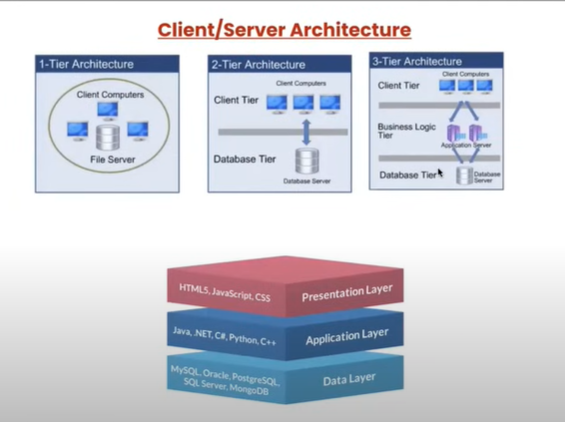
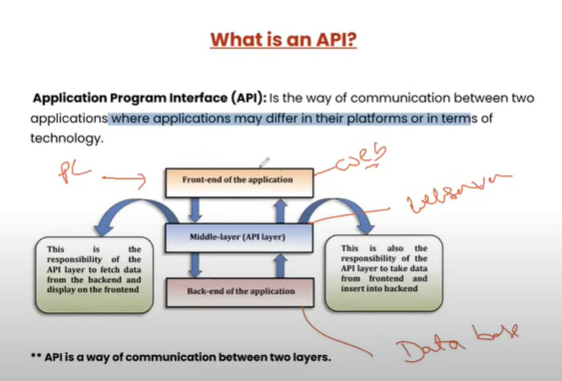
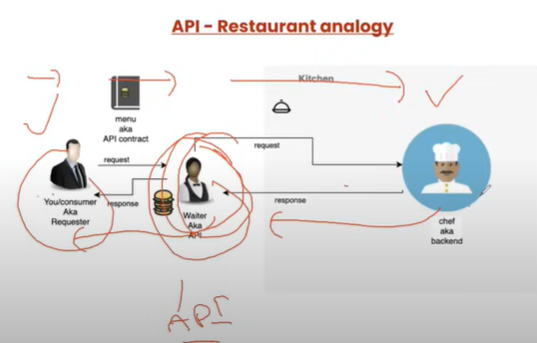
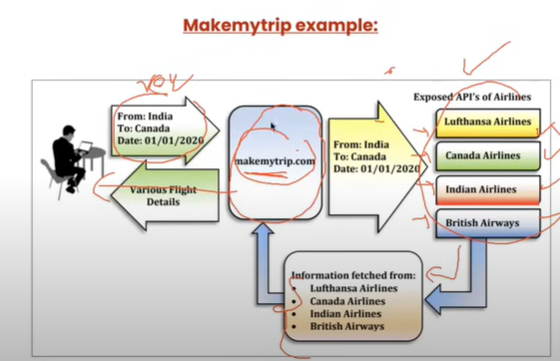
**API Testing: Application Program Interface**

It is way of communication between 2 applications where application may differ in their platforms or in terms of technology.









**Types of API:**

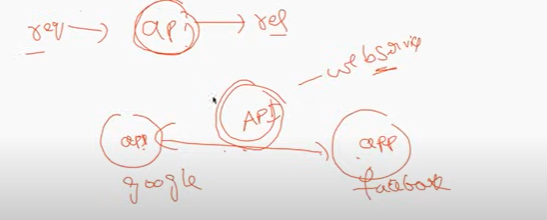
There re 2 types of API’s.

1. Simple Object Access Protocol (SOAP).
2. REST (Representation State Transfer).

Both are the web services.

SOAP => used by old applications.

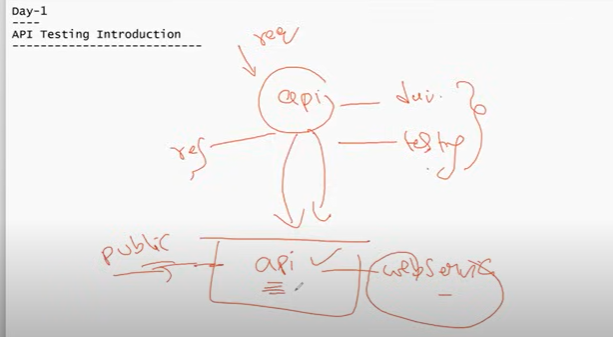
REST => used by new/current applications.



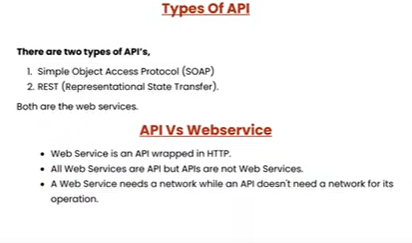
All Web Services are API. But all API are not Web Service.

When API is put in internet then it is called as Web Service.

For development & testing we use Api. And once its available to public over internet its called Webservice.



**Important: API Vs Webservice**



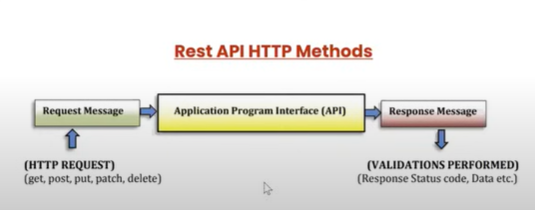
**Rest API methods:** http request

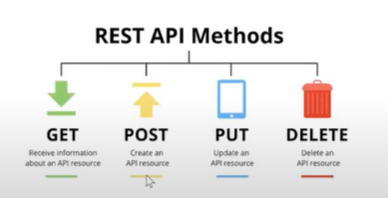
Get: we get data from the server.

Post: we are sending data to server to store.

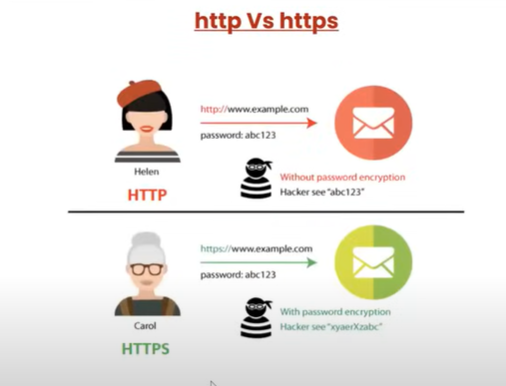
Put: we can create/edit/update data in the server.

Delete: we delete the data from server.

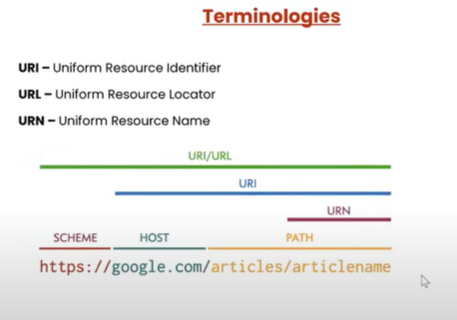




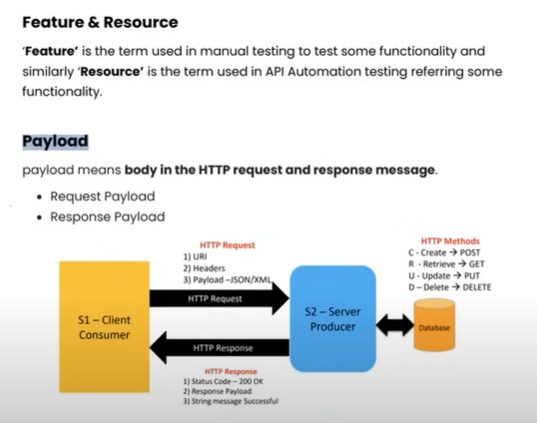
HTTP Vs HTTPS



Terminology:

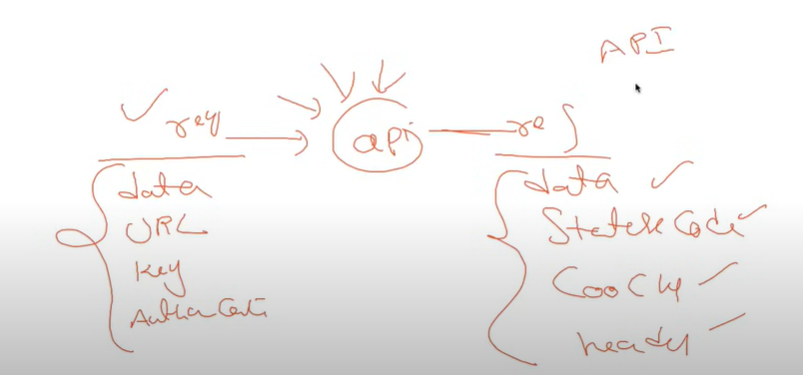


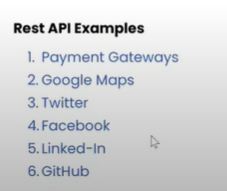
Payload:



Ex: <https://reqres.in/>

Ex Get: <https://reqres.in/api/users?page=2> => <https://reqres.in/api/users?page=2>





Postman – API testing

Desktop/web

Workspace: area where we maintain files and saved.

Workspace – create workspace, rename, delete.

Creating collection = contains number of folders and http requests. Create, rename, delete, run the collection.

We can create any number of collections under workspace.

Request ----🡪 API -------🡪 Response

http Request:

Get => retrieve the resource from database.

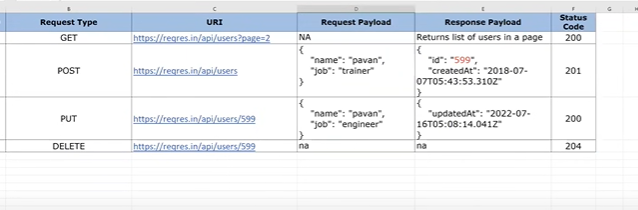
Post => create resource on database.

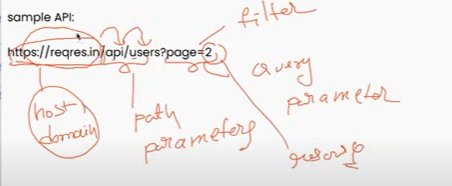
Put => update existing resource on database.

Patch => update partial details of resource.

Delete => delete existing resource from database.

Sample APIs: <https://reqres.in/>





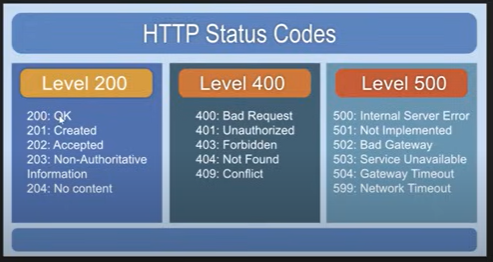
Validations:

1. Status code
2. Time
3. Size data
4. Response body(json/xml)
5. Cookies
6. Headers

HTTP Status code:

3 levels

1. 200
2. 400
3. 500



**How to create own API: JSON**

Create our own API’s

Steps:

1. NodeJS
2. Npm-node package manager

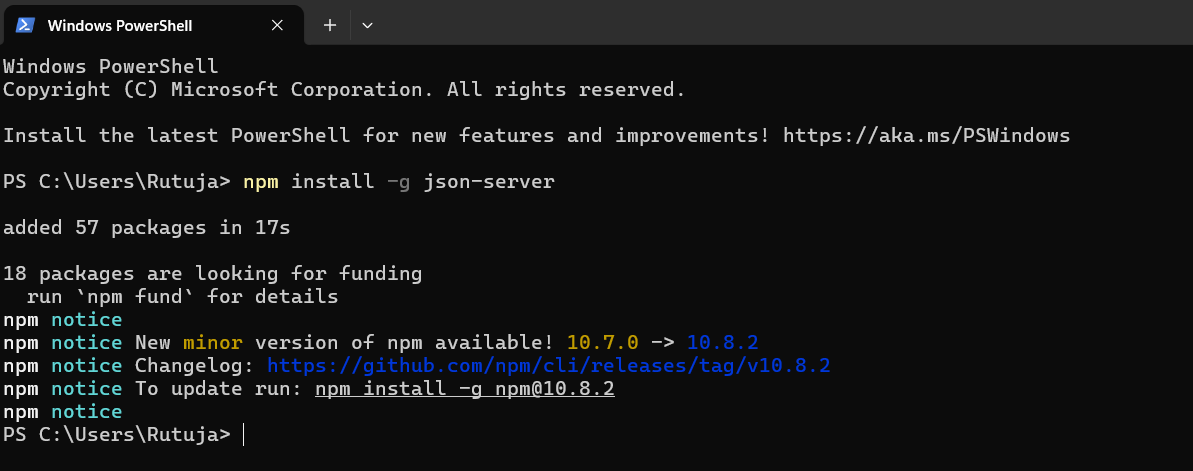
node - -version

npm --version

1. Json-server
2. Install json-server:

Run below command in cmd/terminal

npm install -g json-server



1. Create students.json file with following data.

{

    "students":[

        {

            "id": 1,

            "name": "John",

            "location": "India",

            "phone": "1234567890",

            "courses": [

                "Java",

                "Selenium"

            ]

        },

        {

            "id": 2,

            "name": "Kim",

            "location": "US",

            "phone": "2345678901",

            "courses": [

                "Python",

                "Appium"

            ]

        },

        {

            "id": 3,

            "name": "Smith",

            "location": "Canada",

            "phone": "3456789012",

            "courses": [

                "C#",

                "RestAPI"

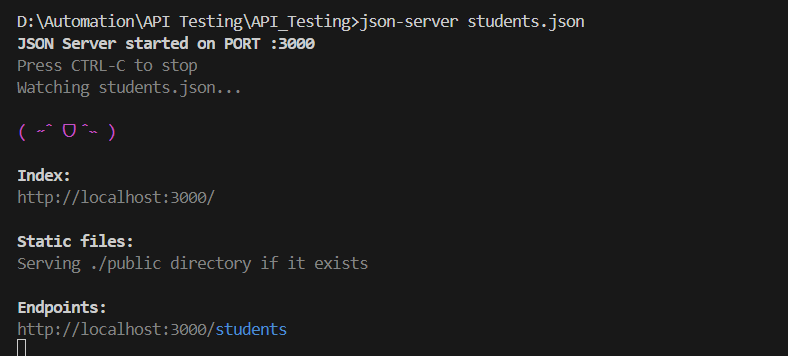
            ]

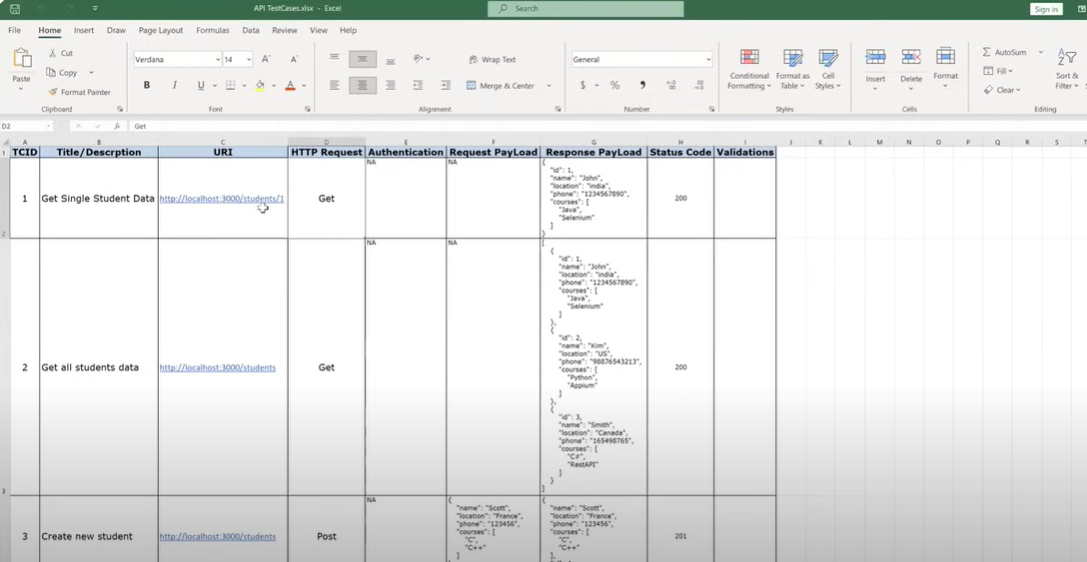
        }

]

}

1. Run using command ‘json-server students.json’



1. <http://localhost:3000/students> 
2. 

**JSON: Java Script Object Notation**

Key value pair

Key: value

**JSON Data Types:**

1. Number
2. String
3. Boolean
4. Null
5. Object
6. Array

{

“name”: “John”

}

**Data Types:**

abcd