SOAPUI and SOAPUI PRO

[ananthgithub@gmail.com](mailto:ananthgithub@gmail.com)

Handgreat99

Data source driven test

Reporting’s are good

Coding is less

In that webservice you have 500 req and you need to fire 300 req more than once with different sets of data

You want to generate good reports .

This is possible in open source as well but manually. But to automate it is hard to automate

REST PROTOCOL

PRO – protocol

Open source also covers REST.

Download SOAPUI PRO

REST – Representational state transfer – Rest is not a protocol .It is a architecture

SOAP – is a protocol

SOAP-Dinner – Ministers – you will be having protocol – spoon and fork – SOAP

Order is fixed

Fixed tag <soap:envelop>

REST – Dinner with friends – you decide the order , in rest there is no fixed tag

Architecture

<http://rest.elkstein.org/>

Sample Soap protocol

<?xml version="1.0"?>

<soap:Envelope

xmlns:soap="http://www.w3.org/2001/12/soap-envelope"

soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:body pb="http://www.acme.com/phonebook">

<pb:GetUserDetails>

<pb:UserID>12345</pb:UserID>

</pb:GetUserDetails>

</soap:Body>

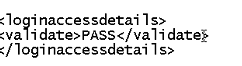
</soap:Envelope>

In rest, you can simply send the request over the URL

http://www.acme.com/phonebook/UserDetails/12345

<http://www.acme.com/phonebook?username=myusername&password=passwd>

Rest Response

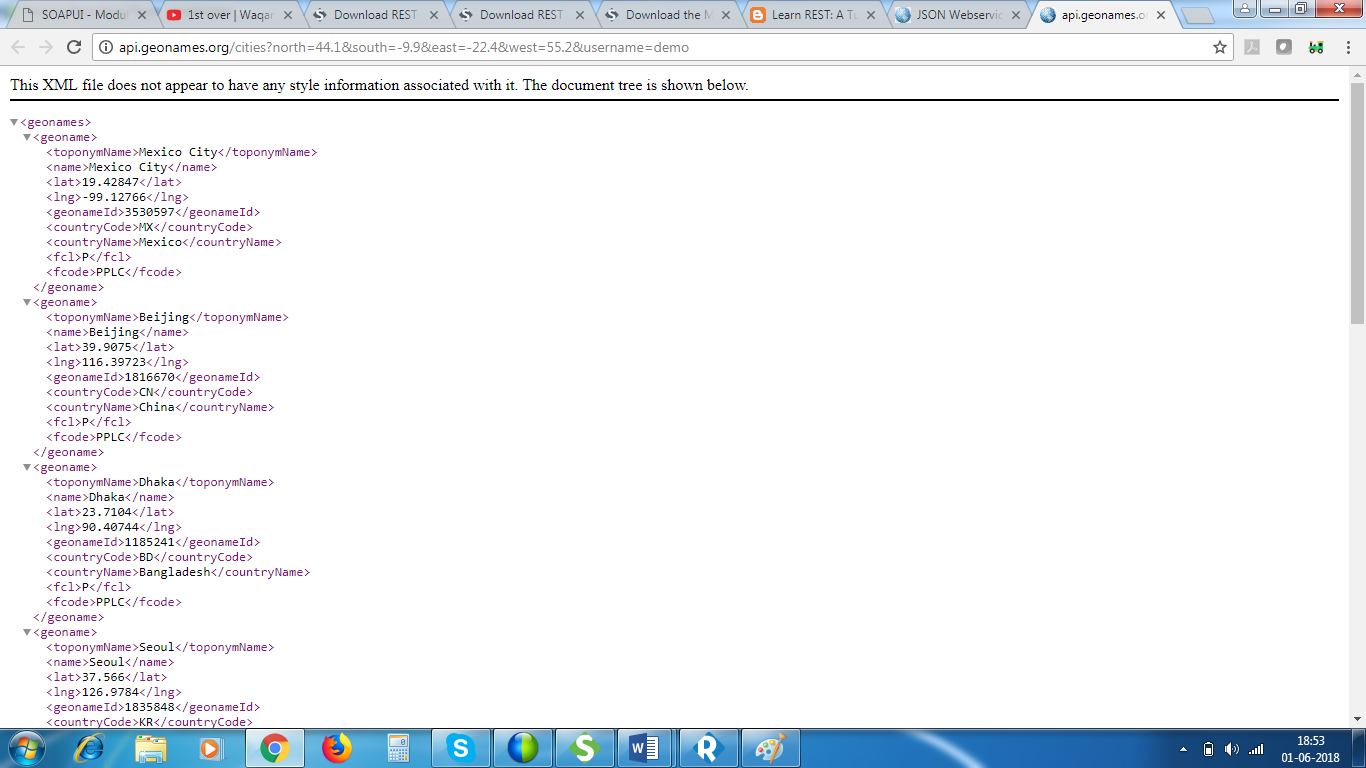


<http://www.geonames.org/export/JSON-webservices.html>

sample req:

<http://api.geonames.org/cities?north=44.1&south=-9.9&east=-22.4&west=55.2&username=demo>

RESPONSE REST:



In SOAP protocol , you need to send the request in soap TOOL. You cannot send in the form of url.

In Rest – response is simplified manner.

SOAP – response – xml document

REST – response – xml and JSON

Rest – HTTP protocol

Request will go in the form of URL over http

Response is xml or json

JSON

Java script option notation

JSON response

<http://api.geonames.org/citiesJSON?north=44.1&south=-9.9&east=-22.4&west=55.2&lang=de&username=demo>

{"geonames":[{"lng":-99.12766456604,"geonameId":3530597,"countrycode":"MX","name":"Mexiko-Stadt","fclName":"city, village,...","toponymName":"Mexico City","fcodeName":"capital of a political entity","wikipedia":"en.wikipedia.org/wiki/Mexico\_City","lat":19.428472427036,"fcl":"P","population":12294193,"fcode":"PPLC"},{"lng":116.397228240967,"geonameId":1816670,"countrycode":"CN","name":"Peking","fclName":"city, village,...","toponymName":"Beijing","fcodeName":"capital of a political entity","wikipedia":"en.wikipedia.org/wiki/Beijing","lat":39.9074977414405,"fcl":"P","population":11716620,"fcode":"PPLC"},{"lng":90.40743827819824,"geonameId":1185241,"countrycode":"BD","name":"Dhaka","fclName":"city, village,...","toponymName":"Dhaka","fcodeName":"capital of a political entity","wikipedia":"en.wikipedia.org/wiki/Dhaka","lat":23.710395616597037,"fcl":"P","population":10356500,"fcode":"PPLC"},{"lng":126.9784,"geonameId":1835848,"countrycode":"KR","name":"Seoul","fclName":"city, village,...","toponymName":"Seoul","fcodeName":"capital of a political entity","wikipedia":"en.wikipedia.org/wiki/Seoul","lat":37.566,"fcl":"P","population":10349312,"fcode":"PPLC"},{"lng":106.84513092041016,"geonameId":1642911,"countrycode":"ID","name":"Jakarta","fclName":"city, village,...","toponymName":"Jakarta","fcodeName":"capital of a political entity","wikipedia":"en.wikipedia.org/wiki/Jakarta","lat":-6.214623197035775,"fcl":"P","population":8540121,"fcode":"PPLC"},{"lng":139.69171,"geonameId":1850147,"countrycode":"JP","name":"Tokio","fclName":"city, village,...","toponymName":"Tokyo","fcodeName":"capital of a political entity","wikipedia":"de.wikipedia.org/wiki/Tokyo","lat":35.6895,"fcl":"P","population":8336599,"fcode":"PPLC"},{"lng":121.531846,"geonameId":1668341,"countrycode":"TW","name":"Taipeh","fclName":"city, village,...","toponymName":"Taipei","fcodeName":"capital of a political entity","wikipedia":"de.wikipedia.org/wiki/Taipei","lat":25.047763,"fcl":"P","population":7871900,"fcode":"PPLC"},{"lng":-74.08175468444824,"geonameId":3688689,"countrycode":"CO","name":"Bogotá","fclName":"city, village,...","toponymName":"Bogotá","fcodeName":"capital of a political entity","wikipedia":"en.wikipedia.org/wiki/Bogot%C3%A1","lat":4.609705849789108,"fcl":"P","population":7674366,"fcode":"PPLC"},{"lng":114.17469,"geonameId":1819729,"countrycode":"HK","name":"Hongkong","fclName":"city, village,...","toponymName":"Hong Kong","fcodeName":"capital of a political entity","wikipedia":"en.wikipedia.org/wiki/Hong\_Kong","lat":22.27832,"fcl":"P","population":7012738,"fcode":"PPLC"},{"lng":100.501444,"geonameId":1609350,"countrycode":"TH","name":"Bangkok","fclName":"city, village,...","toponymName":"Bangkok","fcodeName":"capital of a political entity","wikipedia":"en.wikipedia.org/wiki/Bangkok","lat":13.753979,"fcl":"P","population":5104476,"fcode":"PPLC"}]}

REST – JSON &XML

API – Application programing interface

REST is also used to test API

Ny stock exchange. – API

To this you send the username and stock name

API will return you back the price of the share.

CRUD

IN HTP PROTOCOL / browser what kind of request you send to browser?

In Google , whatever I search searched ,the searched text will be going to url

Whereas in gmail , you will not see the username and pwd in the url

1. You can see the parameters in the url
2. The other you cannot see the parameter in the url coz of security

Get request – see the parameters going in the req

POST – YOU cannot see the parameters ,like username/pwd or any other data

PUT –data retrieval

Delete – delete the request

End user - server

REST

1. Architecture
2. More informal
3. Request is sent over http protocol
4. Reponse – json/xml
5. Rest is used to test API

CRUD –

C- create - post

R – retrieve – post , put

U – update - put

D- delete – delete

<https://id.atlassian.com/signup?application=mac&continue=https://my.atlassian.com>

username : [friendlyananth15@gmail.com](mailto:friendlyananth15@gmail.com)

password : Handgreat99

<https://docs.atlassian.com/DAC/rest/jira/6.1.html>

<https://friendlyananth15.atlassian.net/browse/AN-11?filter=-1>

You can add an issue using webservices rest api

REST – GET

READY – POST/PUT

Endpoints

Resources

Methods-4

https://friendlyananth15.atlassian.net/rest/auth/1/session

https://friendlyananth15.atlassian.net/rest/api/2/project

https://friendlyananth15.atlassian.net/rest/api/2/issue

RED – End point

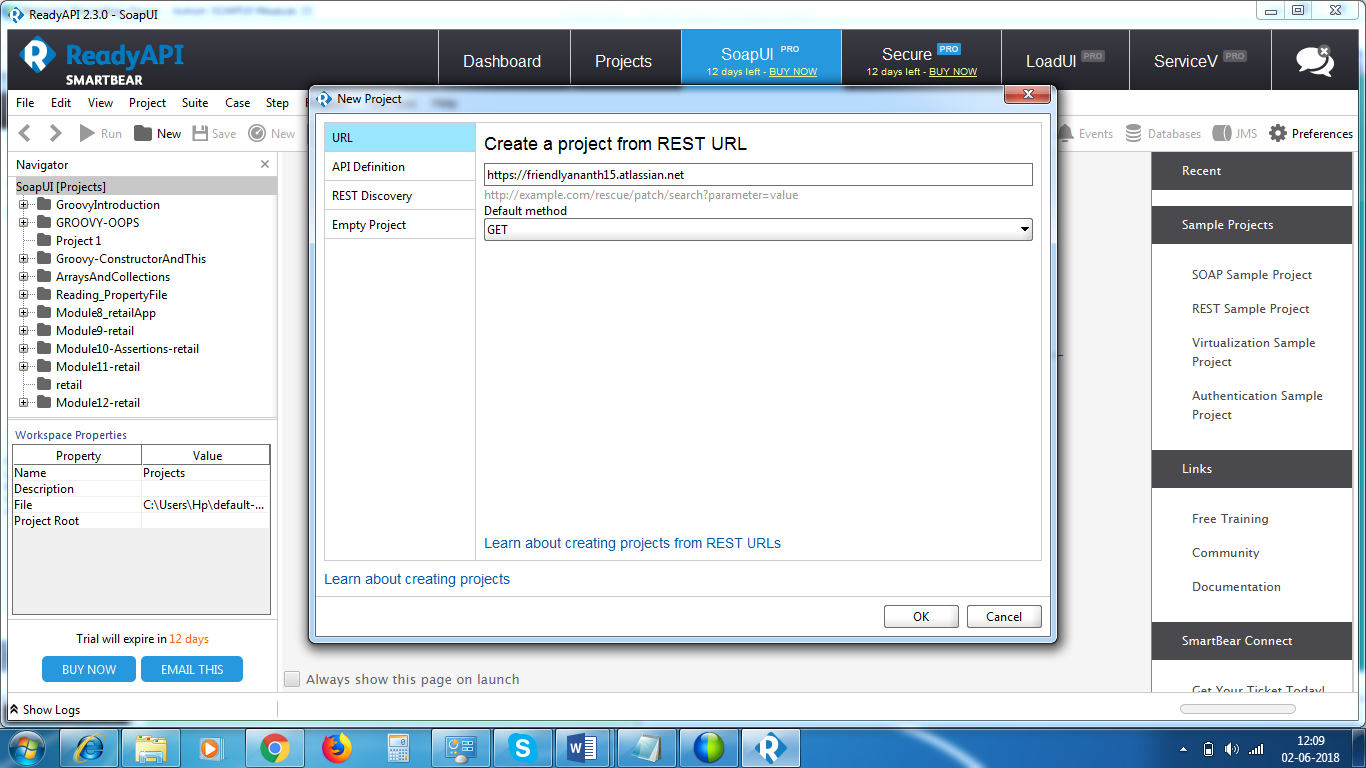
Yellow – resource

In one end point – you can have multiple resources

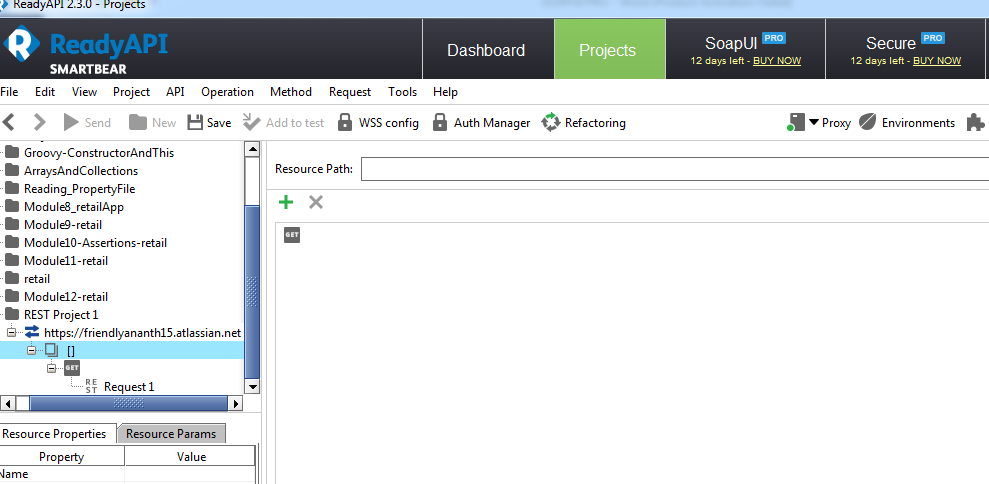
Under every resource – you can fire GET,PUT,POST,DELETE

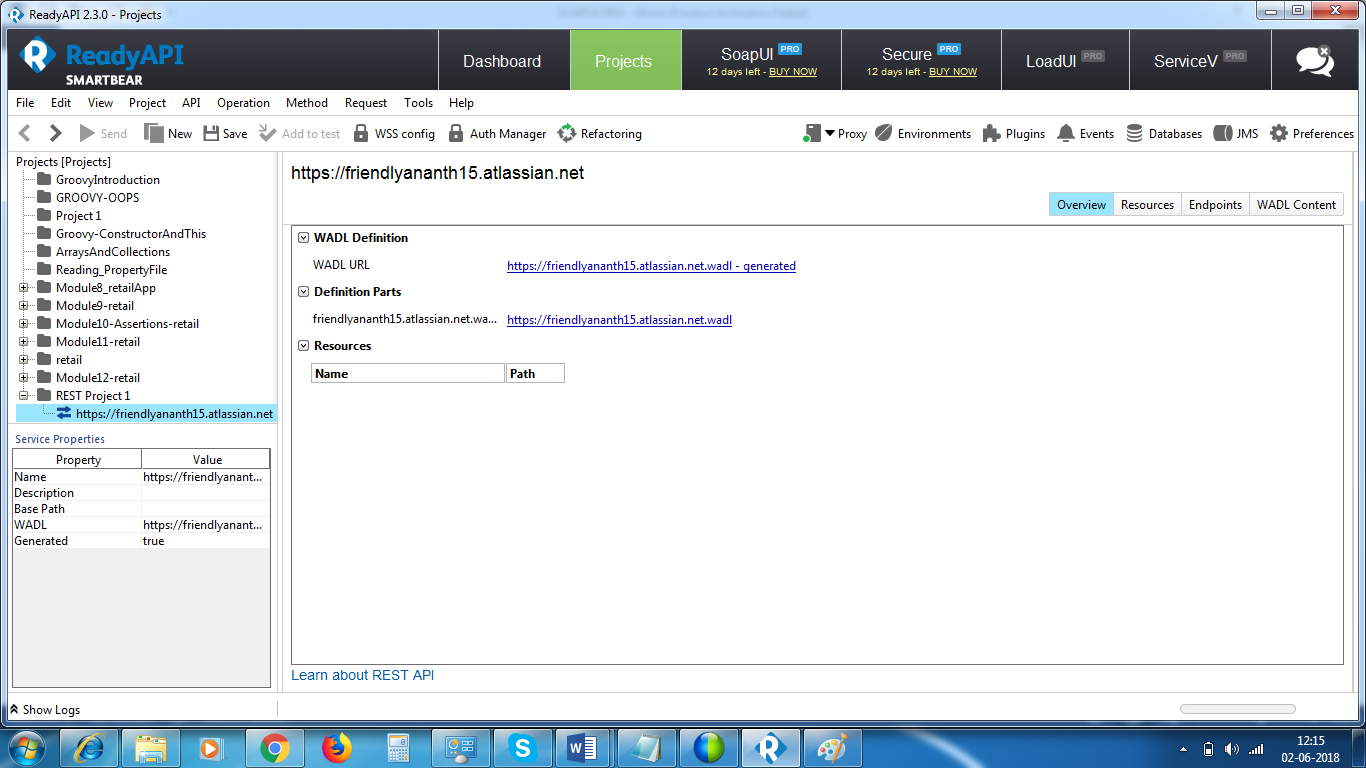
Open SOAPUI pro,create a rest project

Enter the end point

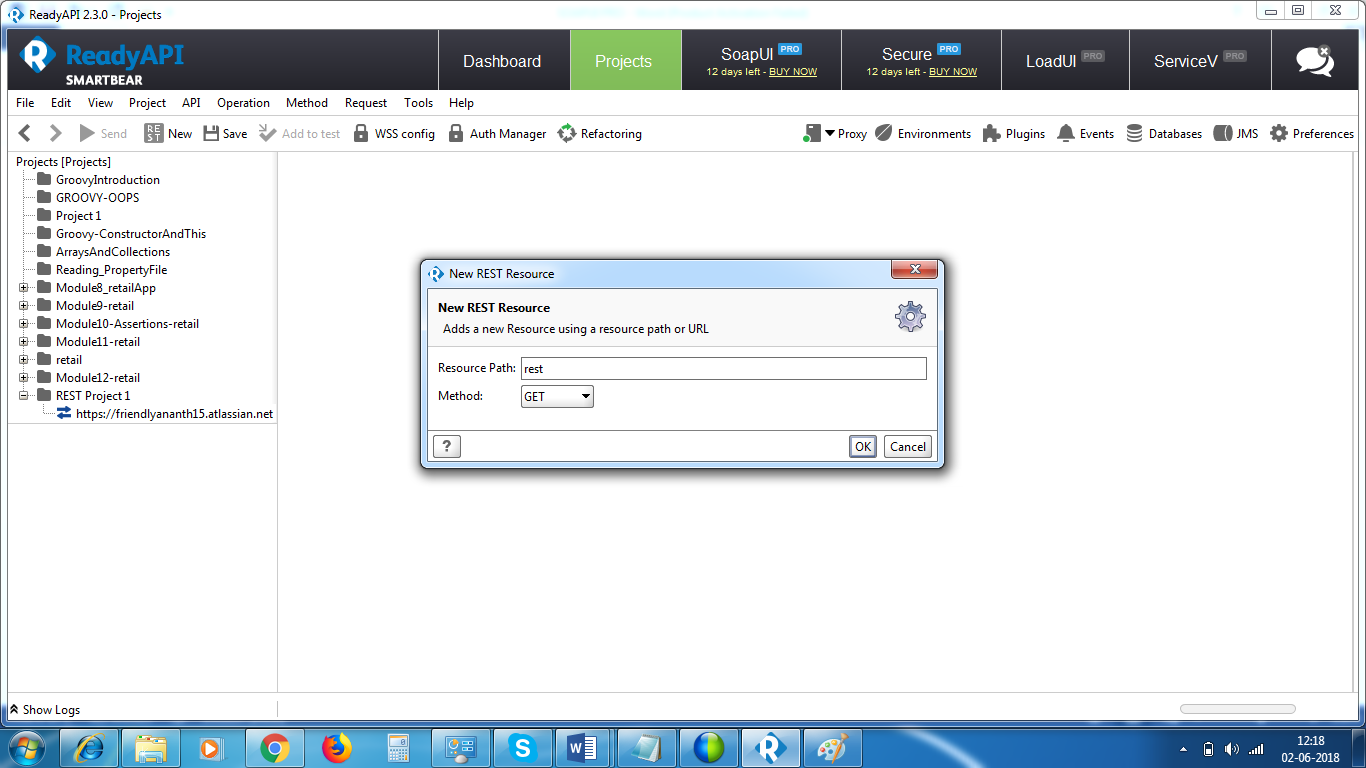


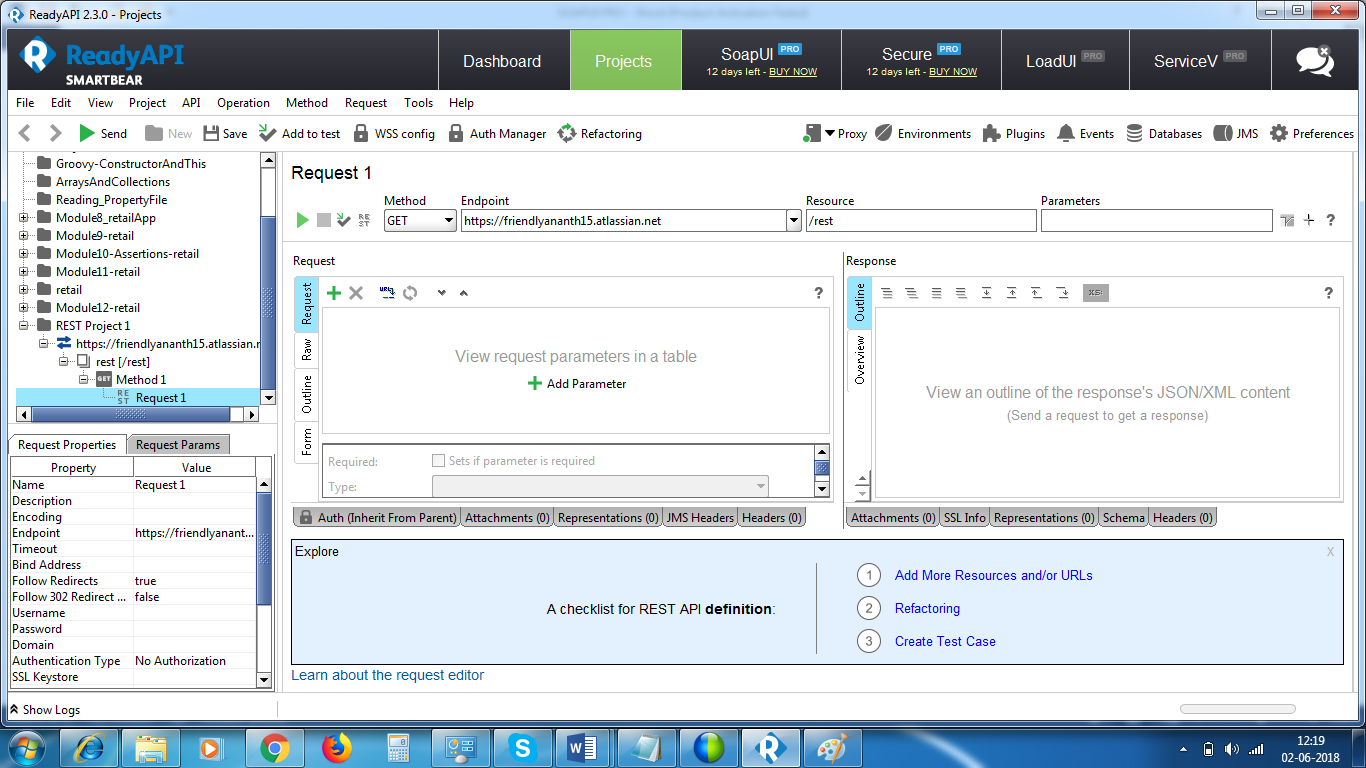
Delete the resources that is by default created –[]





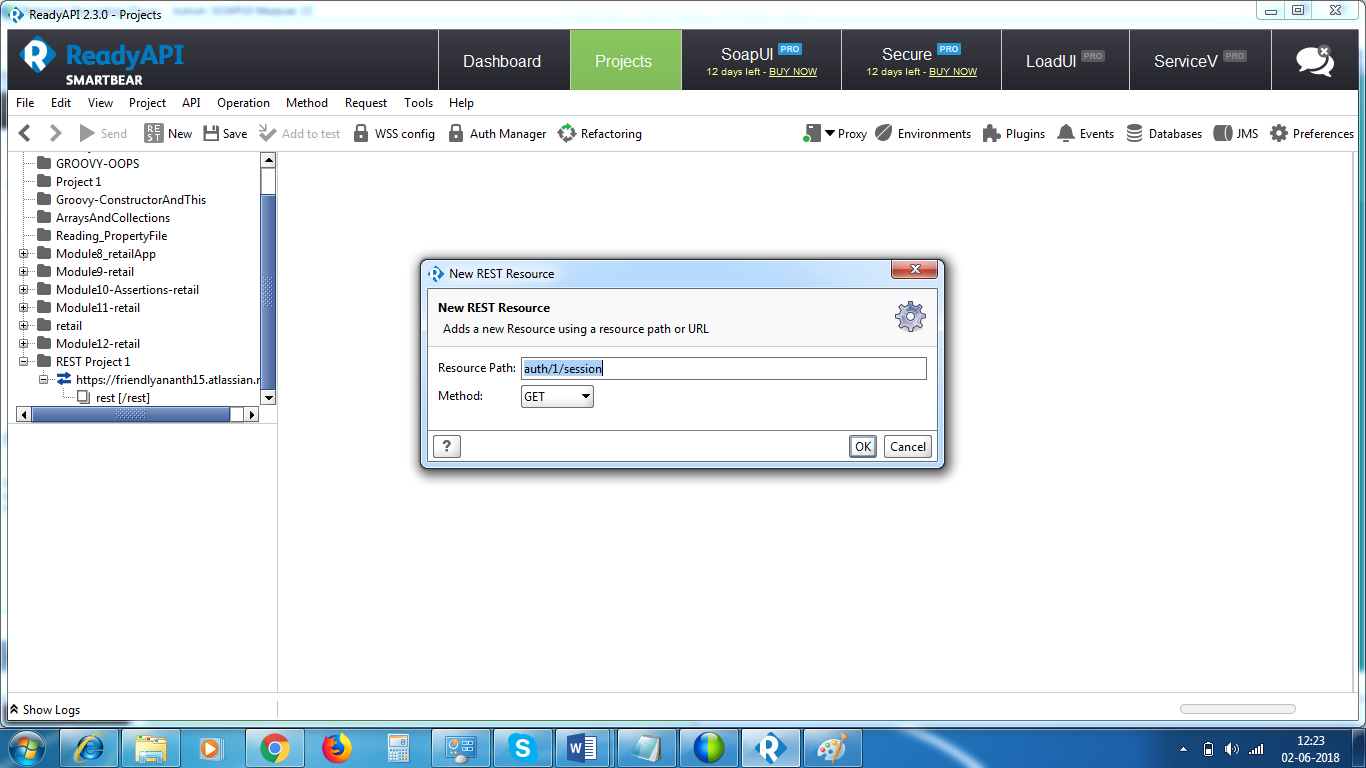
Create a new rest resource - rest





Delete the Method that is created by default

Create a path/ child resource auth/1/session



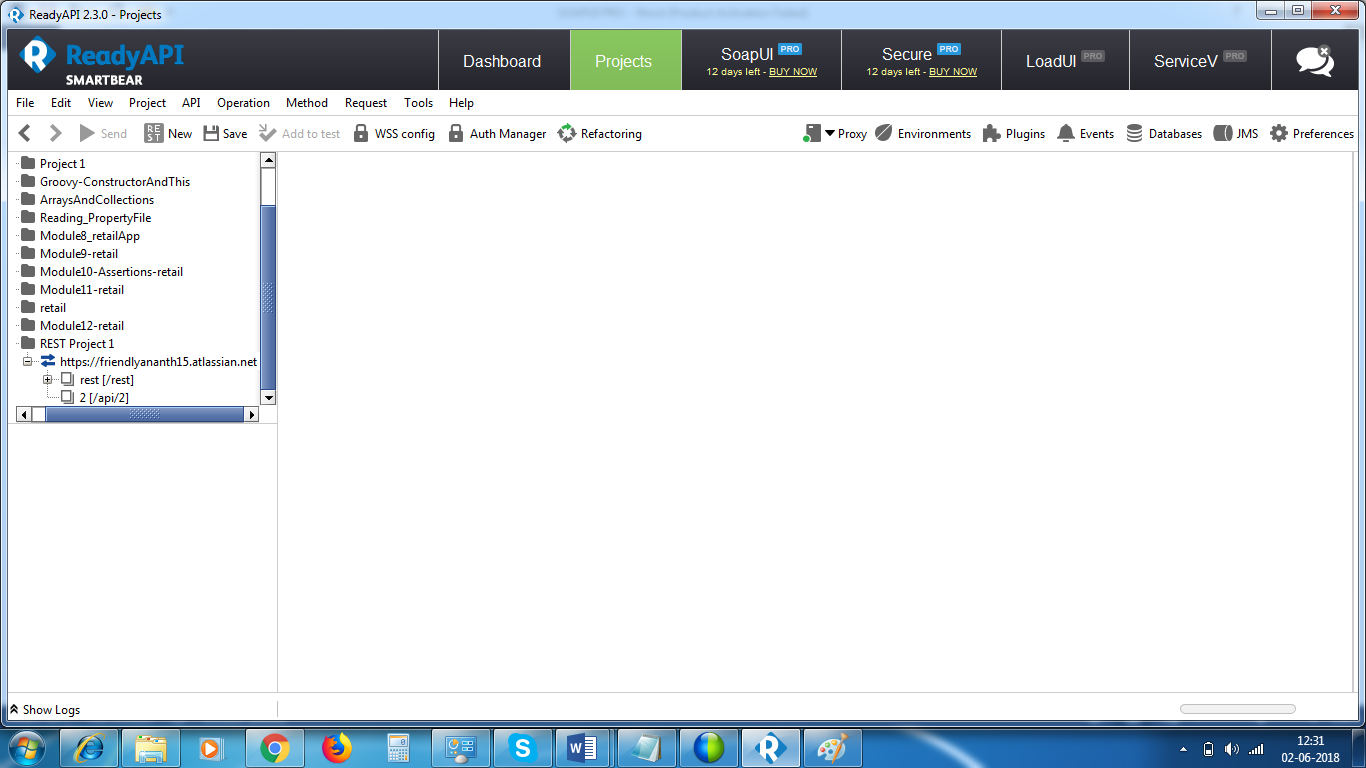
https://friendlyananth15.atlassian.net/rest/auth/1/session

https://friendlyananth15.atlassian.net/rest/api/2/project

<https://friendlyananth15.atlassian.net/rest/api/2/issue>

or create a new resource api/2

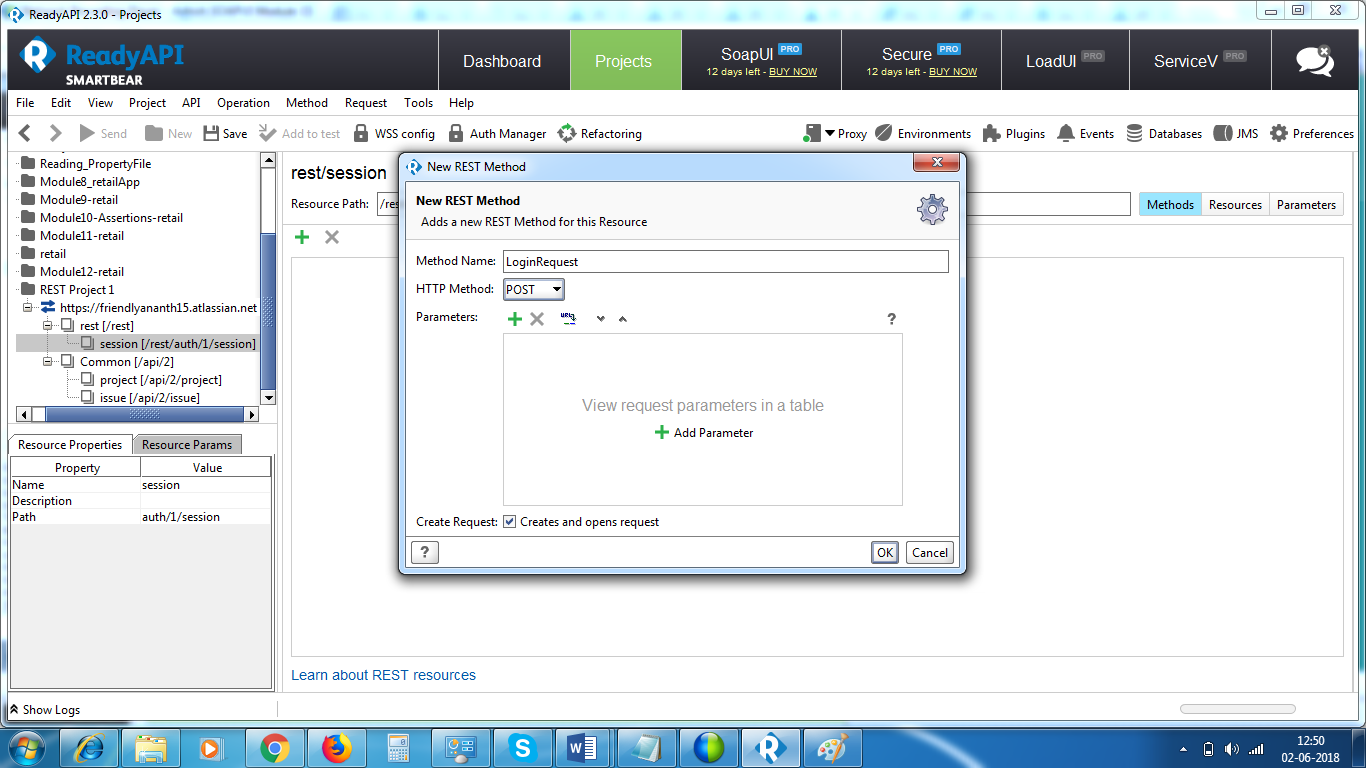
and delete the method

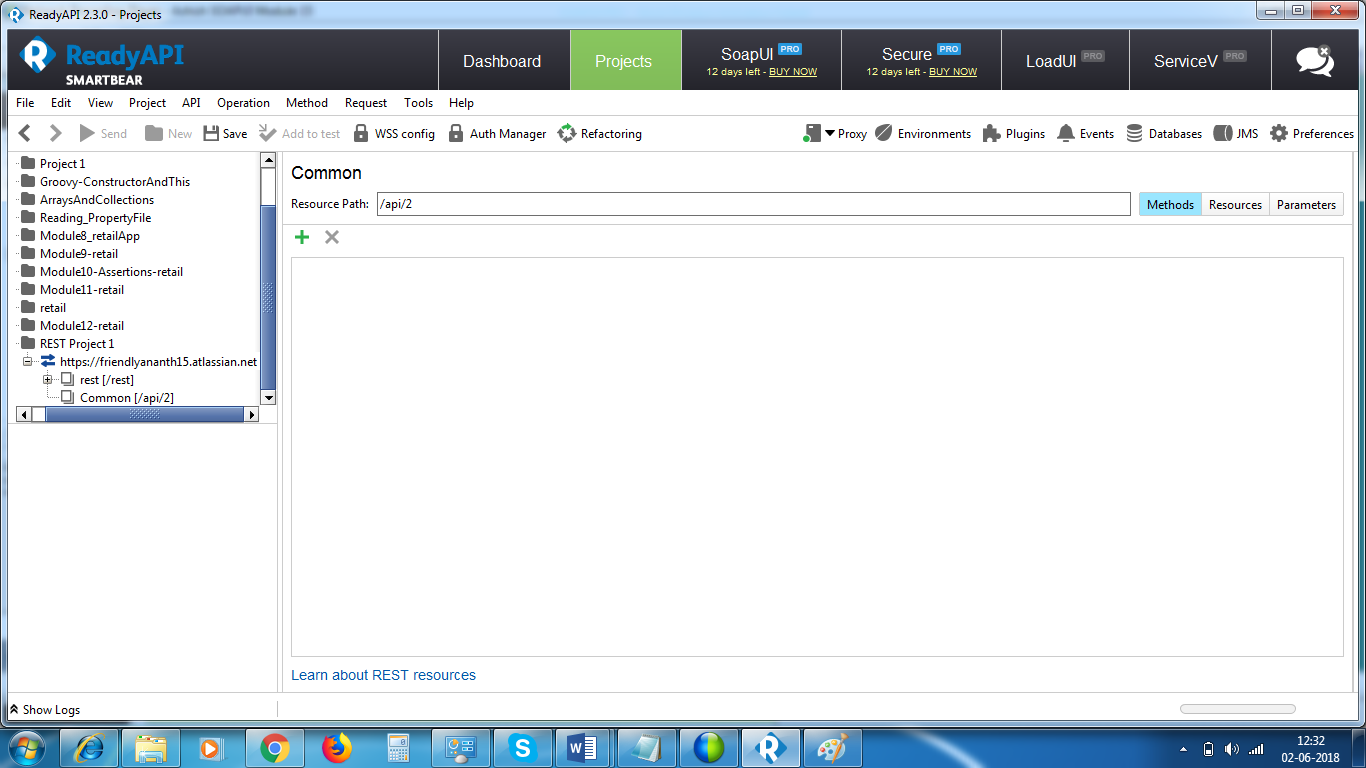


Change 2 to common

Now I can create another resource – project and issue

POST – Session id – username and pwd

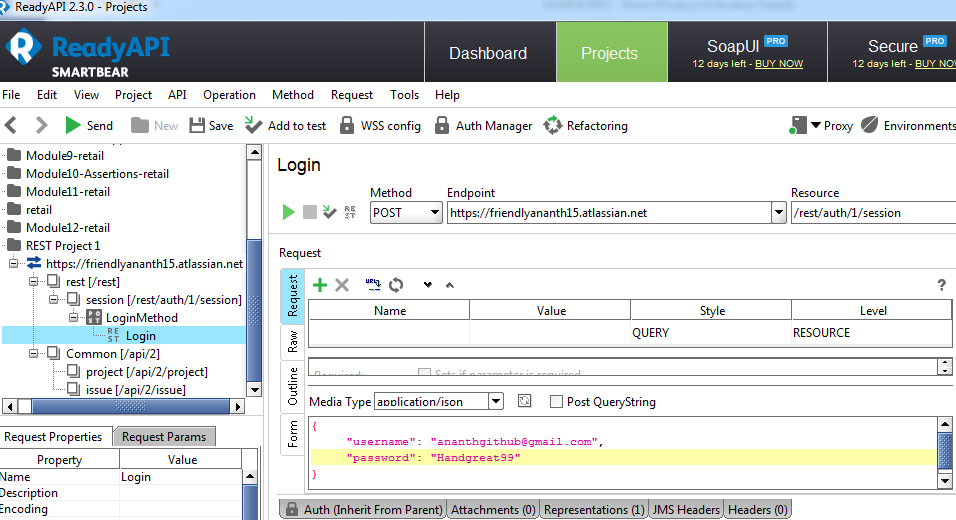




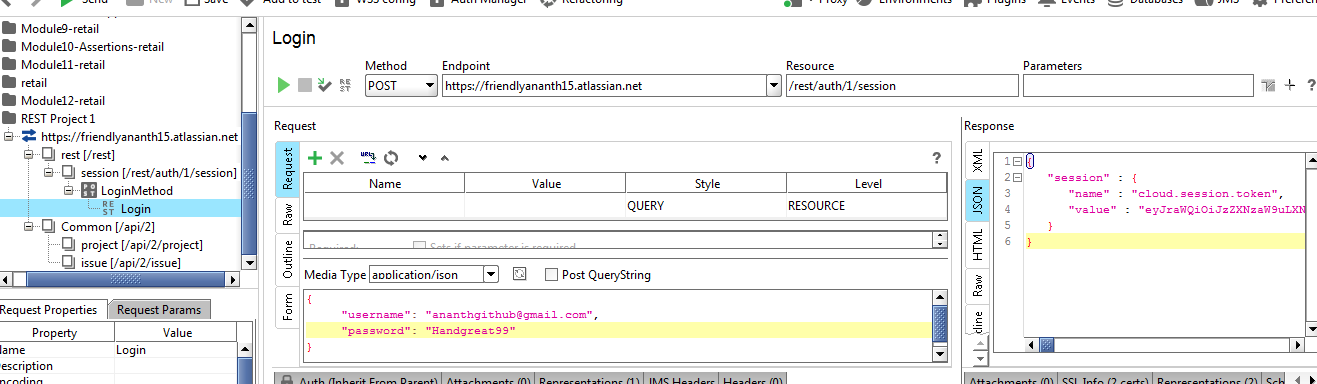
Create project and issue under common

https://friendlyananth15.atlassian.net/rest/api/2/project

<https://friendlyananth15.atlassian.net/rest/api/2/issue>



Fire username and password as an api request



Request:

{

"username": "ananthgithub@gmail.com",

"password": "Handgreat99"

}

RESPONSE

{

"session" : {

"name" : "cloud.session.token",

"value" : "eyJraWQiOiJzZXNzaW9uLXNlcnZpY2VcL3Nlc3Npb24tc2VydmljZSIsImFsZyI6IlJTMjU2In0.eyJhc3NvY2lhdGlvbnMiOltdLCJzdWIiOiI1YjEyMWY4N2IzODZmYzUxZjMyYWUyMWQiLCJhdWQiOiJhdGxhc3NpYW4iLCJpbXBlcnNvbmF0aW9uIjpbXSwibmJmIjoxNTI3OTI1MTU1LCJyZWZyZXNoVGltZW91dCI6MTUyNzkyNTc1NSwiaXNzIjoic2Vzc2lvbi1zZXJ2aWNlIiwic2Vzc2lvbklkIjoiY2JiMzAxZjUtMDNmNC00OTVlLTk5MDEtMmI4NTI5MTAwYjI3IiwiZXhwIjoxNTMwNTE3MTU1LCJpYXQiOjE1Mjc5MjUxNTUsImVtYWlsIjoiYW5hbnRoZ2l0aHViQGdtYWlsLmNvbSIsImp0aSI6ImNiYjMwMWY1LTAzZjQtNDk1ZS05OTAxLTJiODUyOTEwMGIyNyJ9.nCg2Ap9I\_g\_OIo0M4ZS\_2ZQZ\_pffH97URoB7Y\_Vm7qLuGB2zmha8STSxpGxW\_Es6F9xH538xph0HuXl5tJVttCJ817S93NJKOqko6GT9kf7Z0JsP2gQ-qUdaegp9drCHkUHuL46Xhl\_99YaA5gZeUahBJo9Mhbvd4A8YHRFHaohnDiiKtGavldozgrMRcuEM8SFXitWmDnYuiBn3Xw8T2cU9jLcA52Rhe-BnqF9JciFn34mTGJCnSE9Ohips0szG86X2id19raz9J4I8BC-TRd-CLnW0EM5wOfI\_mXRvu2J88T6ykV1SqsVsdTk42Jmh3xl5mcBPpFNyHBDMbPxt3g"

}

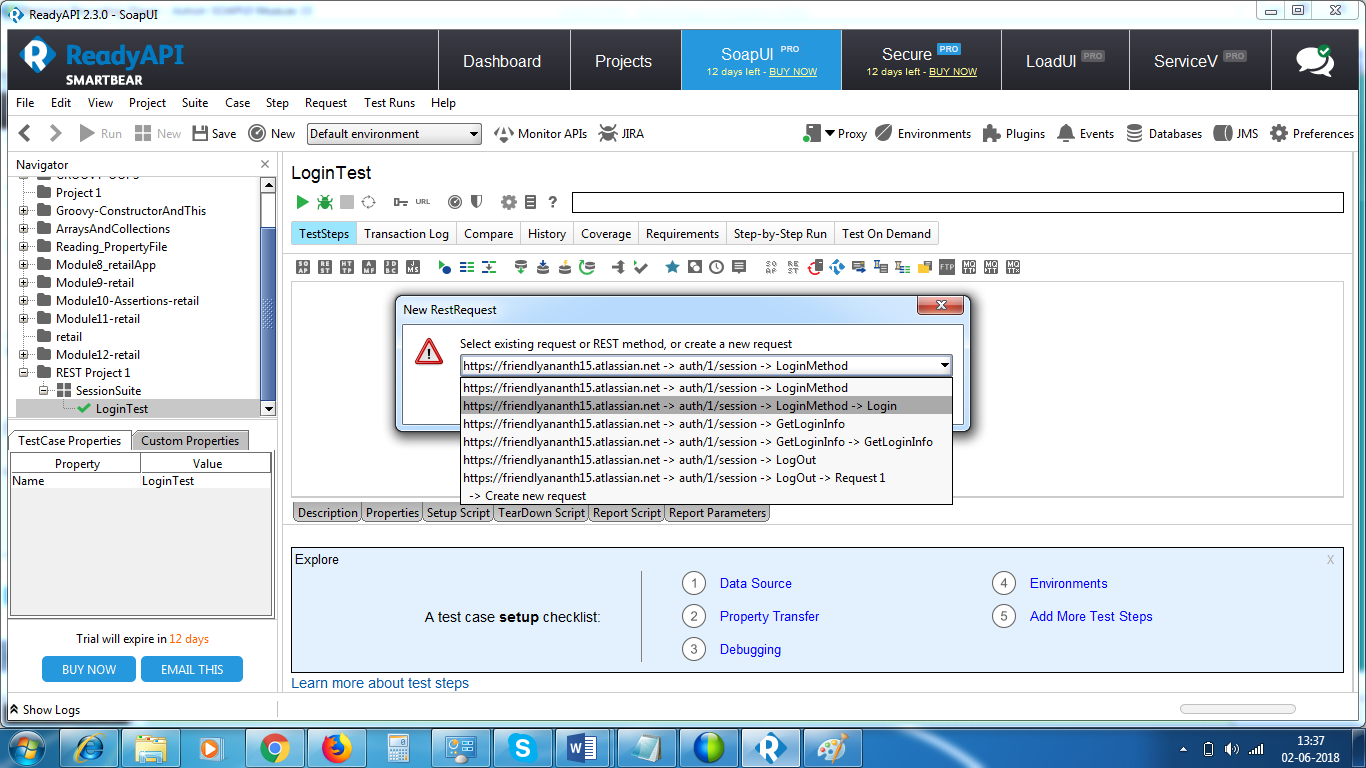
}

Along with response , you will see headers

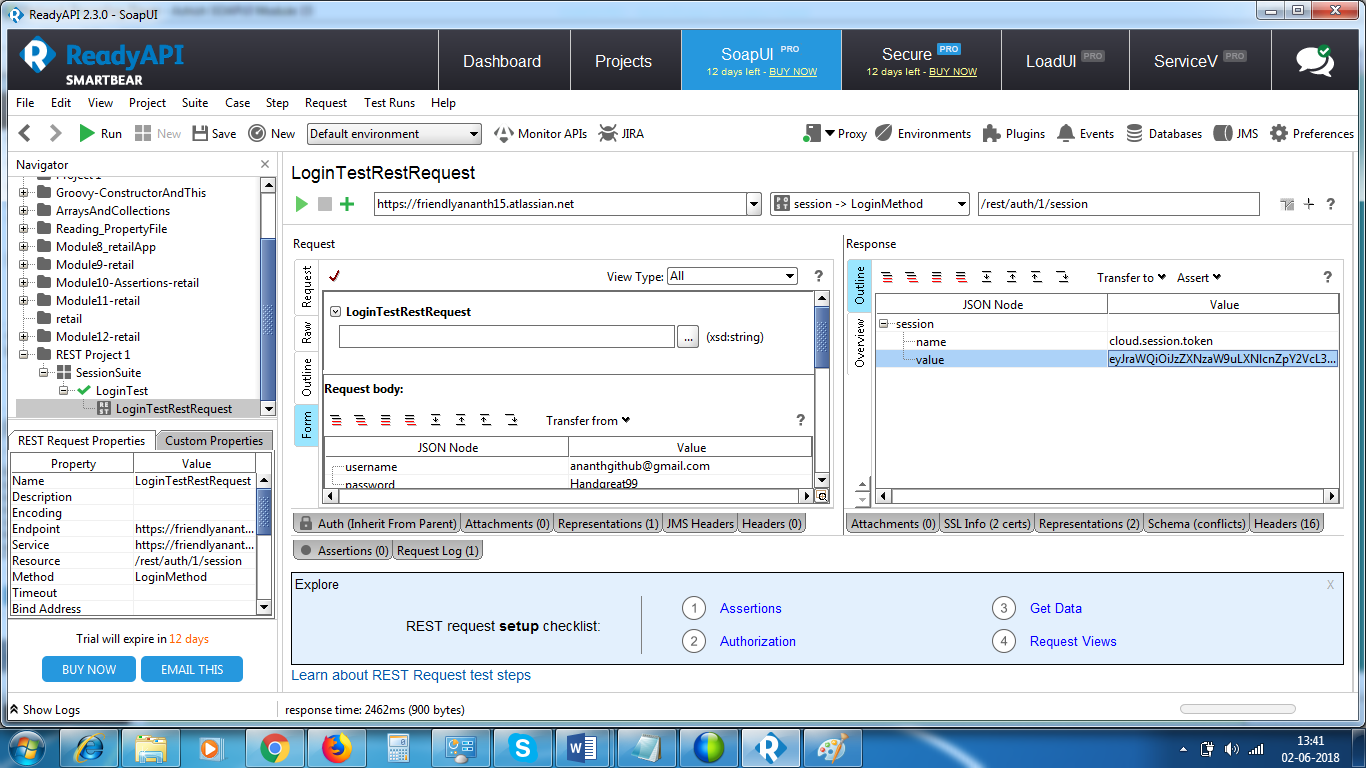
In SOAPUI tab - > RIGHT click on the project and click create test suite

Create a test case - > Login Test

Create a test step - > Rest Request – Login Test Request



Execute it :



If you want to change the project name / or change the request, you need to do it in project tab.

LoginTest – POST

GetLoginInfoTest- GET

LogOut – Delete

Hints:

Cookie

if you send the req , you get headers and message in the response

You need to read the cookie value in the response and pass it to the next request.

Message is the normal response body

String to objectmap

It will read the header – cookie

//set the header

From one test case to another test case.

//read the headers

def headers=testRunner.testCase.testSuite.getTestCaseByName(“LoginTest”).getStepByName(“LoginTestReq”).testRequest.response.responseHeaders (“Set-Cookie”)

log.info headers

log.info headers.get[0]

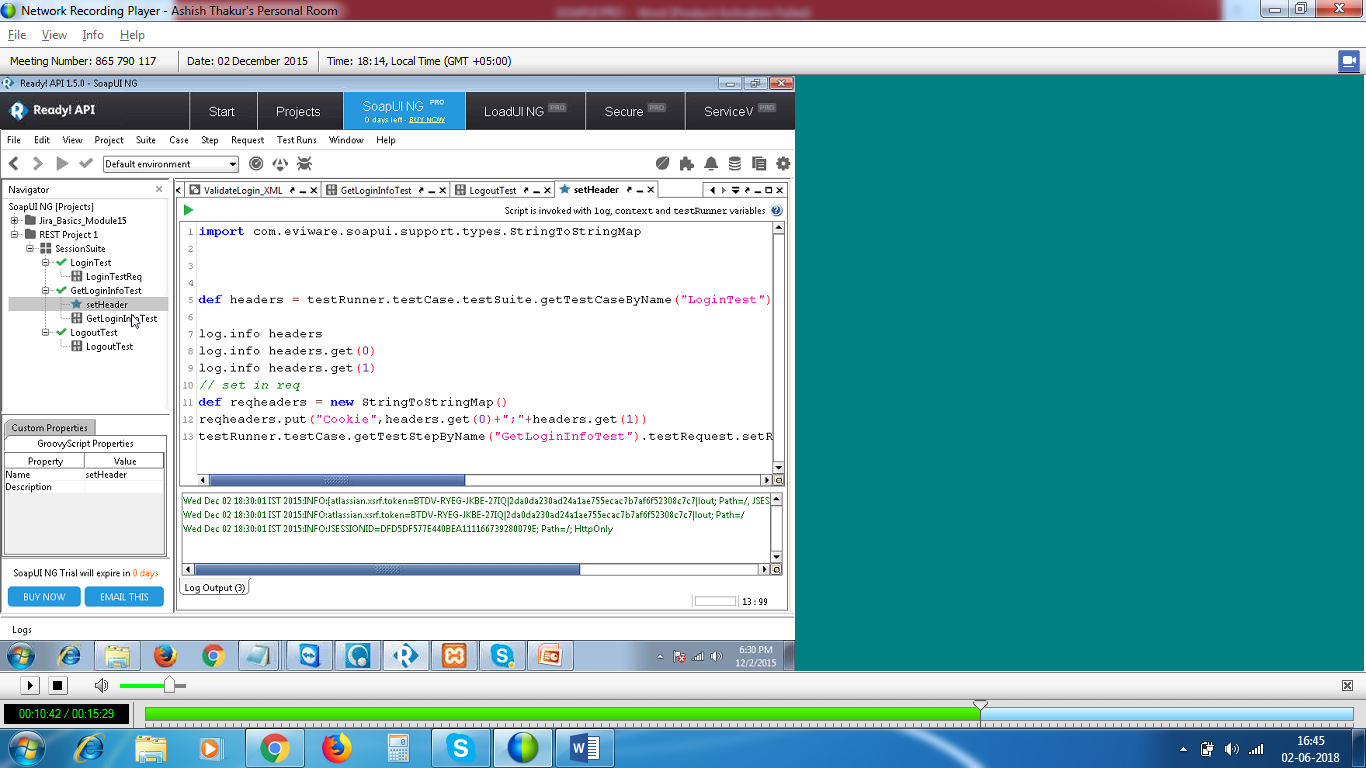
log.info headers.get[1]

//pass the header to another req

Def reqheaders=new StringToStringMap()

reqheaders.put(“Cookie”,headers.get(0)+”;”+headers.get(1))

testRunner.testCase.getTestStepByName(“GetLoginInfoTest”).testRequest.SetRequestHeaders(reqheaders)



Run the setHeader test (groovy test script) and then run GetLoginInfoTest

You will see the header being set in GetLoginInfoTest

Now run the test GetLoginInfoTest , you will get the response

Xml can be ready by xmlholder

How json can be read?

Assertion test step

Add assertion test step

Select source – step1[LoginTestReq]

Add a script assertion

Import groovy.json.JsonSlurper

def res = Message exchange.response.responseContent //this will read the response

log.info res

def slurper=new JsonSlurper();

def json=slurper.parseText(res)

log.info json

//WILL COME BACK TO THIS LATER

//EXAMPLE

def country = new Hashtable()

country.put(“india”,”delhi”)

def phones = new Hashtable()

phones.put(“Samsung”,”Note”)

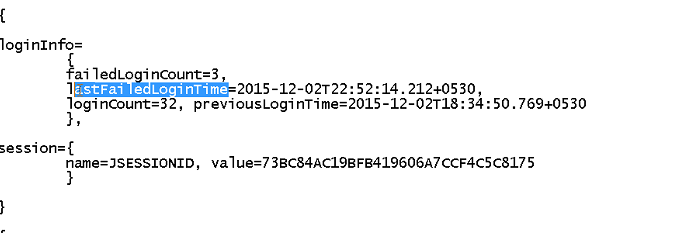
def global=new Hashtable()

global.put(“country”,countries)

global.put(“phone”,phones)

log.info global.get(“country”).get(“India”)

log.info global.country.INDIA



//coming back to where we left

Log.info json.logInfo.lastFailedLoginTime

Log.info json.session.value

Log.info json.size : Ans : 2 there are 2 keys

Log.info json.logininfo.size() : 4

Log.info json.session.size() : 4

Assert json.size()==2 //pass as there are 2 keys

Assert json.size()==3 “ERR – expecting sie to be 2 “

// how to assert session id

Assert.json.session.value.matches(“[0-9A-Z]”) //PASS

Assert.json.session.value.matches(“[0-9A-Z]\*{3}”) //FAIL – as the length should be exactly 3

Assert.json.session.value.matches(“[0-9A-Z]\*{3,}”) //pass

Validate XML

Create script assertion

Import com.eviware.soapUI.support.XmlHolder

def response = messageExchange.response.responseContent AsXml.toString()

log.info response

response=response.replace(“xmlns”,”xmlns:z”) //workaround since namespace is not there

def holder = new XmlHolder(reponse)

holder.getNodeValue(“//loginCount”) --this will print null which is not expected

it is printing null since there is no namespace in the response

like xmlns:a it is not there – we have added for example



Video 16

Parameters