LAPORAN PRAKTIKUM MOBILE PROGRAMMING
MODUL 13

Nama : Faril Isra Albiansyah

NIM : 240605110087

Kelas : Mobile Programming B

Tanggal : 17 – 11 – 2025

JURUSAN TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM

MALANG

GANJIL 2025/2026

I. Tujuan

Tujuan dari praktikum Modul 13 ini adalah:

- Memahami konsep dasar operasi CRUD (Create, Read, Update, Delete) dalam pengembangan aplikasi mobile.
- Mengimplementasikan paket http pada Flutter untuk berkomunikasi dengan server melalui REST API.
- Memahami cara menangani respons dari server dalam format JSON, termasuk parsing data ke dalam objek Model.
- Menangani error handling dan status kode HTTP (seperti 200 OK, 201 Created) dengan benar.
- Membangun antarmuka pengguna (UI) yang interaktif dan dinamis menggunakan setState untuk menampilkan perubahan data secara real-time.

III. Langkah Kerja

Berikut adalah ringkasan langkah-langkah praktikum yang telah dilakukan:

1. Konfigurasi Proyek: Menambahkan dependensi http, intl, dan google_fonts pada file pubspec.yaml serta mengatur izin internet (jika diperlukan pada AndroidManifest).
2. Pembuatan Model Data: Membuat class Post dengan metode factory untuk mengubah data JSON dari API menjadi objek Dart.
3. Implementasi UI Dasar: Membangun tampilan menggunakan Scaffold, AppBar, dan FloatingActionButton.
4. Logika CRUD & HTTP Request:
    a. Membuat fungsi createPost() menggunakan http.post untuk mengirim data baru.
    b. Membuat fungsi updatePost() menggunakan http.put untuk memperbarui data yang ada.
    c. Membuat fungsi deletePost() menggunakan http.delete untuk menghapus data.
5. Interaksi Pengguna (Dialog):

a. Membuat AlertDialog untuk form input data (Tambah & Update).
b. Membuat dialog konfirmasi sebelum melakukan penghapusan data.

6. Integrasi State Management: Menggunakan setState() di setiap fungsi logika untuk memastikan tampilan Card atau pesan status diperbarui sesuai respons dari server.

IV. Screenshot Hasil

1. Kondisi Tampilan Awal :

Aplikasi saat pertama dibuka, belum ada data yang dimuat.

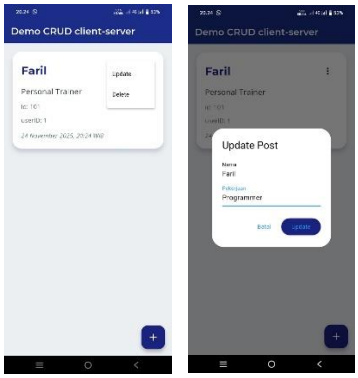
2. Kondisi Dialog Tambah Data

Form input (AlertDialog) saat tombol tambah (+) ditekan.

3. Kondisi Data Berhasil ditambahkan :

Tampilan Card setelah data berhasil ditambahkan (Create).

4. Kondisi Dialog Update Data



Form edit data yang muncul saat menu "Update" dipilih.

5. Kondisi Data Setelah Update

Tampilan data yang berubah setelah proses update berhasil.

6. Konfirmasi Hapus

Dialog peringatan sebelum menghapus data.

7. Kondisi Notifikasi (SnackBar)

Pesan hijau di bawah layar menandakan operasi berhasil.

Pada praktikum ini, aplikasi bekerja dengan cara mengirimkan permintaan (request) ke endpoint API publik (JSONPlaceholder) dan memproses response yang diterima. Berikut adalah penjelasan komponen utamanya:

1. Fungsi createPost(), updatePost(), dan deletePost() Ketiga fungsi ini menggunakan kata kunci async dan await karena proses pengambilan data dari internet membutuhkan waktu (asinkron).

createPost(String title, String body): Mengirim permintaan HTTP POST ke server. Body permintaan berisi data JSON dari input pengguna. Jika server merespons dengan kode 201 (Created), data hasil balikan (JSON) di-decode menjadi objek Post dan disimpan ke variabel createdPost.

updatePost(int id, ...): Mengirim permintaan HTTP PUT ke URL spesifik ID. Ini mensimulasikan penggantian data di server. Jika sukses (kode 200 OK), variabel createdPost diperbarui dengan data baru yang diedit pengguna.

deletePost(int id): Mengirim permintaan HTTP DELETE. Jika sukses (kode 200 OK), variabel createdPost diatur menjadi null, yang menyebabkan UI kembali ke tampilan awal (kosong).

2. Peran Variabel createdPost dan AlertDialog

createdPost: Adalah variabel state bertipe objek Post? (nullable). Variabel ini menjadi penentu tampilan. Jika null, aplikasi menampilkan pesan "Belum ada data". Jika terisi, aplikasi menampilkan widget Card berisi detail data.

AlertDialog: Digunakan sebagai antarmuka input. Widget ini membungkus TextField untuk judul dan pekerjaan. Dialog ini memanggil fungsi logika (createPost atau updatePost) ketika tombol "Simpan" ditekan, lalu menutup dirinya sendiri menggunakan Navigator.pop().

3. Peran setState() dalam Pembaruan UI setState() adalah kunci dari reaktivitas aplikasi ini. Flutter tidak akan membangun ulang tampilan jika variabel berubah tanpa dibungkus fungsi ini.

Saat Create/Update: setState() dipanggil untuk mengisi createdPost dengan data baru, sehingga layar otomatis menampilkan Card data tersebut.

Saat Delete: setState() dipanggil untuk mengubah createdPost menjadi null, sehingga layar otomatis menghapus Card dan kembali menampilkan teks instruksi awal.

IV. Kesimpulan

Berdasarkan praktikum yang telah dilakukan, dapat disimpulkan bahwa:

Flutter dapat dengan mudah diintegrasikan dengan layanan backend menggunakan paket http untuk melakukan operasi CRUD lengkap.

Setiap operasi CRUD dipetakan ke metode HTTP standar: POST (Create), GET (Read), PUT (Update), dan DELETE (Delete).

Pengelolaan state aplikasi sangat bergantung pada pemahaman logika async/await untuk menangani jeda waktu jaringan, serta penggunaan setState() yang tepat untuk menyinkronkan data variabel dengan tampilan antarmuka pengguna.

Penerapan validasi input dan penanganan status kode (seperti pengecekan statusCode == 200) sangat penting agar aplikasi tidak crash saat terjadi kesalahan komunikasi dengan server.