

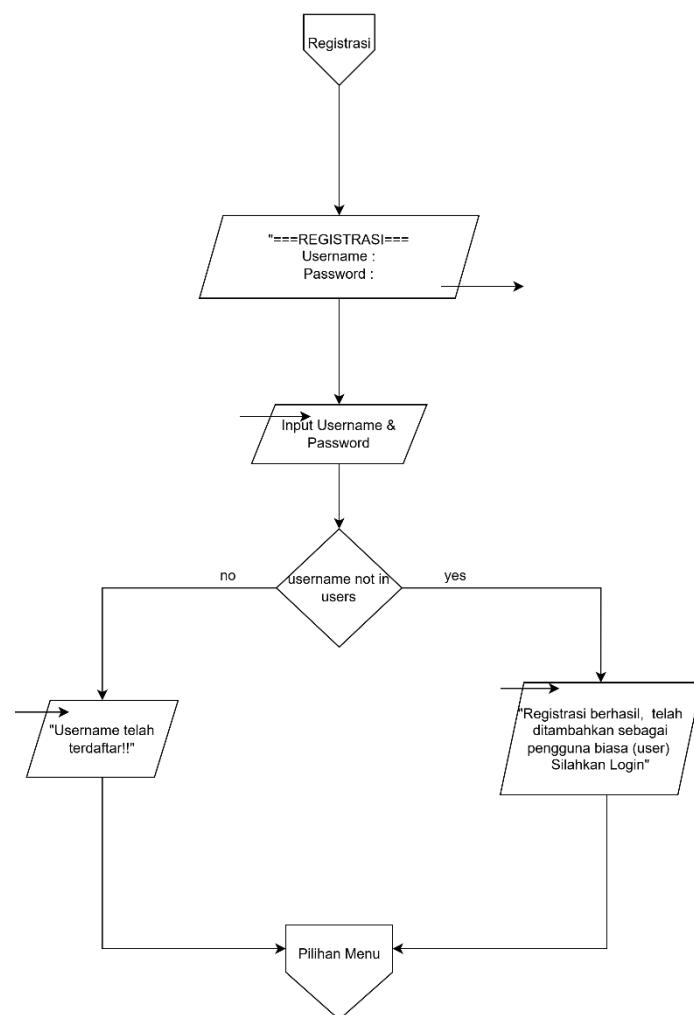
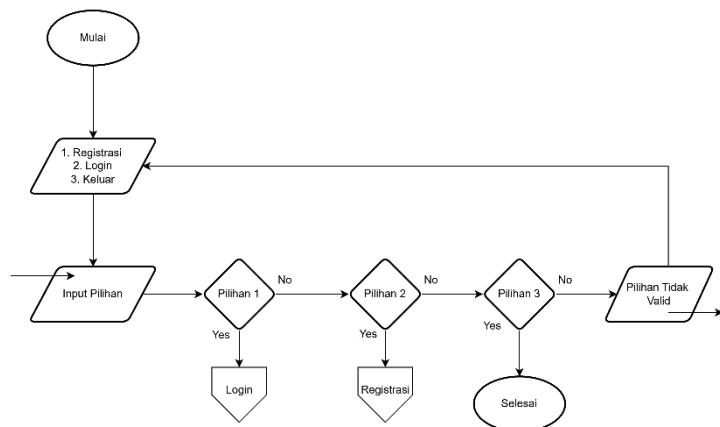
**LAPORAN PRAKTIKUM
POSTTEST 5
ALGORITMA PEMROGRAMAN DASAR**

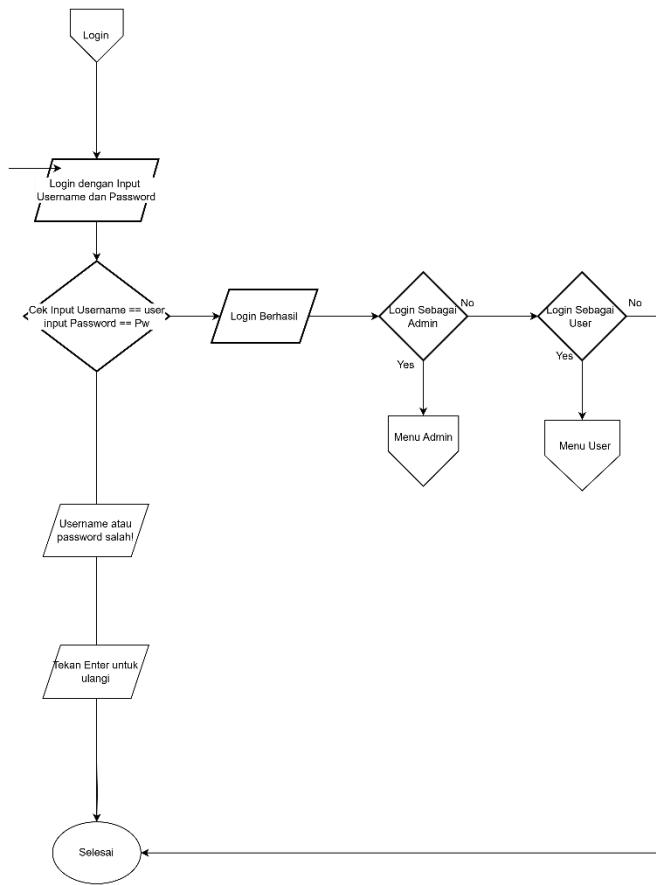


Disusun oleh:
Aditya Fitriansyah (2509106002)
Kelas A1'25

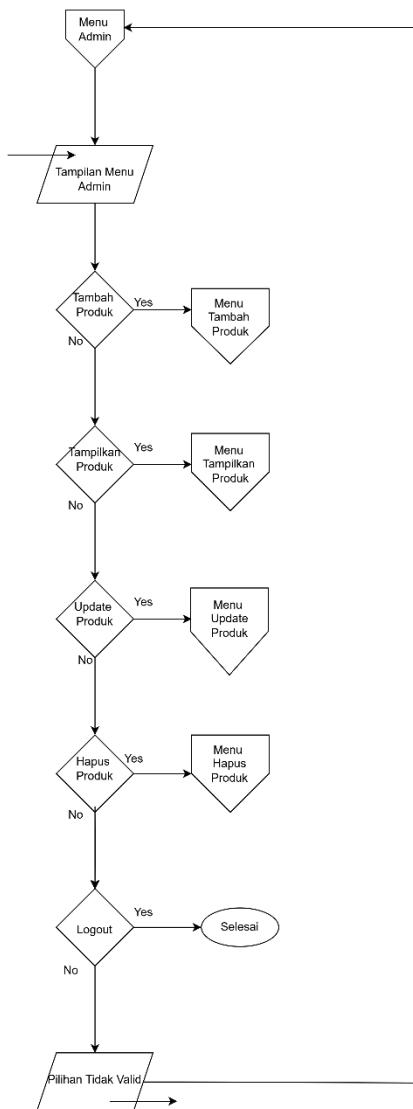
**PROGRAM STUDI INFORMATIKA
UNIVERSITAS MULAWARMAN
SAMARINDA
2025**

1. Flowchart

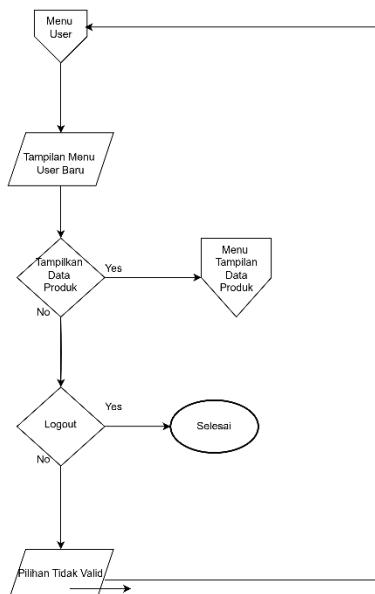




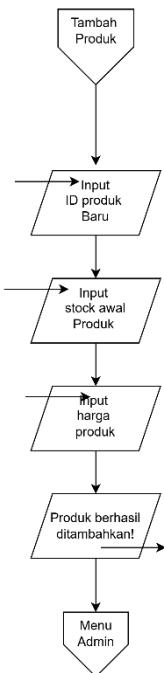
Flowchart 1.3 Login



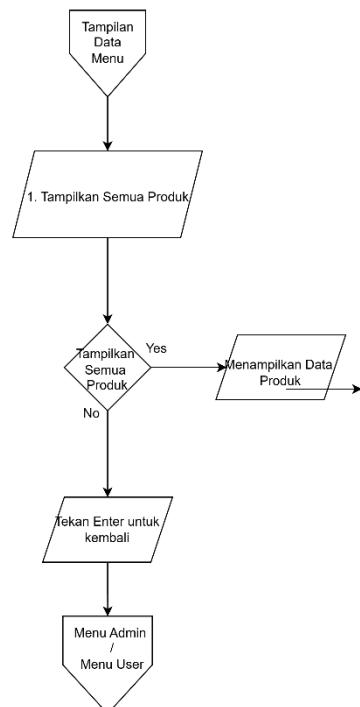
Flowchart 1.4 Menu Admin



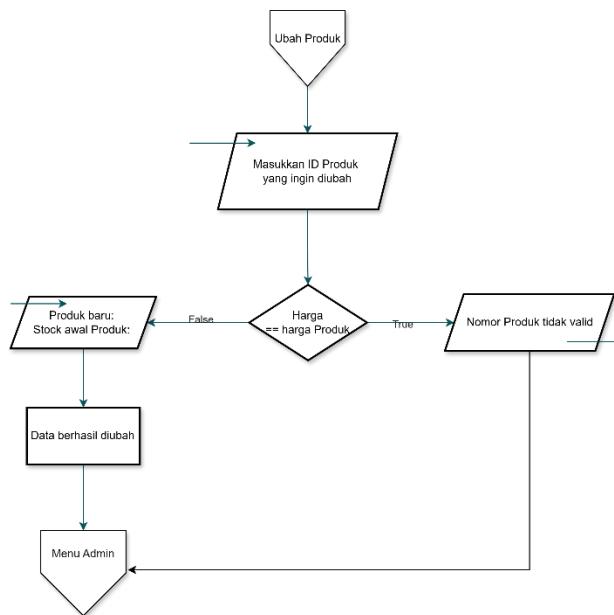
Flowchart 1.5 Menu User



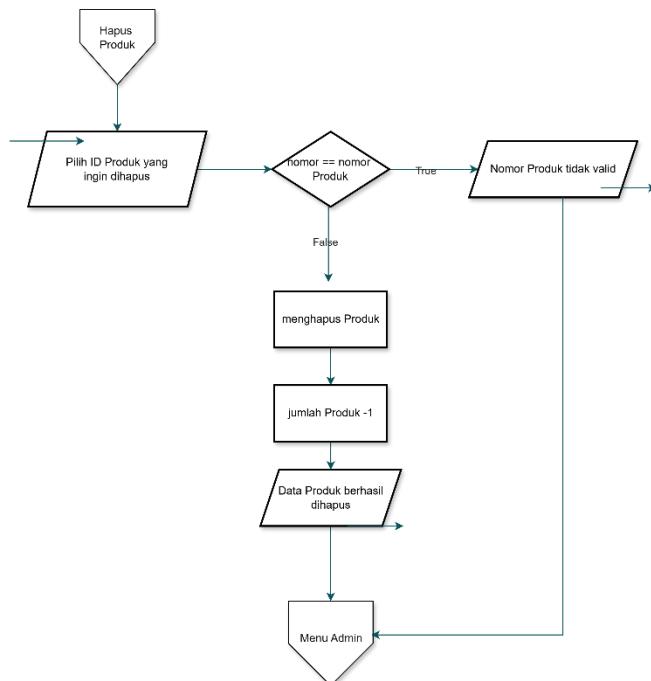
Flowchart 1.6 Tambah Produk (Create)



Flowchart 1.7 Menampilkan Produk (Read)



Flowchart 1.8 Mengubah Produk (Update)



Flowchart 1.9 Menghapus Produk (Delete)

- Program dimulai dengan menampilkan menu Regist/Daftar, Login, dan Keluar program
- Jika memilih menu regis, maka pengguna akan diarahkan mengisi Username dan Password
- Login sebagai User atau Admin dengan memasukkan Username dan Password yang telah ditentukan
- Jika Login sebagai user, Maka Pengguna hanya bisa Melihat tampilan/list produk lalu Logout
- Jika Login sebagai Admin, Maka :
 - Pengguna bisa menambah produk (Create)
 - Pengguna bisa melihat data produk (Read)
 - Pengguna bisa memgubah/memperbarui produk (Update)
 - Pengguna bisa menghapus produk (Delete)
- Jika memilih Tambah Produk (Create), Maka pengguna diminta untuk memasukkan ID produk berupa angka, Stock awal produk, dan Harga produk
- Jika memilih Lihat Produk (Read), Maka pengguna bisa melihat tampilan data produk
- Jika memilih Update Produk, Maka pengguna diarahkan untuk memilih produk mana yang akan diubah dengan memasukkan ID produk, lalu diubah dengan data yang baru
- Jika memilih Hapus produk (Delete), Maka pengguna diarahkan untuk memilih ID produk mana yang akan dihapus.
- Program diakhiri dengan menu Logout dan Keluar.

2. Deskripsi Singkat Program

Program [Sistem Pengelolaan Stok Retail Kecil](#) ini dibuat menggunakan bahasa Python untuk membantu mengelola data produk secara sederhana. Program ini memiliki dua jenis pengguna, yaitu admin dan user. Admin dapat menambah, melihat, mengubah, dan menghapus data produk (CRUD), sedangkan user hanya dapat melihat daftar produk yang tersedia. Setiap data produk disimpan dalam bentuk list yang berisi ID, nama produk, stok, dan harga. Semua data produk dikumpulkan dalam nested list agar mudah dikelola.

Selain itu, program ini juga mendukung fitur register agar pengguna baru dapat membuat akun sendiri, serta login agar sistem dapat membedakan hak akses antara admin dan user. Proses validasi input dilakukan tanpa menggunakan try-except, melainkan dengan pengecekan manual seperti memastikan nilai stok dan harga berupa angka. Untuk mempercantik tampilan dan menjaga keteraturan saat berpindah menu, digunakan perintah os.system('cls' if os.name == 'nt' else 'clear') agar terminal selalu dibersihkan sebelum menampilkan menu baru.

3. Source Code

A. Data Awal → Pilihan Menu (List dan Nested List)

Program menggunakan nested list agar data bisa dikelompokkan rapi. Setiap elemen users berisi [username, password, peran], dan setiap elemen produk berisi [id_produk, nama_produk, stok, harga].

```
users = [
    ["admin", "admin123", "admin"],
    ["user", "user123", "user"]
]

produk = [
    ["001", "Sabun", 15, 3000],
    ["002", "Shampoo", 10, 7000],
    ["003", "Sikat Gigi", 20, 4000]
]
```

B. Fitur Register

Kode ini memungkinkan pengguna baru mendaftar akun. Sistem memeriksa apakah username sudah ada, lalu menambahkan data baru ke list users jika belum terdaftar.

```
for u in users:
    if u[0] == username:
        sudah_ada = True

    if sudah_ada:
        print("X Username sudah terdaftar!")
    else:
        users.append([username, password, "user"])
        print("✓ Akun berhasil dibuat!")
```

C. Fitur Login

Bagian ini melakukan pemeriksaan login dengan mencocokkan username dan password. Jika cocok, sistem mengenali peran pengguna (admin atau user) untuk menentukan menu yang ditampilkan.

```
for u in users:
    if u[0] == username and u[1] == password:
        role = u[2]
```

D. Create (Tambah Produk)

Fitur ini digunakan oleh admin untuk menambah produk baru ke dalam list produk. Sebelum disimpan, input stok dan harga divalidasi agar berisi angka.

```
if (not stok.isdigit()) or (not harga.isdigit()):
    print("X Stok dan harga harus angka!")
else:
    produk.append([idp, nama, int(stok), int(harga)])
    print("✓ Produk berhasil ditambahkan!")
```

E. Read (Lihat Produk)

Menampilkan daftar seluruh produk dalam bentuk tabel rapi. Fitur ini dapat diakses oleh admin maupun user.

```
print("{:<5} {:<15} {:<10} {:<10}".format("ID", "Nama", "Stok", "Harga"))
        print("-"*45)
        for p in produk:
            print("{:<5} {:<15} {:<10} {:<10}".format(p[0], p[1],
p[2], p[3]))
```

F. Update (Ubah Produk)

Bagian ini mencari produk berdasarkan ID, lalu memperbarui isinya. Admin dapat mengubah nama, stok, dan harga produk.

```
for i in range(len(produk)):
    if produk[i][0] == idp:
        index = i

    if index == -1:
        print("X ID produk tidak ditemukan!")
    else:
        nama_baru = input("Nama baru: ")
        stok_baru = input("Stok baru: ")
        harga_baru = input("Harga baru: ")
```

G. Delete (Hapus Produk)

Digunakan untuk menghapus produk dari list berdasarkan ID produk yang dimasukkan.

Jika ID ditemukan, elemen list tersebut dihapus menggunakan `.remove()`.

```
for p in produk:
    if p[0] == idp:
        produk.remove(p)
        ditemukan = True
        print("✓ Produk berhasil dihapus!")
        break
```

4. Hasil Output

```
==== SISTEM PENGELOLAAN STOK RETAIL KECIL ====
1. Login
2. Register
3. Keluar
Pilih menu: █
```

Output 4.1 Pilihan menu

```
==== LOGIN SISTEM ====
Username : admin
Password : admin123
✓ Login berhasil! Selamat datang, admin

Tekan Enter untuk melanjutkan...█
```

Output 4.2 Login (Admin)

```
==== MENU ADMIN ====
1. Tambah Produk (Create)
2. Lihat Produk (Read)
3. Update Produk
4. Hapus Produk
5. Logout
Pilih menu: █
```

Output 4.3 Menu Admin

```
==== TAMBAH PRODUK BARU ====
ID Produk: 004
Nama Produk: Odol
Stok Awal: 25
Harga: 15000
✓ Produk berhasil ditambahkan!

Tekan Enter untuk kembali...█
```

Output 4.4 Tambah Produk (Create)

== DAFTAR PRODUK ==			
ID	Nama	Stok	Harga
001	Sabun	15	3000
002	Shampoo	10	7000
003	Sikat Gigi	20	4000
004	Odol	25	15000

Tekan Enter untuk kembali...█

Output 4.5 Lihat Produk (Read)

== UPDATE PRODUK ==			
ID	Nama	Stok	Harga
001	Sabun	15	3000
002	Shampoo	10	7000
003	Sikat Gigi	20	4000
004	Odol	25	15000

Masukkan ID produk yang ingin diupdate: 003
 Nama baru: Parfum
 Stok baru: 30
 Harga baru: 24000
 Data produk berhasil diperbarui!

Tekan Enter untuk kembali...█

Output 4.6 Update Produk

== HAPUS PRODUK ==			
ID	Nama	Stok	Harga
001	Sabun	15	3000
002	Shampoo	10	7000
003	Parfum	30	24000
004	Odol	25	15000

Masukkan ID produk yang ingin dihapus: 002
 Produk berhasil dihapus!

Tekan Enter untuk kembali...█

Output 4.7 Hapus Produk (Delete)

```
== REGISTER AKUN BARU ==
Masukkan username baru : Aditya
Masukkan password baru : 002
 Akun berhasil dibuat!

Tekan Enter untuk kembali...█
```

Output 4.9 Register

```
==== LOGIN SISTEM ====
Username : user
Password : user123
 Login berhasil! Selamat datang, user

Tekan Enter untuk melanjutkan... █
```

Output 4.10 Login (User)

```
==== MENU PENGGUNA ====
1. Lihat Produk
2. Logout
Pilih menu: █
```

Output 4.11 Tampilan Menu (User)

5. GIT

5.1 GIT Init

Git Init adalah perintah Git yang digunakan untuk menginisialisasi repository Git baru dalam sebuah folder. Cukup ketik “git init” pada terminal VSCode

```
PS D:\praktikum-apd> git init
Initialized empty Git repository in D:/praktikum-apd/.git/
```

Gambar 5.1 Git Init

5.2 GIT Add

Git add adalah perintah Git untuk menambahkan semua perubahan file di folder kerja (working directory) ke staging area, agar siap dicommit ke dalam repository.

```
PS D:\praktikum-apd> git add .
PS D:\praktikum-apd> █
```

Gambar 5.2 Git Add .

5.3 GIT Commit

Adalah perintah Git yang digunakan untuk menyimpan perubahan yang telah ditambahkan ke staging area ke dalam repository. Setiap commit akan memiliki hash unik, pesan commit, dan menyimpan snapshot dari perubahan yang dilakukan.

```
PS D:\praktikum-apd> git commit -m "Finish Posttest 3"
[main 040802a] Finish Posttest 3
 3 files changed, 84 insertions(+)
 create mode 100644 post-test/post-test-apd-3/2509106002-AdityaFitriansyah-PT-3.pdf
 create mode 100644 post-test/post-test-apd-3/2509106002-AdityaFitriansyah-PT-3.py
```

Gambar 5.3 Git Commit

5.4 GIT Remote

Adalah perintah Git yang digunakan untuk menghubungkan repository lokal dengan repository remote (misalnya di GitHub, GitLab, atau Bitbucket). Dengan perintah ini, kita bisa mengelola koneksi ke repository jarak jauh, memungkinkan push, pull, dan fetch dari repository tersebut.

```
PS D:\praktikum-apd> git remote add origin git@github.com:adityafitriansyah/praktikum-apd.git
```

Gambar 5.4 Git Remote

5.5 GIT Push

Adalah perintah Git yang digunakan untuk mengirim (mengunggah) perubahan dari repository lokal ke repository remote (seperti GitHub, GitLab, atau Bitbucket). Perintah ini memastikan bahwa perubahan yang sudah dikomit di lokal tersedia di repository jarak jauh sehingga bisa diakses oleh orang lain atau untuk Cadangan.

```
PS D:\praktikum-apd> git push -u origin main
```

Gambar 5.5 Git Push