Working with APIs in Flutter:

What is an API?

- An API is a set of definitions and protocols for building and integrating application software.
- In other words, if you want to interact with a computer or system to retrieve information or perform a function, an API helps you communicate what you want to that system so that it can understand and fulfil the request.
- You can think of an API as a mediator between the users or clients and the resources or web services they want to get it.
- It's also a way for an organization to share resource and information while maintaining security, controls, and authentication- determining who gets access to what.

Flutter provides **http** package to consume HTTP resources. **http** is a Future-based library and uses *await* and *async* features. It provides many high level methods and simplifies the development of REST based mobile applications.

What is REST API?

- Representational State Transfer.
- Also known as RESTful API.
- Software architectural style that was created to guide the design and development of the architecture for the **World Wide Web**.
- It facilitates caching components to reduce user-perceived latency, enforce security and encapsulation.

More:

- set of architectural constraints, not a protocol or a standard API, developers can implement REST in a variety of ways:
- When a client request is made via a RESTful API, it transfers a representation of the state of the resource to the request or endpoint.
- This information or representation is delivered in one of several formats via HTTP: JSON (JavaScript Object Notation), HTML, XLT, Python, PHP, or plain text.

- JSON is the most generally popular file format to use because despite its name, its languageonstic, as well as readable by both human and machines.
- **Something else to keep in mind**: Headers and parameters are also important in the HTTP methods of a RESTful API HTTP request, as they contain important identifier information as to the request's metadata, authorization, uniform-resource identifier (URI), caching, cookies and more.
- There are request headers and response headers, each with their own HTTP connection.

---

http package provides a high level class and http to do web requests.

- http class provides functionality to perform all types of HTTP requests.
- http methods accept a url, and additional information through Dart Map (post data, additional headers, etc.,). It requests the server and collects the response back in async/await pattern.

Some of the core methods are as follows −

- **read** − Request the specified url through GET method and return back the response as Future<String>
- **get** − Request the specified url through GET method and return back the response as Future<Response>. Response is a class holding the response information.
- **post** − Request the specified url through POST method by posting the supplied data and return back the response as Future<Response>
- **put** − Request the specified url through PUT method and return back the response as Future <Response>
- **head** − Request the specified url through HEAD method and return back the response as Future<Response>
- **delete** − Request the specified url through DELETE method and return back the response as Future<Response>

http also provides a more standard HTTP client class, client. client supports persistent connection. It will be useful when a lot of request to be made to a particular server. It needs to be closed properly using close method. Otherwise, it is similar to http class.