan tanggal dan angka acak. Tanggal diformat lokal (Bahasa Indonesia).

7. Responsif dan Fleksibel di Mobile

Halaman menyesuaikan diri di perangkat kecil, seperti HP atau tablet, agar tetap bisa dibaca dan di-scroll secara horizontal.

8. Tombol Aksi Cetak & Kembali

Memberikan kemudahan bagi pengguna untuk mencetak bukti pesanan atau kembali ke halaman utama.

9. Jika Tidak Ada Item

Jika tidak ada data orderItems di localStorage, maka pengguna akan diberi tahu bahwa tidak ada produk yang dipesan.

Halaman ini adalah bagian penting dari sistem belanja online Vea Commerce, berfungsi sebagai rekap atau invoice digital dari transaksi pembelian. Semua data disusun secara otomatis menggunakan JavaScript dan localStorage, yang menjadikan sistem ini ringan dan mudah digunakan. Desainnya bersih dan responsif, cocok digunakan baik di desktop maupun di perangkat seluler. Kode ini juga bisa menjadi dasar untuk membangun sistem cetak invoice PDF atau integrasi dengan backend (database), jika diinginkan.

Laporan Pembuatan Program Perancangan

Nama: Idham al Qusyairi

Nim: 241130796

Matkul: Perancangan Web

Halaman HTML ini merupakan halaman "Detail Pesanan" dari toko online bernama Vea Commerce yang dirancang untuk menampilkan ringkasan transaksi setelah pelanggan menyelesaikan pembelian. Dalam struktur halaman ini, informasi seperti nomor pesanan, tanggal transaksi, data pelanggan, metode pembayaran, daftar produk yang dipesan, serta ringkasan biaya disusun dengan jelas dan profesional. Fungsionalitas utama ditangani oleh JavaScript yang dijalankan saat halaman dimuat (DOMContentLoaded), dan sebagian besar data dimuat dari localStorage browser, khususnya dari key bernama orderItems.

Penjelasan Fungsionalitas

Begitu halaman dibuka, JavaScript secara otomatis:

Menghasilkan nomor pesanan dinamis berdasarkan tanggal saat ini dan angka acak tiga digit. Hal ini dibuktikan pada baris:

```
const orderNumber = `VC-${now.getFullYear()}${String(now.getMonth()+1).padStart(2,'0')}${S
```

Ini akan menciptakan format seperti VC-20250705-123.

```
now.toLocaleDateString('id-ID', options);
```

Memuat data pelanggan secara statis, yang nantinya bisa digantikan oleh sistem login atau checkout.

```
document.getElementById('customer-name').textContent = 'Pelanggan Vea';
```

Memuat produk pesanan dari localStorage, yang sebelumnya disimpan dari halaman katalog produk menggunakan key orderItems (harus disimpan secara manual saat checkout). Ini dibuktikan melalui:

```
const orderItems = JSON.parse(localStorage.getItem('orderItems')) || [];
```

Jika tidak ada item di dalam keranjang, maka akan ditampilkan baris:

```
<tr><td colspan="4" style="text-align: center;">Tidak ada item pesanan</td></tr>
```

Jika ada data, skrip akan membuat baris <tr> baru di dalam tabel pesanan, menampilkan gambar produk, nama, harga satuan, jumlah, dan subtotal:

```
row.innerHTML = `
    <td>
        <div style="display: flex; align-items: center; gap: 10px;">
            <img src="${item.image}" class="item-image" alt="${item.name}">
            <span>${item.name}</span>
        </div>
    </td>
    <td>Rp ${item.price.toLocaleString('id-ID')}</td>
    <td>${item.quantity}</td>
    <td>Rp ${itemTotal.toLocaleString('id-ID')}</td>
`;
```

Setelah semua item diproses, sistem akan menghitung subtotal dan total akhir dengan menambahkan ongkos kirim (Rp 15.000) seperti ini:

```
const total = subtotal + shippingCost;
```

Secara keseluruhan, halaman ini dirancang sebagai halaman ringkasan checkout yang dinamis, dengan penggunaan localStorage sebagai sumber data dan JavaScript untuk menampilkan informasi secara real-time. Fitur seperti cetak, format mata uang lokal (Rp), serta gaya visual modern menjadikannya layak digunakan dalam platform e-commerce nyata. Kamu bisa membuktikan semua fungsionalitas ini dengan mencoba alur: tambahkan produk ke keranjang dari halaman sebelumnya, simpan ke localStorage, lalu buka halaman ini.

Laporan Pembuatan Program Perancangan

Nama: Sherly

Nim: 241130894

Matkul: Perancangan Web

Struktur HTML

Halaman HTML ini dibuat sebagai katalog online yang menampilkan berbagai produk makeup wanita. Saat halaman dibuka, pengguna langsung disambut dengan bagian atas (header) yang sederhana namun informatif. Di sana tertera judul "Kategori Make-up", memberikan gambaran langsung tentang isi halaman. Tepat di bawahnya, terdapat navigasi kecil yang mengarahkan pengguna kembali ke halaman utama atau dashboard, sehingga pengguna tidak akan merasa tersesat saat menelusuri situs. Bagian utama halaman ini diawali dengan judul besar “KOLEKSI Makeup Wanita” yang memperkenalkan isi katalog. Seluruh produk kosmetik ditampilkan dalam bentuk galeri yang tersusun rapi dalam bentuk grid. Grid ini responsif, artinya susunannya akan otomatis menyesuaikan ukuran layar—baik di komputer, tablet, maupun ponsel—sehingga pengguna tetap bisa menikmati tampilan yang enak dilihat di perangkat apa pun.

Setiap produk ditampilkan dalam sebuah kartu produk yang memuat gambar, nama produk, harga, dan sebuah tombol “Tambah ke Keranjang”. Tombol tersebut tampak seperti fitur belanja, namun belum memiliki fungsi sebenarnya karena belum terhubung dengan sistem pemesanan atau JavaScript. Fitur ini bersifat visual untuk saat ini, namun dapat dengan mudah dikembangkan di masa depan. Katalog ini memuat 14 produk kosmetik dengan variasi yang menarik—mulai dari highlighter, lip matte, makeup set, hingga parfum dan nail polish. Produk-produk tersebut mencerminkan kebutuhan berbagai segmen konsumen, dari yang mencari kosmetik harian hingga yang menginginkan set mewah.

Terakhir, bagian bawah halaman berisi footer yang menampilkan teks hak cipta, menyatakan bahwa semua konten halaman ini merupakan milik Vea dan dilindungi hak cipta. Tahun yang tertera adalah 2025, menunjukkan bahwa halaman ini ditujukan untuk dipublikasikan atau digunakan pada tahun tersebut. Tampilan keseluruhan halaman bersih, modern, dan sangat cocok untuk digunakan sebagai dasar toko online kosmetik yang profesional dan menarik.

1. Produk Dapat Ditambahkan ke Keranjang

Fungsi addToCart() dipanggil saat tombol diklik.

Parameter: nama produk, harga, dan gambar.

2. Fungsi addToCart() Menyimpan ke localStorage

```
let cart = JSON.parse(localStorage.getItem('cart')) || [];
...
localStorage.setItem('cart', JSON.stringify(cart));
```

Data keranjang disimpan dan dibaca dari localStorage, jadi tidak hilang walaupun halaman direfresh.

3. Cek dan Update Jika Produk Sudah Ada

```
const existingItem = cart.find(item => item.name === name);

if (existingItem) {
  existingItem.quantity += 1;
} else {
  cart.push({ name, price, image, quantity: 1 });
}
```

- Jika produk sudah ada di keranjang, jumlah ditambah (quantity += 1).
- Jika belum, produk baru ditambahkan.

4. Notifikasi Saat Produk Ditambahkan

```
alert(`${name} telah ditambahkan ke keranjang!`);
```

- Memberi konfirmasi kepada pengguna bahwa produk berhasil dimasukkan ke keranjang.

5. Animasi Tombol Setelah Klik

```
event.target.textContent = 'Ditambahkan ✓';
event.target.style.backgroundColor = '#4CAF50';
setTimeout(() => {
  event.target.textContent = 'Tambah ke Keranjang';
  event.target.style.backgroundColor = '';
}, 2000);
```

- Tampilan tombol berubah selama 2 detik sebagai efek visual bahwa produk telah ditambahkan.

6. Mengisi Data Produk Secara Otomatis

```
document.querySelectorAll('.add-to-cart').forEach(button => {
  const card = button.closest('.product-card');
  const name = card.querySelector('.product-title').textContent;
  const price = parseInt(card.querySelector('.product-price').textContent.replace(/\D/g, ''));
  const image = card.querySelector('.product-image img').src;

  button.onclick = (e) => addToCart(e, name, price, image);
});
```

- Semua tombol otomatis dihubungkan dengan data produk dari DOM, tidak perlu menulis satu per satu di setiap tombol.

7. Tata Letak Produk Tertata Rapi (Grid Layout)

- Produk ditampilkan secara responsif dengan grid yang otomatis menyesuaikan lebar layar.

8. Styling Tombol Konsisten dan Responsif

- Warna tombol, bentuk, dan efek hover sudah diatur untuk kenyamanan pengguna.

9. Desain Modern dan Responsif

- Halaman akan menyesuaikan tampilan jika dibuka di HP atau layar kecil.

Laporan Pembuatan Program Perancangan

Nama: Steward

Nim: 241130765

Matkul: Perancangan Web

Halaman ini dibangun menggunakan struktur HTML standar dan gaya CSS yang modern dan responsif. Di dalam elemen <body>, terdapat bagian header yang menampilkan logo dan link navigasi ke halaman “Home”. Di bawahnya, ada bagian yang berisi grid katalog produk. Setiap produk ditampilkan menggunakan komponen yang terdiri dari gambar, nama, harga, dan tombol "Tambah ke Keranjang". Ketika pengguna menekan tombol ini, akan terjadi interaksi JavaScript yang penting.

Fungsi utama pada halaman ini adalah addToCart(event, name, price, image). Saat pengguna mengklik tombol, fungsi ini akan dijalankan dan melakukan proses sebagai berikut: Membaca data dari localStorage dengan key 'cart' — jika belum ada, maka akan diinisialisasi sebagai array kosong ([]).Fungsi ini memeriksa apakah produk yang sama sudah ada dalam keranjang menggunakan Array.find(). Jika produk sudah ada, maka hanya menambah kuantitasnya. Jika produk belum ada di keranjang, maka fungsi akan menambahkan objek baru ke dalam array cart. Setelah proses di atas, data cart diperbarui dan disimpan kembali ke localStorage dalam format JSON. Fungsi juga memberikan umpan balik ke pengguna dalam bentuk popup alert dan mengganti teks tombol sementara menjadi "Ditambahkan ✓", lalu mengembalikannya setelah 2 detik.

Bukti – Buktinya

1. Struktur HTML Utama

```
<header>
  <div class="logo">Kategori Baju</div>
  <nav>
    <a href="../DASHBOARD.html">Home</a>
  </nav>
</header>
```

Penjelasan:

Menampilkan nama kategori dan tautan kembali ke halaman Dashboard.

Container Produk

```
<div class="container">
  <h1>KOLEKSI WANITA & PRIA TERBARU</h1>
  <div class="products"> ... </div>
</div>
```

Penjelasan:

- container: membungkus semua produk dan menjaga jarak margin.
- products: grid layout (CSS Grid) untuk menampilkan kartu produk secara rapi dan responsif.

Setiap Produk

```
<div class="product-card">
  <div class="product-image">
    <img src="./PRIA/1.jpg" alt="Kemeja Pria Coklat">
  </div>
  <div class="product-info">
    <h3 class="product-title">Kemeja Pria Coklat</h3>
    <div class="product-price">Rp 400.000</div>
    <button class="add-to-cart" onclick="addToCart(event, 'Kemeja Pria Coklat', 400000, './PRI
  </div>
</div>
```

Bukti:

Produk memiliki nama, harga, gambar, dan tombol Tambah ke Keranjang. Saat tombol ditekan, fungsi addToCart() dipanggil.

2. Fungsi JavaScript: addToCart()

```
function addToCart(event, name, price, image) {
  let cart = JSON.parse(localStorage.getItem('cart')) || [];

  const existingItem = cart.find(item => item.name === name);

  if (existingItem) {
    existingItem.quantity += 1;
  } else {
    cart.push({
      name: name,
      price: price,
      image: image,
      quantity: 1
```

Laporan Pembuatan Program Perancangan

Nama: Wisander Wibowo

Nim: 241130031

Matkul: Perancangan Web

Struktur HTML

Program HTML, CSS, dan JavaScript di atas merupakan sebuah halaman login sederhana untuk aplikasi bernama Vea. Halaman ini dirancang agar responsif dan menarik secara visual, dengan penggunaan gradient hijau sebagai latar belakang serta kontainer putih di tengah layar yang menampilkan form login. Di bagian atas terdapat logo Vea dan judul "Login ke Vea", disusul oleh dua input: satu untuk username dan satu untuk password, serta tombol Login dan opsi Login dengan Google.

Dari sisi tampilan, halaman ini menggunakan CSS internal untuk menciptakan desain yang modern dan ramah pengguna. Form login dibuat dengan elemen yang rapi dan lembut (menggunakan border-radius dan padding), serta memiliki efek hover yang memperindah interaksi pengguna. Tombol "Login dengan Google" juga didesain menarik, menampilkan ikon Google dan efek hover yang memberi kesan profesional.

Logika interaksi halaman ini ditangani oleh JavaScript. Saat halaman selesai dimuat, script akan mendengarkan event saat form dikirim. Ketika pengguna menekan tombol login, JavaScript akan mengecek apakah input username dan password sudah diisi. Kemudian, data tersebut akan divalidasi dengan mencocokkannya terhadap data user statis (dummy) yang disimpan dalam objek: username admin dan password 1234. Jika cocok, maka akan muncul pesan "Login berhasil", data status login disimpan ke localStorage, dan pengguna akan diarahkan ke halaman DASHBOARD.html setelah jeda 1 detik. Namun jika salah, maka akan muncul pesan kesalahan berwarna merah.

Sementara itu, tombol "Login dengan Google" sebenarnya hanyalah sebuah hyperlink biasa yang mengarahkan ke halaman login akun Google. Fitur ini belum terintegrasi dengan sistem autentikasi Google secara langsung, sehingga belum dapat digunakan untuk login sebenarnya.

Secara keseluruhan, program ini adalah prototipe halaman login yang menekankan tampilan visual dan interaksi pengguna dasar. Untuk menjadikannya sistem login nyata, perlu ditambahkan integrasi dengan backend atau layanan autentikasi seperti Firebase, serta menerapkan keamanan seperti penyimpanan password terenkripsi dan manajemen sesi yang baik.

Bukti – Buktinya

1. Tampilan Modern dengan Latar Belakang Gradient Hijau

```
  body {
margin: 0;
padding: 0;
font-family: 'Arial', sans-serif;
background: linear-gradient(135deg, #a8e063, #56ab2f);
display: flex;
justify-content: center;
align-items: center;
height: 100vh;
```

Bukti:

Menunjukkan bahwa latar belakang menggunakan gradien hijau dan konten diposisikan di tengah layar secara vertikal dan horizontal menggunakan flexbox.

2. Kontainer Login Berwarna Putih dan Estetis

```
.login-container {
  background: #fff;
  padding: 30px;
  border-radius: 20px;
  box-shadow: 0 8px 25px rgba(0,0,0,0.2);
  text-align: center;
  width: 320px;
}
```

Bukti:

Desain kontainer login berwarna putih (background: #fff), dengan sudut membulat (border-radius) dan bayangan (box-shadow) agar terlihat lebih profesional dan elegan.

3. Form Login Menggunakan Validasi dan Dummy User

```
const dummyUser = {
  username: "admin",
  password: "1234"
};
```

```
if (username === dummyUser.username && password === dummyUser.password) {
  errorMsg.style.color = "green";
  errorMsg.textContent = "Login berhasil! Mengarahkan...";

  // Simpan status login ke localStorage
  localStorage.setItem('isLoggedIn', 'true');
  localStorage.setItem('username', username);

  setTimeout(() => {
    window.location.href = "DASHBOARD.html";
```

Bukti:

Ini membuktikan bahwa login hanya akan berhasil jika username adalah admin dan password 1234. Jika cocok, akan menampilkan pesan sukses dan redirect ke halaman DASHBOARD.html.

4. Tampilan Error dan Validasi Kosong

```
if (username === "" || password === "") {
  errorMsg.textContent = "Silakan isi semua kolom.";
}
```

```
errorMsg.style.color = "red";
errorMsg.textContent = "Username atau password salah.";
```

Bukti:

Script akan menampilkan pesan error jika kolom tidak diisi atau jika data salah. Ini membuktikan adanya validasi form di sisi klien.

5. Penyimpanan Status Login ke localStorage

```
localStorage.setItem('isLoggedIn', 'true');
localStorage.setItem('username', username);
```

Bukti:

Setelah login berhasil, status login dan nama pengguna disimpan di localStorage, yang memungkinkan pelacakan status login meskipun halaman di-reload.

6. Tombol "Login dengan Google" Bukan Sistem Login Sebenarnya

```
<a href="https://accounts.google.com/v3/signin/identifier?...">
  <span>Login dengan Google</span>
</a>
```

Bukti:

Tombol hanya merupakan link <a> biasa menuju ke halaman login Google. Tidak ada kode OAuth, token, atau proses autentikasi yang sebenarnya, jadi belum bisa disebut sebagai login Google fungsional.

7. Efek Hover untuk Tombol dan Desain Responsif

```
button:hover {
  background-color: #3d8e1e;
}
.google-login:hover {
  background-color: #f5f5f5;
}
```

Bukti:

Memberi efek interaktif saat tombol disentuh kursor (hover), meningkatkan pengalaman pengguna.