

APLIKASI APOTIK



Dosen Pengampu : Basirudin Ansor,M.Kom

Kelompok 2 :

- | | | |
|----|---------------|-----------|
| 1. | Nurul Hidayah | C2C022510 |
| 2. | Ayu Wulandari | C2C022517 |
| 3. | Irfan Muzaki | C2C022534 |

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK DAN ILMU KOMPUTER

UNIVERSITAS MUHAMMADIYAH SEMARANG

KATA PENGANTAR

Puji syukur kita panjatkan kehadirat Allah SWT, karena berkat rahmat dan hidayah-Nya lah kami dapat menyelesaikan tugas yang diberikan kepada kami, sesuai dengan waktu yang telah ditentukan oleh dosen pembimbing mata kuliah AIK. Adapun tugas makalah ini yang berjudul “Aplikasi Apotik”.

Namun tidak lepas dari semua itu, kami menyadari sepenuhnya bahwa ada banyak kekurangan dari makalah ini, baik dari segi penyusunan bahasa maupun segi lainnya, tetapi kami berharap semoga makalah ini dapat bermanfaat bagi kita semua. Untuk itu, kami mengharapkan adanya kritik, saran, dan usulan demi perbaikan di masa yang akan datang, mengingat tidak ada sesuatu yang sempurna tanpa sarana yang membangun.

Semarang, 3 Juli 2024

Kelompok 3

DAFTAR ISI

KATA PENGANTAR.....	2
DAFTAR ISI	3
BAB I PENDAHULUAN.....	
A. Latar Belakang	4
B. Rumusan Masalah.....	4
C. Tujuan.....	4
BAB II PEMBAHASAN	
A. Use Case Diagram.....	6
B. Databse Aplikasi Apotik.....	13
C. Bahasa Pemrograman yang Digunakan.....	24
D. Codingan	28
B. Output Front-End	48
BAB III PENUTUP.....	
A. Kesimpulan	49
B. Saran	50

BAB I

PENDAHULUAN

A. Latar Belakang

Sistem aplikasi apotek bertujuan untuk meningkatkan efisiensi dan efektivitas operasional apotek. Dalam operasional sehari-hari, apotek sering menghadapi berbagai tantangan seperti kesalahan dalam pengelolaan inventaris obat, keterlambatan dalam pelayanan kepada pelanggan, serta kesulitan dalam pencatatan dan pelaporan transaksi. Selain itu, dengan adanya perkembangan teknologi informasi, penggunaan sistem aplikasi dapat membantu apotek untuk lebih terstruktur dan terorganisir dalam mengelola data, sehingga pelayanan kepada pelanggan dapat dilakukan dengan lebih cepat dan akurat. Oleh karena itu, diperlukan sebuah sistem aplikasi apotek yang dapat mengatasi berbagai masalah tersebut dan memberikan manfaat maksimal bagi apotek dalam mengelola operasionalnya.

B. Rumusan Masalah

1. Bagaimana cara mengelola inventaris obat agar dapat meminimalisir kesalahan dan kekurangan stok?
2. Bagaimana meningkatkan kecepatan dan akurasi pelayanan kepada pelanggan di apotek?
3. Bagaimana cara mencatat dan melaporkan transaksi penjualan dengan lebih efisien?
4. Bagaimana menjaga keamanan data dan informasi terkait obat serta transaksi di apotek?
5. Bagaimana mengintegrasikan sistem aplikasi apotek dengan sistem lain yang relevan seperti sistem pembayaran dan sistem informasi kesehatan?

C. Tujuan

1. Meningkatkan efisiensi pengelolaan inventaris obat dengan sistem pencatatan yang akurat dan real-time.

2. Mempercepat proses pelayanan kepada pelanggan dengan sistem transaksi yang cepat dan mudah digunakan.
3. Menyediakan fitur pencatatan dan pelaporan transaksi yang lengkap dan dapat diakses dengan mudah.
4. Menjamin keamanan data dan informasi yang terkait dengan operasional apotek.
5. Mengintegrasikan sistem aplikasi apotek dengan sistem lain yang relevan untuk mendukung operasional yang lebih komprehensif dan terintegrasi.

BAB II

PEMBAHASAN

A. USE CASE DIGRAM

Use Case Diagram adalah satu dari berbagai jenis diagram UML (Unified Modelling Language) yang menggambarkan hubungan interaksi antara sistem dan aktor. Use Case dapat mendeskripsikan tipe interaksi antara si pengguna sistem dengan sistemnya. Use Case Diagram merupakan sesuatu yang mudah dipelajari. Langkah awal untuk melakukan pemodelan, tentu perlunya suatu diagram yang mampu menjabarkan aksi aktor dengan aksi sistem itu sendiri, seperti pada use case diagram.

Fungsi dari Use Case diagram sebagai berikut:

1. Berguna memperlihatkan proses aktivitas secara urut dalam sistem.
2. Mampu menggambarkan proses bisnis, bahkan menampilkan urutan aktivitas pada sebuah proses.
3. Sebagai bridge atau jembatan antara pembuat dengan konsumen untuk mendiskripsikan sebuah sistem.

Manfaat dari Use Case di antaranya:

1. Menggunakannya sebagai kebutuhan verifikasi.
2. Menjadi gambaran interface dari sebuah sistem karena setiap sistem yang dibangun haruslah memiliki interface.
3. Mengidentifikasi siapa saja orang yang dapat berinteraksi dengan sistem, serta apa yang dapat dilakukan oleh sistem.
4. Memberikan kepastian mengenai kebutuhan sistem, sehingga tidak membingungkan.
5. Memudahkan proses komunikasi antar domain expert dan end user.

Komponen Use Case Diagram:

a. Sistem

Sebuah sistem digambarkan ke dalam bentuk persegi. Fungsinya untuk membatasi use case dengan interaksi dari luar sistem.

Sistem pada umumnya diberikan label yang sesuai. Namun, umumnya sistem ini tidaklah diberi gambar karena kita tidak terlalu memberikan makna pada sebuah diagram.

b. Actor

Peran actor sangat penting, tentunya menciptakan use case jadi lebih mudah. Fungsi Actor menjelaskan siapa yang berinteraksi dengan sistem. Actor akan memberikan informasi kepada sistem, serta menerima informasi dari sistem. Keduanya bisa terjadi secara bersamaan.

Aktor tidak memberikan kontrol terhadap sistem, namun hanya memberikan gambaran mengenai hubungannya dengan sistem.





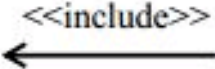
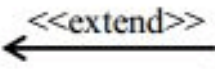
Ternyata, inilah beberapa alasan mengapa actor dapat berhubungan dengan sistem lain:

- Jika terdapat relasi sistem lain dengan sistem yang sedang dibuat.
- Terdapat eksternal resource yang digunakan oleh sistem.
- Adanya kepentingan terhadap sistem, yaitu alur informasi baik penerima maupun arus sistem saling memiliki kepentingan.
- Terdapat seseorang atau pihak lain yang akan mengelola sistem.

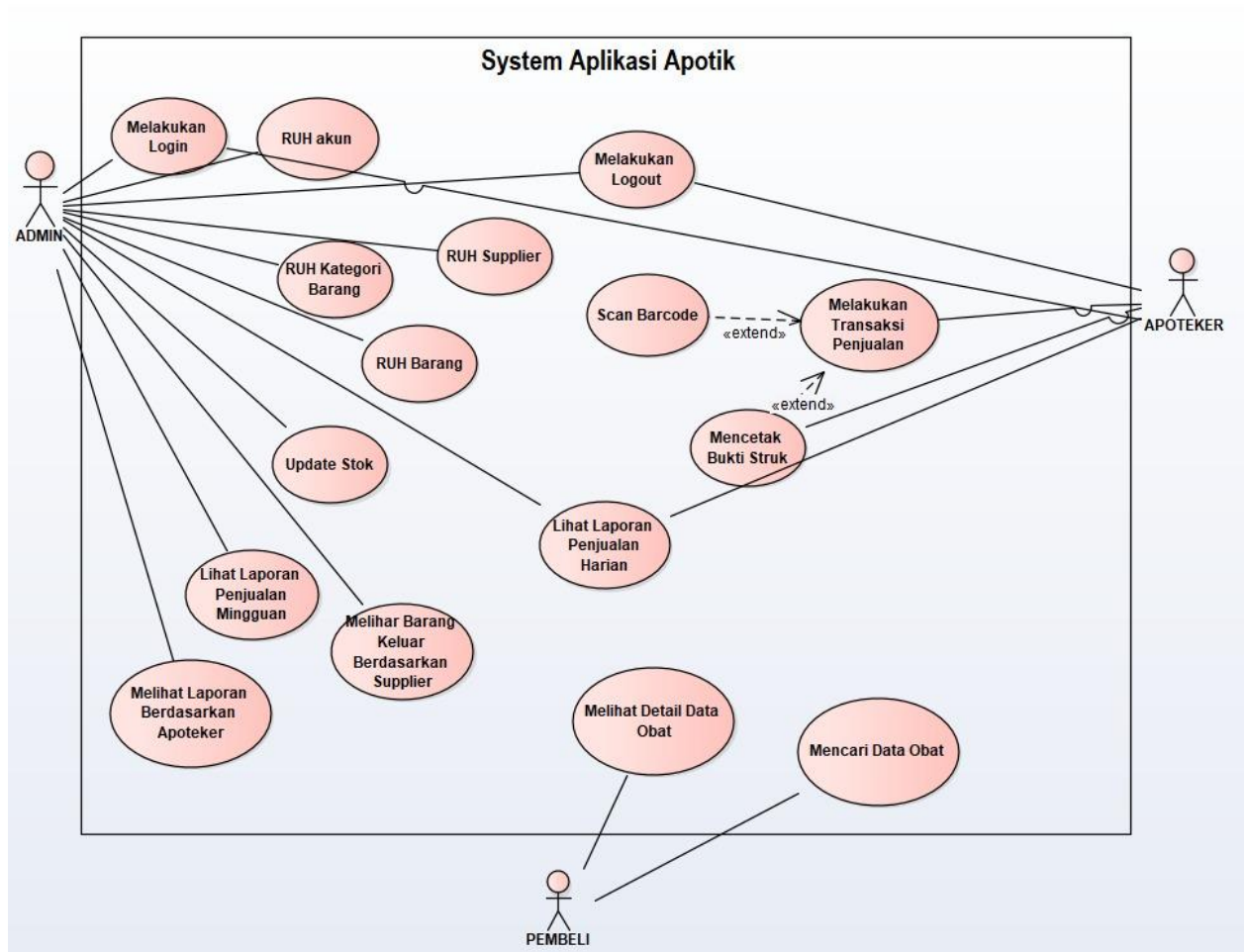
c. Use Case

Use case adalah komponen gambaran fungsional dalam sebuah sistem. Sehingga konsumen maupun pembuat saling mengenal dan mengerti mengenai alur sistem yang akan dibuat.

Simbol-Simbol pada Use Case Diagram

Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan <i>use case</i>
	<i>Use case</i> : Abstraksi dan interaksi antara sistem dan aktor
	<i>Association</i> : Abstraksi dari penghubung antara aktor dengan use case
	<i>Generalisasi</i> : Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case
	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

Di kelompok kami membutuhkan tools/aplikasi **ENTERPRISE ARCHITECT** untuk membuat Use Case Diagram.



Aktor yang terlibat: Admin, Apoteker, dan Pembeli.

Use Case yang ada: Melakukan Login, RUH Akun, RUH Kategori Barang, RUH Barang, RUH Supplier, Update Stok, Lihat Laporan Penjualan Mingguan, Melihat Laporan Berdasarkan Apoteker, Melihat Barang Keluar Berdasarkan Supplier, Melakukan Logout, Lihat Laporan Penjualan Harian, Scan Barcode, Melakukan Transaksi Penjualan, Mencetak Bukti Struk, Melihat Detail Data Obat, Mencari Data Obat.

Di bawah ini beberapa penjelasan lengkap dari use case tersebut:

Aktor:

1. Admin

- Mengelola operasi sistem apotek secara keseluruhan.
- Memiliki hak akses penuh untuk menambah, mengubah, atau menghapus data dalam sistem.

2. Apoteker

- Mengelola stok obat dan melakukan transaksi penjualan.
- Melihat laporan penjualan dan data terkait obat-obatan.

3. Pembeli

- Orang yang membeli obat di apotek.
- Berinteraksi dengan sistem terutama melalui apoteker untuk pembelian obat.

Use Cases:

1. Melakukan Login

- **Aktivitas:** Proses autentikasi pengguna untuk mengakses sistem.
- **Aktor:** Admin, Apoteker.
- **Deskripsi:** Pengguna memasukkan username dan password untuk masuk ke dalam sistem.

2. Melakukan Logout

- **Aktivitas:** Proses mengakhiri sesi pengguna dalam sistem.
- **Aktor:** Admin, Apoteker.
- **Deskripsi:** Pengguna keluar dari sistem untuk mengamankan akun mereka.

3. RUH Akun

- **Aktivitas:** Registrasi, Ubah, dan Hapus akun pengguna.
- **Aktor:** Admin.

- **Deskripsi:** Admin mengelola data akun pengguna (admin dan apoteker) dalam sistem.

4. **RUH Kategori Barang**

- **Aktivitas:** Registrasi, Ubah, dan Hapus kategori barang.
- **Aktor:** Admin.
- **Deskripsi:** Admin mengelola kategori barang yang digunakan untuk pengelompokan obat.

5. **RUH Barang**

- **Aktivitas:** Registrasi, Ubah, dan Hapus data barang (obat).
- **Aktor:** Admin, Apoteker.
- **Deskripsi:** Admin dan apoteker mengelola data obat yang tersedia di apotek.

6. **RUH Supplier**

- **Aktivitas:** Registrasi, Ubah, dan Hapus data pemasok.
- **Aktor:** Admin.
- **Deskripsi:** Admin mengelola informasi tentang pemasok obat-obatan.

7. **Update Stok**

- **Aktivitas:** Memperbarui jumlah stok obat di apotek.
- **Aktor:** Admin, Apoteker.
- **Deskripsi:** Admin dan apoteker memperbarui data stok obat berdasarkan pemasukan dan pengeluaran barang.

8. **Lihat Laporan Penjualan Mingguan**

- **Aktivitas:** Melihat laporan penjualan yang terjadi selama satu minggu.
- **Aktor:** Admin, Apoteker.
- **Deskripsi:** Sistem menghasilkan laporan penjualan mingguan untuk dianalisis.

9. **Lihat Laporan Penjualan Harian**

- **Aktivitas:** Melihat laporan penjualan yang terjadi setiap hari.
- **Aktor:** Admin, Apoteker.

- **Deskripsi:** Sistem menghasilkan laporan penjualan harian untuk dianalisis.

10. Melihat Laporan Berdasarkan Apoteker

- **Aktivitas:** Melihat laporan penjualan yang dilakukan oleh apoteker tertentu.
- **Aktor:** Admin.
- **Deskripsi:** Sistem menghasilkan laporan penjualan berdasarkan transaksi yang dilakukan oleh apoteker tertentu.

11. Melihat Barang Keluar Berdasarkan Supplier

- **Aktivitas:** Melihat data barang yang keluar berdasarkan pemasoknya.
- **Aktor:** Admin.
- **Deskripsi:** Sistem menghasilkan laporan barang keluar yang dikelompokkan berdasarkan pemasok.

12. Scan Barcode

- **Aktivitas:** Memindai barcode obat untuk mempercepat proses pencarian data obat atau transaksi penjualan.
- **Aktor:** Apoteker.
- **Deskripsi:** Apoteker menggunakan pemindai barcode untuk memasukkan data obat secara otomatis dalam sistem.

13. Melakukan Transaksi Penjualan

- **Aktivitas:** Mencatat penjualan obat kepada pelanggan.
- **Aktor:** Apoteker.
- **Deskripsi:** Apoteker memasukkan data penjualan, menerima pembayaran, dan mencetak struk.

14. Mencetak Bukti Struk

- **Aktivitas:** Mencetak bukti transaksi penjualan untuk pelanggan.
- **Aktor:** Apoteker.
- **Deskripsi:** Setelah transaksi penjualan selesai, apoteker mencetak struk sebagai bukti pembayaran untuk pelanggan.

15. Melihat Detail Data Obat

- **Aktivitas:** Melihat informasi lengkap tentang suatu obat.
- **Aktor:** Admin, Apoteker.
- **Deskripsi:** Pengguna dapat melihat detail data obat termasuk nama, kategori, harga, dan stok yang tersedia.

16. Mencari Data Obat

- **Aktivitas:** Mencari informasi obat berdasarkan kriteria tertentu.
- **Aktor:** Admin, Apoteker.
- **Deskripsi:** Pengguna dapat mencari data obat berdasarkan nama, kategori, atau kriteria lainnya.

B. DATA BASE SYSTEM APLIKASI APOTIK

1. Pengertian Data Base

Database adalah kumpulan data yang disimpan dengan sistem tertentu, dan saling berhubungan, sehingga dapat dikelola dengan mudah. Database penting untuk mengatur data yang jumlahnya banyak, dan selalu bertambah. Sebagai contoh pdada program website system Aplikasi Apotik ini sendiri.

Tanpa database, data tersebt hanya akan tersimpan di komputer kita, dan tidak bisa diakses oleh konsumen. Atau, konsumen harus mengakses data dari komputer kita terlebih dahulu secara langsung. Dengan adanya database, data website kita dapat disimpan dalam satu server. Berapapun jumlahnya, bisa disesuaikan dengan kemampuan server tersebut. Dengan begitu data mampu diolah bersamaan sehingga aktivitas browsing untuk melihat produk, memasukkan produk ke keranjang belanja hingga tahap pembayaran bisa berjalan dengan sistem yang baik karena sistem penyimpanan database mampu mengelola data dengan baik. Kita bisa mengatur file sesuai dengan klasifikasinya, misalnya teks, gambar, dan lainnya.

2. Fungsi Database

Terdapat enam fungsi database yang perlu kita ketahui, yaitu:

a. Mempercepat dan Mempermudah Identifikasi Data

Dengan database, Anda bisa membuat sebuah sistem yang dapat mengelompokkan data dan menyimpannya secara terstruktur. Jadi, ketika ada permintaan akses sebuah data, informasinya bisa diberikan dengan cepat sesuai kategori yang sudah ditentukan sebelumnya.

b. Mengontrol Data Secara Terpusat

Tanpa database, data akan terpecah di berbagai penyimpanan secara lokal sesuai dengan pihak yang memiliki data tersebut. Dengan adanya database, semua data bisa dikumpulkan dalam satu tempat, misalnya di server hosting. Jadi, kita pun bisa mengelola berbagai data dari pusat secara lebih efisien.

c. Menghindari Duplikasi Data

Setiap data yang tersimpan dalam database dapat diatur agar terhindar dari data ganda. Sistem database dapat dirancang untuk mengidentifikasi data yang sama, sehingga dapat memberikan warning atau notifikasi ke pengelola database. Misalnya, dengan menerapkan sistem kata kunci atau primary key.

d. Menyimpan Data dengan Lebih Aman

Mengumpulkan data ke dalam satu database, artinya fokus perlindungan keamanannya menjadi lebih baik. Jika data masih tersebar di beberapa perangkat, maka setiap perangkat perlu diamankan. Tapi kalau sudah terpusat di database, Anda cukup mengamankan server dengan perlindungan berlapis.

e. Menghemat Biaya

Dengan database, kita tidak memerlukan banyak tempat untuk menyimpan data. Cukup satu server untuk berbagai kebutuhan data. Secara biaya tentu jauh lebih murah dibanding menyediakan beberapa tempat penyimpanan sendiri.

f. Dapat Diakses Multi-User

Jika data disimpan secara offline di perangkat berbeda, untuk mengakses sebuah file tentu harus menghubungi pemiliknya dulu. Database menyimpan semua data dalam satu sistem. Maka, siapapun bisa mengaksesnya dengan mudah, asalkan memiliki hak akses. Mulai dari programmer, database administrator, hingga pengunjung pada umumnya.

3. Komponen Database

Umumnya, sebuah database akan memiliki empat komponen berikut ini:

1. Data

Data adalah file-file yang berisi informasi, baik teks, log, gambar, dan lainnya. Di dalam database, data akan disimpan dengan struktur tertentu, sehingga mudah dikenali.

Biasanya, struktur tersebut terdiri dari:

- Field – Satuan informasi yang rinci, seperti nama produk, harga, stok, dan lainnya.
- Record – Kumpulan dari field, yang membentuk satu informasi unik. Seperti, harga dari suatu produk.
- Table – Kumpulan dari record, isi dari sebuah file.
- Database – Kumpulan dari tabel atau file.

2. Hardware

Hardware adalah perangkat keras yang digunakan untuk menyimpan dan mengelola data. Kalau untuk penyimpanan secara lokal atau di jaringan tertentu, hardware yang digunakan adalah komputer, disk, memori, dan lainnya. Sedangkan, untuk penyimpanan data online seperti website, server hosting-lah yang digunakan.

3. Sistem Operasi

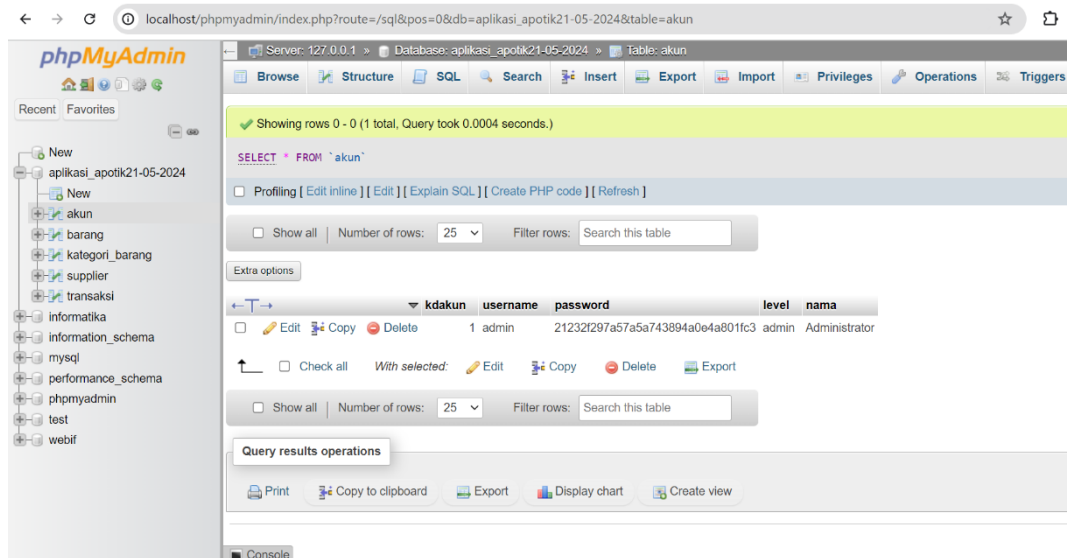
Sistem operasi bertanggung jawab atas semua sistem yang ada di komputer atau server. Pilihlah sistem operasi yang mendukung sistem database yang akan Anda bangun. Bisa menggunakan Windows atau Linux.

4. Database Access Language

Database Access Language adalah bahasa yang digunakan untuk menulis perintah, seperti mengakses, menambah, memperbarui, dan menghapus data di dalam database.

Data Base System Apotik yang kelompok kami buat:

Database:Apotik
Tabel:akun
Kdakun:INT(11)AUTO_INCREMENT PRIMARY KEY
Username_VARCHAR(35)
Password VARCHAR(35)
Level VARCHAR(35)
Nama VARCHAR(50)



1. **Kdakun (INT(11) AUTO_INCREMENT PRIMARY KEY):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **AUTO_INCREMENT:** Nilai kolom ini akan secara otomatis bertambah setiap kali data baru ditambahkan. Ini memastikan setiap baris dalam tabel memiliki nilai unik.
- **PRIMARY KEY:** Kolom ini merupakan kunci utama tabel, yang berarti setiap nilai harus unik dan tidak boleh kosong (NULL). Kunci utama digunakan untuk mengidentifikasi setiap baris data secara unik dalam tabel.

2. **Username (VARCHAR(35)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 35 karakter.
- **Deskripsi:** Kolom ini menyimpan nama pengguna (username) yang digunakan untuk login ke sistem. Nama pengguna harus unik untuk setiap akun.

3. **Password (VARCHAR(35)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 35 karakter.
- **Deskripsi:** Kolom ini menyimpan kata sandi (password) untuk akun pengguna. Kata sandi ini harus dijaga kerahasiaannya dan sebaiknya disimpan dalam bentuk terenkripsi untuk alasan keamanan.

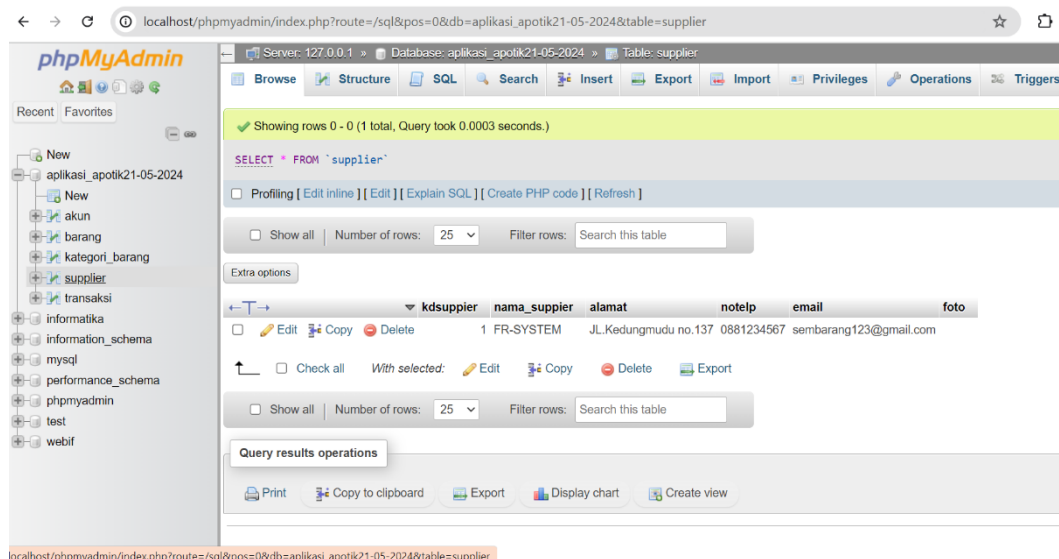
4. **Level (VARCHAR(35)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 35 karakter.
- **Deskripsi:** Kolom ini menyimpan tingkat atau peran pengguna dalam sistem. Misalnya, ini bisa digunakan untuk menentukan apakah pengguna adalah admin, pengguna biasa, atau memiliki peran khusus lainnya.

5. **Nama (VARCHAR(50)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 50 karakter.
- **Deskripsi:** Kolom ini menyimpan nama lengkap pengguna yang terhubung dengan akun tersebut. Nama ini digunakan untuk keperluan identifikasi atau tampilan dalam sistem.

Tabel:Supplier
Kdsupplier INT(11)AUTO_INCREMENT PRIMARY KEY
Nama_supplier VARCHAR(100)
Alamat VARCHAR(35)
Notelp VARCHAR(35)
Email VARCHAR(100)



1. Kdsupplier (INT(11) AUTO_INCREMENT PRIMARY KEY):

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **AUTO_INCREMENT:** Nilai kolom ini akan secara otomatis bertambah setiap kali data baru ditambahkan. Ini memastikan setiap baris dalam tabel memiliki nilai unik.
- **PRIMARY KEY:** Kolom ini merupakan kunci utama tabel, yang berarti setiap nilai harus unik dan tidak boleh kosong (NULL). Kunci utama digunakan untuk mengidentifikasi setiap baris data secara unik dalam tabel.

2. **Nama_supplier (VARCHAR(100)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 100 karakter.
- **Deskripsi:** Kolom ini menyimpan nama pemasok. Nama ini digunakan untuk mengidentifikasi pemasok dalam sistem.

3. **Alamat (VARCHAR(35)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 35 karakter.
- **Deskripsi:** Kolom ini menyimpan alamat pemasok. Informasi alamat ini digunakan untuk tujuan komunikasi dan pengiriman.

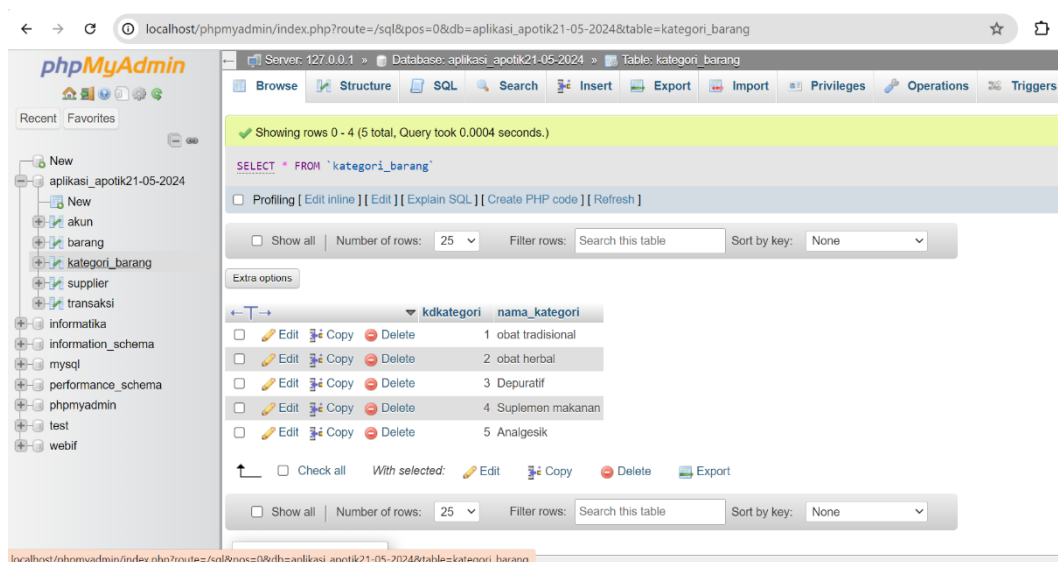
4. **Notelp (VARCHAR(35)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 35 karakter.
- **Deskripsi:** Kolom ini menyimpan nomor telepon pemasok. Informasi ini penting untuk menghubungi pemasok.

5. **Email (VARCHAR(100)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 100 karakter.
- **Deskripsi:** Kolom ini menyimpan alamat email pemasok. Email digunakan untuk komunikasi elektronik dengan pemasok.

Tabel:kategori_barang
Kdkategori INT(11) AUTO_INCREMENT PRIMARY KEY
Nama_kategori VARCHAR(100)



1. **Kdkategori (INT(11) AUTO_INCREMENT PRIMARY KEY):**

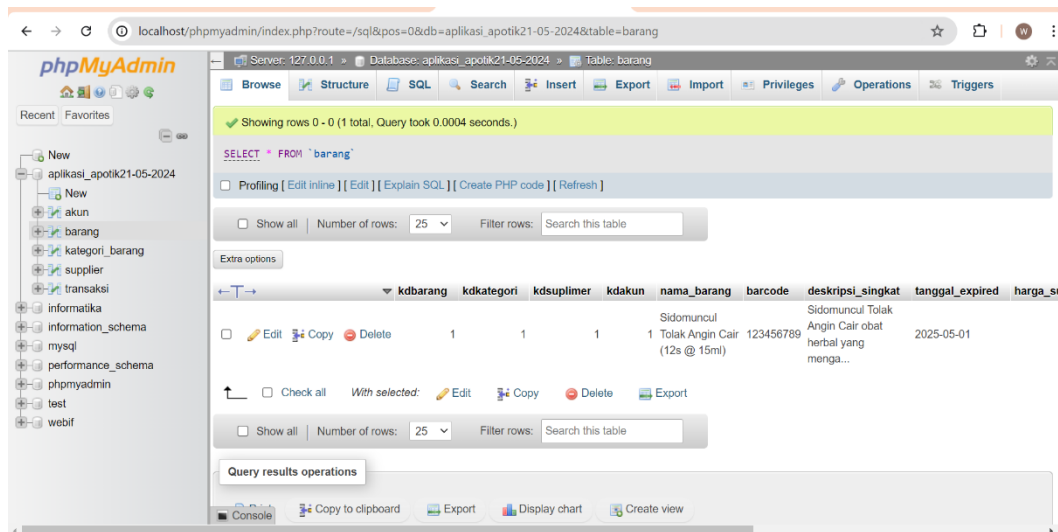
- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.

- **AUTO_INCREMENT**: Nilai kolom ini akan secara otomatis bertambah setiap kali data baru ditambahkan. Ini memastikan setiap baris dalam tabel memiliki nilai unik.
- **PRIMARY KEY**: Kolom ini merupakan kunci utama tabel, yang berarti setiap nilai harus unik dan tidak boleh kosong (NULL). Kunci utama digunakan untuk mengidentifikasi setiap baris data secara unik dalam tabel.

2. **Nama_kategori (VARCHAR(100))**:

- **Tipe Data**: VARCHAR dengan panjang maksimum 100 karakter.
- **Deskripsi**: Kolom ini menyimpan nama kategori barang. Nama ini digunakan untuk mengidentifikasi kategori barang dalam sistem.

Tabel:barang
Kdbarang INT(11) AUTO_INCREMENT PRIMARY KEY
Kdkategori INT(11)
Kdsupplier INT(11)
Kdakun INT(11)
Nama_barang VARCHAR(100)
Barcode VARCHAR(35)
Deskripsi_singkat VARCHAR(255)
Tanggal_expired DATE
Harga_supplier INT(11)
Harga_grosir INT(11)
Harga_user INT(11)
Stok_barang INT(11)



1. **Kdbarang (INT(11) AUTO_INCREMENT PRIMARY KEY):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **AUTO_INCREMENT:** Nilai kolom ini akan secara otomatis bertambah setiap kali data baru ditambahkan. Ini memastikan setiap baris dalam tabel memiliki nilai unik.
- **PRIMARY KEY:** Kolom ini merupakan kunci utama tabel, yang berarti setiap nilai harus unik dan tidak boleh kosong (NULL). Kunci utama digunakan untuk mengidentifikasi setiap baris data secara unik dalam tabel.

2. **Kdkategori (INT(11)):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **Deskripsi:** Kolom ini menyimpan kode kategori barang. Kolom ini merupakan foreign key yang menghubungkan tabel barang dengan tabel kategori_barang. Ini memungkinkan setiap barang dikategorikan sesuai dengan kategori yang ada.

3. **Kdsupplier (INT(11)):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **Deskripsi:** Kolom ini menyimpan kode supplier barang. Kolom ini merupakan foreign key yang menghubungkan tabel barang dengan tabel Supplier. Ini memungkinkan untuk mengetahui pemasok dari setiap barang.

4. **Kdakun (INT(11)):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **Deskripsi:** Kolom ini menyimpan kode akun yang mengelola barang tersebut. Kolom ini merupakan foreign key yang menghubungkan tabel barang dengan tabel akun. Ini memungkinkan

untuk melacak siapa yang mengelola atau menambahkan barang ke dalam sistem.

5. **Nama_barang (VARCHAR(100)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 100 karakter.
- **Deskripsi:** Kolom ini menyimpan nama barang. Nama ini digunakan untuk mengidentifikasi barang dalam sistem.

6. **Barcode (VARCHAR(35)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 35 karakter.
- **Deskripsi:** Kolom ini menyimpan kode barcode barang. Barcode digunakan untuk mempermudah identifikasi dan pengelolaan barang secara otomatis.

7. **Deskripsi_singkat (VARCHAR(255)):**

- **Tipe Data:** VARCHAR dengan panjang maksimum 255 karakter.
- **Deskripsi:** Kolom ini menyimpan deskripsi singkat mengenai barang. Deskripsi ini memberikan informasi tambahan tentang barang.

8. **Tanggal_expired (DATE):**

- **Tipe Data:** DATE.
- **Deskripsi:** Kolom ini menyimpan tanggal kedaluwarsa barang. Informasi ini penting untuk barang-barang yang memiliki masa simpan tertentu, seperti makanan atau obat-obatan.

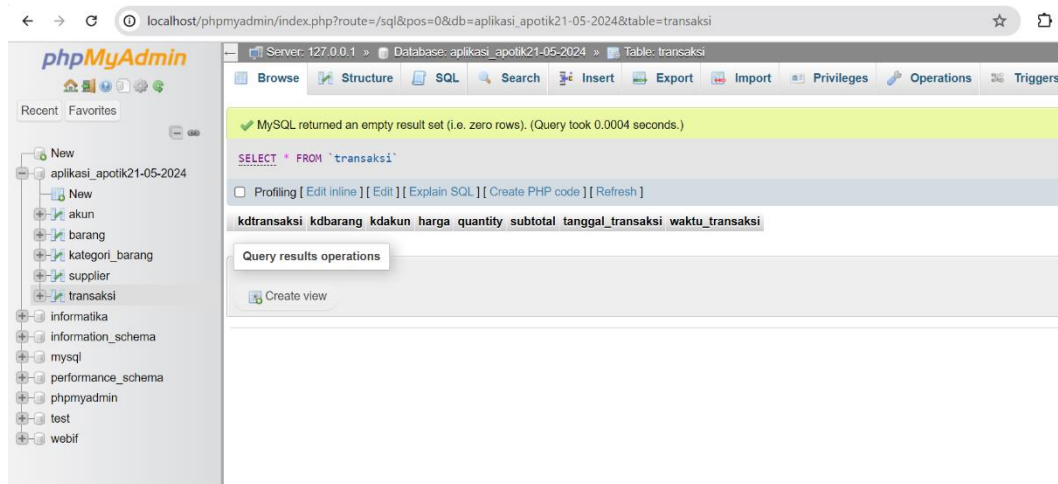
9. **Harga_supplier (INT(11)):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **Deskripsi:** Kolom ini menyimpan harga barang dari supplier. Informasi ini digunakan untuk mencatat biaya pembelian barang dari pemasok.

Tabel:transaksi
Kdtransaksi INT(11)
Kdbarang INT(11)
Kdakun INT(11)
Harga INT(11)
Jumlah_pembelian INT(11)
Subtotal INT(11)

Tanggal_transaksi DATE

Waktu_transaksi TIME



1. **Kdtransaksi (INT(11)):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **Deskripsi:** Kolom ini menyimpan kode transaksi. Biasanya, ini adalah kunci utama tabel untuk mengidentifikasi setiap transaksi secara unik, namun dalam struktur ini tidak ada penanda PRIMARY KEY atau AUTO_INCREMENT.

2. **Kdbarang (INT(11)):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **Deskripsi:** Kolom ini menyimpan kode barang yang ditransaksikan. Kolom ini merupakan foreign key yang menghubungkan tabel transaksi dengan tabel barang. Ini memungkinkan pelacakan barang apa yang terlibat dalam transaksi.

3. **Kdakun (INT(11)):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **Deskripsi:** Kolom ini menyimpan kode akun yang melakukan transaksi. Kolom ini merupakan foreign key yang menghubungkan tabel transaksi dengan tabel akun. Ini memungkinkan pelacakan siapa yang melakukan transaksi.

4. **Harga (INT(11)):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **Deskripsi:** Kolom ini menyimpan harga barang per unit saat transaksi. Ini digunakan untuk menghitung total harga berdasarkan jumlah pembelian.

5. **Jumlah_pembelian (INT(11)):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **Deskripsi:** Kolom ini menyimpan jumlah barang yang dibeli dalam transaksi.

6. **Subtotal (INT(11)):**

- **Tipe Data:** INT (Integer) dengan panjang maksimum 11 digit.
- **Deskripsi:** Kolom ini menyimpan total harga untuk transaksi tersebut. Subtotal dihitung sebagai hasil dari Harga dikalikan Jumlah_pembelian.

7. **Tanggal_transaksi (DATE):**

- **Tipe Data:** DATE.
- **Deskripsi:** Kolom ini menyimpan tanggal ketika transaksi terjadi. Ini digunakan untuk pelacakan dan pelaporan transaksi berdasarkan tanggal.

8. **Waktu_transaksi (TIME):**

- **Tipe Data:** TIME.
- **Deskripsi:** Kolom ini menyimpan waktu ketika transaksi terjadi. Ini digunakan untuk pelacakan lebih detail mengenai waktu transaksi.

C. BAHASA PEMROGRAMAN YANG DIGUNAKAN

1. Python

Python merupakan bahasa pemrograman komputer yang biasa dipakai untuk membangun situs, software/aplikasi, mengotomatiskan tugas dan melakukan analisis data. Bahasa pemrograman ini termasuk bahasa tujuan umum. Artinya, ia bisa digunakan untuk membuat berbagai program berbeda, bukan khusus untuk masalah tertentu saja.

Kelompok kami menggunakan bahasa *python* ini karena sifatnya yang serba guna dan mudah digunakan, ia menjadi bahasa pemrograman yang paling banyak digunakan. Terutama untuk kami yang masih pemula. Bahasa pemrograman ini bahkan bisa digunakan oleh pemula untuk mengotomatiskan tugas-tugas sederhana di komputer. Misalnya, mengganti nama file, mencari atau mengunduh konten online hingga mengirimkan email atau pesan dengan interval waktu yang diinginkan. Selain itu, *library* yang berfokus pada data seperti pandas, NumPy dan matplotlib membuatnya sangat mudah memproses,

memanipulasi serta memvisualisasikan data. Itulah sebabnya kelompok kami suka menggunakannya dalam analisis data.

2. JAVASCRIPT

JavaScript adalah bahasa pemrograman yang sangat populer untuk pengembangan aplikasi web dan memiliki banyak keunggulan dalam pembuatan sistem aplikasi apotik. Berikut adalah beberapa alasan mengapa JavaScript merupakan pilihan yang baik:

1. Interaktivitas dan Responsivitas

JavaScript memungkinkan pengembangan antarmuka pengguna yang interaktif dan responsif. Ini sangat penting untuk aplikasi apotik, di mana pengguna mungkin perlu berinteraksi dengan berbagai elemen UI seperti formulir, tabel produk, dan grafik stok barang.

2. Pengembangan Front-End dan Back-End

JavaScript dapat digunakan untuk pengembangan front-end dan back-end:

- **Front-End:** JavaScript bersama dengan HTML dan CSS memungkinkan pembuatan antarmuka pengguna yang dinamis dan menarik. Framework dan pustaka seperti React, Angular, dan Vue.js mempermudah pengembangan aplikasi front-end.
- **Back-End:** Dengan Node.js, JavaScript juga dapat digunakan untuk mengembangkan server dan API, memungkinkan Anda untuk menggunakan satu bahasa pemrograman untuk seluruh stack aplikasi.

3. Ekosistem dan Pustaka yang Kaya

JavaScript memiliki ekosistem yang sangat besar dengan banyak pustaka dan framework yang tersedia untuk berbagai kebutuhan, seperti:

- **React:** Untuk pembuatan antarmuka pengguna yang dinamis.
- **Angular:** Framework yang lengkap untuk pengembangan aplikasi web.
- **Vue.js:** Framework progresif untuk membangun antarmuka pengguna.

- **Express:** Framework untuk mengembangkan aplikasi back-end dengan Node.js.

3. HTML

HTML (HyperText Markup Language) bukanlah bahasa pemrograman, melainkan bahasa markup yang digunakan untuk membuat dan menyusun struktur halaman web. Namun, HTML sangat penting dalam pengembangan aplikasi apotik berbasis web karena beberapa alasan:

1. Struktur Dasar Halaman Web

HTML menyediakan kerangka dasar dari setiap halaman web, memungkinkan Anda untuk menentukan elemen-elemen seperti:

- **Judul (Headings):** <h1>, <h2>, <h3>, dll.
- **Paragraf (Paragraphs):** <p>
- **Daftar (Lists):** , ,
- **Tabel (Tables):** <table>, <tr>, <td>
- **Formulir (Forms):** <form>, <input>, <button>

2. Integrasi dengan CSS dan JavaScript

HTML bekerja sama dengan CSS (Cascading Style Sheets) dan JavaScript untuk memberikan tampilan dan interaktivitas pada halaman web:

- **CSS:** Mengatur gaya dan tata letak elemen HTML, seperti warna, font, dan penempatan elemen.
- **JavaScript:** Menambahkan interaktivitas dan dinamika pada halaman web, seperti validasi formulir, animasi, dan interaksi pengguna.

3. Kompatibilitas dan Keterbacaan

HTML didukung oleh semua browser web modern, menjadikannya pilihan universal untuk membuat aplikasi web. Keterbacaan HTML juga memudahkan pengembang untuk bekerja sama dan memelihara kode.

4. Aksesibilitas

HTML mendukung pembuatan halaman web yang dapat diakses oleh semua pengguna, termasuk mereka yang menggunakan teknologi bantu. Penggunaan tag semantik (seperti <header>, <nav>, <main>, <footer>) membantu dalam menciptakan struktur yang jelas dan dapat diakses.

5. SEO (Search Engine Optimization)

HTML memungkinkan Anda untuk mengoptimalkan konten web untuk mesin pencari, sehingga aplikasi apotik Anda lebih mudah ditemukan oleh pengguna.

6. Pengembangan Front-End

Dalam pembuatan sistem aplikasi apotik, HTML digunakan untuk membangun antarmuka pengguna (UI) di mana pengguna dapat:

- Melihat informasi produk dan stok barang.
- Melakukan pencarian produk.
- Melihat dan mengelola transaksi.
- Mengelola data akun dan supplier.

4. CSS

Alasan kelompok kami menggunakan CSS yaitu:

1. Mempercepat Loading Halaman Web

Jika mengatur tampilan website dengan CSS, kecepatan loading website bisa meningkat. Karena kita bisa menuliskan satu rangkaian kode untuk beberapa halaman website sekaligus, jumlah kode bisa diminimalkan. Dengan begitu, beban pada saat proses loading website lebih kecil.

2. Memudahkan Pengelolaan Kode

Dengan CSS, kita tidak perlu merubah kode di setiap halaman apabila ingin mengganti tampilan website. Sebagai contoh, kita ingin mengubah latar belakang semua halaman website. Maka, cukup edit kode CSS terkait latar belakang, perubahan itu akan diterapkan di semua halaman.

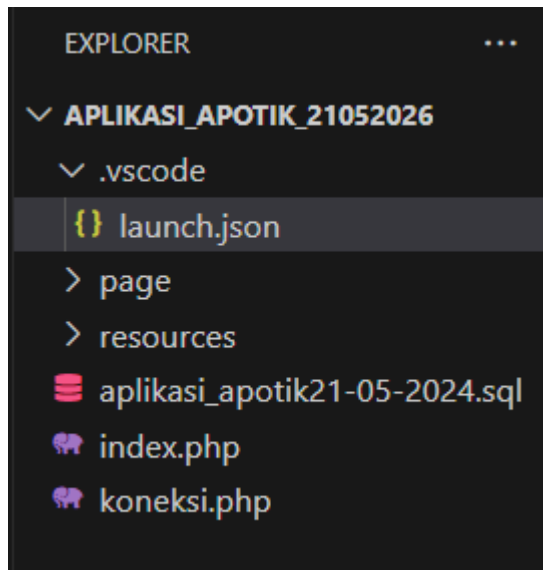
3. Menawarkan Lebih Banyak Variasi Tampilan

HTML adalah bahasa yang dapat untuk mengatur tampilan halaman website, tetapi terbatas. Nah, CSS menawarkan lebih banyak *style* tampilan, sehingga kita bisa lebih bebas membuat antarmuka website. Contohnya, Anda bisa menggunakan CSS untuk membuat tombol dengan warna yang Anda inginkan, atau membuat menu dropdown CSS.

4. Membuat Website Tampil Rapi di Semua Ukuran Layar

Fungsi CSS yang tidak kalah menarik adalah membuat tampilan website optimal di berbagai ukuran layar. Baik itu di laptop maupun di smartphone, karena CSS memiliki berbagai *property* untuk mengatur tampilan konten sesuai kebutuhan layar, misalnya dengan `max-width`. Ketika menggunakan *property* ini mengubah ukuran elemen HTML sesuai ukuran layar yang digunakan untuk menampilkan website.

D. CODINGAN



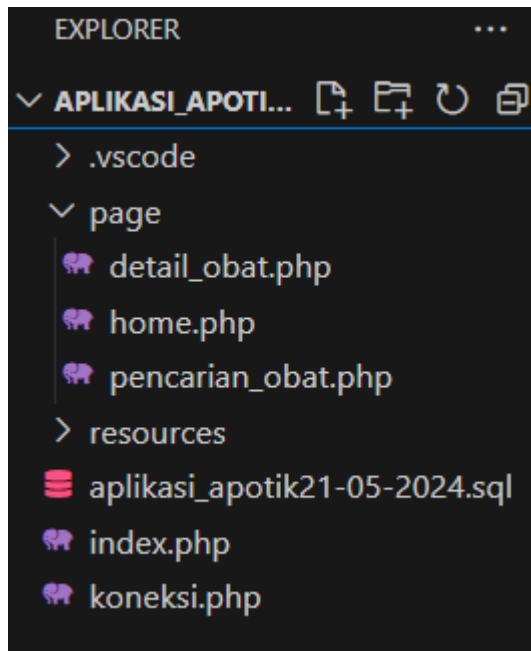
FILE .VSCODE

```
{  
  
  // Use IntelliSense to learn about possible attributes.  
  // Hover to view descriptions of existing attributes.  
  // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387  
  "version": "0.2.0",
```

```

"configurations": [
  {
    "type": "chrome",
    "request": "launch",
    "name": "Launch Chrome against localhost",
    "url": "http://localhost:8080",
    "webRoot": "${workspaceFolder}"
  }
]
}

```



FILE PAGE

detail_obat.php:

```

<?php
include "koneksi.php";

if (isset($_GET['kdbarang'])) {
    $kdbarang = mysqli_real_escape_string($koneksi,
    $_GET['kdbarang']);
    $barang = mysqli_query($koneksi, "SELECT a.*, b.nama_kategori
    FROM barang a LEFT JOIN kategori_barang b ON
    a.kdkategori=b.kdkategori WHERE a.kdbarang='$kdbarang'");
    $cekdata = mysqli_num_rows($barang);

    if ($cekdata == 0) {

```

```

        echo "<h3 style='color:red'>Maaf!! Data yang Anda cari tidak
ditemukan</h3>";
    } else {
        $stampildata = mysqli_fetch_array($barang);
    }
} else {
    echo "<h3 style='color:red'>Data tidak valid</h3>";
}
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Detail Obat</title>
    <link rel="stylesheet" href="resources/css/style.css">
</head>
<body>
    <div class="detail-container">
        <h2>Detail Obat</h2>
        <div class="row">
            <div class="col-6">
                
            </div>
            <div class="col-6">
                <div class="row">
                    <label class="col-4">Nama Barang</label>
                    <div class="col-8 produk_judul"><?php echo
$stampildata['nama_barang']; ?></div>
                </div>
                <div class="row">
                    <label class="col-4">Kategori Barang</label>
                    <div class="col-8"><?php echo
$stampildata['nama_kategori']; ?></div>
                </div>
                <div class="row">
                    <label class="col-4">Deskripsi Singkat</label>
                    <div class="col-8 produk_deskripsi"><?php echo
$stampildata['deskripsi_singkat']; ?></div>
                </div>
                <div class="row">
                    <label class="col-4">Tanggal Expired</label>
                    <div class="col-8 produk_deskripsi"><?php echo
$stampildata['tanggal_expired']; ?></div>
                </div>
                <div class="row">
                    <label class="col-4">Kode Barcode</label>
                    <div class="col-8 produk_deskripsi"><?php echo
$stampildata['barcode']; ?></div>

```

```

        </div>
        <div class="row">
            <label class="col-4">Sisa Stok</label>
            <div class="col-8 produk_deskripsi"><?php echo
isset($tampildata['stok_barang']) ? $tampildata['stok_barang'] :
'Data tidak tersedia'; ?></div>
        </div>
        <div class="row">
            <label class="col-4">Harga User</label>
            <div class="col-8 produk_harga">Rp. <?php echo
number_format($tampildata['harga_user'], 0, ',', '.'); ?></div>
        </div>
    </div>
    </div>

    <div class="action-links">
        <a href="edit_barang.php?kdbarang=<?php echo
$tampildata['kdbarang']; ?>">Edit Barang</a>
        <a href=".">KEMBALI</a>
    </div>
</div>
</body>
</html>

```

home.php:

```

<h1>Obat terbaru</h1>

<div class="produk_grid">
    <?php
    // Query untuk mendapatkan daftar obat terbaru, dibatasi 6 item
    $home_produk = mysqli_query($koneksi, "SELECT DISTINCT
nama_barang, kdbarang, gambar, deskripsi_singkat, harga_user FROM
barang ORDER BY kdbarang");
    while ($home_produk_tampil = mysqli_fetch_array($home_produk)) {
        ?>
        <div class="produk_kotak">
            
            <a href="?page=detail_obat&&kdbarang=<?php echo
$home_produk_tampil['kdbarang']; ?>">
                <div class="produk_judul"><?php echo
$home_produk_tampil['nama_barang']; ?></div>
                </a>
                <div class="produk_deskripsi"><?php echo
$home_produk_tampil['deskripsi_singkat']; ?>....</div>
                <div class="produk_harga">Rp. <?php echo
number_format($home_produk_tampil['harga_user'], 0, ',', '.');
?></div>
            </div>
        <?php
    }
}

```

```

    }
    ?>
</div>

```

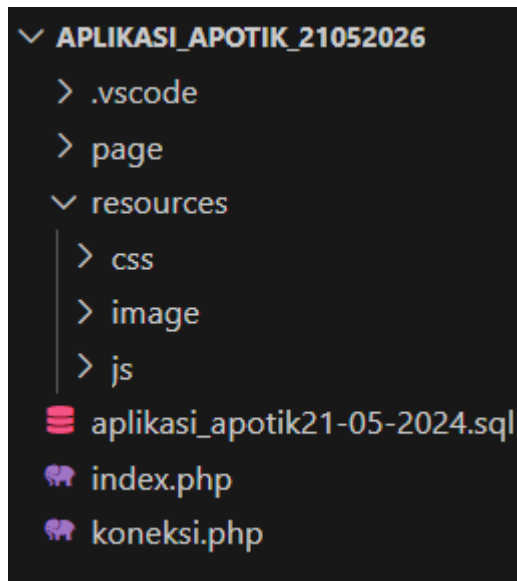
pencarian_obat.php:

```
<h3>Pencarian obat</h3>
```

```

<?php
    // Mengambil nilai nama_barang dari input pencarian dan
    mengamankannya
    $nama_barang = mysqli_real_escape_string($koneksi,
    @$_POST['nama_barang']);
    // Query untuk mencari obat berdasarkan nama_barang
    $home_produk = mysqli_query($koneksi, "SELECT * FROM barang
    WHERE nama_barang LIKE '%$nama_barang%' ORDER BY kdbarang DESC ");
    $cekproduk = mysqli_num_rows($home_produk);
    if ($cekproduk == 0) {
        echo "<h3 style='color:red'>Maaf!! Data Yang anda cari tidak
    di temukan</h3>";
    }
    while ($home_produk_tampil = mysqli_fetch_array($home_produk)) {
        ?>
        <div class="col-3">
            <div class="produk_kotak">
                
                <a href="?page=detail_obat&&kdbarang=<?php echo
    $home_produk_tampil['kdbarang']; ?>">
                    <div class="produk_judul"><?php echo
    $home_produk_tampil['nama_barang']; ?></div>
                    </a>
                </div>
                <div class="produk_deskripsi"><?php echo
    $home_produk_tampil['deskripsi_singkat']; ?>....</div>
                <div class="produk_harga">Rp. <?php echo
    number_format($home_produk_tampil['harga_user'], 0, ',', '.');
    ?></div>
            </div>
        <?php
    }
    ?>
</div>

```

FILE RESOURCE

CSS: (style css)

```
/* Reset box-sizing */
* {
  box-sizing: border-box;
  margin: 0;
  padding: 0;
  font-family: "Arial", sans-serif;
}

/* Utility classes */
.row::after {
  content: "";
  clear: both;
  display: table;
}

[class*="col-"] {
  float: left;
  padding: 15px;
}

.col-1 {
  width: 8.33%;
}
.col-2 {
  width: 16.66%;
}
.col-3 {
  width: 25%;
}
.col-4 {
  width: 33.33%;
}
```

```

.col-5 {
    width: 41.66%;
}
.col-6 {
    width: 50%;
}
.col-7 {
    width: 58.33%;
}
.col-8 {
    width: 66.66%;
}
.col-9 {
    width: 75%;
}
.col-10 {
    width: 83.33%;
}
.col-11 {
    width: 91.66%;
}
.col-12 {
    width: 100%;
}

/* Header section */
.header {
    background: url("../image/anime1.jpeg") no-repeat center center;
    background-size: cover;
    background-attachment: fixed;
    padding: 20px;
    height: auto; /* Adjust height to fit content */
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
    display: flex;
    flex-direction: column;
    align-items: center; /* Center horizontally */
    text-align: center;
}

.header_title {
    font-size: 24px;
    color: white;
    font-weight: bold;
    text-transform: uppercase;
    margin-bottom: 20px;
}

.header_search {
    display: flex;
    justify-content: center;
    align-items: center;
    gap: 10px; /* Space between input and button */
}

```

```

}

.header_pencarian_input {
  padding: 10px 15px;
  border: 2px solid #00bfff; /* Bright blue */
  border-radius: 25px;
  width: 300px; /* Set a fixed width for better alignment */
  transition: all 0.3s ease;
}

.header_pencarian_input:focus {
  border-color: #87cefa; /* Light sky blue */
  box-shadow: 0 0 5px rgba(135, 206, 250, 0.5);
}

.header_pencarian_button {
  padding: 10px 20px;
  background: #00bfff; /* Bright blue */
  border: none;
  border-radius: 25px;
  color: white;
  cursor: pointer;
  transition: background 0.3s ease;
  font-size: 16px; /* Adjust font size */
}

.header_pencarian_button:hover {
  background: #87cefa; /* Sky blue */
}

/* Navigation menu */
.menu_atas {
  background: #4682b4; /* Steel blue */
  padding: 15px 0;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

.menu_atas_link {
  color: white;
  text-decoration: none;
  padding: 15px 20px;
  transition: all 0.3s ease;
}

.menu_atas_link:hover {
  background: #87cefa; /* Sky blue */
  color: black;
  border-radius: 5px;
}

/* Content section */
.konten {

```

```

    background: #f1f1f1;
    padding: 20px 0;
}

.konten_data {
    background: #fff;
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}

/* Grid layout for products */
.produk_grid {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
    gap: 20px;
}

/* Product box */
.produk_kotak {
    padding: 10px;
    border: 1px solid #f1f1f1;
    border-radius: 10px;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
    transition: transform 0.3s ease, box-shadow 0.3s ease;
}

.produk_kotak:hover {
    transform: translateY(-5px);
    box-shadow: 0 8px 12px rgba(0, 0, 0, 0.2);
}

.produk_judul {
    font-size: 20px;
    color: #00bfff; /* Bright blue */
    font-weight: bold;
    margin: 10px 0;
    transition: color 0.3s ease;
}

.produk_judul:hover {
    color: #87cefa; /* Sky blue */
}

.produk_deskripsi {
    padding: 10px 0;
    color: #666;
    font-size: 12px;
}

.produk_harga {
    font-size: 24px;

```

```

        color: #00bfff; /* Bright blue */
        font-weight: bold;
    }

    /* Background for edit_barang.php */
    body {
        background: url("../image/anime1.jpeg") no-repeat center center
        fixed;
        background-size: cover;
        color: white; /* Ensure text is visible against the background */
        padding: 20px;
    }

    h3 {
        text-align: center;
        color: #fff;
    }

    form {
        background: rgba(
            0,
            0,
            0,
            0.7
        ); /* Semi-transparent background for better text visibility */
        padding: 20px;
        border-radius: 10px;
    }

    .row {
        margin-bottom: 15px;
    }

    .row label {
        display: inline-block;
        width: 25%;
        color: #fff;
    }

    .row input,
    .row textarea {
        width: 70%;
        padding: 10px;
        border-radius: 5px;
        border: none;
    }

    /* Background for detail_obat.php */
    body.detail_obat {
        background: url("../image/anime1.jpeg") no-repeat center center
        fixed;
        background-size: cover;
    }

```

```

    color: white; /* Ensure text is visible against the background */
    padding: 20px;
}

/* Background for index.php */
body.index {
    background: url("resources/image/anime1.jpeg") no-repeat center
    center fixed;
    background-size: cover;
}

```

IMAGE (PRODUK):





Edit_barang.php:

```
<?php
include "koneksi.php";
////////////////////////////////////
if (isset($_GET['kdbarang'])) {
    $kdbarang = mysqli_real_escape_string($koneksi,
$_GET['kdbarang']);
    $query = mysqli_query($koneksi, "SELECT * FROM barang WHERE
kdbarang='$kdbarang'");
    $data = mysqli_fetch_array($query);
}
?>

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <title>Edit Barang</title>
    <link rel="stylesheet" type="text/css"
href="resources/css/style.css">
</head>
<body>
    <h3>Edit Barang</h3>
    <form action="proses_edit_barang.php" method="post"
enctype="multipart/form-data">
        <input type="hidden" name="kdbarang" value="<?php echo
$data['kdbarang']; ?>">
        <div class="row">
            <label class="col-4">Nama Barang</label>
            <input type="text" name="nama_barang" value="<?php echo
$data['nama_barang']; ?>" required>
        </div>
```

```

        <div class="row">
            <label class="col-4">Kategori Barang</label>
            <input type="text" name="kdkategori" value="<?php echo
$data['kdkategori']; ?>" required>
        </div>
        <div class="row">
            <label class="col-4">Deskripsi Singkat</label>
            <textarea name="deskripsi_singkat" required><?php echo
$data['deskripsi_singkat']; ?></textarea>
        </div>
        <div class="row">
            <label class="col-4">Tanggal Expired</label>
            <input type="date" name="tanggal_expired" value="<?php
echo $data['tanggal_expired']; ?>" required>
        </div>
        <div class="row">
            <label class="col-4">Kode Barcode</label>
            <input type="text" name="barcode" value="<?php echo
$data['barcode']; ?>" required>
        </div>
        <div class="row">
            <label class="col-4">Harga User</label>
            <input type="number" name="harga_user" value="<?php echo
$data['harga_user']; ?>" required>
        </div>
        <div class="row">
            <label class="col-4">Sisa Stok</label>
            <input type="number" name="stok_barang" value="<?php
echo $data['stok_barang']; ?>" required>
        </div>
        <div class="row">
            <label class="col-4">Gambar</label>
            <input type="file" name="gambar">
        </div>
        <div class="row">
            <input type="submit" name="update" value="Update
Barang">
        </div>
    </form>
</body>
</html>

```

Index.php:

```

<?php
// file untuk koneksi ke database
include "koneksi.php";
?>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">

```



```

    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <title>Aplikasi Apotik</title>
    <!-- file CSS eksternal untuk styling -->
    <link rel="stylesheet" type="text/css"
href="resources/css/style.css">
</head>
<body>

    <!-- Bagian header -->
    <section class="header">
        <div class="container">
            <div class="row">
                <!-- ini adalah Judul apotek di sisi kiri header -->
                <div class="col-6">
                    <div class="header_title">Apotek Anime</div>
                </div>
                <!-- syntax pada form pencarian di sisi kanan header
-->
                <div class="col-6">
                    <form action="?page=pencarian_obat"
method="post">
                        <input type="text" name="nama_barang"
placeholder="Pencarian Data Obat Apotik Anime....."
class="header_pencarian_input">
                        <input type="submit" name="Cari"
value="Cari" class="header_pencarian_button">
                    </form>
                </div>
            </div>
        </div>
    </section>

    <!-- Bagian menu atas -->
    <section class="menu_atas">
        <div class="container">
            <a href="." class="menu_atas_link">Beranda</a>
            <a href="?page=kategori" class="menu_atas_link">Kategori
Obat</a>
            <a href="?page=informasi"
class="menu_atas_link">Informasi Apotik</a>
            <a href="?page=kontak" class="menu_atas_link">Hubungi
Kami</a>
            <a href="tambah_barang.php"
class="menu_atas_link">Tambah Barang</a> <!-- Menambahkan link untuk
halaman tambah barang -->
        </div>
    </section>

    <!-- Bagian konten utama -->
    <div class="konten">
        <div class="container">

```

```

        <div class="konten_data">
            <?php
                // Mendapatkan nilai parameter 'page' dari URL
                dan mengamankannya dengan mysqli_real_escape_string
                $page = isset($_GET['page']) ?
                mysqli_real_escape_string($koneksi, $_GET['page']) : 'home';

                // Memuat halaman yang sesuai dengan nilai
                parameter 'page'
                if ($page == "home") {
                    include "page/home.php";
                } elseif ($page == "detail_obat") {
                    include "page/detail_obat.php";
                } elseif ($page == "pencarian_obat") {
                    include "page/pencarian_obat.php";
                } elseif ($page == "kategori") {
                    include "page/kategori.php";
                } elseif ($page == "informasi") {
                    include "page/informasi.php";
                } elseif ($page == "kontak") {
                    include "page/kontak.php";
                } else {
                    include "page/home.php";
                }
            ?>
        </div>
    </div>
</div>

</body>
</html>

```

Koneksi.php:

```

<?php
// Mengatur koneksi ke database
$koneksi=mysqli_connect("localhost","root","","aplikasi_apotik21-05-
2024") or die("Maaf Koneksi ke database gagal");
?>

```

Proses_edit_barang.php:

```

<?php
include "koneksi.php";

if (isset($_POST['update'])) {
    $kdbarang = mysqli_real_escape_string($koneksi,
    $_POST['kdbarang']);
    $nama_barang = mysqli_real_escape_string($koneksi,
    $_POST['nama_barang']);

```

```

        $kdkategori = mysqli_real_escape_string($koneksi,
$_POST['kdkategori']);
        $deskripsi_singkat = mysqli_real_escape_string($koneksi,
$_POST['deskripsi_singkat']);
        $tanggal_expired = mysqli_real_escape_string($koneksi,
$_POST['tanggal_expired']);
        $barcode = mysqli_real_escape_string($koneksi,
$_POST['barcode']);
        $harga_user = mysqli_real_escape_string($koneksi,
$_POST['harga_user']);
        $stok = mysqli_real_escape_string($koneksi,
$_POST['stok_barang']);

        // Cek jika ada file gambar yang di-upload
        if ($_FILES['gambar']['name']) {
            $gambar = $_FILES['gambar']['name'];
            $tmp_name = $_FILES['gambar']['tmp_name'];
            $target_dir = "resources/image/produk/";
            move_uploaded_file($tmp_name, $target_dir.$gambar);

            // Update dengan gambar baru
            $query = "UPDATE barang SET nama_barang='$nama_barang',
kdkategori='$kdkategori', deskripsi_singkat='$deskripsi_singkat',
tanggal_expired='$tanggal_expired', barcode='$barcode',
harga_user='$harga_user', stok_barang='$stok', gambar='$gambar'
WHERE kdbarang='$kdbarang'";
        } else {
            // Update tanpa mengubah gambar
            $query = "UPDATE barang SET nama_barang='$nama_barang',
kdkategori='$kdkategori', deskripsi_singkat='$deskripsi_singkat',
tanggal_expired='$tanggal_expired', barcode='$barcode',
harga_user='$harga_user', stok_barang='$stok' WHERE
kdbarang='$kdbarang'";
        }

        $result = mysqli_query($koneksi, $query);

        if ($result) {
            echo "Barang berhasil diupdate.";
        } else {
            echo "Gagal mengupdate barang: " . mysqli_error($koneksi);
        }

        // Tambahkan tombol atau link untuk kembali ke beranda
        echo '<br><br>';
        echo '<a href="index.php" style="text-decoration: none; padding:
10px 20px; background-color: #4CAF50; color: white; border-radius:
5px;">Kembali ke Beranda</a>';
    }
?>

```

Proses_tambah_barang.php:

```

<?php
include "koneksi.php";

// Mengambil data dari form
$nama_barang = mysqli_real_escape_string($koneksi,
$_POST['nama_barang']);
$kdkategori = mysqli_real_escape_string($koneksi,
$_POST['kdkategori']);
$deskripsi_singkat = mysqli_real_escape_string($koneksi,
$_POST['deskripsi_singkat']);
$tanggal_expired = mysqli_real_escape_string($koneksi,
$_POST['tanggal_expired']);
$barcode = mysqli_real_escape_string($koneksi, $_POST['barcode']);
$harga_user = mysqli_real_escape_string($koneksi,
$_POST['harga_user']);
$gambar = $_FILES['gambar']['name'];
$target_dir = "resources/image/produk/";
$target_file = $target_dir . basename($gambar);

// Memindahkan file gambar ke direktori tujuan
if (move_uploaded_file($_FILES['gambar']['tmp_name'], $target_file))
{
    // Menyimpan data ke dalam database
    $query = "INSERT INTO barang (nama_barang, kdkategori,
deskripsi_singkat, tanggal_expired, barcode, harga_user, gambar)
VALUES ('$nama_barang', '$kdkategori',
'$deskripsi_singkat', '$tanggal_expired', '$barcode', '$harga_user',
'$gambar')";

    if (mysqli_query($koneksi, $query)) {
        echo "<script>alert('Barang berhasil ditambahkan');
window.location.href='index.php';</script>";
    } else {
        echo "<script>alert('Gagal menambahkan barang');
window.location.href='tambah_barang.php';</script>";
    }
} else {
    echo "<script>alert('Gagal mengupload gambar');
window.location.href='tambah_barang.php';</script>";
}
?>

```

Tambah_barang.php

```

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <title>Tambah Barang</title>

```

```

        <link rel="stylesheet" type="text/css"
href="resources/css/style.css">
</head>
<body>

    <section class="header">
        <div class="container">
            <div class="row">
                <div class="col-12">
                    <div class="header_title">Tambah Barang
Baru</div>
                </div>
            </div>
        </div>
    </section>

    <div class="konten">
        <div class="container">
            <div class="konten_data">
                <form action="proses_tambah_barang.php"
method="post" enctype="multipart/form-data">
                    <div class="row">
                        <label class="col-4">Nama Barang</label>
                        <input type="text" name="nama_barang"
required class="col-8">
                    </div>
                    <div class="row">
                        <label class="col-4">Kategori Barang</label>
                        <input type="text" name="kdkategori"
required class="col-8">
                    </div>
                    <div class="row">
                        <label class="col-4">Deskripsi
Singkat</label>
                        <input type="text" name="deskripsi_singkat"
required class="col-8">
                    </div>
                    <div class="row">
                        <label class="col-4">Tanggal Expired</label>
                        <input type="date" name="tanggal_expired"
required class="col-8">
                    </div>
                    <div class="row">
                        <label class="col-4">Kode Barcode</label>
                        <input type="text" name="barcode" required
class="col-8">
                    </div>
                    <div class="row">
                        <label class="col-4">Harga User</label>
                        <input type="number" name="harga_user"
required class="col-8">
                    </div>
                </form>
            </div>
        </div>
    </div>

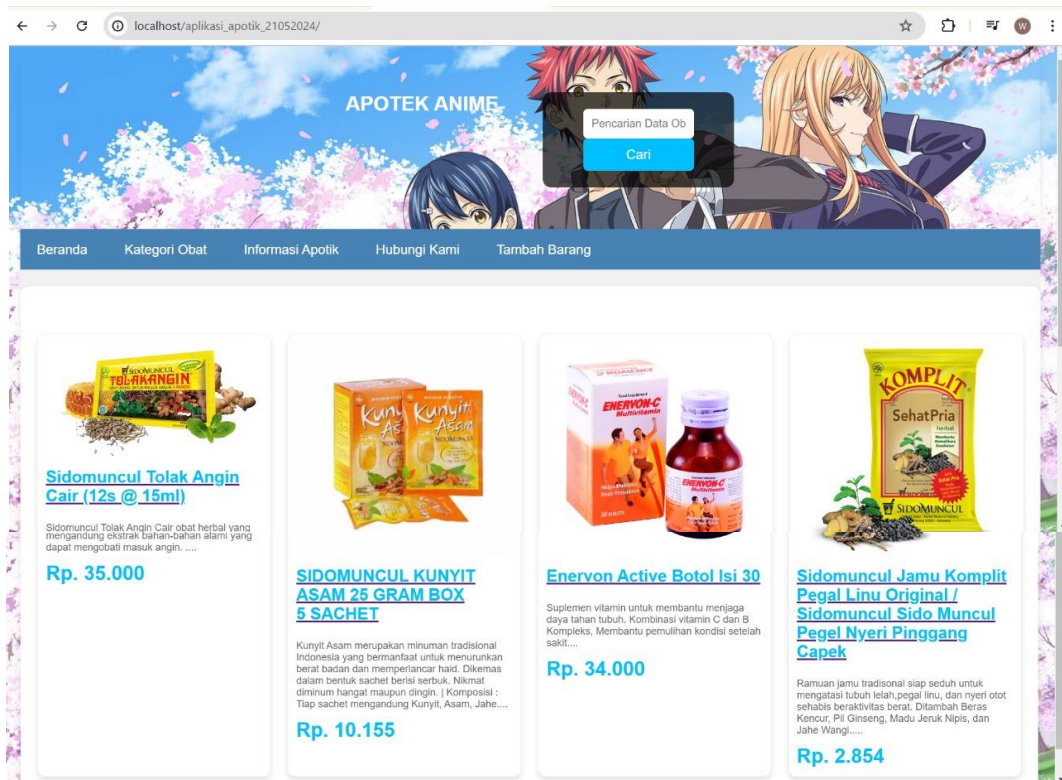
```

```

<div class="row">
  <label class="col-4">Gambar Barang</label>
  <input type="file" name="gambar" required
class="col-8">
</div>
<div class="row">
  <input type="submit" value="Tambah Barang"
class="col-12 header_pencarian_button">
</div>
</form>
</div>
</div>
</div>
</body>
</html>

```

E. OUTPUT FRONT END



TAMBAH BARANG

localhost/aplikasi_apotik_21052024/tambah_barang.php

Nama Barang	<input type="text"/>
Kategori Barang	<input type="text"/>
Deskripsi Singkat	<input type="text"/>
Tanggal Expired	hh/bb/tttt <input type="checkbox"/>
Kode Barcode	<input type="text"/>
Harga User	<input type="text"/>
Gambar Barang	<input type="button" value="Pilih File"/> Tidak ada file yang dipilih
<input type="button" value="Tambah Barang"/>	


TAMBAH BARANG BARU

Nama Barang	Paracetamol 500 Mg
Kategori Barang	Analgesik
Deskripsi Singkat	Paracetamol adalah obat yang digunakan untuk meredakan nyeri ringan hingga sedang. Obat ini juga berfungsi sebagai penurun demam. Perlu diketahui, paraceta
Tanggal Expired	16/05/2026 <input type="checkbox"/>
Kode Barcode	6009602980011
Harga User	22000
Gambar Barang	<input type="button" value="Choose File"/> paracetamol.jpg
<input type="button" value="Tambah Barang"/>	

DETAIL OBAT

localhost/Aplikasi_apotik_21052024/?page=detail_obat&&kdbarang=1

Beranda Kategori Obat Informasi Apotik Hubungi Kami Tambah Barang



Tolak Angin Cair Dus 12's Herbal - Masuk Angin Daya Tahan Tubuh

Tolak Angin merupakan Obat Herbal Terstandar (OHT) yang diproduksi di pabrik yang terstandar GMP (Good Manufacturing Product), ISO (International Organization of Standardization), dan HACCP (Hazard Analysis Critical Control Point).

2025-02-01

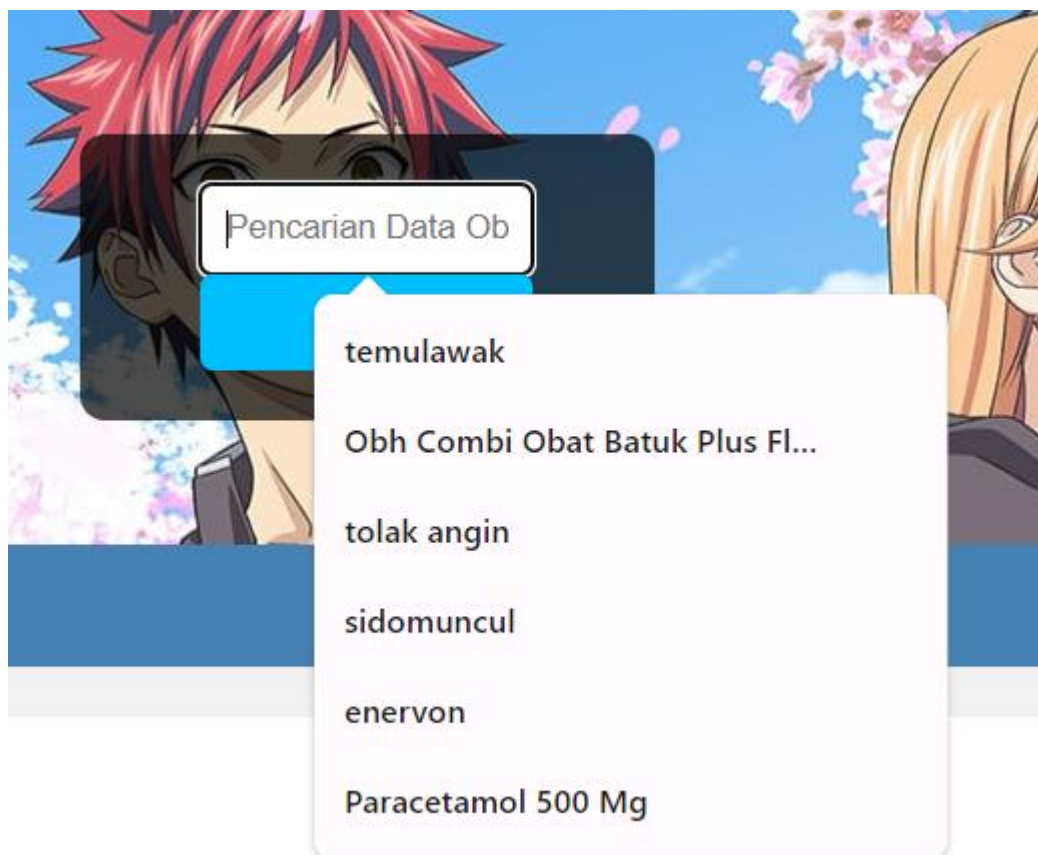
57894367

300

Rp. 44.500

[Edit Barang KEMBALI](#)

PENCARIAN DATA OBAT



Pencarian Data Ob

- temulawak
- Obh Combi Obat Batuk Plus Fl...
- tolak angin
- sidomuncul
- enervon
- Paracetamol 500 Mg

BAB III

PENUTUP

A. KESIMPULAN

Pengembangan sistem aplikasi apotik adalah upaya penting untuk meningkatkan efisiensi operasional, manajemen persediaan, dan pelayanan pelanggan di apotik. Melalui pembuatan aplikasi berbasis web, berbagai aspek operasional apotik dapat diotomatisasi dan disederhanakan. Dari kajian dan pengembangan yang telah dilakukan, dapat disimpulkan bahwa:

1. **Kemudahan Penggunaan:** Dengan menggunakan bahasa pemrograman seperti Python, JavaScript, dan HTML, aplikasi apotik dapat dibuat dengan antarmuka yang user-friendly, mudah dipahami dan digunakan oleh staf apotik.
2. **Efisiensi Operasional:** Aplikasi ini memungkinkan pengelolaan data yang lebih efisien, termasuk data produk, supplier, transaksi, dan akun pengguna. Otomatisasi ini mengurangi risiko kesalahan manusia dan meningkatkan kecepatan operasional.
3. **Keamanan dan Skalabilitas:** Dengan menggunakan framework yang tepat seperti Django untuk Python dan Express untuk Node.js, aplikasi apotik dapat dibangun dengan keamanan yang baik dan kemampuan untuk menangani pertumbuhan data dan pengguna di masa depan.
4. **Interaktivitas dan Responsivitas:** Penggunaan JavaScript, khususnya melalui framework seperti React atau Vue.js, memungkinkan aplikasi untuk memberikan pengalaman pengguna yang interaktif dan responsif.
5. **Integrasi dan Komunikasi Real-Time:** Aplikasi dapat diintegrasikan dengan berbagai layanan dan perangkat lain, serta mendukung komunikasi real-time untuk notifikasi stok dan transaksi.

B. SARAN

Untuk lebih meningkatkan kualitas dan manfaat dari sistem aplikasi apotik yang telah dikembangkan, berikut adalah beberapa saran yang dapat dipertimbangkan:

1. **Pengujian dan Pemeliharaan:** Lakukan pengujian secara menyeluruh dan berkelanjutan untuk memastikan aplikasi berfungsi dengan baik dan bebas dari bug. Juga, siapkan rencana pemeliharaan rutin untuk menangani masalah teknis dan memperbarui fitur sesuai kebutuhan.
2. **Pelatihan Pengguna:** Berikan pelatihan kepada staf apotik tentang cara menggunakan aplikasi dengan efektif. Ini akan memastikan bahwa mereka dapat memanfaatkan semua fitur yang ada dan mengurangi kesalahan operasional.
3. **Pengembangan Fitur Tambahan:** Pertimbangkan untuk menambahkan fitur tambahan seperti analisis data penjualan, integrasi dengan sistem pembayaran digital, dan fitur manajemen inventaris yang lebih canggih.
4. **Keamanan Data:** Pastikan bahwa semua data yang disimpan dan diproses oleh aplikasi dilindungi dengan baik. Implementasikan enkripsi data, kontrol akses yang ketat, dan pemantauan keamanan untuk mencegah kebocoran data dan akses tidak sah.
5. **Feedback Pengguna:** Kumpulkan umpan balik dari pengguna aplikasi secara berkala untuk memahami kekurangan dan area yang perlu diperbaiki. Umpan balik ini sangat berharga untuk pengembangan versi aplikasi berikutnya.
6. **Dokumentasi yang Baik:** Sediakan dokumentasi yang lengkap dan mudah diakses untuk semua aspek aplikasi, termasuk panduan pengguna, dokumentasi teknis, dan panduan instalasi. Dokumentasi yang baik akan membantu dalam pemeliharaan dan pengembangan lebih lanjut.