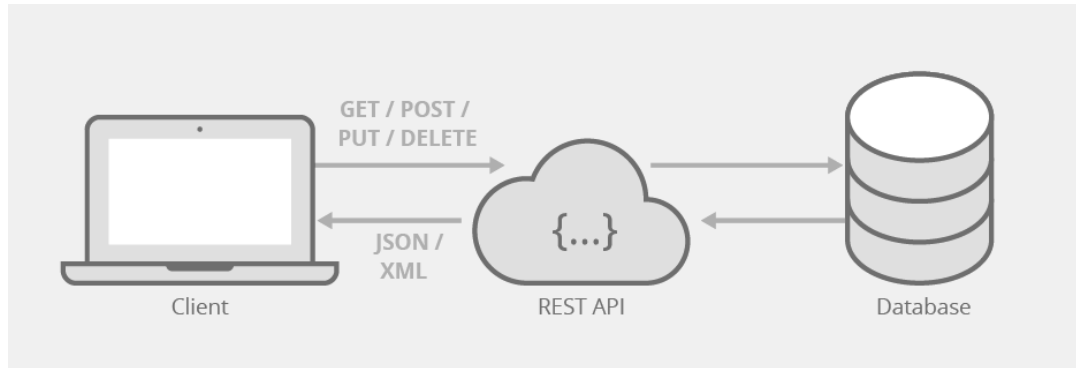


# Day 10

1. Testing API endpoints with Postman
2. Ensure Internet Permission in Manifest
3. Make GET and POST Requests
4. Restructure using MVC approach

# API requests



- GET Request to fetch data
- POST request to send data (specify params)
- UPDATE for updation and DELETE for deletion

# API endpoints

- Find a sample API
- Setup API as Postman Collection
- Make GET and POST requests

# Postman Collections

The screenshot displays the Postman REST client interface. At the top, there are three active request tabs: 'GET Get all employees', 'POST Add an employee', and 'GET Get an employee'. The current view is for the 'GET Get all employees' request. The URL bar shows 'https://dummy.restapiexample.com/api/v1/employees' with a 'Send' button to the right. Below the URL bar, there are tabs for 'Params', 'Authorization', 'Headers (6)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. The 'Params' tab is active, showing a table for 'Query Params' with columns for KEY, VALUE, and DESCRIPTION. The 'Body' tab is also visible, showing a status of '200 OK' and a response size of '1.9 KB'. The response body is displayed in a code editor with a 'Pretty' view selected, showing a JSON object with pagination and employee data.

Sample API / **Get all employees** Save

GET `https://dummy.restapiexample.com/api/v1/employees` Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	Bulk Edit
Key	Value	Description	

Body Cookies Headers (19) Test Results Status: 200 OK Time: 173 ms Size: 1.9 KB Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "page": 1,
3   "per_page": 6,
4   "total": 12,
5   "total_pages": 2,
6   "data": [
7     {
8       "id": 1,
9       "email": "george.bluth@reqres.in",
10      "first_name": "George",
11      "last_name": "Bluth",
12      "avatar": "https://reqres.in/img/faces/1-image.jpg"
```

# Restructure using MVC approach

- Create **Model class** for Entity (Employee)
- Add JSON **encode and decode methods**
- Add **Controller class** for API functions
- **FutureBuilder** widget to render data on screens

# Implementation

GET and POST API requests

- Model (**Employee class**)
- Controller (**API functions**)
- Render data using **FutureBuilder**

# Assignment

- Test API with Postman
- Implement **GET and POST Requests** and render data using FutureBuilder

# Reference Links

- **Generate Dart class from JSON**

[https://javiercbk.github.io/json\\_to\\_dart/](https://javiercbk.github.io/json_to_dart/)

- **Sample APIs for testing**

<https://reqres.in>

<https://jsonplaceholder.typicode.com>