

Modul Pembelajaran Integrasi Python dan Database

Durasi program kelas : 21 Jam

Pertemuan 1: Pengantar Python (3,5 jam)

Struktur data dasar: Python menyediakan tipe data sederhana seperti *integer*, *float*, *string*, dan *boolean*.

Misalnya, umur = 20 (int) atau nama = "Naja" (string). Struktur data koleksi penting antara lain **list** (daftar urut) dan **dictionary** (pasangan kunci-nilai). Contoh list:

```
buah = ["apel", "jeruk", "mangga"]  
print(buah[0]) # Output: apel
```

List menyimpan item berurutan yang dapat diubah.

Dictionary misalnya `mhs = {"nama": "Naja", "sem": 2}`

Dictionary berguna untuk mengasosiasikan kunci ke nilai.

Kontrol alur (flow control): Untuk membuat keputusan, Python menggunakan `if`, `elif`, `else`; dan untuk pengulangan, gunakan `for` atau `while`.

Contoh kondisi sederhana:

```
nilai = 85  
if nilai >= 90:  
    print("Nilai A")  
elif nilai >= 80:  
    print("Nilai B")  
else:  
    print("Nilai C atau D")
```

Pada contoh di atas, `if/elif/else` memeriksa kondisi dan mengeksekusi blok yang sesuai. Kode hanya akan di eksekusi apabila kondisi nya terpenuhi.

Kita bisa melakukan proses Looping dengan for misalnya: `for i in range(3):`
`print(i)` maka python akan mencetak 0,1,2.

atau Loop while yang akan mengulang terus menerus selama kondisi nya benar/terpenuhi.

contoh sintaks While Loop :

```
countdown = 5
while countdown > 0:
    print(f"T-minus {countdown}...")
    countdown -= 1
print("Lift off! 🚀")
```

Kontrol alur semacam ini penting agar program *dinamis* dan fleksibel.

Function (Fungsi) : Fungsi adalah blok kode yang diberi nama untuk tugas tertentu. Didefinisikan dengan def:

```
def salam():
    print("Halo, Naja!")
salam() # Output: Halo, Naja!
```

Parameter fungsi bisa ditambahkan di dalam kurung def, misal `def tambah(a, b):`
`return a + b.`

Fungsi akan mempermudah organisasi kode agar lebih rapi dan mengantisipasi pemakaian kode yang berulang ulang.

Tips untuk siswa & Kesalahan Umum yang sering terjadi :

1. **Indentasi.** Python bergantung indentasi. Pastikan indentasi konsisten (misal 4 spasi) atau akan muncul *IndentationError*.
2. **Titik dua (:).** Jangan lupa menambahkan : setelah kondisi if, loop for, atau definisi fungsi.
3. **Kesalahan penamaan.** Penamaan variabel sensitif huruf besar/kecil. Variabel sama persis.

4. **Tipe data:** Perhatikan operasi antar tipe, misalnya menambahkan integer dengan string menyebabkan `TypeError`. Cek kembali tipe sebelum operasi.

Teknik Debugging Dasar Python

Membaca pesan error (traceback): Saat Python menemukan kesalahan, ia menampilkan *traceback* yang menunjukkan baris bermasalah. Contoh `NameError` ketika variabel tak terdefinisi. Pesan ini menunjukkan baris dan jenis error, sehingga membantu melacak sumber masalah.

Teknik ini penting; bacalah pesan error dengan seksama untuk menemukan sumber bug.

Menggunakan `print()` untuk debug: Sisipkan perintah `print()` di bagian kode yang diragukan untuk memeriksa nilai variabel atau alur eksekusi. Misalnya, cetak hasil sementara di dalam loop atau fungsi untuk memastikan program berjalan seperti harapan. Seperti dijelaskan, menambah `print()` membantu melacak nilai variabel dan alur kode, mempercepat penemuan error.

Breakpoint dan debugger: Dalam IDE (mis. VS Code atau PyCharm), kita bisa menempatkan breakpoint (titik henti) untuk menghentikan program dan memeriksa nilai variabel secara interaktif. Python juga menyediakan modul bawaan `pdb` untuk debugging interaktif. Namun, untuk pemula cukup dengan membaca *traceback* dan menggunakan `print()`.

Kesalahan umum & Tips Debugging:

Lupa definisi: Error seperti “name ‘x’ is not defined” berarti variabel belum dibuat; pastikan variabel dan fungsi sudah dideklarasikan sebelum dipakai.

Indentasi tidak konsisten: Kombinasi tab/spasi atau level indentasi yang salah menyebabkan error. Terapkan satu gaya indentasi.

Typo sederhana: Typo pada nama variabel atau kata kunci (`pront` vs `print`) sering terjadi; periksa kembali ejaan.

Logika salah: Kadang-kadang bug disebabkan oleh logika program, bukan sintaks. Cek dengan contoh kecil atau *unit test*.

Pertemuan 2: Instalasi & Penggunaan XAMPP, phpMyAdmin, Dasar MySQL (3,5 jam)

Instalasi XAMPP:

XAMPP adalah paket server lokal yang meliputi Apache (web server), MySQL/MariaDB (database server), PHP, dan alat lain. Unduh XAMPP dari situs Apache Friends (sesuaikan OS) dan ikuti langkah instalasi.

Download link : <https://www.apachefriends.org/download.html>

Setelah terinstal (biasanya di C:\xampp), jalankan aplikasi XAMPP.



Gambar di samping menunjukkan panel XAMPP. Untuk menjalankan website di localhost, aktifkan dua modul utama: **Apache** dan **MySQL**. Klik tombol *Start* pada keduanya sampai indikator menyala hijau. *Apache* berfungsi sebagai web server, sedangkan *MySQL* (atau MariaDB) menangani database. Pastikan kedua layanan ini aktif sebelum melanjutkan.

Akses phpMyAdmin: Setelah Apache dan MySQL aktif, buka browser dan akses localhost/phpmyadmin. phpMyAdmin adalah antarmuka web untuk mengelola database MySQL. Di sini Anda dapat membuat, mengubah, dan menghapus database dan tabel.

Membuat Database & Tabel (CRUD dasar):

1. **Membuat Database:** Di phpMyAdmin, klik tab *New*. Masukkan nama database (misalnya db_naja) dan klik *Create*. Database baru akan muncul di sidebar. Anda juga bisa menggunakan perintah SQL dengan sintaks seperti berikut :

```
CREATE DATABASE db_naja;
```

Perintah SQL di atas akan membuat database baru.

2. **Membuat Tabel:** Pilih database baru, klik *Create Table*. Misalnya buat tabel tasks dengan kolom id (INT AUTO_INCREMENT, PK), judul (VARCHAR), selesai (BOOLEAN atau TINYINT). Setelah entri kolom, klik *Save*. Contoh SQL untuk membuat tabel:

```
CREATE TABLE tasks (  
  id INT NOT NULL AUTO_INCREMENT,  
  judul VARCHAR(255),  
  selesai TINYINT(1) NOT NULL DEFAULT 0,  
  PRIMARY KEY (id)  
);
```

Perintah CREATE TABLE seperti di atas membuat struktur tabel baru. Pada kode tersebut, id adalah kunci utama (primary key) dan bertambah otomatis.

3. **Operasi CRUD dasar:** CRUD adalah singkatan dari CREATE, READ, UPDATE & DELETE. Untuk menyimpan data (C), gunakan *Insert* di phpMyAdmin atau SQL:

```
INSERT INTO tasks (judul, selesai) VALUES ('Belajar Python', 0);
```

Untuk membaca data (R), gunakan SQL:

```
SELECT * FROM tasks;
```

Untuk memperbarui (U):

```
UPDATE tasks SET selesai = 1 WHERE id = 1;
```

Dan untuk menghapus (D):

```
DELETE FROM tasks WHERE id = 1;
```

Perhatikan sintaks SQL (INSERT, SELECT, UPDATE, DELETE) dan struktur tabel yang benar.

Tips untuk siswa & Kesalahan Umum:

- *Port Apache/MySQL*: Jika Apache gagal start, mungkin port 80/443 sudah dipakai aplikasi lain. Ubah port Apache (di Config) atau hentikan aplikasi konflik (misal Skype). *Folder htdocs*: Simpan file website (HTML/PHP) di folder `xampp/htdocs/your_project` untuk diakses via `localhost/your_project`.
- *Pengaturan ulang phpMyAdmin*: Jika phpMyAdmin tidak muncul, pastikan modul *mysql* berjalan dan alamat `localhost/phpmyadmin` benar.
- *Tipe data kolom*: Pilih tipe data sesuai kebutuhan (VARCHAR untuk teks pendek, INT untuk angka). Misalnya jangan gunakan VARCHAR terlalu pendek hingga data terpotong.
- *Primary key*: Jangan lupa centang *AI* (Auto Increment) pada kolom *id* agar unik. Tanpa primary key, data bisa duplikat dan susah dibedakan.

Pertemuan 3: Konsep Database Relasional, ERD, dan Normalisasi (3,5 jam)

Pengertian Database

1. Database adalah kumpulan informasi yang disimpan di dalam komputer secara sistematis untuk memperoleh informasi dari basis data tersebut.
2. Database adalah representasi kumpulan fakta yang saling berhubungan disimpan secara bersama, untuk memenuhi berbagai kebutuhan.
3. Database merupakan sekumpulan informasi yang saling berkaitan pada suatu subjek tertentu untuk tujuan tertentu pula.
4. Database adalah susunan record data operasional lengkap dari suatu organisasi atau perusahaan, yang diorganisir dan disimpan secara terintegrasi dengan menggunakan metode tertentu sehingga mampu memenuhi informasi yang optimal yang dibutuhkan oleh para pengguna.

Asal Mula Istilah Database

Istilah “database” berawal dari ilmu komputer. Meskipun kemudian artinya semakin luas, memasukkan hal-hal yang di luar bidang elektronika, artikel mengenai database

komputer. Catatan yang mirip dengan database sebenarnya sudah ada sebelum revolusi industri yaitu dalam bentuk buku besar, kuitansi dan kumpulan data yang berhubungan dengan bisnis.

Konsep Dasar Database

Konsep dasar database adalah kumpulan dari catatan, atau potongan dari pengetahuan. Sebuah database memiliki penjelasan terstruktur dari jenis fakta yang tersimpan di dalamnya: penjelasan ini disebut skema. Ada banyak cara untuk mengorganisasi skema, atau memodelkan struktur database: ini dikenal sebagai database model atau model data. Model yang umum digunakan sekarang adalah model relasional, yang menurut istilah yaitu mewakili semua informasi dalam bentuk tabel yang saling berhubungan dimana setiap tabel terdiri dari baris dan kolom (definisi yang sebenarnya menggunakan terminologi matematika). Dalam model ini, hubungan antar tabel diwakili dengan menggunakan nilai yang sama antar tabel.



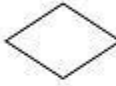

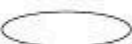


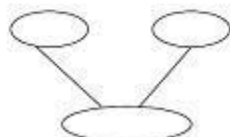

Perangkat Untuk Membuat Database.

Database dapat dibuat dan diolah dengan menggunakan suatu program komputer, yaitu yang biasa disebut dengan software (perangkat lunak). Software yang digunakan untuk mengelola dan memanggil kueri (query) database disebut Database Management System (DBMS) atau jika diterjemahkan kedalam bahasa Indonesia berarti "Sistem Manajemen Basis Data". DBMS terdiri dari dua komponen, yaitu Relational Database Management System (RDBMS) dan Overview of Database Management System (ODBMS). RDBMS meliputi Interface Drivers, SQL Engine, Transaction Engine, Relational Engine, dan Storage Engine. Sedangkan ODBMS meliputi Language Drivers, Query Engine, Transaction Engine, dan Storage Engine.

Database Relasional: Basis data relasional menyimpan informasi dalam bentuk tabel yang saling berhubungan. Setiap tabel merepresentasikan entitas (seperti tasks, users) dan kolomnya adalah atribut. Relasi antar tabel dicapai dengan kunci (key) yang sama, misalnya `id_user` di tabel tasks dan users. Model ini mudah dipahami karena menyerupai tabel spreadsheet.

ERD (Entity Relationship Diagram): ERD adalah diagram yang menggambarkan entitas (tabel) dan relasinya. Komponen utama ERD meliputi entitas (kotak persegi panjang), atribut (di dalam kotak atau elips), dan relasi (garis yang menghubungkan entitas). Contoh ilustrasi ERD sederhana (sistem perhotelan) ditampilkan di bawah:

Komponen ERD

<u>Notasi</u>	<u>Arti</u>
1. 	1. Entity
2. 	2. Weak Entity
3. 	3. Relationship
4. 	4. Identifying Relationship
5. 	5. Atribut
6. 	6. Atribut Primary Key
7. 	7. Atribut Multivalue
8. 	8. Atribut Composite
9. 	9. Atribut Derivatif

Contoh ERD Sistem Perhotelan

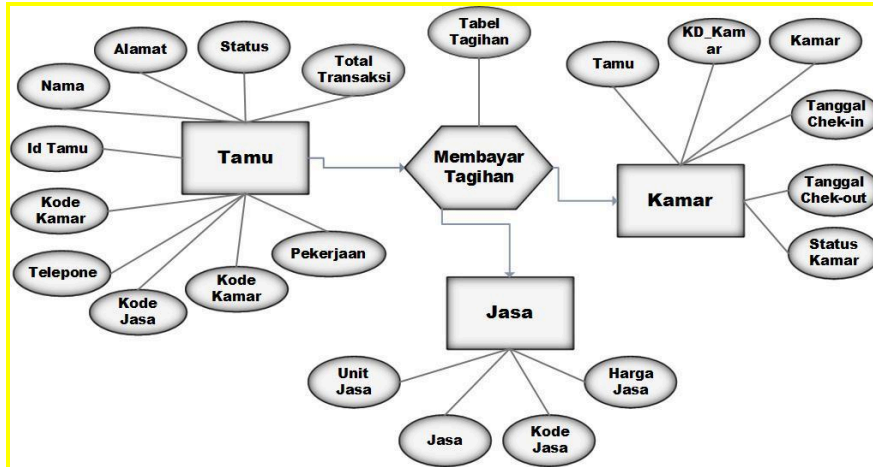


Diagram di atas menunjukkan bagaimana entitas terhubung. Misalnya, satu *Tamu* dapat memiliki banyak *Transaksi*, sedangkan satu *Kamar* dapat ditempati banyak *Tamu* (relasi 1:M). Tabel *tasks* pada studi kasus nanti dapat dihubungkan ke entitas lain misalnya *users* melalui *foreign key*.

Normalisasi Basis Data

Normalisasi adalah proses menyusun tabel agar data terorganisir dengan baik, menghindari duplikasi (redundansi), dan mempermudah pemeliharaan. Tujuannya membentuk entitas “non-redundant” yang stabil.

Contoh sederhana: jika satu tabel menyimpan data karyawan dan proyek (terduplikasi bila karyawan mengerjakan banyak proyek), proses normalisasi akan menguraikannya menjadi dua tabel terpisah (tabel karyawan, tabel proyek) yang kemudian dihubungkan.

Tahapan normalisasi yang umum adalah:

- **1NF (First Normal Form)**: Setiap kolom hanya mengandung nilai atomik (tidak ada daftar/multivalued di satu sel). Contohnya, kolom *hobi* yang berisi beberapa nilai harus dipecah.
- **2NF (Second Normal Form)**: Memenuhi 1NF dan setiap atribut non-primer bergantung penuh pada kunci utama. Jika tabel gabungan (dekomposisi) diperlukan, pecah menjadi tabel yang lebih kecil.
- **3NF (Third Normal Form)**: Memenuhi 2NF dan tidak ada ketergantungan transitif antar kolom. Data bukan kunci tidak bergantung pada kolom non-kunci lainnya.

Mengikuti normalisasi membantu menghindari pemborosan ruang dan inkonsistensi data.

Tips untuk siswa & Kesalahan umum:

- **Duplikasi data**: Hindari menyimpan informasi yang sama di banyak tabel. Gunakan tabel terpisah dan hubungan kunci (misal, simpan data pengguna sekali dalam *users*, lalu hubungkan di tabel *tasks* dengan *user_id*).

- *Kunci utuh*: Pastikan menetapkan **primary key** pada setiap tabel (biasanya kolom id unik. Ini mencegah data duplikat dan mempermudah referensi antar tabel.
- *Skema dan ERD*: Buat sketsa ERD sebelum membuat tabel di MySQL. Ini membantu memvisualisasikan entitas dan relasi sehingga struktur database lebih jelas.
- *Tidak over-normalisasi*: Untuk latihan, cukup sampai 3NF. Over-normalisasi (terlalu banyak tabel) dapat membuat query menjadi kompleks tanpa banyak keuntungan. Fokus pada kebutuhan aplikasi.

Pertemuan 4: Integrasi Python dengan MySQL (3,5 jam)

Driver MySQL untuk Python: Untuk menghubungkan Python ke MySQL, perlu menginstal library `mysql-connector-python` atau `PyMySQL`. Misalnya, dengan pip:

```
pip install mysql-connector-python
```

Python memerlukan *driver* untuk akses MySQL, dan dalam contoh ini digunakan *MySQL Connector*. Setelah pemasangan selesai, driver siap digunakan.

Membuat Koneksi & Menjalankan Query: Berikut contoh kode Python sederhana untuk menyambung dan mengambil data:

```
import mysql.connector

# Membuat koneksi ke database
conn = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="db_naja"
)
cursor = conn.cursor()

# Contoh: Membaca data dari tabel tasks
cursor.execute("SELECT id, judul, selesai FROM tasks")
for (id, judul, selesai) in cursor:
    print(f"{id}. {judul} – Selesai: {selesai}")

# Jangan lupa tutup koneksi
conn.close()
```

Kode di atas membuka koneksi ke server localhost, menggunakan user root (password biasanya kosong di lokal), lalu `conn.cursor()` digunakan untuk menjalankan perintah SQL. Metode `execute()` menjalankan query, dan loop for membaca hasilnya. Setelah selesai, panggil `conn.close()` untuk menutup koneksi.

Cara ini memungkinkan program Python berinteraksi dengan database MySQL secara langsung.

Memasukkan dan Mengubah Data: Selain SELECT, Anda dapat menjalankan INSERT, UPDATE, dan DELETE lewat `cursor.execute()`.

Contohnya, untuk menambah tugas baru:

```
baru = ("belajar python", 0)
sql = "INSERT INTO tasks (judul, selesai) VALUES (%s, %s)"
cursor.execute(sql, baru)
conn.commit() # simpan perubahan ke database
```

Setelah `execute`, panggil `conn.commit()` untuk memastikan perubahan ditulis ke database. Jika tidak memanggil `commit`, data tidak akan tersimpan.

Tips untuk siswa & Kesalahan Umum:

- *Instalasi driver:* Pastikan pip sudah di-*update* dan Python yang digunakan adalah versi yang sama dengan yang di-pip. Jika muncul error `ModuleNotFoundError`, cek kembali instalasi `mysql-connector-python`.
- *Penulisan query:* Dalam `cursor.execute()`, beri format string dengan `%s` dan memasukkan tuple argumen untuk menghindari SQL Injection. Jangan lupa ubah ejaan nama kolom/tabel sesuai yang ada di database.
- *Close dan Commit:* Lupa menutup koneksi (`conn.close()`) dapat mengakibatkan sumber daya tidak terlepas. Lupa `conn.commit()` (untuk operasi tulis) mengakibatkan perubahan tak tersimpan.
- *Error koneksi:* Jika muncul error koneksi (Access denied), periksa kredensial MySQL (user/password) dan pastikan MySQL sudah aktif di XAMPP.

Pertemuan 5: Studi Kasus – Aplikasi Terminal To-Do List CRUD (3,5 jam)

Deskripsi Kasus: Kita akan membuat aplikasi tugas sederhana di terminal (console) untuk menambah, melihat, mengubah, dan menghapus entri tugas (To-Do List). Tugas disimpan dalam tabel `tasks` MySQL. Tiap tugas memiliki `id`, `judul`, dan status `selesai` (0/1).

Langkah-langkah:

1. **Buat tabel:** Pastikan tabel `tasks` sudah dibuat (seperti di pertemuan sebelumnya).
2. **Buat antarmuka menu:** Tampilkan pilihan sederhana di terminal:
 - 1. Tampilkan tugas (READ)
 - 2. Tambah tugas (CREATE)
 - 3. Tandai selesai/ubah tugas (UPDATE)
 - 4. Hapus tugas (DELETE)
 - 5. Keluar
3. **Implementasi fungsi CRUD:** Tulis fungsi Python untuk tiap operasi. Contoh fungsi menambah tugas:

```
def tambah_tugas(conn, judul):  
    cursor = conn.cursor()  
    sql = "INSERT INTO tasks (judul, selesai) VALUES (%s, %s)"  
    cursor.execute(sql, (judul, 0))  
    conn.commit()  
    print("Tugas ditambahkan!")
```

dan fungsi menampilkan tugas:

```
def tampilkan_tugas(conn):  
    cursor = conn.cursor()  
    cursor.execute("SELECT id, judul, selesai FROM tasks")  
    for (id, judul, selesai) in cursor:  
        status = "✓" if selesai else "X"  
        print(f"{id}. {judul} [{status}]")
```

4. **Loop utama:** Gunakan `while True:` untuk terus menampilkan menu hingga pengguna memilih keluar. Baca input dengan `input()` dan panggil fungsi sesuai pilihan. Contoh kerangka program:

```
while True:
    print("Menu To-Do List:")
    print("1. Tampilkan tugas")
    print("2. Tambah tugas")
    print("3. Tandai selesai")
    print("4. Hapus tugas")
    print("5. Keluar")
    pilihan = input("Pilih (1-5): ")
    if pilihan == "1":
        tampilkan_tugas(conn)
    elif pilihan == "2":
        judul = input("Masukkan judul tugas: ")
        tambah_tugas(conn, judul)
    elif pilihan == "3":
        tid = input("ID tugas yang selesai: ")
        tandai_selesai(conn, int(tid))
    elif pilihan == "4":
        tid = input("ID tugas yang dihapus: ")
        hapus_tugas(conn, int(tid))
    else:
        break
```

5. **Jalankan aplikasi:** Pastikan program terhubung ke database (`conn = mysql.connector.connect(...)` seperti sebelumnya) sebelum loop utama. Kemudian ujicoba berbagai operasi di terminal.

Tips untuk siswa & Kesalahan Umum:

- *Konversi input:* `input()` menghasilkan string; jika akan digunakan sebagai angka (misalnya `id`), konversi ke `int`. Ketika lupa mengkonversi, operasi SQL yang mengharapkan angka bisa gagal.
- *Penanganan kesalahan:* Selalu validasi masukan pengguna. Misalnya, jika ID tidak ditemukan, tangani dengan pesan error yang baik, agar program tidak *crash*.
- *Commit setelah CRUD:* Lupa memanggil `conn.commit()` setelah `INSERT/UPDATE/DELETE` membuat perubahan tidak tersimpan.
- *Koneksi berulang:* Sebaiknya buat satu koneksi (`conn`) di awal program lalu pakai untuk semua operasi, daripada buka-tutup berulang-ulang.

Pertemuan 6: Finalisasi Proyek dan Review Kode (3,5 jam)

Rekap dan Finalisasi: Di pertemuan terakhir, fokus pada penyempurnaan aplikasi dan review seluruh kode. Pastikan semua fitur To-Do List berjalan: memasukkan tugas, menampilkan list, memperbarui status selesai, dan menghapus. Lakukan pengujian ulang untuk setiap fungsi.

Review Kode: Baca kembali kode bersama, perhatikan:

- Struktur kode (apakah sudah modular dengan fungsi-fungsi terpisah).
- Pemberian nama variabel dan fungsi (deskriptif dan konsisten).
- Komentar: Tambahkan komentar singkat di bagian kompleks agar mudah dipahami.
- Pemformatan: Pastikan indentasi konsisten dan spasi di dalam operator.
- Penanganan error: Cek apakah ada kemungkinan error yang belum tertangani (misalnya koneksi gagal, query error).

Praktik Langsung: Berikan kesempatan untuk memperbaiki kesalahan pada kode secara langsung (misal typo, bug logika). Diskusikan kenapa terjadi dan solusinya. Misalnya, jika fungsi update tak berfungsi, periksa kembali query SQL-nya.

Tips Penting:

Backup database: Sebelum perubahan besar, simpan backup (export) database agar data tidak hilang saat eksperimen.

Testing incremental: Lakukan pengujian setelah menambah fitur kecil daripada sekaligus di akhir. Ini mempermudah menemukan bagian yang salah.

Konsistensi kode: Pakai satu gaya kode (misal PEP8) untuk memudahkan pemeriksaan.

Dokumentasi ringan: Buat dokumentasi singkat file `README.md` yang menjelaskan cara menjalankan aplikasi.

Tidak Ada Evaluasi Formal: Modul ini berfokus pada praktik langsung tanpa ujian tertulis. Setiap pertemuan peserta dapat mencoba sendiri (hands-on) dengan bimbingan, dan pertanyaan diskusi dipersilakan. Tujuannya memahami sintaks dan konsep, serta mampu mengaplikasikannya secara mandiri.
