REST

Created	@May 5, 2022 8:00 PM
Class	
Type	
Materials	
Reviewed	☐

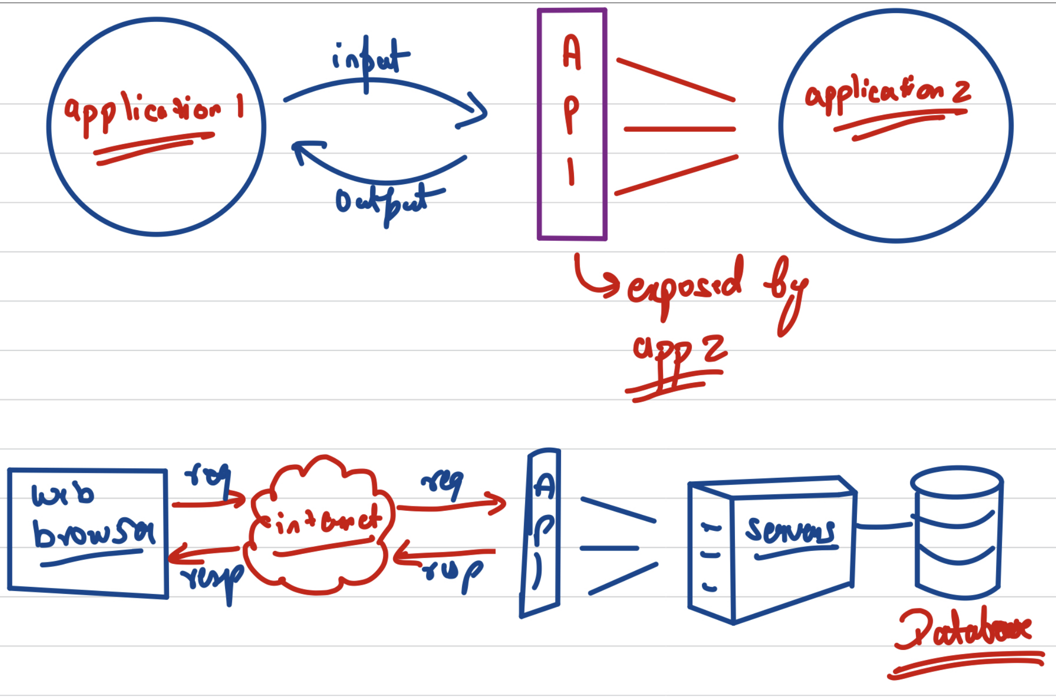
What is an API ?

The term API stands for Application Programming Interface. So what exactly it is ?

Let’s take an example of a Bank. If you want to deposit money in the bank, then you just don’t go inside the bank locker and put the money yourself. That now how it is done.

There is a cashier who is present at the front desk, who receives the money on behalf of the bank from you, counts it and return you a slip for it. For submitting the money, you have to also fill a form and give it to the cashier. Now how cashier handles the money to deposit in the bank is not your headache. All we are concerned about is, to handover the money to cashier by submitting a form and receiving a slip for it.

So in this case, the cashier is acting as an API, i.e. an interface between us the user and the bank. The cashier is the exposed entity that we communicate to and get out money submitted, we don’t directly communicate with the bank internals.



So the same thing happens in case applications running in different systems, in order to communicate in between, we have to expose the functionalities through API's that the other application can consume.

Even on web, a lot of web applications/mobile applications follow the same architecture, that they don't directly consume data from databases or services for server but instead they connect with the API's exposed by the server and then do the corresponding tasks.

So the API's expose the signature (format) of how to provide input to the api, how to call the api, and what response to expect.

REST

REST stands for Representational State Transfer.

In a nutshell, REST is a set of conventions or recommendations on how applications on web, should communicate between each other. It is often regarded as `language of internet`. It is more or less like an architectural style, which proposes approach for communication between apps.

And an API created by following the REST conventions are called as `REST API`.

Note:

Apart from REST also there are other conventions like SOAP and GRPC. Some application follow these also.

Principles of REST

Client Server Architecture

REST assumes every call / communication exists between a client (someone expecting some information) and a server (someone providing some information).

`Client` ——————————> `Server`

Method of data transfer

So as humans generally when we communicate we use languages like English, Hindi, German etc to communicate our thoughts. And our thoughts are just form of data. Similarly, Every time when there will be exchange of data between client and server. REST assumes the data transfer to happen with `JSON`.

JSON

JSON stands for Javascript Object Notation. JSON looks more or less like a JS object only.

```
{
  message: 'Successfully fetched user emails',
  success: true,
  code: 200,
  data: [
    'a@b.com',
    'c@d.com'
  ]
}
```

So in REST we send and receive data in the form of `JSON`.

REST is stateless

The server will never store the data about the client after completion of request-response.

Routes are resource oriented

In REST the routes are resource focussed. So, In rest the primary data representation is called resource.

Example:

- http://www.relevel.com/courses
- http://www.relevel.com/user
- http://www.relevel.com/admin/login

So here, course , user, admin all these are resources. In a nutshell, resource means an entity for rest.

So let's assume we are making a blog application, then the REST routes will look like this:

- http://www.blog.com/user
- http://www.blog.com/blog
- http://www.blog.com/blog
- http://www.blog.com/blog/comments

Request Methods:

In http, we have different request methods. By saying request method, we define the type of the request. The differences which we have written below are based on REST conventions. In GET and POST there are more difference which we will discuss later. HTTP defines a set of **request methods** to indicate the desired action to be performed for a given resource.

- GET: this request should only retrieve data from a resource or read the data.

- POST: this method submits data for a resource and causes changes in state of resource, so suitable for creating a resource
- PUT: this method completely updates the resource
- PATCH: this method partially updates the resource
- DELETE: this method deletes a resource

CRUD Routes

CRUD (CREATE READ UPDATE DELETE)

What the hell are crud routes ? So CRUD in a nutshell means the operations of creating a resource, reading a resource, updating a resource and deleting a resource.

Example:

- http://www.twitter.com/tweet , GET → This will read all the tweets
- http://www.twitter.com/tweet/1 , GET → This will read the first tweet
- http://www.twitter.com/tweet/2, GET → This will read the second tweet
- http://www.twitter.com/tweet**/1, DELETE → This will delete the first tweet**

Let’s make few CRUD routes for a resource TWEET

- CREATE A TWEET: http://www.twitter.com/tweet , POST
- READ ALL TWEETS: http://www.twitter.com/tweet, GET
- READ A PARTICULAR TWEET: http://www.twitter.com/tweet/tweet_id , GET
- DELETE A PARTICULAR TWEET: http://www.twitter.com/tweet/tweet_id , DELETE
- UPDATE A PARTICULAR TWEET WITH ALL PARAMETERS: http://www.twitter.com/tweet/tweet_id , PUT
- UPDATE A TWEET WITH FEW PARAMETERS: http://www.twitter.com/tweet/tweet_id , PATCH

URL: https://www.booking.com/hotel , POST : Create new hotel

URL: http://www.zomato.com/restaurant/2 , PATCH : Partially Update the restaurant with id 2

URL: http://www.zomato.com/restaurant/5/food , POST

URL: http://www.zomato.com/user/login , GET → this will be used for getting user login detail

Sending Data in HTTP Requests

So, in order to send data on a route we have 3 ways:

- Request params/URL Params
- Request Body
- Query Params

Query Params :

URL: https://www.fliokart.com/products?min_price=300&max_price=30000&rating=5 , GET

URL Params:

URL: http://www.flipkart.com/products/:category_name

URL: http://www.flipkart.com/products/electronics → electronics is the url param

URL: http://www.flipkart.com/categories/2 → this 2 is the URL param

Request body

Every http request has a lot of data values, like request code, request method, url etc.

One more such parameter is a JSON object called as body.

In any request we can send data in request body also.

REST generally prefers,

POST to be handled by Request body

GET , DELETE to be handled by URL params

PUT , PATCH : URL params and Request body both, so in the URL params we will get what resource to be updated and the request body will keep the data about what values of the resource mentioned to be updated

For any extra filters we can use query params.