

**LAPORAN TEORI MOBILE PROGRAMMING**  
**MODUL 13**



Nama : Erai Bagusalim  
NIM : 240605110088  
Kelas : Mobile Programming B  
Tanggal : 18 November 2025

**PROGRAM STUDI TEKNIK INFORMATIKA**  
**FAKULTAS SAINS DAN TEKNOLOGI**  
**UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG**  
**GANJIL 2025/2026**

**i. Tujuan**

1. Memahami cara implementasi operasi CRUD di Flutter.
2. Penerapan paket http untuk komunikasi client-server.
3. Penanganan respons dari REST API (termasuk error handling).
4. pembuatan antarmuka interaktif untuk manajemen data dinamis.

## **ii. Langkah Kerja**

1. Penambahan dependensi (http, intl, google\_fonts) pada file pubspec.yaml.
2. Pembuatan UI menggunakan Scaffold, Card, AlertDialog, dan PopupMenuButton.
3. Implementasi fungsi createPost(), updatePost(), dan deletePost() beserta logika untuk mengirim permintaan HTTP dan mem-parsing respons JSON.
4. Pembuatan dialog interaktif untuk input data (tambah dan update) serta untuk konfirmasi (hapus).
5. Penggunaan setState() untuk memperbarui tampilan secara reaktif berdasarkan perubahan pada state data (createdPost).

### iii. Screenshot Hasil

#### a. Kode Program

##### Main.dart

```
1 import 'package:flutter/material.dart';
2 import 'package:http/http.dart' as http;
3 import 'dart:convert';
4 import 'package:intl/intl.dart';
5 import 'package:google_fonts/google_fonts.dart';
6
7 class Post {
8   final int id;
9   final int userId;
10  final String title;
11  final String body;
12  final DateTime createdAt;
13
14  Post({
15    required this.id,
16    required this.title,
17    required this.body,
18    required this.createdAt,
19    required this.userId
20  });
21
22  factory Post.fromJson(Map<String, dynamic> json) {
23    return Post(
24      id: json['id'],
25      userId: json['userId'] ?? 1,
26      title: json['title'],
27      body: json['body'],
28      createdAt: DateTime.now(),
29    );
30  }
31 }
```

```
1 void main() {
2   runApp(const MyApp());
3 }
4
5 class MyApp extends StatelessWidget {
6   const MyApp({super.key});
7
8   @override
9   Widget build(BuildContext context) {
10    return MaterialApp(
11      title: 'Flutter CRUD Demo',
12      theme: ThemeData(
13        primaryColor: Colors.indigo[900],
14        colorScheme: ColorScheme.fromSwatch().copyWith(secondary: Colors.lightBlueAccent),
15        scaffoldBackgroundColor: Colors.blueGrey[50],
16        appBarTheme: AppBarTheme(
17          backgroundColor: Colors.indigo[900],
18          elevation: 4,
19          titleTextStyle: GoogleFonts.montserrat(
20            textStyle: const TextStyle(
21              color: Colors.white,
22              fontSize: 22,
23              fontWeight: FontWeight.w600,
24            ),
25          ),
26        ),
27      ),
28      home: const HomePage(),
29      debugShowCheckedModeBanner: false,
30    );
31  }
32 }
33 }
```

```
1 class HomePage extends StatefulWidget {
2   const HomePage({super.key});
3
4   @override
5   State<HomePage> createState() => _HomePageState();
6 }
7
8 class _HomePageState extends State<HomePage> {
9   Post? createdPost;
10  Future<void> createPost(String title, String body) async {
11    final url = Uri.parse('https://jsonplaceholder.typicode.com/posts');
12
13    try {
14      final response = await http.post(
15        url,
16        headers: <String, String>{'Content-Type': 'application/json; charset=UTF-8'},
17        body: jsonEncode(<String, dynamic>{
18          'title': title,
19          'body': body,
20          'userId': 1,
21        })),
22    );
23
24    if (!mounted) return;
25
26    if (response.statusCode == 201) {
27      final data = jsonDecode(response.body);
28      setState(() {
29        createdPost = Post.fromJson(data);
30      });
31    } else {
32      ScaffoldMessenger.of(context).showSnackBar(
33        const SnackBar(content: Text('Gagal menambahkan post')),
34      );
35    }
36  } catch (e) {
37    ScaffoldMessenger.of(context).showSnackBar(
38      SnackBar(content: Text('Error: $e')),
39    );
40  }
41 }
```

```
1 Future<void> updatePost(int id, String title, String body) async {
2   final url = Uri.parse('https://jsonplaceholder.typicode.com/posts/1');
3
4   try {
5     final response = await http.put(
6       url,
7       headers: <String, String>{'Content-Type': 'application/json; charset=UTF-8'},
8       body: jsonEncode({
9         'id': id,
10        'title': title,
11        'body': body,
12        'userId': createdPost!.userId,
13      })),
14    );
15
16    if (!mounted) return;
17
18    if (response.statusCode == 200) {
19      final data = jsonDecode(response.body);
20      setState(() {
21        createdPost = Post(
22          id: data['id'],
23          userId: data['userId'] ?? createdPost!.userId,
24          title: data['title'],
25          body: data['body'],
26          createdAt: createdPost!.createdAt,
27        );
28      });
29      ScaffoldMessenger.of(context).showSnackBar(
30        const SnackBar(content: Text('Post berhasil diperbarui'),
31          backgroundColor: Colors.green,
32        ),
33      );
34    } else {
35      ScaffoldMessenger.of(context).showSnackBar(
36        const SnackBar(content: Text('Gagal memperbarui post')),
37      );
38    }
39  } catch (e) {
40    ScaffoldMessenger.of(context).showSnackBar(
41      SnackBar(content: Text('Error: $e')),
42    );
43  }
44 }
```

```

1 Future<void> deletePost(int id) async {
2   final url = Uri.parse('https://jsonplaceholder.typicode.com/posts/$id');
3
4   try{
5     final response = await http.delete(
6       url,
7       headers: {'Content-Type': 'application/json; charset=UTF-8'},
8     );
9
10    if(!mounted) return;
11
12    if (response.statusCode == 200) {
13      ScaffoldMessenger.of(context).showSnackBar(
14        const SnackBar(
15          content: Text('Post berhasil dihapus!'),
16          backgroundColor: Colors.green,
17        ),
18      );
19      setState(() {
20        createdPost = null;
21      });
22    } else {
23      ScaffoldMessenger.of(context).showSnackBar(
24        const SnackBar(
25          content: Text('Gagal menghapus post'),
26          backgroundColor: Colors.red
27        ),
28      );
29    }
30  } catch (e) {
31    ScaffoldMessenger.of(context).showSnackBar(
32      SnackBar(
33        content: Text('Error: $e'),
34        backgroundColor: Colors.red
35      ),
36    );
37  }
38 }

```

```

1 void showDeleteConfirmationDialog(int id) {
2   showDialog(
3     context: context,
4     builder: (context) => AlertDialog(
5       title: const Text('Konfirmasi Hapus'),
6       content: const Text('Apakah Anda yakin ingin menghapus post ini?'),
7       actions: [
8         TextButton(
9           onPressed: () => Navigator.pop(context),
10          child: const Text('Batal'),
11        ),
12        ElevatedButton(
13          onPressed: () {
14            deletePost(id);
15            Navigator.pop(context);
16          },
17          style: ElevatedButton.styleFrom(
18            backgroundColor: Colors.red,
19          ),
20          child: const Text('Hapus'),
21        ),
22      ],
23    ),
24  );
25 }

```

```

1 void showUpdatePostDialog(Post currentPost){
2   final titleController = TextEditingController(text: currentPost.title);
3   final bodyController = TextEditingController(text: currentPost.body);
4
5   showDialog(
6     context: context,
7     builder: (context) => AlertDialog(
8       title: const Text('Update Post'),
9       content: Column(
10        mainAxisAlignment: MainAxisAlignment.min,
11        children: [
12          TextField(
13            controller: titleController,
14            decoration: const InputDecoration(labelText: 'Nama'),
15          ),
16          TextField(
17            controller: bodyController,
18            decoration: const InputDecoration(labelText: 'Pekerjaan'),
19          ),
20        ],
21      ),
22      actions: [
23        TextButton(
24          onPressed: () => Navigator.pop(context),
25          child: const Text('Batal'),
26        ),
27        ElevatedButton(
28          onPressed: () {
29            final updatedTitle = titleController.text.trim();
30            final updatedBody = bodyController.text.trim();
31
32            if(updatedTitle.isNotEmpty && updatedBody.isNotEmpty) {
33              updatePost(currentPost.id, updatedTitle, updatedBody);
34              Navigator.pop(context);
35            } else {
36              ScaffoldMessenger.of(context).showSnackBar(
37                const SnackBar(content: Text('Nama dan Pekerjaan harus diisi')),
38              );
39            }
40          },
41          style: ElevatedButton.styleFrom(
42            backgroundColor: Colors.indigo[900],
43          ),
44          child: const Text('Update'),
45        ),
46      ],
47    );
48  }
49 }

```

```

1 void showAddPostDialog() {
2   final titleController = TextEditingController();
3   final bodyController = TextEditingController();
4
5   showDialog(
6     context: context,
7     builder: (context) => AlertDialog(
8       title: const Text('Tambah Post'),
9       content: Column(
10        mainAxisAlignment: MainAxisAlignment.min,
11        children: [
12          TextField(
13            controller: titleController,
14            decoration: const InputDecoration(labelText: 'Nama'),
15          ),
16          TextField(
17            controller: bodyController,
18            decoration: const InputDecoration(labelText: 'Pekerjaan'),
19          ),
20        ],
21      ),
22      actions: [
23        TextButton(
24          onPressed: () => Navigator.pop(context),
25          child: const Text('Batal'),
26        ),
27        ElevatedButton(
28          onPressed: () {
29            final title = titleController.text.trim();
30            final body = bodyController.text.trim();
31            if(title.isNotEmpty && body.isNotEmpty) {
32              createPost(title, body);
33              Navigator.pop(context);
34            } else {
35              ScaffoldMessenger.of(context).showSnackBar(
36                const SnackBar(content: Text('Nama dan Pekerjaan harus diisi')),
37              );
38            }
39          },
40          style: ElevatedButton.styleFrom(
41            backgroundColor: Colors.indigo[900],
42          ),
43          child: const Text('Simpan'),
44        ),
45      ],
46    );
47  }
48 }

```

```

1  @override
2  Widget build(BuildContext context) {
3    return Scaffold(
4      appBar: AppBar(
5        title: const Text('Demo CRUD client-server'),
6      ),
7      body: Center(
8        child: createdPost == null
9        ? Text(
10          'Demo CRUD client-server di Flutter',
11          style: GoogleFonts.montserrat(
12            fontSize: 18,
13            fontWeight: FontWeight.w500,
14            color: Colors.indigo[900],
15          ),
16        )
17        : ListView(
18          padding: const EdgeInsets.symmetric(vertical: 16),
19          children: [
20            Padding(
21              padding: const EdgeInsets.symmetric(horizontal: 16),
22              child: Card(
23                elevation: 8,
24                shadowColor: Colors.indigo.withOpacity(0.3),
25                shape: RoundedRectangleBorder(
26                  borderRadius: BorderRadius.circular(20),
27                ),
28                child: Padding(
29                  padding: const EdgeInsets.all(20),
30                  child: Column(
31                    crossAxisAlignment: CrossAxisAlignment.start,
32                    children: [
33                      Row(
34                        mainAxisAlignment: MainAxisAlignment.spaceBetween,
35                        children: [
36                          Expanded(
37                            child: Text(
38                              createdPost!.title,
39                              style: GoogleFonts.montserrat(
40                                fontSize: 26,
41                                fontWeight: FontWeight.w700,
42                                color: Colors.indigo[900],
43                              ),
44                            ),
45                          PopupMenuButton<String>(
46                            itemBuilder: (context) => [
47                              const PopupMenuItem(
48                                value: 'update',
49                                child: Text('Update'),
50                              ),
51                              const PopupMenuItem(
52                                value: 'delete',
53                                child: Text('Delete'),
54                              ),
55                            ],
56                            onSelect: (value) {
57                              if (value == 'update') {
58                                if (createdPost != null) {
59                                  showUpdatePostDialog(createdPost!);
60                                }
61                              } else if (value == 'delete') {
62                                if (createdPost != null) {
63                                  showDeleteConfirmationDialog(createdPost!.id);
64                                }
65                              }
66                            },
67                          ),
68                        ],
69                      ),
70                      const SizedBox(height: 12),
71                      Text(
72                        createdPost!.body,
73                        style: GoogleFonts.openSans(
74                          fontSize: 16,
75                          height: 1.5,
76                          color: Colors.blueGrey[900],
77                        ),
78                      ),
79                      const SizedBox(height: 12),
80                      Text('id: ${createdPost!.id}',
81                        style: GoogleFonts.openSans(
82                          fontSize: 16,
83                          height: 1.5,
84                          color: Colors.blueGrey[600],
85                        ),
86                      ),
87                      const SizedBox(height: 12),
88                      Text('userID: ${createdPost!.userId}',
89                        style: GoogleFonts.openSans(
90                          fontSize: 16,
91                          height: 1.5,
92                          color: Colors.blueGrey[600],
93                        ),
94                      ),
95                      const SizedBox(height: 16),
96                      Text(
97                        DateFormat("dd MMMM yyyy, HH:mm 'WIB'").format(createdPost!.createdAt),
98                        style: GoogleFonts.openSans(
99                          fontSize: 14,
100                         color: Colors.blueGrey[600],
101                         fontStyle: FontStyle.italic
102                       ),
103                     ),
104                   ),
105                 ),
106               ),
107             ),
108           ),
109         ),
110       ),
111       floatingActionButton: FloatingActionButton(
112         onPressed: showAddPostDialog,
113         backgroundColor: Colors.indigo[900],
114         shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(16)),
115         child: const Icon(Icons.add),
116       ),
117     );
118   }
119 }
120 }
121 }

```

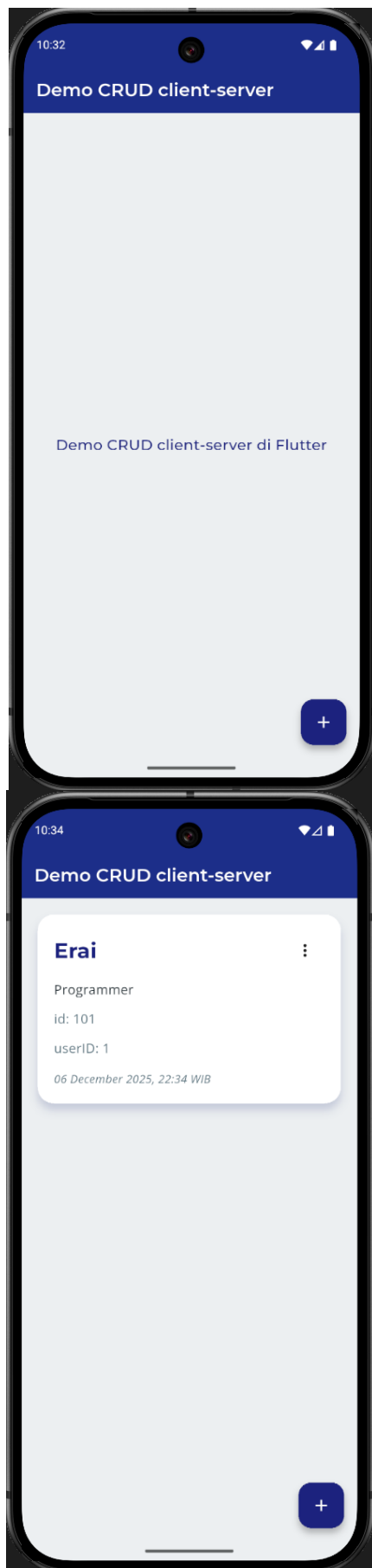
## **b. Penjelasan Kode Program**

Aplikasi Flutter ini menerapkan konsep pengelolaan data dinamis berbasis client–server dengan memanfaatkan API dari [jsonplaceholder.typicode.com](https://jsonplaceholder.typicode.com). Proses komunikasi data dilakukan menggunakan paket `http`, sehingga aplikasi dapat mengirim dan menerima data dari server secara langsung. Melalui konsep CRUD (Create, Update, Delete), pengguna dapat menambahkan, mengubah, dan menghapus data secara real time tanpa perlu me-restart aplikasi.

Method `createPost()` digunakan untuk mengirim data baru ke server melalui metode POST. Data yang dimasukkan melalui `AlertDialog` kemudian dikirim dalam format JSON, dan jika berhasil, hasil respon dari server disimpan ke dalam variabel state `createdPost`. Selanjutnya, method `updatePost()` berfungsi untuk memperbarui data menggunakan metode PUT, di mana pengguna dapat mengubah informasi yang sudah ada melalui dialog `update`. Sedangkan method `deletePost()` digunakan untuk menghapus data dengan metode DELETE, yang diawali dengan dialog konfirmasi agar tidak terjadi penghapusan secara tidak sengaja.

Variabel `createdPost` berperan sebagai penyimpan utama data yang ditampilkan pada layar aplikasi. Jika variabel ini bernilai null, maka aplikasi hanya menampilkan teks awal. Namun ketika data sudah tersedia, informasi post langsung ditampilkan dalam bentuk kartu. Sementara itu, widget `AlertDialog` berfungsi sebagai media interaksi pengguna dalam memasukkan data, memperbarui data, serta mengonfirmasi penghapusan. Peran `setState()` sangat penting karena bertugas memperbarui tampilan secara otomatis setiap kali terjadi perubahan data, baik saat data dibuat, diubah, maupun dihapus, sehingga tampilan aplikasi selalu menyesuaikan dengan kondisi terbaru.

Output :



#### **iv. Kesimpulan**

Berdasarkan hasil pembuatan dan pengujian aplikasi, dapat disimpulkan bahwa proses pengelolaan data dinamis berbasis client–server pada Flutter dapat berjalan dengan baik melalui penerapan konsep CRUD (Create, Update, Delete). Penggunaan metode `createPost()`, `updatePost()`, dan `deletePost()` mampu menghubungkan aplikasi dengan server secara efektif menggunakan paket `http`. Variabel state `createdPost` berperan penting dalam menyimpan dan menampilkan data, sedangkan `AlertDialog` mempermudah pengguna dalam memberikan input dan mengatur alur interaksi. Selain itu, penggunaan `setState()` menjadikan tampilan aplikasi dapat diperbarui secara otomatis setiap kali terjadi perubahan data, sehingga aplikasi menjadi lebih responsif, interaktif, dan efisien dalam pengelolaan informasi.