**MAKALAH**

**FINAL PROJECT DASAR PEMROGRAMAN PENGADAAN BARANG MENGGUNAKAN BAHASA PEMROGRAMAN PYTHON**

**DAN DATABASE MYSQL**

Dosen Pengajar : Walim, M.Kom



Disusun oleh :

1. Abdul Aziz (19241791)
2. Ahmad Maulana Zuhdi (19240878)
3. Defa Raihan Agis (19240506)
4. Muhammad Nadhif Fadhilah Toziri (19240559)
5. Nero Aziz Saputra (19240522)

**JURUSAN SISTEM INFORMASI**

**FAKULTAS TEKNIK DAN INFORMATIKA**

**UNIVERSITAS BINA SARANA DAN INFORMATIKA**

**2024/2025**

# KATA PENGATAR

Puji dan Syukur kami panjatkan kepada kehadirat Allah SWT / Tuhan Yang Maha Esa, karena beliaulah yang selalu senantiasa melimpahkan Rahmatnya dan Karunia-Nya sehingga kami dapat menyusun laporan makalah ini secara baik dan benar. Kami juga tidak lupa mengucapkan banyak terima kasih kepada dosen pengajar kami,Bapak. Walim, M.Kom yang telah membimbing kami dalam mengerjakan laporan makalah ini.

Makalah ini berisikan tentang Final Project Dasar Pemrograman “Program Sederhana Membedakan Penyakit Covid-19 dan Gejala Tifus“. Adapun maksud dan tujuan kami di buatnya makalah ini adalah untuk memenuhi salah satu tugas mata kuliah yakni Dasar Pemrograman.

Kami menyadari bahwa masih banyak kekurangan yang mendasar atas proses pembuatan makalah ini baik dari segi apapun. Oleh karena itu kami mohon maaf sebesar-besarnya dan kami mengharapkan saran serta kritik yang dapat membantu kami untuk penyempurnaan makalah ini selanjutnya.

Cikarang, 20 November 2024

Kelompok 6

# DAFTAR ISI

[KATA PENGATAR ii](#_Toc183373225)

[DAFTAR ISI iii](#_Toc183373226)

[DAFTAR GAMBAR iv](#_Toc183373227)

[BAB I PENDAHULUAN 4](#_Toc183373228)

[1.1 Latar Belakang 4](#_Toc183373229)

[1.2 Identifikasi Masalah 4](#_Toc183373230)

[1.3 Rumusan Masalah 4](#_Toc183373231)

[1.4 Tujuan 4](#_Toc183373232)

[1.5 Ruang Lingkup 5](#_Toc183373233)

[BAB II ISI MAKALAH 6](#_Toc183373234)

[2.1 Pengertian Software 6](#_Toc183373236)

[2.2 Program Pengadaan Barang 6](#_Toc183373237)

[2.3 Algoritma Program 7](#_Toc183373238)

[2.4 Flowchart Program 9](#_Toc183373239)

[BAB III PROGRAM PYTHON 10](#_Toc183373240)

[3.1 Listing Program 10](#_Toc183373242)

[3.2 Hasil Running 18](#_Toc183373243)

[3.2.1 Form login 18](#_Toc183373244)

[3.2.2 Jika login gagal 18](#_Toc183373245)

[3.2.3 Jika login berhasil 18](#_Toc183373246)

[3.2.4 Menu utama 18](#_Toc183373247)

[3.2.5 Tambah produk baru 18](#_Toc183373248)

[3.2.6 Update jumlah produk 18](#_Toc183373249)

[3.2.7 Tambah jumlah produk 19](#_Toc183373250)

[3.2.8 Kurangi jumlah produk 19](#_Toc183373251)

[3.2.9 Hapus produk 19](#_Toc183373252)

[3.2.10 Lihat semua produk 19](#_Toc183373253)

[3.2.11 Keluar aplikasi 19](#_Toc183373254)

[BAB IV PENUTUP 20](#_Toc183373255)

[4.1 Kesimpulan 20](#_Toc183373257)

# DAFTAR GAMBAR

[Gambar 2.4‑1: Flowchart aplikasi 9](#_Toc183373447)

[Gambar 3.2‑1: Form login 18](#_Toc183373448)

[Gambar 3.2‑2: Tampilan login gagal 18](#_Toc183373449)

[Gambar 3.2‑3: Tampilan login berhasil 18](#_Toc183373450)

[Gambar 3.2‑4: Tampilan menu utama 18](#_Toc183373451)

[Gambar 3.2‑5: Form menambah produk baru 18](#_Toc183373452)

[Gambar 3.2‑6: Form pilihan update produk 18](#_Toc183373453)

[Gambar 3.2‑7: Form tambah jumlah produk 19](#_Toc183373454)

[Gambar 3.2‑8: Form kurangi jumlah produk 19](#_Toc183373455)

[Gambar 3.2‑9: Form hapus produk 19](#_Toc183373456)

[Gambar 3.2‑10: Form lihat semua produk 19](#_Toc183373457)

[Gambar 3.2‑11: Tampilan keluar aplikasi 19](#_Toc183373458)

# BAB I PENDAHULUAN

## Latar Belakang

Seiring dengan perkembangan teknologi, solusi digital mulai diadopsi untuk menyelesaikan permasalahan dalam proses pengadaan. Salah satu pendekatan yang menarik adalah pengembangan aplikasi pengadaan barang berbasis teknologi digital. Aplikasi ini dirancang untuk mengotomatiskan proses pengadaan, meminimalkan kesalahan, meningkatkan transparansi, serta mempercepat waktu pemrosesan.

Pengembangan aplikasi pengadaan barang dapat dilakukan menggunakan berbagai teknologi, termasuk Python sebagai struktur aplikasi dan MySQL untuk pengelolaan basis data, Python menjadi pilihan yang kuat untuk membangun aplikasi yang handal dan fleksibel karena kemudahan dalam memahami sintaks programnya.

Karena itu, dengan kemajuan di bidang teknologi saat ini, dibuatnya system untuk pengadaan barang agar memudahkan dalam update serta menganalisis stok barang/produk.

## Identifikasi Masalah

1. Pengelolaan data pengadaan barang yang manual
2. Kurangnya efisiensi dalam proses pengadaan
3. Kesulitan memantau stok barang

## Rumusan Masalah

## Bagaimana cara membuat sistem pengadaan barang yang terotomatisasi menggunakan Python untuk memudahkan proses pencatatan data barang?

## Bagaimana cara menyimpan dan mengelola data barang, seperti stok, harga, dan pemasok, secara terstruktur menggunakan Python?

## Bagaimana sistem dapat membantu pengguna memantau stok barang dan menghasilkan laporan pengadaan secara otomatis?

## Tujuan

1. Mengotomatisasi proses pencatatan pengadaan barang
2. Meningkatkan efisiensi manajemen stok
3. Menyediakan laporan pengadaan

## Ruang Lingkup

Dalam penelitian ini kelompok kami membatasi permasalahan terkait pengadaan barang dengan menggunakan Bahasa Pemrograman Python dan basis data MySQL untuk menampilkan beberapa fitur yang dapat memperbarui data dengan cepat.

# BAB II ISI MAKALAH



## Pengertian Software

1. Pengertian Python

Menurut Dafa Raditya Denishtsany (2024), Python adalah bahasa pemrograman tingkat tinggi yang dirancang dengan sintaksis sederhana dan mudah dipahami, menjadikannya cocok untuk pemula. Python mendukung berbagai bidang, seperti pengembangan web, analisis data, kecerdasan buatan, dan pembelajaran mesin. Keunggulannya termasuk fleksibilitas, pustaka yang kaya, dan kemampuan untuk diintegrasikan dengan bahasa lain.

1. Pengertian MySQL

Menurut Ahmad (2021), MySQL adalah sistem manajemen basis data relasional (RDBMS) yang terkenal karena kecepatan dan efisiensinya. MySQL sering digunakan dalam pengembangan aplikasi berbasis web untuk menyimpan dan mengelola data secara terstruktur

1. Pengertian Pengadaan Barang

Menurut Putri dan Sembiring (2021), Pengadaan barang adalah proses pembelian atau penyediaan barang dan jasa yang dibutuhkan oleh suatu organisasi. Proses ini melibatkan identifikasi kebutuhan, pemilihan pemasok, dan pengelolaan pengiriman barang dengan tujuan efisiensi dan penghematan biaya

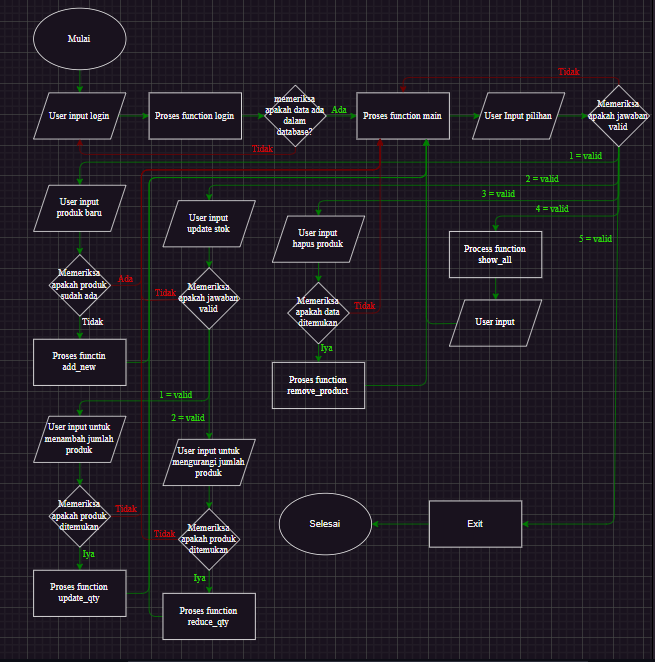
## Program Pengadaan Barang

Program Pengadaan Barang menggunakan Python dan MySQL adalah sebuah program yang dirancang untuk mengelola proses pengadaan barang secara efisien dengan menggunakan Python sebagai bahasa pemrograman dan MySQL sebagai basis data untuk penyimpanan informasi.

## Algoritma Program

1. Mulai
2. Proses login
3. User diminta untuk meng-input username dan password
4. Memeriksa apakah data valid
5. **Jika valid**, lanjut ke langkah 3
6. **Jika tidak valid**, looping langkah 2
7. Proses menu utama aplikasi
8. User diminta untuk meng-input pilihan (1-5)
9. Jika input **1** maka akan lanjut ke langkah 4
10. Jika input **2** maka akan lanjut ke langkah 5
11. Jika input **3** maka akan lanjut ke langkah 6
12. Jika input **4** maka akan lanjut ke langkah 7
13. Jika input **5** maka akan lanjut ke langkah 8
14. Jika input selain diatas maka akan kembali ke langkah 3
15. Menambah produk baru
16. User diminta meng-input data produk
17. Data dikirim ke database untuk dicocokkan apakah data sudah ada
18. **Jika ada**, maka data tidak akan dimasukkan ke database, dan kembali ke langkah 3
19. **Jika tidak ada**, maka akan memasukkan data ke dalam database, lalu kembali ke langkah 3
20. Update jumlah produk
21. User meng-input pilihan (1-2)
22. Jika input 1 maka akan lanjut ke langkah b
23. Jika input 2 maka akan lanjut ke langkah c
24. Jika input selain diatas maka akan kembali ke langkah 3
25. Menambah jumlah produk
26. User diminta meng-input data produk
27. Memeriksa apakah produk ditemukan
28. **Jika ditemukan**, maka akan menambah jumlah produk dan kembali ke langkah 3
29. **Jika tidak ditemukan**, maka akan kembali ke langkah 3
30. Mengurangi jumlah produk
31. User diminta meng-input data produk
32. Memeriksa apakah produk ditemukan
33. **Jika ditemukan**, maka akan mengurangi jumlah produk dan kembali ke langkah 3
34. **Jika tidak ditemukan**, maka akan kembali ke langkah 3
35. Hapus produk
36. User diminta untuk meng-input data produk
37. Memeriksa apakah produk ditemukan
38. **Jika ditemukan**, maka akan menghapus produk dan kembali ke langkah 3
39. **Jika tidak ditemukan**, maka akan kembali ke langkah 3
40. Lihat semua produk
41. Melihat semua produk
42. User diminta meng-input value apapun untuk kembali ke langkah 3
43. Keluar
44. Keluar dari aplikasi dan menjalankan langkah 2

## Flowchart Program



Gambar 2.4‑1: Flowchart aplikasi

# BAB III PROGRAM PYTHON



## Listing Program

*# Aplikasi pengadaan barang dengan operasi SQL CRUD (Create, Read, Update, Delete)*

*# Install Xampp*

*# Nyalakan Apache dan Mysql*

*# search di browser localhost*

*# pilih phpMyAdmin*

*# masuk ke menu sql*

*# Paste sintaks dibawah ini (digunakan hanya untuk petama kali saja untuk mengkonfigurasi database):*

*# CREATE DATABASE db\_minimarket;*

*# USE db\_minimarket;*

*# CREATE TABLE tb\_product (*

*#     id INT(11) AUTO\_INCREMENT PRIMARY KEY,*

*#     name VARCHAR(255) NOT NULL,*

*#     category VARCHAR(255) NOT NULL,*

*#     weight INT(11) NOT NULL,*

*#     quantity INT(11) NOT NULL,*

*#     supplier VARCHAR(255) NOT NULL*

*# );*

*# CREATE TABLE users (*

*#     id INT(11) AUTO\_INCREMENT PRIMARY KEY,*

*#     username VARCHAR(16) NOT NULL,*

*#     password VARCHAR(16) NOT NULL,*

*#     position VARCHAR(25) NOT NULL*

*# );*

*# INSERT INTO `users` (`id`, `username`, `password`, `position`) VALUES (NULL, 'admin', 'admin', 'admin');*

*# INSERT INTO tb\_product (id)*

*# VALUES (*

*#     CONCAT(*

*#         CHAR(FLOOR(65 + (RAND() \* 26))),  -- Huruf acak (A-Z)*

*#         LPAD(FLOOR(RAND() \* 9999) + 1, 4, '0')  -- Nomor acak (1-9999)*

*#     )*

*# );*

*# Install library MySQL:*

*# pip install mysql.connector*

*# import library*

import mysql.connector *# library untuk menghubungkan kedalam database*

import time *# Library untuk waktu*

import getpass *# Library untuk password*

import os *# Library untuk berinteraksi dengan OS*

*# Function untuk clear terminal*

def clear():

    if os.name == 'nt':  *# Identitas OS windows*

        os.system('cls')  *# Sintaks windows untuk membersihkan layar terminal*

    else:

        os.system('clear') *# Sintaks untuk OS Unix membersihkan layar terminal*

*# Function untuk membuat koneksi kedalam database*

def db\_connection():

    try: *# Mencoba membuat koneksi kedalam database*

*# Membuat koneksi dengan format:*

        return mysql.connector.connect(

            host = 'localhost',

            user = 'root',

            password = '',

            database = 'db\_minimarket'

        )

*# Sintaks yang berjalan jika terjadi error pada kode try*

    except mysql.connector.Error as e: *# Menampilkan bagian yang error dari try koneksi ke MySQL*

        print(f"Error: {e}")

        time.sleep(5) *# Membuat delay selama 5 detik sebelum mengeskekusi sintaks selanjutnya*

        clear() *# Function clear*

*# Function untuk user login*

def login(username, password):

    try: *# Mencoba untuk fetch data sesuai dengan yang diinput oleh user (jika data True)*

        db = db\_connection() *# Mendeklarasikan variable untuk function db\_connection*

        cursor = db.cursor() *# Membuat cursor kedalam SQL*

        cursor.execute("SELECT \* FROM users WHERE username = %s AND password = %s", (username, password)) *# Cursor mengeksekusi query untuk memilih column yang sesuai dengan yang diinput user berdasarkan parameter function*

        get\_user = cursor.fetchone() *# Fetch 1 data dari kolom table*

        if get\_user: *# Jika data berhasil di fetch*

            clear() *# Function clear*

            print("\n\033[32m\033[1m!--Login Berhasil--!\033[0m\033[37m\n") *# Alert keterangan untuk user*

            time.sleep(1) *# Membuat delay waktu selama 1 detik kepada terminal sebelum mengeksekusi sintaks selanjutnya*

            clear() *# Function clear*

            main()  *# Function main*

        else: *# Jika data yang di fetch tidak cocok*

            clear() *# Function clear*

            print("\n\033[31m\033[1m!--Login Gagal--!\nusername atau password salah\033[0m\033[37m\n") *# Alert keterangan untuk user*

            time.sleep(1) *# Membuat delay waktu selama 1 detik kepada terminal sebelum mengeksekusi sintaks selanjutnya*

            clear() *# Function clear*

    except ValueError as e: *# Menampilkan value yang error dari kolom try login*

        print(f"Error: {e}")

    finally: *# setelah selesai, tutup koneksi dan kursor untuk menghemat RAM*

        cursor.close() *# Menutup cursor*

        db.close() *# Menutup koneksi untuk menghemat memori*

*# Function untuk menambah product baru*

def add\_new(name, category, weight, quantity, supplier):

    try: *# Mencoba mengeksekusi*

        db = db\_connection() *# Mendeklarasikan variable untuk function db\_connection*

        cursor = db.cursor() *# Membuat cursor kedalam SQL*

        cursor.execute("SELECT \* FROM tb\_product WHERE name = %s AND weight = %s AND supplier = %s", (name, weight, supplier)) *# Cursor mengeksekusi query untuk select semua column pada table*

        cursor.fetchall() *# Fetch semua data*

        if cursor.rowcount == 0: *# Jika produk yang ditambahkan belum ada pada column*

            cursor.execute("INSERT INTO tb\_product (name, category, weight, quantity, supplier) VALUES(%s, %s, %s, %s, %s)", (name, category, weight, quantity, supplier)) *# Cursor mengeksekusi query untuk menambahkan produk kedalam database*

            db.commit() *# Menyimpan perubahan kedalam database*

            print("\n\033[32m\033[1m!--Product Berhasil Ditambahkan--!\033[0m\033[37m\n") *# Alert keterangan produk berhasil ditambahkan*

            time.sleep(2) *# Membuat delay selama 2 detik sebelum mengeksekusi sintaks berikutnya*

            clear() *# Function clear*

        else: *# Jika Produk sudah ada dalam database*

            print(f"\033[31m\033[1m!--Gagal Menambah Product Baru--!\nProduct Sudah Ada\033[0m\033[37m") *# Alert Keterangan produk gagal ditambahkan*

            time.sleep(2) *# Membuat delay selama 2 detik sebelum mengeksekusi sintaks berikutnya*

            clear() *# Function clear*

            main() *# Function main*

    except ValueError as e: *# Jika terdapat sintaks error pada try*

        print(f"Error: {e}")

        time.sleep(2) *# Membuat delay selama 2 detik sebelum mengeksekusi sintaks selanjutnya*

        clear() *# Function clear*

        main()

    finally: *# Jika sudah selesai*

        cursor.close() *# Menutup cursor*

        db.close() *# Menutup koneksi kedatabase untuk menghemat memori*

*# Function +qty product*

def update\_qty(id, name, quantity, weight, supplier):

    try: *# Mencoba mengeksekusi*

        db = db\_connection() *# Mendeklarasikan variable untuk function db\_connection*

        cursor = db.cursor() *# Membuat cursor kedalam SQL*

        cursor.execute("SELECT id FROM tb\_product WHERE name = %s AND weight = %s AND supplier = %s", (name, weight, supplier)) *# Cursor mengeksekusi query untuk memilih column id berdasarkan name, weight, dan supplier*

        select = cursor.fetchone() *# Fetch 1 data yang match*

        if select: *# Jika select true*

            cursor.execute("UPDATE tb\_product SET quantity = quantity + %s WHERE id = %s AND name = %s AND weight = %s AND supplier = %s", (quantity, id, name, weight, supplier)) *# Cursor mengeksekusi query untuk + quantity*

            db.commit() *# Menyimpan perubahan kedalam database*

            print("\n\033[32m\033[1m!--Berhasil Menambahkan Jumlah Product--!\033[0m\033[37m\n") *# Alert keterangan berhasil menambahkan jumlah produk*

            time.sleep(2) *# Membuat delay selama 2 detik sebelum mengeksekusi sintaks selanjutnya*

            clear() *# Function clear*

        else: *# Jika select false*

            print(f"\n\033[31m\033[1m!--Gagal Menambah Jumlah Product--!\033[0m\033[37m\n") *# Alert keterangan gagal menambah jumlah produk*

            time.sleep(2) *# Membuat delay selama 2 detik sebelum mengeksekusi sintaks selanjutnya*

            clear() *# Function clear*

    except ValueError as e: *# Jika terdapat error pada try*

        print(f"Error: {e}")

    finally: *# Jika selesai*

        cursor.close() *# Menutup cursor*

        db.close() *# Menutup koneksi untuk menghemat memori*

*# Function untuk - quantity*

def reduce\_qty(id, name, quantity, weight, supplier):

    try: *# Mencoba mengeksekusi*

        db = db\_connection() *# Mendeklarasikan variable untuk function db\_connection*

        cursor = db.cursor() *# Membuat cursor pada SQL*

        cursor.execute("SELECT id FROM tb\_product WHERE name = %s AND weight = %s AND supplier = %s", (name, weight, supplier)) *# Cursor mengeksekusi query untuk memilih column id berdasarkan name, weight, supplier*

        select = cursor.fetchone() *# Fetch 1 data yang macth*

        if select: *# Jika select true*

            cursor.execute("UPDATE tb\_product SET quantity = quantity - %s WHERE id = %s AND name = %s AND weight = %s AND supplier = %s", (quantity, id, name, weight, supplier)) *# Cursor mengeksekusi query untuk mengurangi quantity*

            db.commit() *# Menyimpan perubahan kedalam SQL*

            print("\n\033[32m\033[1m!--Berhasil Mengurangi Jumlah Product--!\033[0m\033[37m\n") *# Alert keterangan berhasil mengurangi jumlah produk*

            time.sleep(2) *# Membuat delay selama 2 detik sebelum mengeksekusi sintaks selanjutnya*

            clear() *# Function clear*

        else: *# Jika select false*

            print(f"\n\033[31m\033[1m!--Gagal Mengurangi Jumlah Product--!\033[0m\033[37m\n") *# Alert keterangan gagal mengurangi jumlah produk*

            time.sleep(2) *# Membuat delay selama 2 detik sebelum mengeksekusi sintaks selanjutnya*

            clear() *# Function clear*

    except ValueError as e: *# Jika terdapat error pada try*

        print(f"Error: {e}")

    finally: *# Jika selesai*

        cursor.close() *# Menutup cursor*

        db.close() *# Menutup koneksi untuk menghemat memori*

*# Function untuk menghapus produk*

def remove\_product(id, name, weight, supplier):

    try: *# Mencoba mengeksekusi*

        db = db\_connection() *# Mendeklarasikan variable untuk function db\_connection*

        cursor = db.cursor() *# Membuat cursor pada SQL*

        cursor.execute("SELECT \* FROM tb\_product WHERE id = %s AND name = %s AND weight = %s AND supplier = %s", (id, name, weight, supplier)) *# Cursor mengeksekusi query untuk menghapus produk sesuai dengan id, name, weight, dan supplier*

        select = cursor.fetchone() *# Fetch data yang sesuai*

        if select:

            cursor.execute("DELETE FROM tb\_product WHERE id = %s AND name = %s AND weight = %s AND supplier = %s", (id, name, weight, supplier))

            db.commit()

            print("\n\033[32m\033[1m!--Product berhasil dihapus--!\033[0m\033[37m\n") *# Alert keterangan produk berhasil dihapus*

            time.sleep(2) *# Membuat delay selama 2 detik sebelum mengeksekusi sintaks selanjutnya*

            clear() *# Function clear*

        else:

            print(f"\033[31m\033[1m!--Gagal Menghapus Product--!\n\033[0m\033[37m") *# Alert keterangan gagal menghapus produk*

            time.sleep(2) *# Membuat delay selama 2 detik sebelum mengeksekusi sintaks selanjutnya*

            clear() *# Function clear*

    except ValueError as e: *# Jika terdapat error pada try*

        print(f"Error: {e}")

    finally: *# Jika selesai*

        cursor.close() *# Menutup cursor*

        db.close() *# Menutup koneksi untuk menghemat memori*

*# Function melihat semua produk*

def show\_all():

    try: *# Mencoba mengeksekusi*

        db = db\_connection() *# Mendeklarasikan variable untuk function db\_connection*

        cursor = db.cursor() *# Membuat cursor pada SQL*

        cursor.execute("SELECT \* FROM tb\_product") *# Cursor mengeksekusi query untuk select semua column*

        show\_result = cursor.fetchall() *# Fetch semua data*

*# Format untuk output*

        print("--- DAFTAR PRODUK ---".center(100))

        print("="\*100)

        print(f"{'ID':<5} {'Name':<25} {'Categpry':<20} {'Weight':<10} {'Quantity':<10} {'Supplier':<20}") *# Membuat table dengan format lebar colomn*

        print("="\*100)

        for cell in show\_result: *# looping data sesuai dengan jumlah rows di database*

            print(f"{cell[0]:<5} {cell[1]:<25} {cell[2]:<20} {cell[3]:<10} {cell[4]:<10} {cell[5]:<20}") *# Membuat table dengan format cell sesuai dengan database*

        print("="\*100)

        answer = input("Kembali ke manu? ")

        if answer:

            time.sleep(2)

            clear()

            main()

    except ValueError as e: *# Jika terdapat error pada try*

        print(f"Error: {e}")

    finally: *# Jika selesai*

        cursor.close() *# Menutup cursor*

        db.close() *# Menutup koneksi untuk menghemat memori*

*# Function main(utama)*

def main():

    while True: *# Jika login berhasil / True*

        print("\n\033[35m\033[1mSelamat Datang Di Minimarket\033[0m\033[37m\n")

        print("1. Tambah Product Baru")

        print("2. Update Jumlah Product")

        print("3. Hapus Product")

        print("4. Lihat Semua Product")

        print("5. Keluar")

        user\_answer = int(input("Masukkan pilihan kamu(1/2/3/4/5): ")) *# Variable untuk menampung jawaban user*

        if user\_answer == 1: *# Jika user menginput 1*

            clear() *# Function clear*

            print("Menambahkan  Product Baru\n")

            nama\_product = input("Nama Product: ")

            catg\_product = input("Kategoti Product: ")

            weig\_product = int(input(("Berat Product(gram): ")))

            qtyy\_product = int(input(("Jumlah Product: ")))

            splr\_product = input("Supplier Product: ")

            add\_new(nama\_product, catg\_product, weig\_product, qtyy\_product, splr\_product) *# Menjalankan function add\_new dengan format*

        elif user\_answer == 2: *# Jika user menginput 2*

            clear() *# Function clear*

            print("1. Tambahi Jumlah Product ")

            print("2. Kurangi Jumlah Product ")

            upd\_user\_answer = int(input("Masukkan pilihan kamu(1/2): ")) *# Variable untuk menampung jawaban user*

            if upd\_user\_answer == 1: *# Jika user menginput 1*

                clear() *# Function clear*

                print("Menambah Jumlah Product\n")

                frId\_product = int(input("ID Product: "))

                nama\_product = input("Nama Product: ")

                weig\_product = int(input("Berat Product(gram): "))

                splr\_product = input("Supplier Product: ")

                qtyy\_product = int(input("Tambah Jumlah Product: "))

                update\_qty(frId\_product, nama\_product, qtyy\_product, weig\_product, splr\_product) *# Menjalankan function update\_qty dengan format*

            elif upd\_user\_answer == 2: *# Jika user menginput 2*

                clear() *# Function clear*

                print("Mengurangi Jumlah Product\n")

                frId\_product = int(input("ID Product: "))

                nama\_product = input("Nama Product: ")

                weig\_product = int(input("Berat Product(gram): "))

                splr\_product = input("Supplier Product: ")

                qtyy\_product = int(input("Tambah Jumlah Product: "))

                reduce\_qty(frId\_product, nama\_product, qtyy\_product, weig\_product, splr\_product) *# Menjalankan function reduce\_qty dengan format*

            else: *# Jika jawaban tidak valid*

                clear() *# Clear*

                print("\033[31mJawaban tidak valid !\033[37m") *# Alert keterangan jawaban tidak valid*

                time.sleep(2) *# Membuat delay selama 2 detik sebelum mengeksekusi sintaks selanjutnya*

                clear() *# Function clear*

        elif user\_answer == 3: *# Jika user menginput 3*

            clear() *# Function clear*

            print("Menghapus Product\n")

            frId\_product = int(input("ID Product: "))

            nama\_product = input("Nama Product: ")

            weig\_product = int(input("Berat Product(gram): "))

            splr\_product = input("Supplier Product: ")

            remove\_product(frId\_product, nama\_product, weig\_product, splr\_product) *# Menjalankan function remove\_product dengan format*

        elif user\_answer == 4: *# Jika user menginput 4*

            clear()

            show\_all() *# Menjalankan function show\_all*

        elif user\_answer == 5: *# Jika user menginput 5*

            clear()

            print("\033[31mKeluar dari Aplikasi !\033[37m")

            time.sleep(2)

            clear()

            break *# menghentikan function main*

        else: *# Jika jawaban tidak valid*

            clear() *# Function clear*

            print("\033[31mJawaban tidak valid !\033[37m") *# Alert keterangan jawaban tidak valid*

            clear() *# Function clear*

*# if \_\_name\_\_ == "\_\_main\_\_":*

clear() *# Function clear*

while True: *# looping login jika gagal maka akan terus loop sampai meminta agar login berhasil*

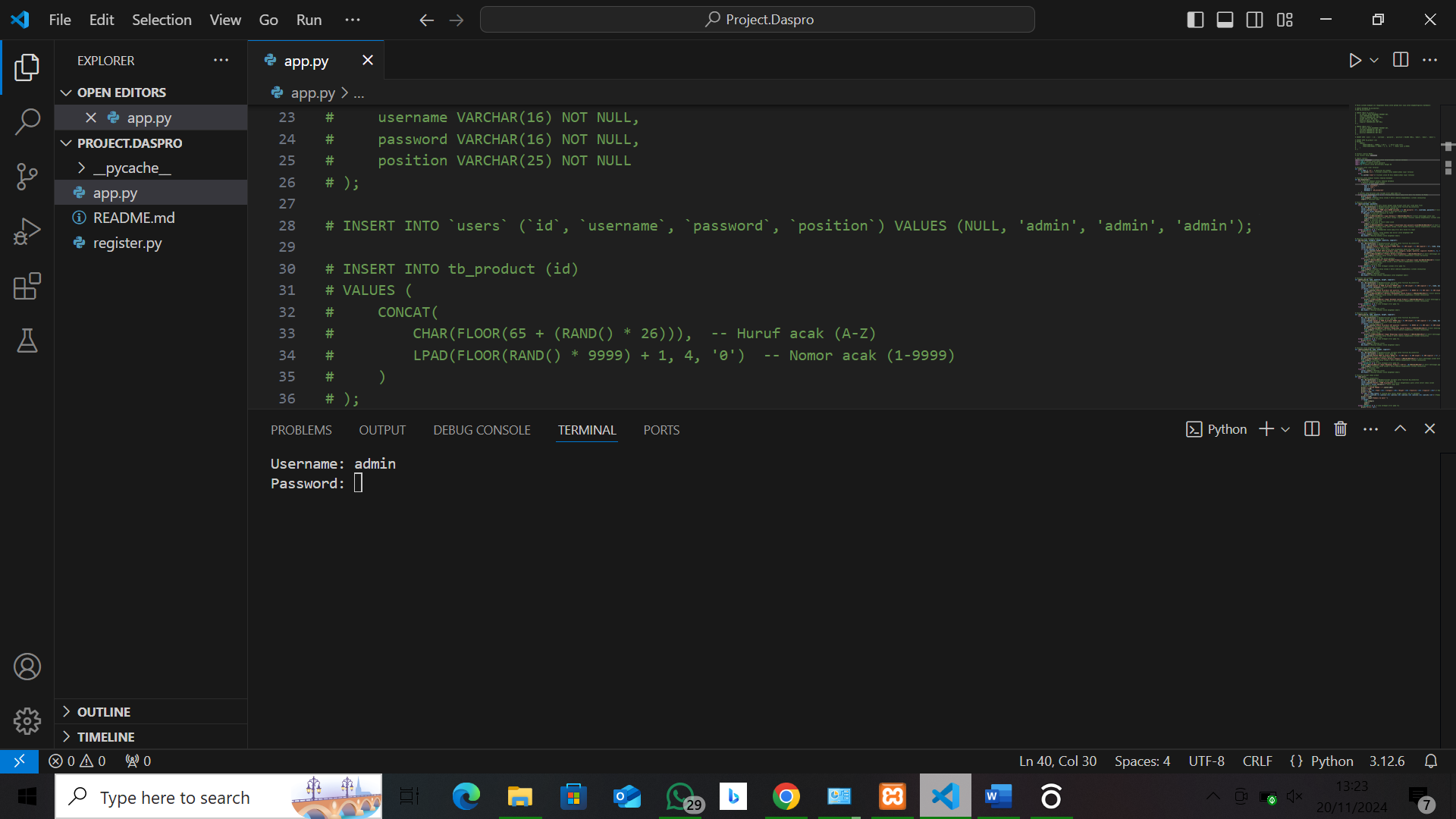
    username = input("Username: ")

    password = getpass.getpass("Password: ")

    login(username, password) *# Menjalankan function login dengan format*

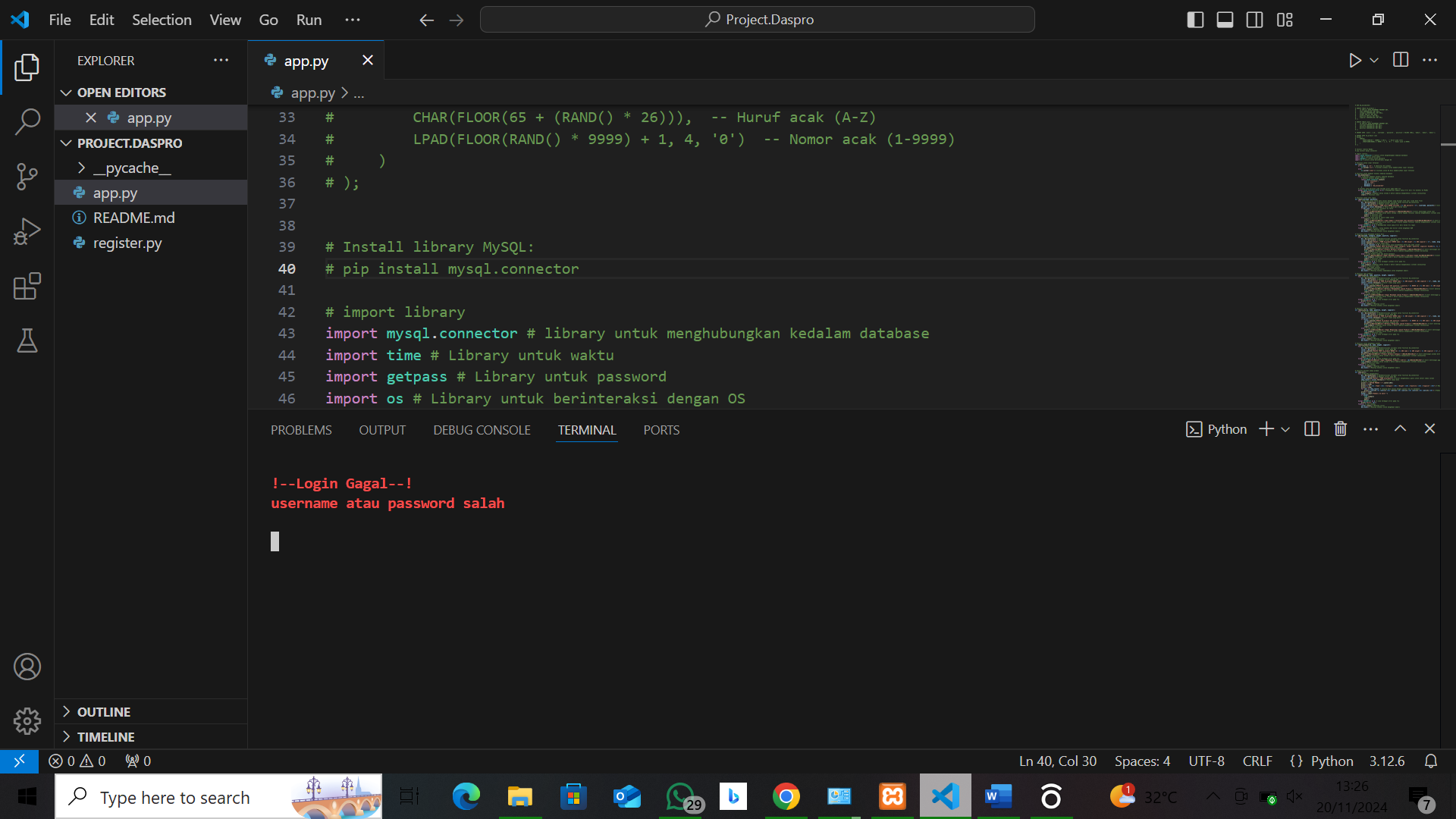
## Hasil Running

### Form login



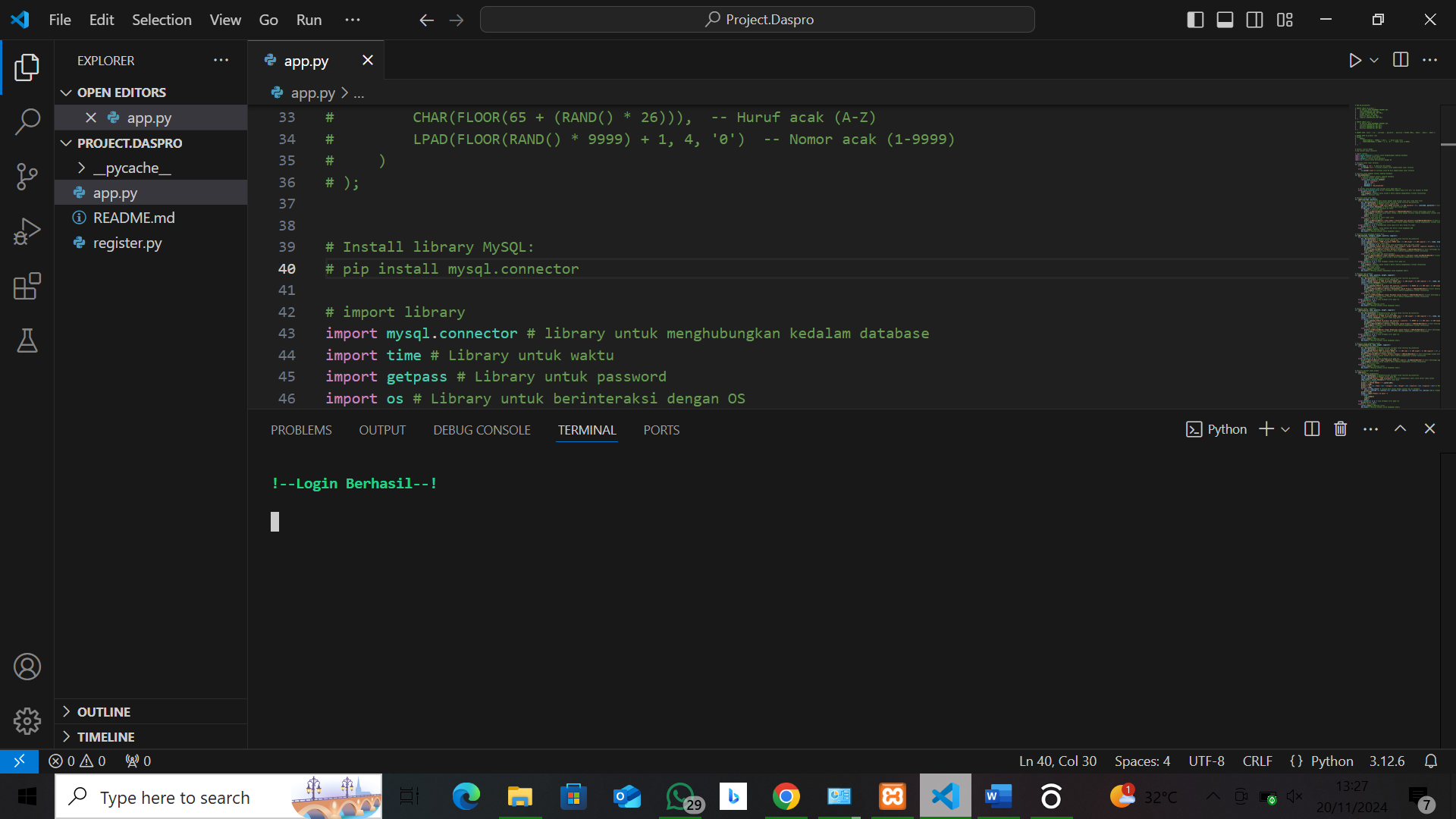
Gambar 3.2‑1: Form login

### Jika login gagal



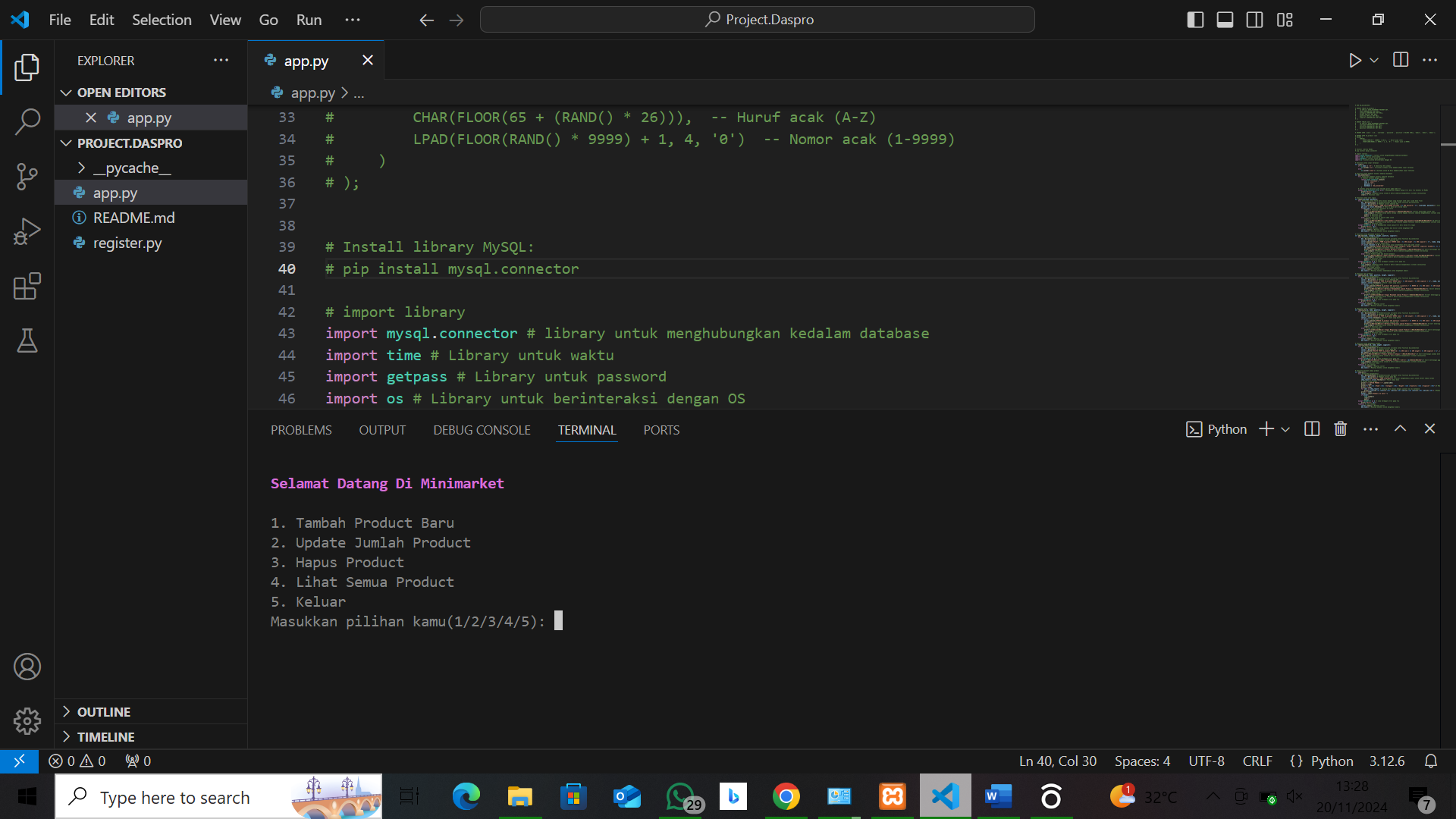
Gambar 3.2‑2: Tampilan login gagal

### Jika login berhasil



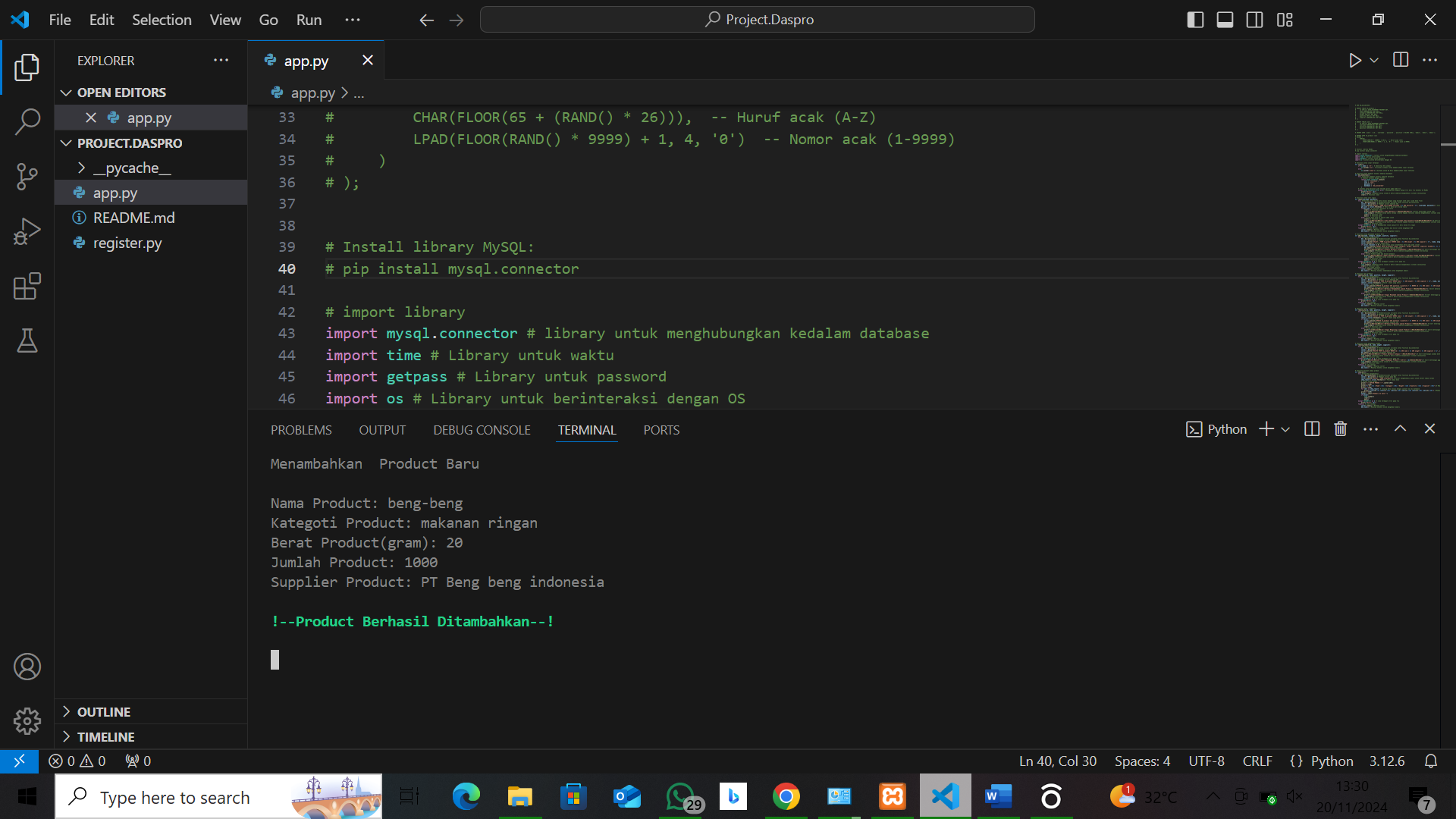
Gambar 3.2‑3: Tampilan login berhasil

### Menu utama



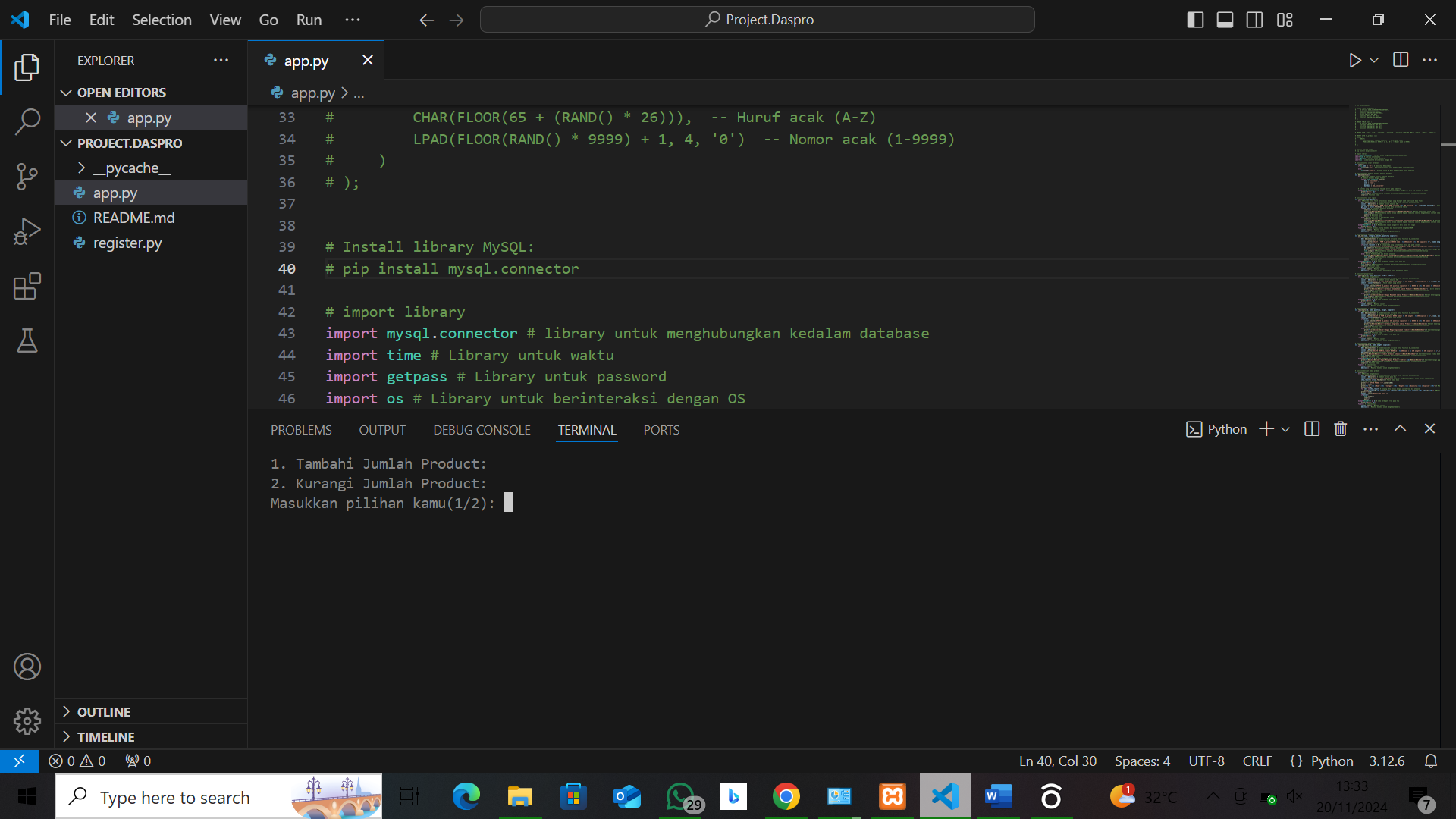
Gambar 3.2‑4: Tampilan menu utama

### Tambah produk baru



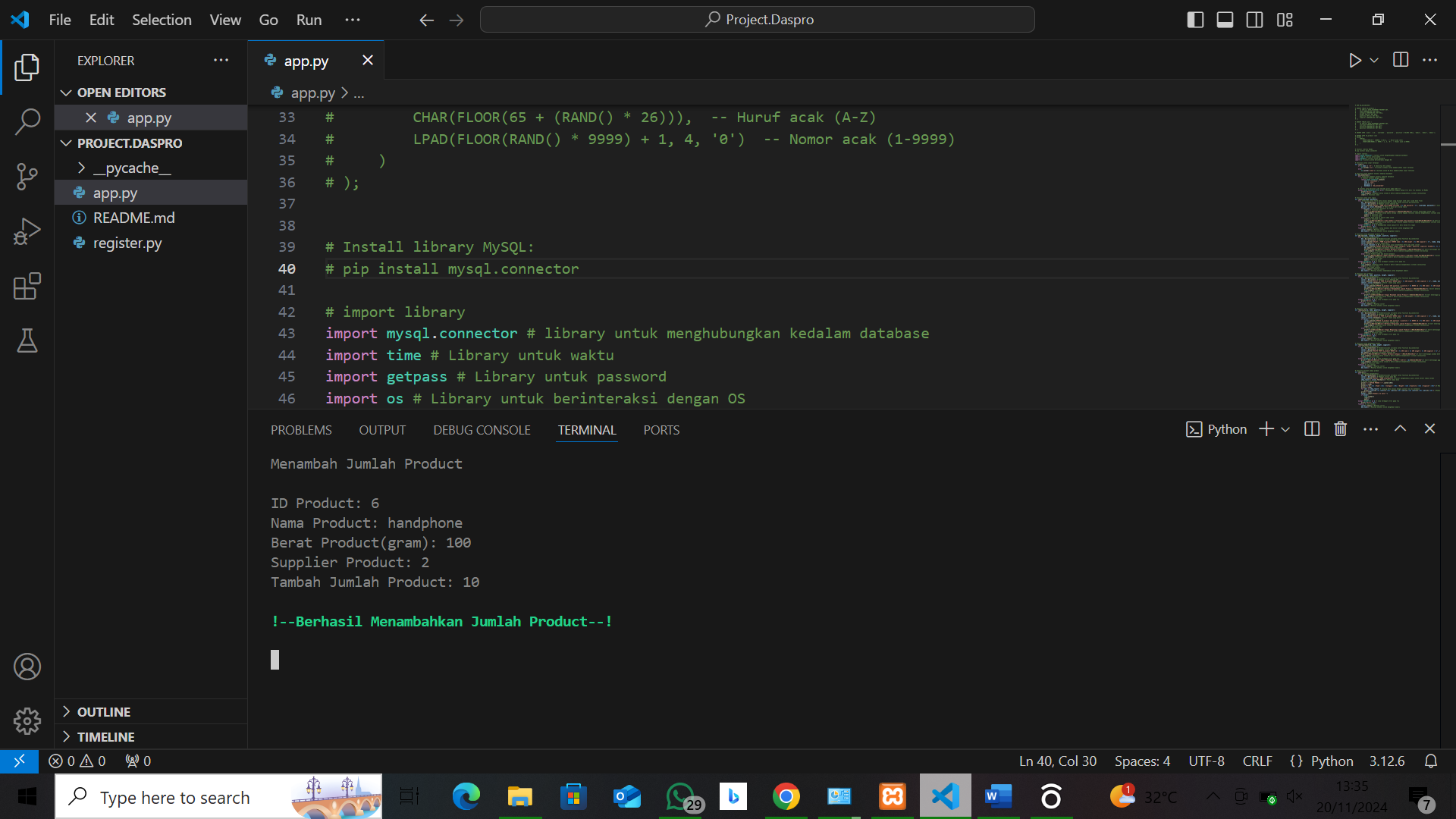
Gambar 3.2‑5: Form menambah produk baru

### Update jumlah produk



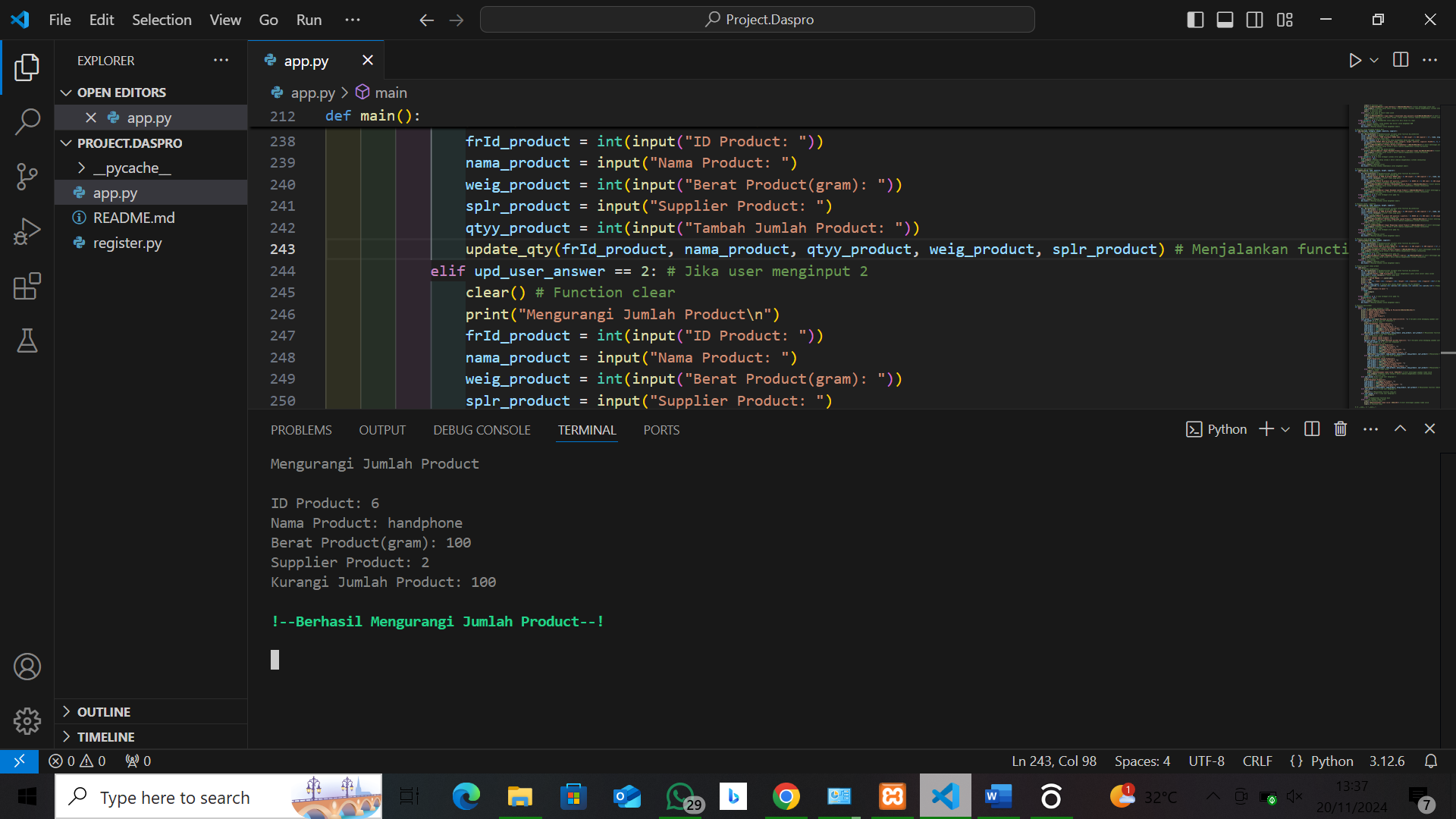
Gambar 3.2‑6: Form pilihan update produk

### Tambah jumlah produk



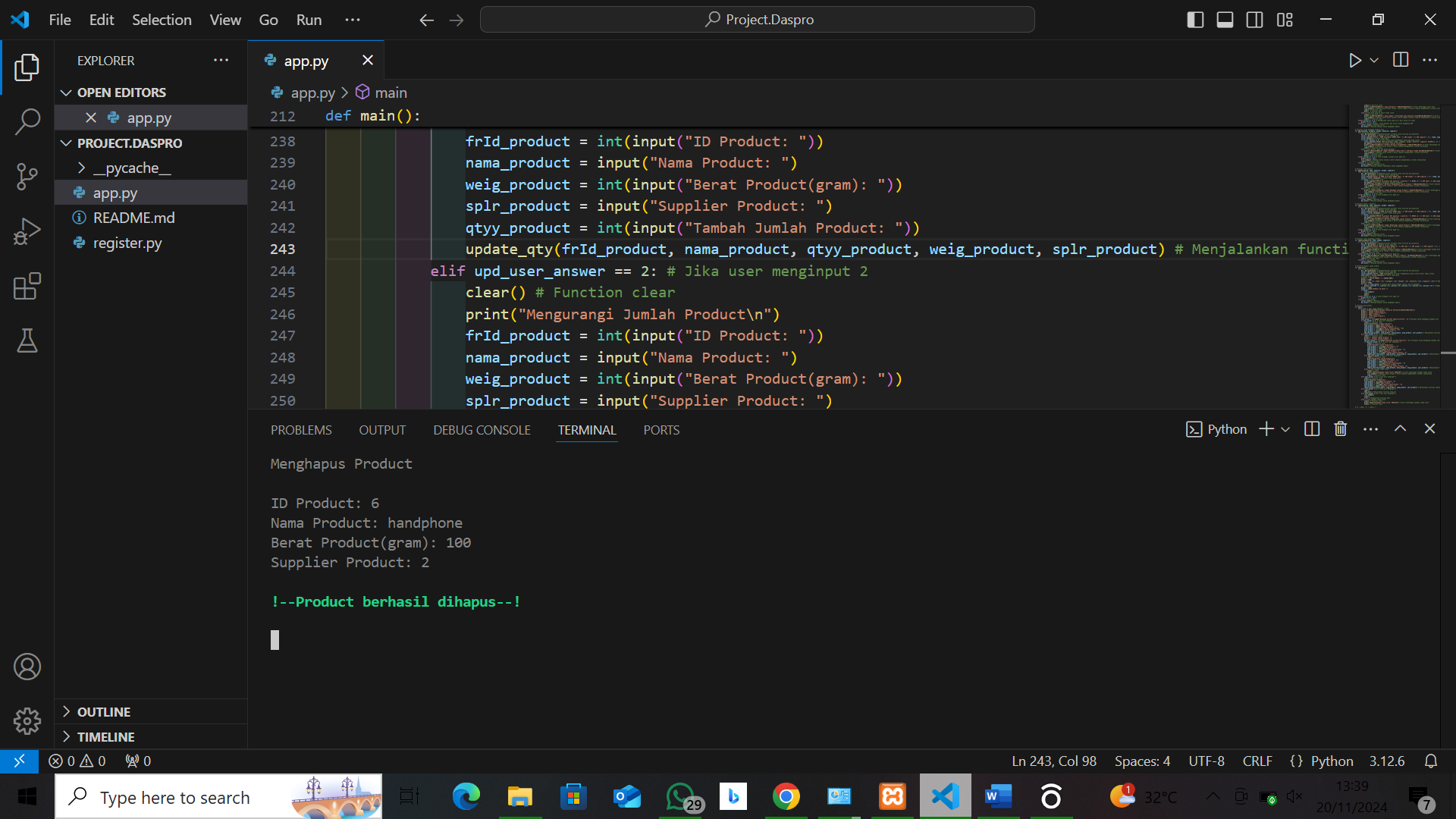
Gambar 3.2‑7: Form tambah jumlah produk

### Kurangi jumlah produk



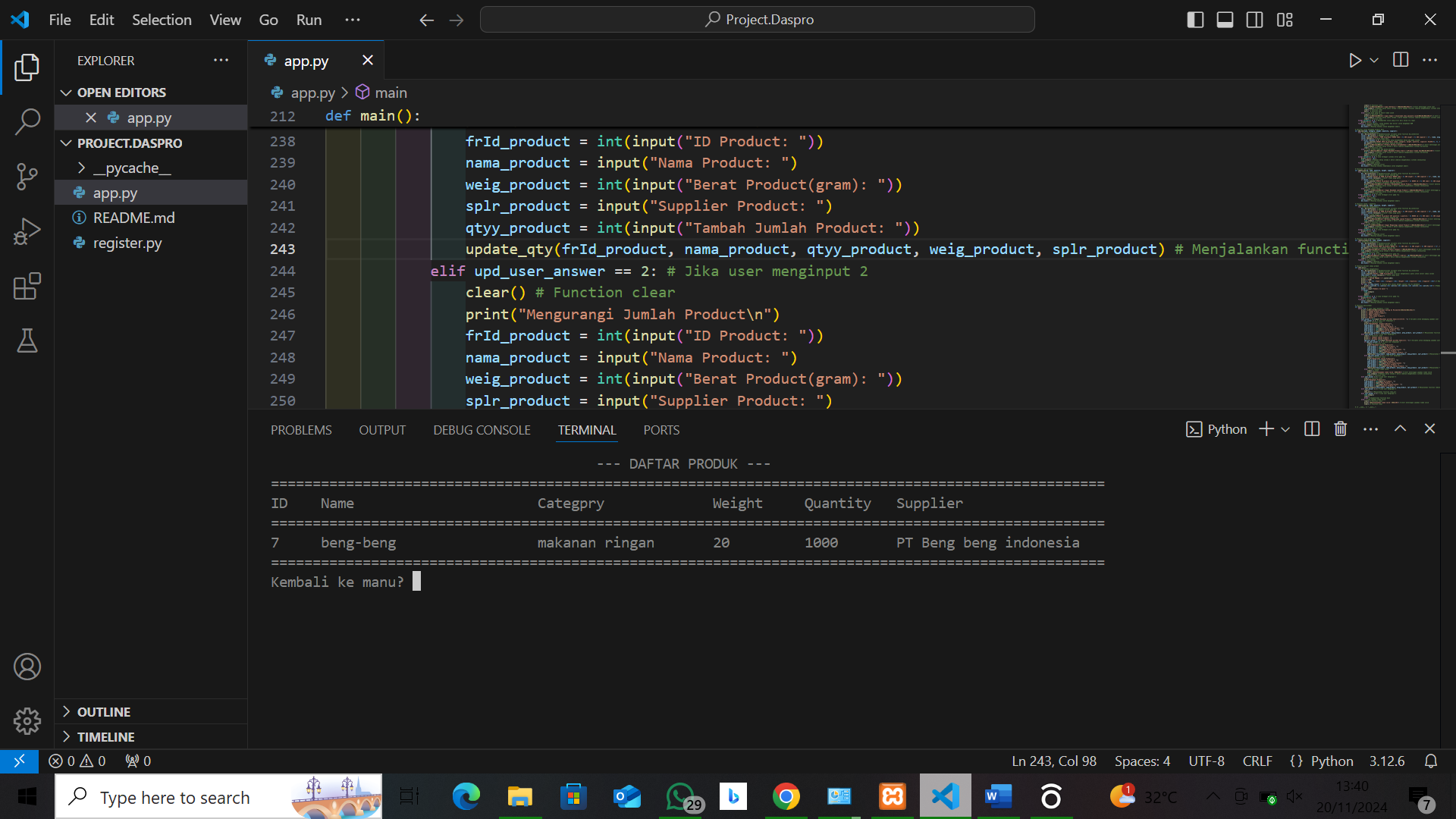
Gambar 3.2‑8: Form kurangi jumlah produk

### Hapus produk



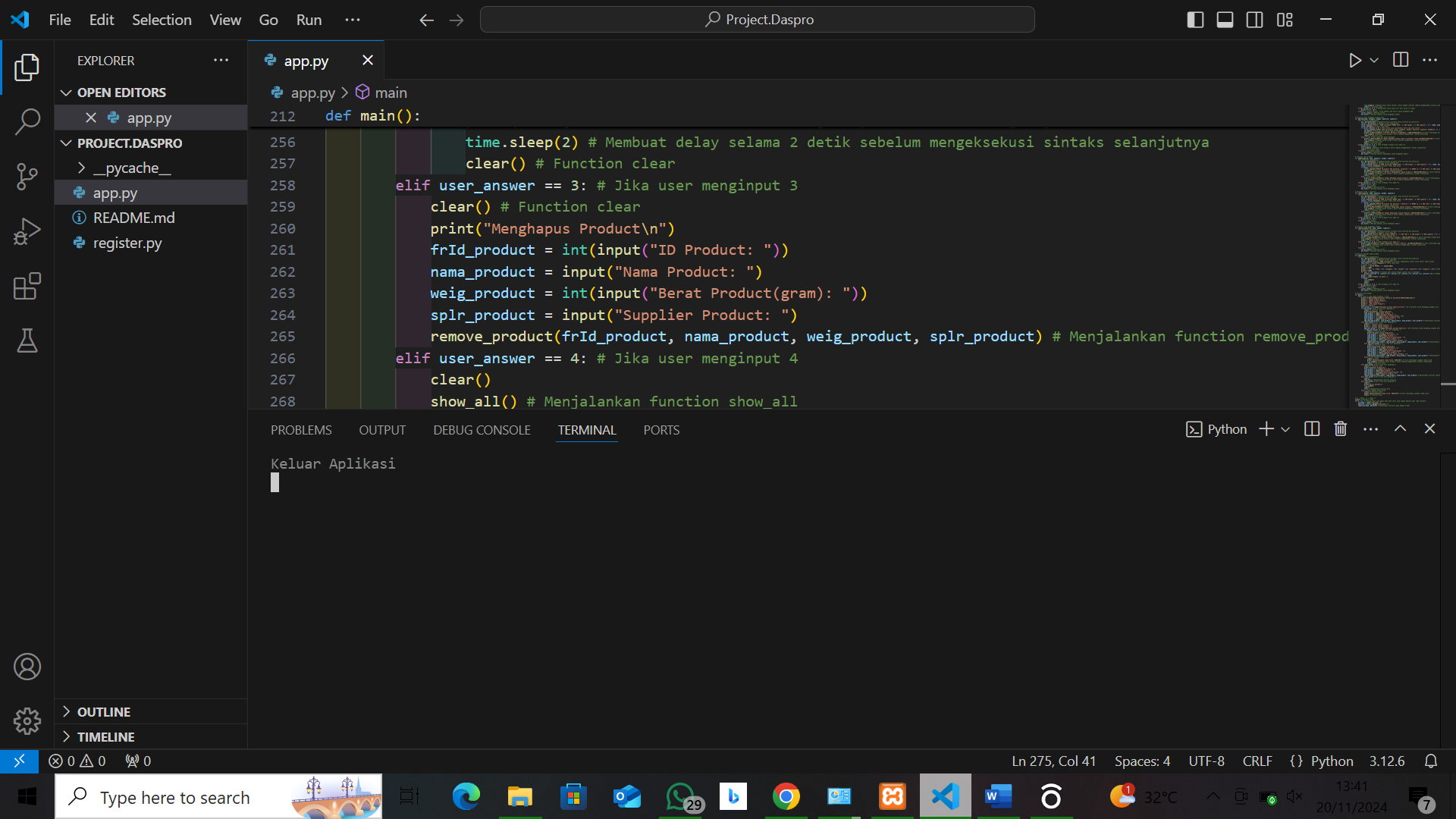
Gambar 3.2‑9: Form hapus produk

### Lihat semua produk



Gambar 3.2‑10: Form lihat semua produk

### Keluar aplikasi



Gambar 3.2‑11: Tampilan keluar aplikasi

# BAB IV PENUTUP



## Kesimpulan

Pada akhir makalah ini, dapat disimpulkan bahwa implementasi projek akhir mata kuliah Dasar Pemrograman dengan judul **Pengadaan Barang Menggunakan Bahasa Pemrograman Python dan Database MySQL** telah memberikan pemahaman kepada penulis tentang pengembangan aplikasi sederhana berbasis database. Penggunaan Python sebagai bahasa pemrograman memungkinkan proses pengolahan data berjalan efisien, sedangkan MySQL digunakan sebagai database untuk menyimpan dan mengelola data barang dengan terstruktur. Dengan aplikasi ini, proses pengadaan barang menjadi lebih mudah, terorganisasi, dan terhindar dari kesalahan data manual.

Melalui pengerjaan proyek ini, peserta memperoleh pengalaman langsung dalam:

1. Mengintegrasikan bahasa pemrograman dengan database.
2. Membuat alur logika program yang efektif melalui algoritma dan *flowchart*.
3. Mengasah keterampilan pemrograman dasar seperti input/output, perulangan, kondisi, dan manipulasi data.