Flutter REST API Integration: Fetching & Displaying Data

Subtitle: Learn how to connect your Flutter app to REST APIs and display dynamic data in your UI.

Website Name: haas.dev
Website Link: https://dev-roast-app.vercel.app

Introduction

Most modern apps require **dynamic data from servers**. Flutter makes it easy to **fetch, parse, and display API data** using HTTP requests. This guide teaches practical API integration.

Step 1: Add HTTP Dependency

```
dependencies:
  http: ^1.1.0
```

Run:

```
flutter pub get
```

Import in Dart:

```
import 'package:http/http.dart' as http;
import 'dart:convert';
```

Step 2: Fetch Data from API

```
Future<List> fetchUsers() async {
  final response = await http.get(Uri.parse('https://jsonplaceholder.typicode.co

  if (response.statusCode == 200) {
    return jsonDecode(response.body);
  } else {
    throw Exception('Failed to load data');
  }
}
```

Exercise: Fetch a list of users and print their names in console.

Step 3: Display Data in a ListView

```
FutureBuilder<List>(
  future: fetchUsers(),
  builder: (context, snapshot) {
    if (snapshot.connectionState == ConnectionState.waiting) {
      return CircularProgressIndicator();
    } else if (snapshot.hasError) {
      return Text('Error: ${snapshot.error}');
    } else {
      final users = snapshot.data!;
      return ListView.builder(
        itemCount: users.length,
        itemBuilder: (context, index) {
          return ListTile(
            title: Text(users[index]['name']),
            subtitle: Text(users[index]['email']),
          );
        },
      );
    }
  },
)
```

Exercise: Display fetched user data in a scrollable list.

Step 4: Handle POST Requests

```
Future<void> addUser(String name, String email) async {
  final response = await http.post(
    Uri.parse('https://jsonplaceholder.typicode.com/users'),
    headers: {'Content-Type': 'application/json'},
    body: jsonEncode({'name': name, 'email': email}),
  );

  if (response.statusCode == 201) {
    print('User added');
  } else {
    throw Exception('Failed to add user');
```

```
  }
}
```

Exercise: Create a form to add a new user using POST request.

Step 5: Mini Project

Build a **Flutter API App**:

- Fetch data from JSONPlaceholder or any free API
- Display data in a `ListView` with proper loading/error handling
- Add a form to POST new data
- Bonus: Implement pull-to-refresh to reload API data

Key Takeaways

- REST APIs allow apps to work with **dynamic data from servers**
- `http` package provides **simple GET and POST methods**
- Proper error handling and UI updates are critical for **robust apps**
- API integration is essential for **real-world, scalable Flutter apps**

Skipping API integration makes your app **static and unable to interact with external data sources**, limiting functionality.

Visit **haas.dev** for more Flutter guides, tutorials, and complete project resources.

Website Name: haas.dev
Website Link: https://dev-roast-app.vercel.app
