API Testing → sdet ① (Part 1)

⇒ API | Webservices
⇒ Postman UI
⇒ Types of HTTP Requests

→ MT of API can be done by using POSTMAN.

What is web services ?

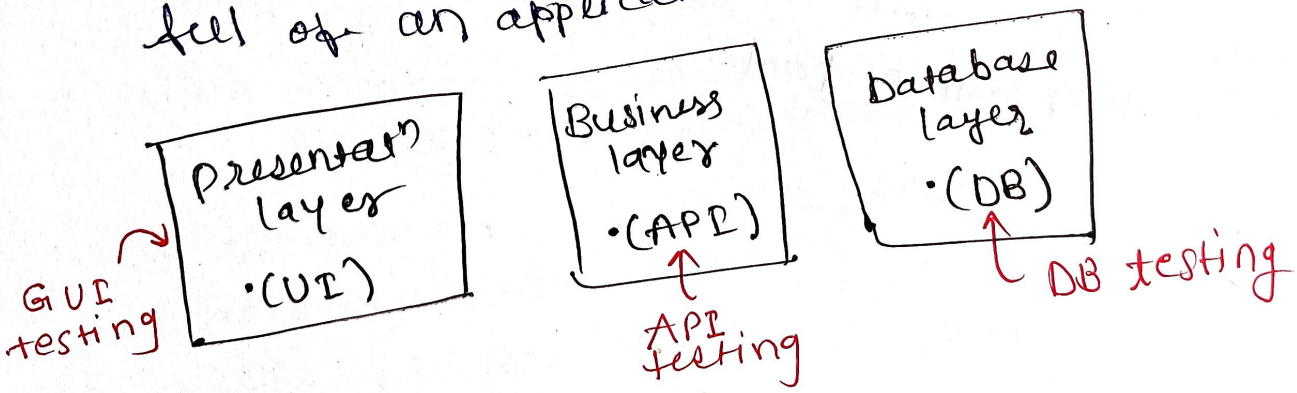
→ Web service - service available over the web.
- enable communication bet^n applications over the web.
- provides a standard protocol / format for communication.

Why we use it ?

- Platform independent communication - using web services two diff applications can talk to each other and exchange data / information.

What is API testing ?

- It is entirely different from GUI Testing and mainly concentrates on the business logic layer of the s/w architecture.
The testing won't concentrate on the look and feel of an applicatn.



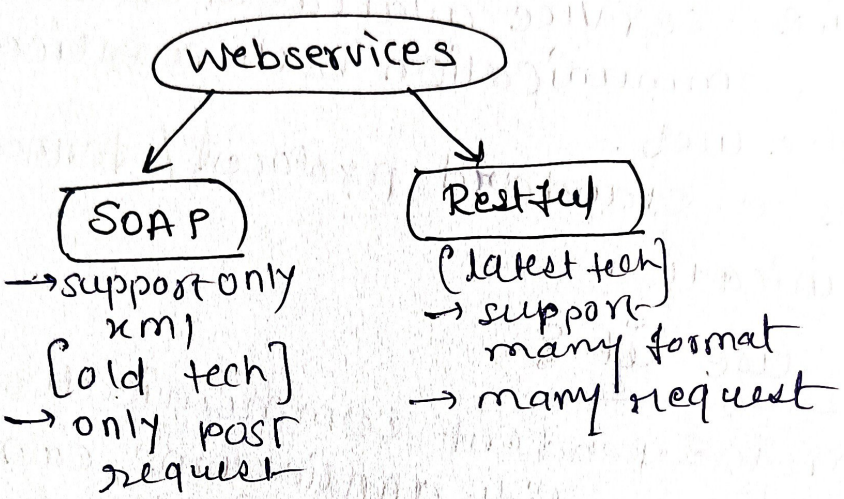
All Webservices are API
but All API are not Webservices

Types of web services

there are mainly two types of web services.

① SOAP Web service. [Simple Object Access Protocol]

② RESTful web services. [Representation State Transfer]



4 types of Request [HTTP request perform on API's]

① GET – select

② POST – Create

③ PUT – Update

④ Delete – delete

⇒ postmal tool – used to test API manually
It is open source tool.

Sdet-②

→ Install postman → ① download from google
exe file → double click - install

② Sign up (gmail)

③ whatever task doing in postman it will save in workspace

④ create new workspace [New project]

⑤

for test Rest API ⇒

We require ① Endpoint (url)

② What type of request (HTTP method type) (Get, Post..)
for post ⇒ required some data

③ URI

④ Body ⇒ data pass to post request

⑤ Response ⇒ for each request there will be response

⑥ failure Response ⇒ if any failure

⑦ Status code

⑧ comments

⇒ ① create collection and add new request in it.

collection → It consist of multiple request
* every request considered as one test case

New collection → give name (Rest API's) → create.

URI (Uniform Resource ~~Locator~~ Identifier)

URI ⇒ https://reqres.in/api/users?page=2

https://reqres.in ---> Domain } must contain in URI
/api/users ---> path parameter }

page=2 ---> query paramter (filter parameter)

URL maybe domain, path parameter or query parameter.

① GET => (only url pass)
Send request then some response will get in body.
then validate ① data ② Headers ③ Status code
④ How much time in getting data.

→ it will get you the data from the server
→ whatever request you are doing, you are getting some response from server.

POST → it will go create new resource in server.
eg. you want to create new record in server then send POST request also you need to pass request body.
pass request body (Request payload)

PUT → update already data created by using POST request update that record
it is also required, Request payload.

Delete

every request will get you response in JSON format / XML / HTML / Text ... etc.

⇒ Get, POST, PUT & Delete requests (Part 2)

⇒ Adding Tests / Validations

search
↳⇒ JSON path finder chrome extention ⇒ add in chrome

⇒ for locate particular value in JSON

⇒ we will get path of json data

```
var response = JSON.parse(responseBody);
tests["verify firstname"] = response.data.first_name="janet";
```

⇒ We can send multiple set of data in one request

sdet-③

In Response, there need to check Headers.

Header → verify headers are proper not.

Cookies →

Body →

Test Results →

Postman support JS (part 2)

① var response = JSON.parse(responseBody);

tests["page No"] = response.page == 2;

(description can write anything) (keyword)

② tests["validating status code"] = responseCode.code == 200;

③ tests["validating response body"] = responseBody.has("2");

→ one validation fail whole test will fail.

→ for some APIs, you will verify the responses against the db, & for some others, it is better to verify the responses against other APIs.

JSON path finder Chrome extension

part 3

→ Data Driven Testing using JSON & CSV files
→ How To Run Collections
→ How to generate HTML Reports
→ Exporting & Importing Collections.

* Data Driven Testing

→ execute same request with multiple set of data. ⇒ prepare this data in ~~xml~~ csv / json file (prepare test data)
→ post request → go and create new record in server with data body (data of the file)

Practical ⇒ new collection → save with "data driven testing"
⇒ inside that create request → type url. → body section → ~~text~~ raw → choose text format (eg. json, xml...)
→ type data in it. → save with name

(*) test

① tests["check status code"] = responseCode.code = 200
② tests["check status in response] = responsebody.has("success")
③ var response = JSON.parse(responseBody);

test(" ") = response.status == "success"
test["check status in the response in exact postn"] = response.status == "success"

Headers → content type (xml etc)
we can validate Headers - write js in tests ← javascript

```
pm.test("check content-Type header", function()
  {
    pm.response.to.be.header("content-Type", ". ");
  });
```

⇒ test & data will same only file will change
iteration = 2 means 2 data set execute होगा।

S4

Send request with multiple set of data
- Prepare data in xml/json file
Can perform data driven testing on csv/json format

(*) Request Body/Request Payload: => abc.csv

```
{
    "name": "john"
    "salary": "100000",
    "age": "35"
}
```

Data format:- csv, json

(*) (1) csv → tests => in postman → Request Body format

```
{
  "name": "{{name}}",
  "salary": "{{salary}}",
  "age": "{{age}}"
}
```

capture data from csv/json file

Run by using test Runner.
Runner → select file
→ Same url can execute with diff type of data.

New file → New Runner window => collection runner window => Select Collection => Select file

convert csv ↔ JSON

⇒ one collection is contain many request
so we can run whole collection suit.

collection ⇒ suit
request ⇒ test case

We gave 1 request for 4 Validation test ⇒ it will run since we have 4 iteration for 4 test ⇒ 4×4 = 16

⇒ We can export results in JSON format.

(1) How to run single collection (which includes multiple requests)
→ go to Runner → Select collection requests → Run

(2) ⇒ Run collection through command prompt
→ Install tool Node.js (npm)
→ Install newman
→ Export collection & then run from cmd prompt.
↓
click on 3dots - export

your workspace is not on your local m/c, it is present in cloud, so need to export collection in json format

→ download nodejs for windows (3rd party tool)
→ refer document then cross check ⇒ cmd prompt ⇒ `node -v`

② npm (node package manager)

`npm -V` → execute this command on cmd prompt
(component of nodejs)

→ install newman plugin – for reporting purpose

cmd → `npm install -g newman`
↳ install newman on top of nodejs

→ export & Run

How many ways we can run collection through command prompt ?

method 1 → syntax: `newman run <<exported collect^n file.json>>`

open cmd in that folder where your json file
then fire cmd ⇒ eg: `necoman run xyz.json` ↵
execute all request in collection.

* Generate HTML Report ⇒

cmd ⇒ `newman run xyz.json -r html` # it will generate Html report

method 2

install plugin for ~~this~~ ⇒ `npm install newman-reporter-html`
generate HTML

go to give path & open newman folder & open file.
you can see all report.

method 3 executing collection remotely (without exporting & importing)

(3dots) Share collection → (select collection) get public link → after getting that link run through cmd prompt "paste link instead of json file"

" you have to share your collection & get the URL"
then can execute collect^n remotely & also create html report using cmd.

(55)

→ Fake API creation & Requests
→ Postman Variables
→ Workflows

(part 4)

How to create own API's (Fake API's)

① Install NodeJS
download link :- https://nodejs.org/en/download/
after installation give below command:
node --version

② npm (comes with node.js)
check version ⇒ npm --version

③ Install json-server (JSON server component)
npm install -g json-server // install json server in local mic

④ create info.json file with sample data (json format)
Loc → C:\Users\admin\info.json # popular some data on json server

(locate your cmd prompt)

⑤ Run the below command to make your API's up & running.
json-server info.json # start ~~AppServer~~ API
get url then hit that url

postman variables

① collection variables
② Environment variables / Global variables

Collection variables ⇒

Right click on -- collection - Edit - Variables - Define variables (key, value) - Update Usage : {{key}}

(or) postman.setGlobalVariable ("key", "value");
postman.clearGlobalVariable ("key");

Environment variables / Global variables

Manage Environments - Global - key, value -- save Usage : {{key}}

Workflows ⇒
postman.setNextRequest ("Request Name")
postman.setNextRequest ("null") # when no user

⇒ If same url is required in every request then replace it with variable.

⇒ So, create one collection variable, & have this data in that collection variable.

Can use variable within collection.

Specify this created variable in url ⇒ {{variable}}

② ~~Environment~~

`postman.setGlobalVariable("url","....")`

pre request Scripts in postman

→ before sending ~~request~~ request this Script will be executed.

→ define variable in scripts:-

→ run time variable will be created

② Environment ⇒

create this variable at postman tool level.

We can use environment variable in multiple collection.

Practical ⇒ postman setting → manage environment → click on Add → QA Environment → create variable.

→ Single Environment holds multiple variables.

Workflows ⇒ control, execution order based on requirements. (flow of)

by writing Script in prerequisit Script

HOW to Chain API Requests (Part-5)

⇒ Create Get Requests (Listings all users)
⇒ create Global variable "username" is xyz at collection level
⇒ write test script in Request1 to update "username"

```
jsonData = JSON.parse(responseBody)
value = jsonData.data[0].first_name
pm.globals.set("username", value);
```

⇒ Create put Request (update user by name)
⇒ parametrize username variable in to 2nd request (PUT)

```
"name": "{{username}}"
```

⇒ Run Collection which contains both the requests.
⇒ Authorizations

Chain ⇒ send API request
Get some response

API request1 --> response
API request2 --> response.

O/P of API request1 will become I/P for API request2
this is called API request chaining.

open ⇒ JSON path finder

```
jsonData = JSON.parse(responseBody) # parse whole response into partical variable
```

then extract data from response to variable
for extract exact value, we need to use
JSON path finder → plugin
Capture response & paste on "JSON path finder"
⇒ easily get Node from JSON file
Specify Node ⇒ data.submit

[we need to extract exact value from JSON]

we can set this variable/value as global Variable.
We can set global variable by using snippet (which is already given)

⇒ sending ~~on~~ data / variable / response from one request to another

Steps ⇒ ~~e~~ extract variable from JSON set it as global

Script ⇒

```
jsonData = JSON.parse(responseBody) # extract data from JSON
value = jsonData.data[0].first_name # extract data from JSON
pm.globals.set("username", value); # set that variable as global
```

Create global variable at collection level & set extracted variable to that global variable through script.

अगर तुझे value dynamically use करनी है कहीं पर भी या फिर दूसरे request में तो फिर use करें {{ }} this syntax में curly brace में variable से value आएगी

Req(1) ⇒ extract variable

Req(2) ⇒ use this variable (global variable)

can use this global variable in multiple request.

API Request 1 → Response

i/p to 2nd request

API Request 2 → Response

i/p to 3rd request

API Request 3 → Response

⇒ Postman Scripting / Response Validations
⇒ Script Execution Order
⇒ Single / Multipart files upload & download

There are 2 types of scripts → ① Pre-request
② Test script

We can specify this scripts in different levels ⇒
① collection → folder → Requests
Order of Execution ⇒ Pre-request will execute first for collection → folder → Request level then actual request will execute then Test-script will execute in above given sequence.

Postman scripting using Javascript & chai BDD

→ Internally postman used JS.

Postman scripting ⇒
① You can set global variable & pass it run time to request
Pre-request script ⇒ pm.globals.set("userid", 2);
① can verify Status code.

Validate Response

#multiline Assertion (one funct^n can have multiple Assertion)

```
pm.test("test description", function(){
var responseJson = pm.response.json(); # parse response
```

see Json path finder

```
pm.expect(responseJson.data.id).to.eql(2); # verify response id with our expected id
pm.expect(responseJson.data.email).to.eql("---");
```

one assertion is fail then all assertion will fail.

→ sending

Handling responses that don't parse

* verify something (text/value) is present in the body/response

```
pm.test("Body contains string", function(){
    pm.expect(pm.response.text()).to.include("from respons")
});
```

verify status code from list/one of set

```
pm.test("Successful status code", function(){
    pm.expect(pm.response.code).to.be.oneOf([200, 201])
}); # Status code being one of a set
```

Testing headers

```
pm.test("Content-Type present", function(){
    pm.response.to.have.header("Content-Type");
}); # verify it is type of header or not
```

verify value of header type

```
pm.test("Content-Type value", function() {
pm.expect(pm.response.headers.get("Content-Type)).to.eql("application/json")
});
```

verify cookies

cookies present or not

```
pm.test("Cookie is present", function() {
    pm.expect(pm.cookies.has("cookie name")).to.be.true;
```

pm → postman; expect → expecting; pm.cookies → from All cookie; has → is our; "cookie name" → of cookie

```
});
```

verify value of cookie → right or wrong

```
pm.test("cookie value check", function(){
  pm.expect(pm.cookies.get("name of cookie")).to.eql("--");
});
```

Part 6-②

Verify response time of API

```
pm.test("Response Time less than 20ms", function(){
pm.expect(pm.response.responseTime).to.be.below(20);
});
```

learning.postman.com

* File upload and download

⇒ Export & import collections (part 7)

⇒ Documenting Collections

⇒ Run options

⇒ Postman Integration with Jenkins

Export & import collections

① Export collection as file & Import as file.

② Export collection to a folder & import from folder

③ Share collection → Get public link then import as link.

④ import as Raw Text → Curl command → Import

Documenting Collections

Run options ⇒

① within the postman using Runner window

② Run Collection in the command prompt (CLI)

- ~~uploaded~~ Export collection file in Json format

or

- Shared collection link

Install Node.js – NPM – npm install newman

③ Run through Jenkins

can run through jenkins by using Links or collection file

⇒ install jenkins
generic jenkins (war file)

⇒ Jenkin will start on tomcat server

`Java -jar jenkins.war` # Run through cmd prompt

Authorizations

whenever we require key or token to pass request.

Create New collection → create some request (by simple copy past)

① Basic Authorizat^n → just we need to pass username & password.

② ~~Basic Auth~~ ⇒ https: postman-echo.com/basic-auth
username : postman
password : password

② ~~API key~~ APP key

③ Bearer Token / OAuth 2.0 :

learning.postman.com

Test cases for API Testing

1. verify that HTTP status code is 200 OK
2. verify that the response is valid JSON.
3. verify that response data has important keys like name, private, fullname, description, url
4. verify that request-/response has a content type "Content-Type" key
5. verify that the response has id generated & it is not null.
6. verify the request does not have errors.
7. verify that without basic Auth, request will works (as this is open call) 200 OK.
8. verify that time taken by Request is less than 2000 ms
9. verify that HTTP status code is 201 created
10. verify that the response is valid JSON.
11. verify that response is with body.
12. verify the name of repo created is not private.
13. verify that full-name, url, for, description key is present.

Test cases → ① check status code
② verify value in data

1xx (Informational) → The request is received & continues to be processed.

2xx (Successful) → The request is successfully received, understood, & accepted

3xx (Redirection) → further action needs to be taken to complete the request.

4xx (Client Error) → The request contains the wrong ~~status~~ syntax or cannot be fulfilled

5xx (Server Error) → The server fails to fulfil an apparently valid request.

⇒ 5 imp things should be considered for ~~API~~ tested that should be always in list to be tested

① complete suit start from data, data should be created at the same time uniquely, it should not be previously save data, so we cant check the data properly then.
data should be generated uniquely with unique Id.
Id should be generated uniquely at real time.
randomly generated data.

⇒ ~~verificat° of data~~, when we are sending request response we are verifying creating data first, so all creat° API will check first verification of data, verificat° of action whatever we are performing like post, patch, delete
if we delete resource ~~then~~ verify then we should try to get resource again then verify that resource is exist or not. verify that error msg is correct or not.

① verify functionality of API

② can run multiple threads on the same API to check that if you are creating one user & how much time it is taking
- we can parameterized
perform some performance testing on API's

③ Schema validation

What is API ? (pramod)

Basically a collectn of functions & procedures which allows us to communicate two applications or libraries.

web API

& OAP - Simple Object Access Protocol
RPC - Remote Procedure Call
REST - Representational State Transfer

* REST API ⇒

→ In Simple terms REST API is nothing but a design pattern for Web API.

→ The REST architecture style describes six Constraints

scrollTest.com

REST Constrains ⇒ Uniform Interface, stateless, Cacheable, Client-server, Layered System, Code on demand

Diff. betn SOAP & REST

Difference	SOAP	REST
Style	Protocol	Architectural style
Function	Function driven - transfer structured info	Data-driven- access a resource for data
Data format	only uses XML	permits many data formats, including plain text, HTML, XML & JSON
Security	Support WS-Security & SSL	Supports SSL & HTTPS
Bandwidth	Requires more resources & bandwidth	Requires fewer resources & its lightweight
Data cache	can not be cached	can be cached
Payload Handling	Has a strict communican contract & needs knowledge of everything before any interact	Needs no knowledge of the API
ACID Compilance	Has built-in ACID compilance to reduce anomalies	Lacks ACID compilance

HTTP status code

reqres.in ⇒ dummy API

learn JSON from W3School

Headers ⇒ meta information (key-value pair)

imp ⇒ API & right click → copy as (curl → for bash)

then go to postman → Raw text ⇒ paste here

⇒ Continue

API Testing → task → API Bugs & test cases for API testing

→ Resource → All task → http://scrolltest.com/api/task
Thread → https://scrolltest.co/30days

→ Learn → Common Bugs
① Wrong error codes
② Missing keys
③ Empty post or Update are not handled properly.

Test cases for API Testing

① Validate the keys with the Min & Max range of APIs (eg. maximum & minimum length)

② keys verification. if we have JSON, XML APIs we should verify its that all the keys are coming. (if request failed)

③ have a test cases to do XML, JSON schema validation.
- mentioned min no. of keys.
- can mentioned the properties, type of response, required & non required properties.

④ verify the parse the Response data.
JSON is invalid / null

⑤ verify the JSON Schema Validation, verify the field type, verify the mandatory fields.

⑥ verify valid response headers & Negative test cases response

⑦ verify that how the API's error codes handled.

⑧ verify the response HTTP status code.

⑨ Valid Response payload

chaining Request verificatn.

Verification of APIs with Data Parameters.

(12) End to End CRUD flows
(13) Database Integrity Test cases
(14) file upload testcases

Postaman Basics ⇒

Exposeting code, managing cookies in postman, share Collection, monitor collection, publish docs, collection Runner, data driven testing

⇒ We can add monitor → 3 dot of collection ⇒ monitor ⇒ create monitor ⇒ Run
we can run it daily.

How que ⇒ API monitoring

Publish docs ⇒ 3 dot ⇒ documentation ⇒ publish

Collection Runner ⇒ Run ⇒ Html Run

Data Driven Testing ⇒

CRUD with local Dummy server

CRUD ⇒ ① Install Nodejs.org (by default npm will install)
② Install JSON server
③ npm i -g json-server@v0.12.2
④ create db.json (in one folder)
⑤ Run Collection → json-server --watch db.json

① install nodejs.org ⇒ check in cmd prompt ⇒ node -v
② npm → use to install diff. packages

* JSON server will create fake REST API's

⇒ hit get call in json ⇒ you will get info. according their id

⇒ POST ⇒ create one data in server
Header ⇒ by it is metadata go with request, which will automatically add.

⇒ Patch ⇒ Partially update

⇒ get → get data from server
POST → create data in server
Patch → update some data (Partially update)
PUT → update whole data
Del → delete data

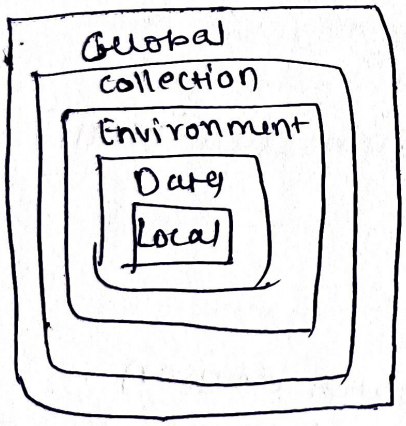
Variables

If you want to hit the url after sending request through tests :-

```
pm.sendRequest("https://postman-echo.com/get", function (err, response)
{
  console.log(response.json()); });
```

Variable Scopes ⇒

Postman supports the following variable scopes :-
i) Global ii) Collection iii) Environment iv) Data v) Local



① Global ⇒ (key value)
Tests ⇒ pm.globals.set("gvar", "pramod"); #create
console.log(pm.globals.get("gvar")); # retrive

② collection ⇒ 3 button ⇒ edit ⇒ tests ⇒ set a collectn variable

③ Environment variable ⇒
var Id = pm.environment.get("id");
pm.environment.set("id", 5);
console.log(pm.environment.get("id"));

JSON schema validation

we need to check data type of variable.

→ use ajv

schema → is a blueprint of a response

search for → JSON schema validator, json schema creator

Settings ⇒ jsonschema.net/home

jsonschemavalidator.net

① validate → type

→ validate required fields

JSON Schema : JSON schema is a document or structure or format that can be used to validate & format json documents. You can use JSON schema for formatting validating the formatting of JSON msg the syntax the data types & to validate the entire structure & content of a JSON message so here JSON schema is written in JSON.

ee JSON Schema are JSON Documents that describes other JSON Documents"

JSON Schema → defines the structure of a JSON message
can use to validate a JSON message.

a json message and in case of API's that have a json request & response JSON schema can be used to validate the API request and response so, if we take an eg. of API requests when we run our API requests or we do any process or action on the request we get the output

↓ API request
validate format
process
output
validate format

JSON schema

validate format syntax or data types of the API request, we can catch it right at the stage of validating the format even before we do any processing or action on the request & even before send out the response

response we can again validate the format of the response or the O/P. & this validatn of the format is done by Json Schema in case of Json messages of API request.

give Json data & it will convert into Json Schema

Ramod

Postman
CRUD
Variables
Upload data
workflow
History tab
JSON schema validat
data driven testing
monitor
mocking API
graphQL
Types of defects

Raghav

postman
webservice
API request
CRUD
own API
collections
variables
scripting & Testing
debugging & trouble shooting

Data driven testing

what are the types of Bugs & defects that API testing detects?

- Duplicate or missing functionality
- Improper messaging
- Error handling mechanism is incompatible.
- Multi threaded issues
- Security, performance & security issues.
- Reliability issues

Test Scenarios

① Verify that HTTP

Interview Question on API Testing

1. What is API?
2. what are HTTP status code's
3. what is path & query parameter?
4. diff. betn SOAP & REST
5.

API Testing Interview Question

1. What is API?
2. What is web services?
3. what is Rest API?
4. Diff. betn Rest API and Soap based API?
5. what is Rest Client?
6. Diff. HTTP methods (GET | POST | PUT | DELETE)
7. What is POSTMAN?
8. what is URI & URL?
9. what is Header?
10. What is Response?
11.

API Testing - Naveen

API → Application programming interface

API webservices → communicating over the NIW over the HTTP call

through HTTP protocol, we are calling these APIs and over the NIW.

these type of web services/API are called web services.

→ API's contain some sort of information.

→ To access API, need some sort ~~information~~ of authentication or key or token key

→ sometimes it can have username/password

→ Request → JSON/XML

→ Method → HTTP method - CRUD

→ GET calls

Important factors while calling API

① Authentication - session/token/ID

② Username/password

③ Request - JSON/XML

④ HTTP Method - CRUD

⑤ GET calls (Retrieve)

⇒ rest api is provide some security also over the NIW also

⇒ If you are passing XML file or JSON file different types of files passing that is called your rest API or soap API

⇒ Java code/eclipse are calling those API.

⇒ same thing happening over the NIW ⇒ in the form of HTTP protocol then that becomes web services.

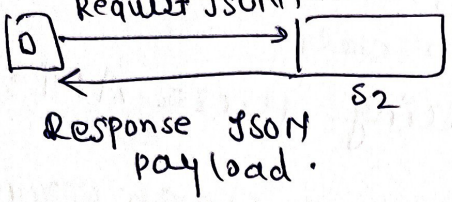
⇒ getting API over HTTP or over the NIW then it called/becomes webservices

2 types of API ⇒ ① Soap based ⇒ simple object access protocol

② Rest based

front end testing → selenium / QTP
Back end testing → Postman

JSON / XML => data collector, lightwt, standard format to communicate betn 2 diff. interface Platform / System.



If you want to interact / use my system, you have to send, JSON file
request payload in the form of JSON

API => use when two systems are interacting with each other.

when API =>
↳ interact with diff. system or independent system or access diff features then use API.
[through jar files (locally)]

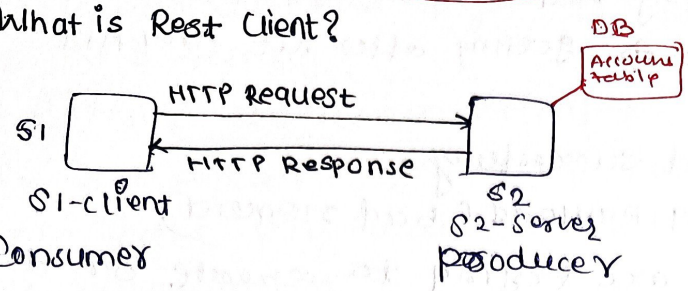
=> communication happen through API webservices.

Topics covered

1. REST API concepts
2. Difference betn API & services
3. REST API concepts
4. Web Services
5. SOAP vs REST APIs
6. REST

Naveen

What is Rest Client?



C - create - POST (creat data)
R - Retrieve - GET
U - Update - PUT
D - Delete - DELETE

① HTTP Request → 1. URI
2. Headers
3. Payload (JSON/XML)

② HTTP Response → 1. Status - 200 OK
2. Response Payload
3. String message - successful

(1) Create an Account → POST call → used to create new entity
>> JSON/XML (Passing info in the form of JSON) Headers
Account ID = 1
Account name = "Tom"
Address = "New-York"
Account type = "Savings"

(2). whatever I have created, I want to fetch
Get an account by ID/Name
>> URI
>> Path parameter
>> Query Parameter
>> Headers

Post → create entity
PUT → create + update

(3) Update an account - PUT call (internally call POST)
create → entity + update
Account id = 1
Account name = Tom
Address = New York
AccountType = Current

(4) Delete an account (Delete) → delete account
JSON
Account id = 100
Account name = "Tom"
Address
AccountType

→ from each and every request, we get one response.
Status Response Code ⇒ getting after hit any API

① 200 OK
② 201 → created successfully
③ 400 → missing payload (bad request)
④ 404 → You are trying to create or delete entity (Not found)
page is not found on server
⑤ 500 → server is down (server internal error)
⑥ 401 → Authentication error
⑦ 403 → forbidden

HTTP Method - GET / POST / PUT / DELETE

How to access this API ⇒ by using URI & URL
end point ⇒ URL
what is URI ⇒ URL (endpoint URL) + API URL (or service (backend service)) = URI

what is Postman? ⇒ tool for API testing,
other tools ⇒ SOAP UI, Advance Rest Client, JMeter, Browser.

? ⇒ query parameter → used to filter your results.
/ ⇒ path parameter → specific information

POST → create new user
JSON → key value pair

⑥ What all challenges are included under API testing?

① API documentation →

To test an API you need lot of things like parameters, payload. can not start testing without any documentatn.
without knowing what data we have what data we have to send it to server, & how response being received so without knowing all these details its hard to perform API testing.

② Access DB ⇒

you hit API request & you get a response back so how do you validate that response what you got is expected or not. so for validatn you have to compare response values what API gives you with
this happens when API's are from third party Company where their database details are not shared. so it will challenge.

③ Authorizatn overhead ⇒ when you talk to API & get response in general it will not be that easy because there is some Authorizatn & authentications involved so you have to authenticate yourself so that API understands you authenticatn request & then only it process your request.
if API request for oAth Authenticatn then you should know the tokens being generated & which service give you the tokens which will help you to give that info. to API sending you request. so there is some overhead is involved when its API expects you to authorize right.

2Que ⇒ What is the difference between PUT & POST methods?

POST request → creating a new object on the server

Send an API with HTTP post request that means ~~we are~~ it is going to create some new object or new data on the server

PUT request → updating something which is already on the server.

update the object in server with new value.

3 what are commonly used HTTP methods?

send data using that HTTP method

In general when we talk about REST API so there are four different HTTP methods which are commonly used

① GET ⇒ It enables you to retrieve data from a server.

② POST ⇒ It enables you to add data to an existing file or resource in a server.

③ PUT ⇒ It lets you replace an existing file or resource in a server

④ Delete ⇒ it let you delete data from a server.

4 List out few Authentication techniques used in API's?

authenticate yourself first that you are right person to share the data

① authenticate based on cookie authentication/session authentication

② Basic Authentication (simple username & password

③ Digest Authentication

④ OAuth

5 why API testing is determined as the most suitable form for automation testing?

because it is lighter ui testing & its more stable than UI testing

⑥ what is Rest API?

REST stands for Representational State Transfer. It is a set of functions helping developers in performing requests & receive responses. Interaction is made through HTTP protocol in REST API.

⑦ what exactly needs to verify in API Testing?

In API testing, we send a request to API with the known data & then analysis the response.

① We will verify the accuracy of the data
② Will see the HTTP status code
③ We will see the response time
④ Error codes in case API returns any errors.
⑤ Authorization would be check.
⑥ Non functional testing such as performance testing, security testing.

before ? ⇒ path parameter
after ? ⇒ query parameter

⑧ what are the core components of an HTTP request?

① HTTP request methods like:- PUT, POST, DELETE
② Base Uniform Resource Identifier (URI)
③ Resources and parameters
④ Request Header, which carries metaData (as a key-value pairs) for the HTTP Request msg.
⑤ Response Body, which indicates the message content or resource representation.

payload ⇒ request body

API testing allows the communication between s/w systems.
API testing works on backend also known as backend testing.

Restful web services uses the HTTP protocol. They use the HTTP protocol as a medium of communication betn the client & the server.

SOAP webservices?
SOAP → Simple object Access Protocol. It is an xml based messaging protocol. It helps in exchanging information among computers.

How do we represent a resource in REST?
using HTTP Methods.

GET → allows read only allows

POST is super set of all other HTTP methods except GET

payload ⇒ It is request body. that is secured JIP data which is sent to API to process the request. payload is generally represented in JSON format in Rest APIs

* query parameter constructed at the end of API
? ⇒ ↑
/ ⇒ path parameter

Rest Assured ⇒ Java library which can automate Rest API.

How would we define API details in Rest Assured Automat?
we shall define all the request details & send it to server in GIVEN, WHEN, THEN methods.

what is serialization and Deserialization in Rest Assured?

Serialization in Rest Assured context is a process of converting Java object into Request body (payload)
Rest assured also supports deserialization by converting Response body back to Java object.

List out few common json paising techniques used in Rest Assured Automation?

Json path
Deserialization of json using pojo classes

multipart method → send attachements to API using Rest Assured Test.

Different status code ⇒

Template for API test cases

TCID | ENDPOINT | HTTP method type | Status Code | URI | Body (Payload) | Response | failure response

comments | validation

TCI	Title	Request URL	HTTP method	Request Body	Auth keys	Response
Validations	Bugs					

Swagger =>

1. install NodeJs
2. git clone to particular project

Explain the API testing approach?

① Write appropriate test cases for the APIs & use testing techniques like BVA, equivalence class, etc for verifying the functionality.

② Verify the calls of the combination of two or more value-added parameters.

③ Define the scope and basic functionality of API program.

④ Define the accurate input parameter.

⑤ Test case execution & comparison of the results with expected results.

⑥ Determining API behaviour under cond^n like the connection with files etc.

What is API? Application Programming Interface

- It helps in communication, bet^n different SW systems.
- It facilitates data exchange between systems and data exchange located in different remote place.
- API is a collection of functions which can be executed by another SW program.
- API act as an abstraction.

API's Work ⇒

- The general workflow of API is that it takes a request, process it which might involve data validation, database interaction, data processing and then the resultant of this is sent back to the source.
- APIs provide an abstraction to the internal business logic as they are not exposed to the world.

- API acts as a Abstraction.

What is API Testing?

API testing is a type of integration testing used to test API to validate

- API testing is a type of SW testing that involves testing API's directly.
- API is a part of integrating to check whether the API meets expectations in terms of functionality, reliability, performance and security of application.
- In API testing, our primary focus is on Business logic layer of the SW architecture.

What are the types of API testing?

Unit testing
functional testing
load testing - for testing performance under load.
Runtime / Error detection
Security testing → how authorized ↳ passing headers, authentication, username, pass, session id
UI testing
penetration testing - security point of view
Fuzz testing
validation testing - interoperability - Integration

What are the protocol used in API testing?

HTTP
REST
SOAP
UDDI

What are the tools used for API testing?

tools used for API testing are

parasoft SOAtest
Postman
Alert Site API monitoring

What are the Advantages of API testing?

① **Test for core functionality :-**

API testing helps in core functionality by giving direct access to the application, without user interface.
The core functionality of the application will be tested before GUI tests.
This will help to detect minor issues which can become bigger during GUI testing.

② **Time effective :-**

- API testing is less time consuming than GUI testing.
- API test requires less code for testing functionalities so, it can provide better and faster test coverage compare to GUI test automation.

This will reduce the cost for the testing project.

③ **Language Independent :-**

In API testing data is exchange using XML or JSON. these transfer mode are completely language independent, which allows users to select any code language or when adopting automation test service for the project.

④ **Easy Integration with GUI :-**

API can be easily integrated with GUI.

What are the Advantages of API testing?

① Test for core functionality :-

API testing helps in core functionality by giving direct access to the application, without user interface.
The core functionality of the application will be tested before GUI tests.
This will help to detect minor issues which can become bigger during GUI testing.

② Time effective :-

- API testing is less time consuming than GUI testing.
- API test requires less code for testing functionalities so, it can provide better and faster test coverage compare to GUI test automation.
This will reduce the cost for the testing project.

③ Language Independent :-

In API testing data is exchange using XML or JSON. these transfer mode are completely language independent, which allows users to select any code language o when adopting automation test service for the project.

④ Easy Integration with GUI :-

API can be easily integrated with GUI.

What exactly needs to verify in API testing?

① We will verify the accuracy of the data.
② will see the HTTP Status code of resultant API
③ Will see the response time of API
④ Error codes in case API returns any errors.
⑤ Authorization would be check. Schema validation
⑥ Non functional testing such as PT, ST
⑦ data accuracy of the actual response with expected response.
⑧ verification of the API whether it is updating any data structure.
⑨ Based on I/P cond^n, returned values from the APIs are checked.
⑩ verify if the API does not return anything.
⑪ Verificat^n of the API whether it triggers some other event or calls another API.

How do you test the API security?

for testing the security of API during API testing, we need to validate 2 things:-

① Authentication ⇒ whether the identity of the end user is correct.
② Authorization ⇒ whether the user has access to the resource.

What are the major challenges face during API testing?

The first & foremost challenge is selecting an appropriate parameter and then its combination.

* Proper parameter selection
* Proper parameter combination
* Call sequencing – proper sequencing of call is required
* Proper knowledge of Output verification and Validation
* A major challenge is providing I/P values which are very difficult because GUI is not available.
* Knowing which API needs to be called in what sequence
* Knowing what are the proper I/P values that needs to be provided to the API I/P.
* updating schema

What are the types of bug that can be found during API testing?

API testing help us to find many types bugs which are =>

Stress
Security
Duplicate or missing functionality
Reliability
Unused flags
performance
Incompatible error handling
Improper errors / Non implemented errors
multithreaded issues

How to do API testing?

1. select suit in which you went to add test cases
2. Choose the test development mode.
   develope required test cases for testing the APIs by making
3. clearly define scope & functionality of API.
4. configure applicatn control parameter, test cond^n & validatn method
5. verify test cases by passing the I/P parameters.
6. configure test cond^n & method validatn
   compare the results of diff test cases based on their expected behaviour.
7. execute API test / perform API test
8. analyze / review test reports
   ① filter & sequence API test cases.

Define Test Data ?

Test Data is the i/p data to perform test cases.
This data can be prepared either manually or by using different tools
for eg. to test login functionality of an applicatn, the i/p data will be username & password which constitutes test data.

Define test coverage ? provide fair idea about what the testers need to cover in their test cases

It is amount of testing performed by making use of test cases. It can be either functional testing or non functional testing.

What is API framework ?

1. It is a platform for developing s/w applicatn.
2. It is foundation on which s/w developer can build applications for a specific platform
3. It can include predefined classes & functions that can be used to process i/p, manage h/w devices & interact with system s/w.
4. It is defined by configuration file which consists the list of all API's that is required to be activated for a particular program run.

What is API test environment ?

It is quite complex method where the configurat of server & db is done as per the requirement of the s/w applicatn
API is checked for its proper functioning after installatn

What is API documentation ?

① A good documentat^n is most for any foundation.

② API documentation serves as quick reference for accessing library or working within a program.

③ when we use any such documents, it must consists of proper plan, content source, proper layout, information related to each funct^n.

④ there are various documentation tools like Doxygen → for .net code, JavaDoc → for java code, Swagger UI etc.
Sapper Dox

API documentat^n template ⇒ Swagger, Slate, API blueprint, RestDOC, webservice API specificat^n

How to document each funct^n

① Description ⇒ small description of what a function does
Syntax ⇒ syntax about the parameter of the code, sequence in which they occur, required and optional elements etc.
Parameters ⇒ functions parameters
Error messages ⇒ Syntax of Error messages
example code ⇒ small snippet of code
Related links ⇒ Related functions.

What are the difference between API & web services

API	Web Service
① APIs don't need a NW for operation	① web services always need a NW for operations.
② All APIs are not web services	② All web services are APIs
③ API is a set of protocol & definitions which allow one application to interact with another application.	③ A web service is a way for two machines to interact with each other over a NW.
④ API can be communicated through SOAP, REST, XML-RPC : and CURL calls as well	④ web services can be communicated through SOAP, REST, AND RPC

Restful web services:
① SOAP web services
② Restful Web Services.

① SOAP => Simple object Access Protocol is a XML based method which is used in web Services.

→ It is helps in exchanging information among computers
→ It is platform and language independent.

Restful Web Services ⇒

→ To implement the concept of REST architecture HTTP method is used.

Restful Web Services defines URI and also provides resource representation like JSON and set of HTTP method.

Resources in REST ⇒

Rest architecture treats any content as resource, which can be text files, HTML pages, images, videos or dynamic business information.

REST server gives the functionality to access the resources and modifies them.

⇒ Rest uses different representation to define the resources like text, JSON and XML.

The most popular representation of resources is JSON and XML

⇒ Restful Web services uses the HTTP protocol. They use the HTTP protocol as a medium of communication betn client & server.

⇒ Restful web services are REST architecture based Web Services.

Restful Web Services use HTTP as a communication betn the server & the client.

Rest API ⇒

① It is a set of function helps the developers performing requests when the response is receiving.

Rest is standard for API creation.

⇒ Rest (Representational State transfer) API is defined as set of functions that helps a developer in sending requests & receiving responses.

⇒ In this protocol, the interactn is always made through an HTTP protocol.

SOAP API

① SOAP stands as Simple Object Access Protocol.

② SOAP is a protocol

③ SOAP can work with XML format. In SOAP all the data passed in xml format.

REST API

① REST stands as Representational State Transfer.

② REST is an architectural Pattern.

③ Rest permits diff. data format such as plain text, HTML, XML, JSON etc. But the most preferred format for transferring data is in JSON.

API => It's SW/program/interface helps to communicate small funct" interact & help to get O/P.

eg. Google Maps API, Twitter API, Youtube API.

API & web services

- All web services are API but not all API's are web services.
- A web service uses only three styles of use:- SOAP, REST & XML-RPC for communicat" whereas API may be exposed to multiple ways.

A web service always needs a n/w to operate while API's don't need a n/w for operat"

Core engine is API

2 systems interact with each other.

Limits of API usage?

many API have a certain limit set up by provider

Every API has their own adv, limitat" based on our business requirement/need we can go with google API or other APIs.

we use some architectural styles for creating a web API's

1. HTTP for client server communicat"
2. XML/JSON as formatting language
3. Simple URL as the address for the service
4. Stateless communicat"

API testing → functional
→ Non functional

Reliable - ① functional → all +ve -ve
Performance - ② Non functional ⇒ performance security

API services
webservices
Backend services

XML or JSON are communicatn media to transfer data

protocols ⇒ JMS (messaging protocol), REST, HTTP, UDDI and SOAP

~~Test~~ Environment ⇒
- It's not easy task.
- the test environment of API is a bit complex & requires configuratn of db & server, depending on the SW requirements.

No GUI is available in this test form

→ deploy code using Jenkins

When the installation process is complete, API is verified for the proper operatn. Throughout the process, the API called from the original environment is set up with different parameters to study the test results.

API errors

1. missing module errors - access some module resour
2. parameter validation error
3. Documentation errors - whatever acceptance criteria, what will be request parameters if not properly documented
4. error handling / warning

→ alias is not created, some of the functionality is missing.

what Bugs ⇒

1. missing or duplicate functionality
2. error condn gracefully not handled properly [proper error handled]
3. Stress & load issues.
   API is reliable in terms of load testing
   code is not getting crash
4. Reliability -
   if give multiple / simultaneous / concurrent request - then how it is handling
5. Security - imp only authenticated API's only can access
6. Not implemented errors
7. Inconsistent error handling
8. performance
9. multithreaded issue
10. Improper errors

API documentation

API documentation should be proper, complete, accurate technical writing giving instructions on how to effectively use & integrate with an API.
It has all the information needed to work with the API, & helps you answers all the API testing questions with details on functions, classes, return types, arguments & also eg. & tutorials.

templates ⇒ (1) Swagger
(2) postman collections
(3) mirodot
(4) Slate
(5) Hatelo G
(6) API blueprint
(7) RestDOC

document ⇒ (1) source of the content - url, url.
(2) document plan or sketch
(3) delivery layout - flowchart
(4) info needed for every funct° in the document
(5) Automatic document creat° programs.

Principles of an API test design §

- Setup – create objects, start services, initialize data etc
- Execution – steps to apply API or the scenarios, including logging
- Verification – Oracles to evaluate the result of the execution
- Reporting – pass, failed, or blocked
- Clean up – Pre-test state, post state data

what must be check?

During the API testing process, a request is raised to the API with the known data.

if client hitting API from client, so we are hitting HTTP request to the server. then we will receive some response from the server.

what kind of data you are passing it is json/xml basic string, query parameter, path parameters.

correct/incorrect data the send request

Rest API – get/post-
data
headers

what if not passing headers what will be the response?

① accuracy of data

② Schema validation ⇒

data schema

⇒ type of data is very imp

prepare schema according to their requirement

data come in chronological order

③ Authorization checks →

always pass token

Integration testing functional testing, correct/incorrect parameter, path/query parameter, +ve/-ve scenarios, i/p parameters, search parameter, wrong headers, correct headers, get/post/delete, create with multiple test design techniques.

- comparing the test result with the expected result

API testing tool → postman, SoapUI pro,

Automation → Rest Assured, HTTP client

API testing	Unit testing
→ done by QA	→ done by development
→ Black box testing	→ White box testing
→ access full functionality of the system	→ verify each unit in isolation performs as expected or not test small small module
→ run after the build is ready	→ each of the code modules must be ensured to pass the unit test before being built by developers.

→ parameter select^{n} ⇒ Sometimes we don't have proper documentation we should know all functionality of API. What kind of API's are there weather it is path or query parameter. what's diff combination are there what is call sequencing of API.

Output verification or validation → lot of discuss with developer

provide i/p to API because of no GUI is there

Rest ⇒ Representational State transfer.

→ used to Http protocol to send a request & get the response for the integral purpose

Http methods like put, get, post, update, delete (CRUD operation)

→ API — Multiple System

→ client accessing with Authentication to access resources with the help URI with authentication with passing some JSON to server.

→ Get - retrieve
Post - create
retrieve -
Update - update
delete - delete

Restful web services
- ① SOAP — XML based (simple object access protocol) methods to expose web services.
- ② Rest

webservices developed in the REST style are referred to as Restful web services.

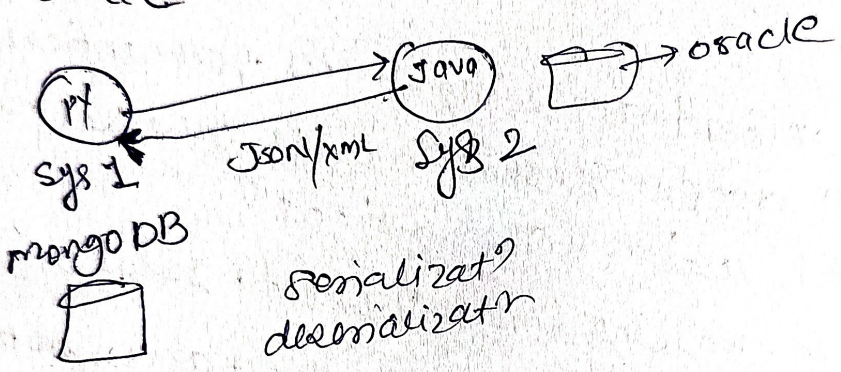
These web services use HTTP methods to implement the concept of REST architecture.

A Restful web services usually defines URI, uniform resource identifier a service, provide resource representation like JSON & set of HTTP methods.

Resources in REST ⇒

Rest architecture treats any content as a resource, which can either be text files, HTML pages, images, videos or dynamic business information. Rest server gives access to resources & modifies them, where each resource is identified by URIs / global IDs

Rest uses diff representatn to define a resource like text, JSON & XML.
XML & JSON are popular representatn of resource because it is lightwt.



WSR → raw logs
→ analysed logs