Groovey Script:

```
import groovy.json.JsonSlurper

import jxl.*

import jxl.write.*

import org.apache.poi.xssf.usermodel.*

Workbook work = Workbook.getWorkbook(new
File("C:\\Users\\M1073124\\Desktop\\Compreihensive\\soup\\SBAspreadSheetData.xls"));

Sheet sh = work.getSheet(0)

rc = sh.getRows();

cc = sh.getColumns();

log.info"rows count="+rc

log.info"column count="+cc

for(int i=1;i<rc;i++)

{

String option=sh.getCell(0,i).getContents()

switch(option)

{

case "post":

log.info "post"

def data=testRunner.testCase.testSuite

def testStep = testRunner.testCase.testSteps["post_request"]

for(int ii=testStep.assertionCount-1; ii>=0; ii--) {

testStep.removeAssertion(testStep.getAssertionAt(ii));

}

def Assertion = testRunner.testCase.testSteps["post_request"].addAssertion("Contains")

for(int j=1;j<cc;j++)

{

if(j==1)

{
```

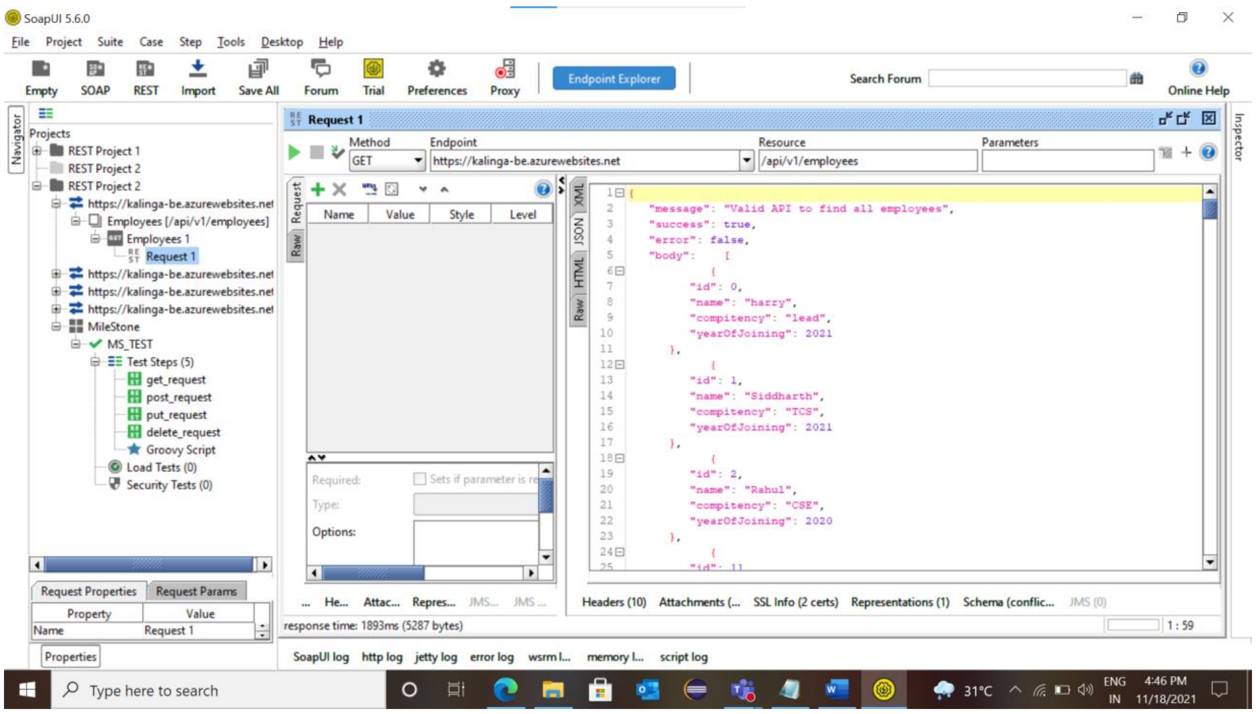
```
data.setPropertyValue("compitency",sh.getCell(j,i).getContents())

Assertion.setToken(sh.getCell(j,i).getContents())

}

else if(j==2)

{

data.setPropertyValue("id",sh.getCell(j,i).getContents())

Assertion.setToken(sh.getCell(j,i).getContents())

}

else if(j==3)

{

data.setPropertyValue("name",sh.getCell(j,i).getContents())

Assertion.setToken(sh.getCell(j,i).getContents())

}

else

{

data.setPropertyValue("yearOfJoining",sh.getCell(j,i).getContents())

Assertion.setToken(sh.getCell(j,i).getContents())

}

}




def tStep=data.testCases["MS_TEST"].testSteps["post_request"]

tStep.run(testRunner,context)

def json=tStep.testRequest.response.responseContent

def tJson=new JsonSlurper()

def object=tJson.parseText(json)

//assert object.toString().contains(sh.getCell(1,i).getContents())==true

log.info (object)
```

```
Thread.sleep(5000)

break

case "get":

log.info "get"

def data=testRunner.testCase.testSuite

def testStep = testRunner.testCase.testSteps["get_request"]

for(int ii=testStep.assertionCount-1; ii>=0; ii--) {

testStep.removeAssertion(testStep.getAssertionAt(ii));

}

def Assertion = testRunner.testCase.testSteps["get_request"].addAssertion("Contains")

for(int j=1;j<cc;j++)

{

if(j==1)

{

Assertion.setToken(sh.getCell(j,i).getContents())

}

else if(j==2)

{

Assertion.setToken(sh.getCell(j,i).getContents())

}

else if(j==2)

{

Assertion.setToken(sh.getCell(j,i).getContents())

}

else

{

Assertion.setToken(sh.getCell(j,i).getContents())

}

}
```

```
def tStep=data.testCases["MS_TEST"].testSteps["get_request"]

tStep.run(testRunner,context)

def json=tStep.testRequest.response.responseContent

def tJson=new JsonSlurper()

def object=tJson.parseText(json)

//assert object.toString().contains(sh.getCell(1,i).getContents())==true

log.info (object)

Thread.sleep(3000)

break

case "put":

log.info "put"

def data=testRunner.testCase.testSuite

def testStep = testRunner.testCase.testSteps["put_request"]

for(int ii=testStep.assertionCount-1; ii>=0; ii--) {

testStep.removeAssertion(testStep.getAssertionAt(ii));

}

def Assertion = testRunner.testCase.testSteps["put_request"].addAssertion("Contains")

for(int j=1;j<cc;j++)

{


if(j==1)

{

data.setPropertyValue("compitency",sh.getCell(j,i).getContents())

Assertion.setToken(sh.getCell(j,i).getContents())

}

else if(j==2)

{

data.setPropertyValue("id",sh.getCell(j,i).getContents())

Assertion.setToken(sh.getCell(j,i).getContents())
}
else if(j==3)
{
data.setPropertyValue("name",sh.getCell(j,i).getContents())
Assertion.setToken(sh.getCell(j,i).getContents())
}
else
{
data.setPropertyValue("yearOfJoining",sh.getCell(j,i).getContents())
Assertion.setToken(sh.getCell(j,i).getContents())
}
}
def tStep=data.testCases["MS_TEST"].testSteps["put_request"]
tStep.run(testRunner,context)
def json=tStep.testRequest.response.responseContent
def tJson=new JsonSlurper()
def object=tJson.parseText(json)
//assert object.toString().contains(sh.getCell(2,i).getContents())==true
log.info (object)
Thread.sleep(3000)
break
case "delete":
log.info "delete"
def data=testRunner.testCase.testSuite
def testStep = testRunner.testCase.testSteps["delete_request"]
for(int ii=testStep.assertionCount-1; ii>=0; ii--) {
testStep.removeAssertion(testStep.getAssertionAt(ii));
}
def Assertion = testRunner.testCase.testSteps["delete_request"].addAssertion("Contains")
```

```
Assertion.setToken("DELETED")


def tStep=data.testCases["MS_TEST"].testSteps["delete_request"]
tStep.run(testRunner,context)
def json=tStep.testRequest.response.responseContent
def tJson=new JsonSlurper()
def object=tJson.parseText(json)
//assert object.toString().contains("DELETED")==true
log.info (object)
break
}
}
work.close();
```

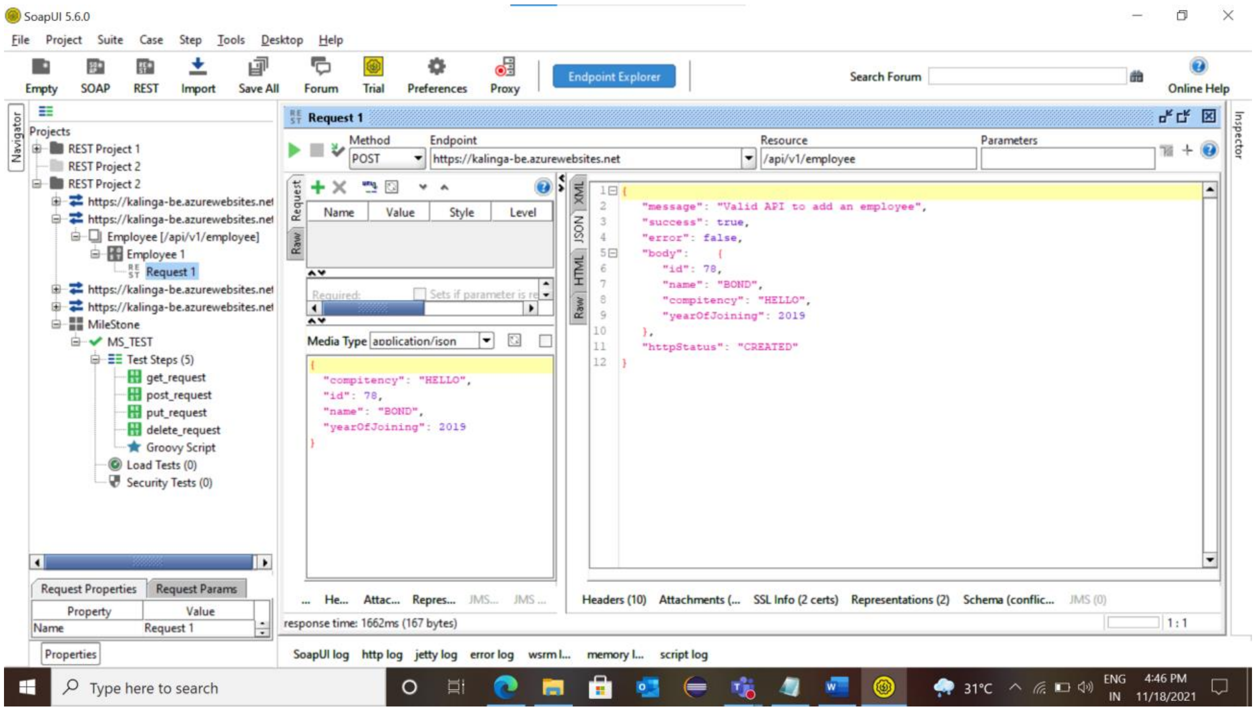
OUTPUT:



Method: GET
Endpoint: https://kalinga-be.azurewebsites.net
Resource: /api/v1/employees

```
{
   "message": "Valid API to find all employees",
   "success": true,
   "error": false,
   "body":    [
            {
         "id": 0,
         "name": "harry",
         "compitency": "lead",
         "yearOfJoining": 2021
      },
            {
         "id": 1,
         "name": "Siddharth",
         "compitency": "TCS",
         "yearOfJoining": 2021
      },
            {
         "id": 2,
         "name": "Rahul",
         "compitency": "CSE",
         "yearOfJoining": 2020
      },
            {
```

response time: 1893ms (5287 bytes)



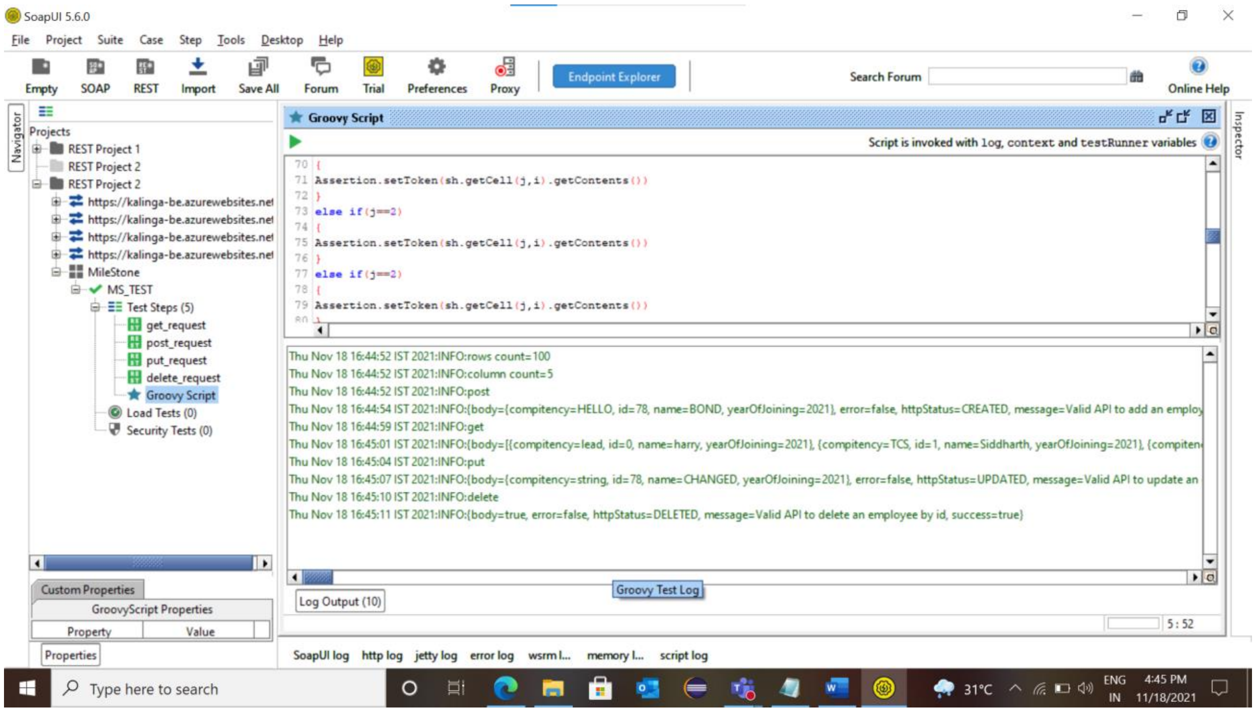
Request 1 — Method: POST — Endpoint: https://kalinga-be.azurewebsites.net — Resource: /api/v1/employee

Media Type: application/json

```
{
  "compitency": "HELLO",
  "id": 78,
  "name": "BOND",
  "yearOfJoining": 2019
}
```

```
{
   "message": "Valid API to add an employee",
   "success": true,
   "error": false,
   "body":    {
      "id": 78,
      "name": "BOND",
      "compitency": "HELLO",
      "yearOfJoining": 2019
   },
   "httpStatus": "CREATED"
}
```

response time: 1662ms (167 bytes)



Groovy Script

Script is invoked with log, context and testRunner variables

```
{
Assertion.setToken(sh.getCell(j,i).getContents())
}
else if(j==2)
{
Assertion.setToken(sh.getCell(j,i).getContents())
}
else if(j==2)
{
Assertion.setToken(sh.getCell(j,i).getContents())
```

```
Thu Nov 18 16:44:52 IST 2021:INFO:rows count=100
Thu Nov 18 16:44:52 IST 2021:INFO:column count=5
Thu Nov 18 16:44:52 IST 2021:INFO:post
Thu Nov 18 16:44:54 IST 2021:INFO:{body={compitency=HELLO, id=78, name=BOND, yearOfJoining=2021}, error=false, httpStatus=CREATED, message=Valid API to add an employ
Thu Nov 18 16:44:59 IST 2021:INFO:get
Thu Nov 18 16:45:01 IST 2021:INFO:{body=[{compitency=lead, id=0, name=harry, yearOfJoining=2021}, {compitency=TCS, id=1, name=Siddharth, yearOfJoining=2021}, {compiten
Thu Nov 18 16:45:04 IST 2021:INFO:put
Thu Nov 18 16:45:07 IST 2021:INFO:{body={compitency=string, id=78, name=CHANGED, yearOfJoining=2021}, error=false, httpStatus=UPDATED, message=Valid API to update an
Thu Nov 18 16:45:10 IST 2021:INFO:delete
Thu Nov 18 16:45:11 IST 2021:INFO:{body=true, error=false, httpStatus=DELETED, message=Valid API to delete an employee by id, success=true}
```