Rest API Automation

Soap Vs Rest

Soap is a standard because it is always bind to WSDL definitions

Language,platform and transport independent

Soap uses XML for all messages

Soap has built in error handling – ws security

Every request or response is embedded within soap envelope making it heavy weight even for simple data transfers.

User should have prior understanding on how WSDL works before learning SOAP

Webservices.

Rest is architecture - it uses simple http methods to create resources

Language and platform independent, but depends on http transport

Rest works with any formats xml, json, csv or plain text

Rest can be easily used in presence of firewalls and it can be made secure by

different external authentication mechanisms.

Rest is easy to learn and no expensive tools required.

Rest API’s are faster.

<soap:envelope>

<orderid>

Rest means representational state transfer.

It depends on stateless, client-server, cacheable communications protocol – http protocol.

It is an architectural style – It uses simple http methods to make calls unlike

Other complex mechanisms like Corba,RPC, Soap which follow certain standards.

URL is subset of URI:

URI – url+ resource

A url can always be URI i but not all URI’s are URL’s

Rest API URI examples:

<http://talentscreen.api.com/candidates>

<http://talentscreen.api.com/subjects>

endpoint- talentscreen.api.com

resources- candidates,subjects

resource – noun

actions performed on resources- verbs

http://api.citibank.com/accounts/openAccount

HttpMethods:

CRUD operations- most frequently used as part of Rest API’s

Post – creating a resource

Using order details provided by customer an order is created with certain order id

Get- retrieving the data-

Send order id and get order details

Put- update the resource

Update the exisiting order with new order details by using previous order id

Delete- delete the resource

Delete order using order id

Options- gives all the supported operations

Head- gives all header details

Safe: get ,head

The particular method action do not affect the existing resource as it is just retrieval.

Idempotent methods: get same response for N >0 request

Get,head,put,delete

http://amazon.com/orderid/387689789

http://amazon.com/orderid/387689789

Get vs Post:

In general- get is used for retrieval, post is to create the resource

Some of the secure retrievals like userLogin(get) are done through

Post methods because the secure should not be exposed in url.

Get do not have request body and all req params are sent through url.

Length of url is limited – so we cannot have unlimited request data in get request

But post request data is unlimited.

Put vs post:

Post is to create and put to update the resource but when there is no existing resource put acts as post and creates the resource.

Post can act as put if the resource already exists

Put can act as post if resource do not exists

Http status codes:

100 - informational

200 – success

201- resource created- post

200- ok- get

204- no content

300- redirecting

400-client side error

400- bad request

401- unauthorised

403- client forbidden

404- resource not found

500- server side error

500- internal server error

503-service unavaialble

Authentication:

BasicAuthentication – username and pwd is entered and token is generated based on uname and pwd.

OAuth 1

OAuth 2

Api key

What do you test in an API:

Functional testing:

Status code

Imp header info

Response message

Create Request data – valid data/invalid data/blank/boundary values

-Mandatory/optional params

Authentications test

Endpoint test

Resources within endpoint- mostly crud operations

Actual response: element present, element has exact value, data count,

Data order

After post request – execute the sql query and make sure the data is posted properly in db.

Security testing

Performational testing

What you require to start RestAPI testing:

API specs/specification docs- details about endpoint, resources, parameters,

Functionality details of resources.

Sample request/response – from dev team or in specs

Tools – SoapUI , Postman- chrome,AdvancedRestClient- chrome, RestClient-firefox

Automation Framework-

Java- HttpClient, RestAssured

JS- chakram

Frequently coordinate with Dev team to resolve queries.

Request /response has header and payload(body)

Payload –

{

orderId:3682837

}

We can set request data in post using HttpEntity interface-implementation classes.

HttpEntity implemenataions frequently used are:

StringEntity – string is input

UrlEncodedFormEntity – uses BasicNameValuePair class values in list object

401- unauthorized

200-ok

201-resource created

400-bad req

BDD – behaviour driven development

Eg: login scenario

given user navigates to gmail.com

when user enters valid uname and pwd

then user should be successfully

This is generally written in cucumber- feature files using easy English codes- gherkin language

feature files are implemented using step definition files and

step definitions call webdriver methods or api methods.

Basics of cucumber:

http://toolsqa.com/cucumber/cucumber-tutorial/

RestAssured : Rest api automation fw which internally uses httpclient builder but

It has more easy to use methods than HttpClient.

It has BDD style methods which makes code more readable and easy to understand.

It has easy json parsing methods which can be used with help of

Hamcrest matchers.

https://github.com/rest-assured/rest-assured/wiki/GettingStarted

<https://github.com/rest-assured/rest-assured/tree/master/examples>

<https://github.com/rest-assured/rest-assured/wiki/Usage>

AuthenticationTest:

Pass valid auth details : correct response

Invalid auth details: u should not response

When req data is complex- they try to create java POJO entity class for reqdata.

Using Jackson jars we can convert a JSON object to JAVA pojo - Marshalling

JavaPojo to JSON Object – Unmarshalling

RestAssured internally it uses Jackson when we pass java pojos

httpClient needs external conversion:

https://www.mkyong.com/java/how-to-convert-java-object-to-from-json-jackson/