MODUL 5
RESULT AND REPORTING

A. Tujuan

1. Mengetahui apa itu test execution reports pada katalon.
2. Mengetahui bagaimana cara melihat hasil test execution pada katalon.
3. Mengetahui bagaimana cara melihat hasil report pengujian dalam format tertentu.
4. Mengetahui bagaimana cara melihat hasil report pengujian pada email.
5. Mengetahui apa itu analytic katalon pada kataon studio.

B. Dasar Teori

Test execution report merupakan cara untuk melihat hasil pengujian yang telah dilakukan, dimana hasil pengujian ini akan ditampilkan dalam bentuk laporan.

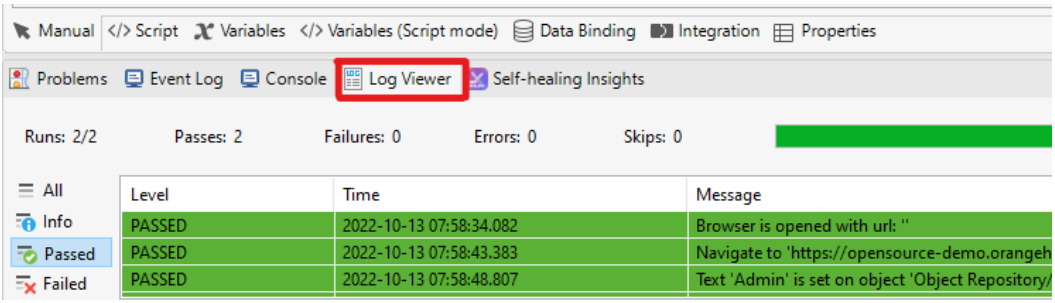
5.1 TEST EXECUTION REPORTS

Didalam katalon studio terdapat dua fitur yang menyajikan hasil eksekusi dari setiap test yang dijalankan.

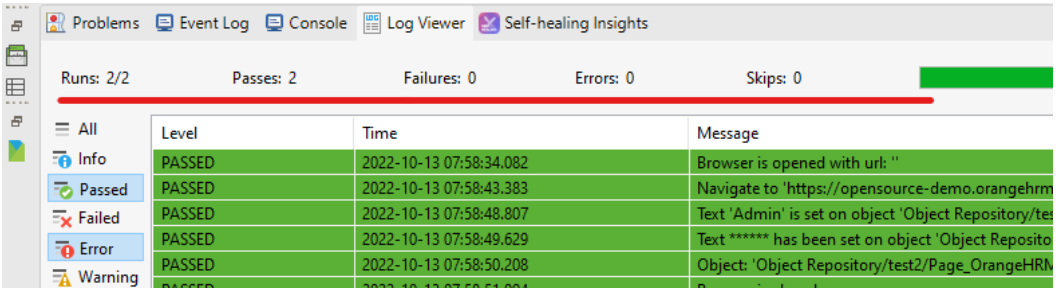
- Log : untuk menyajikan hasil eksekusi dari test case, test suite, dan test suite collection
- Report : untuk menyajikan hasil eksekusi dari test suite dan test suite collection

5.1.1. Log Viewer

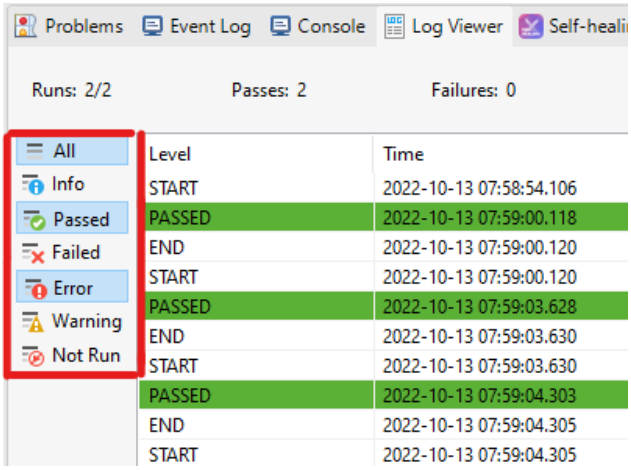
1. Kita dapat menggunakan tools log viewer untuk melihat log dari setiap test case.



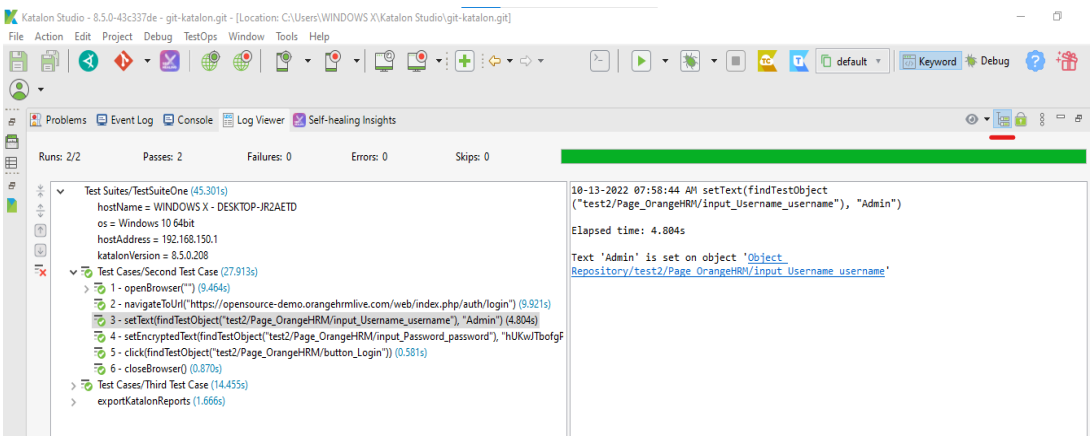
2. Dalam jendela log viewer kita dapat melihat tes yang dijalankan, tes yang lulus uji, kesalahan, error, dan test yang dilewati.



3. Jika ingin memfilter log berdasarkan hasil kita dapat menggunakan menu pada jendela log sebelah kiri



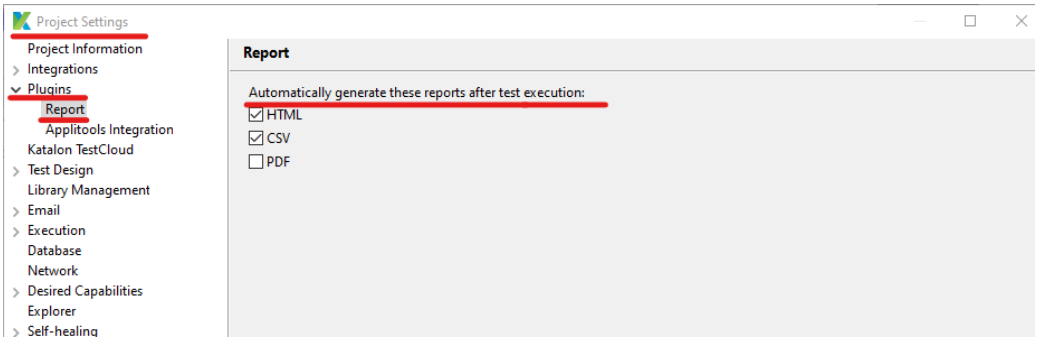
4. Katalon juga menyediakan tampilan hirarki yang didalamnya terdapat jendela log dan jendela detail dari setiap tahapan pengujian



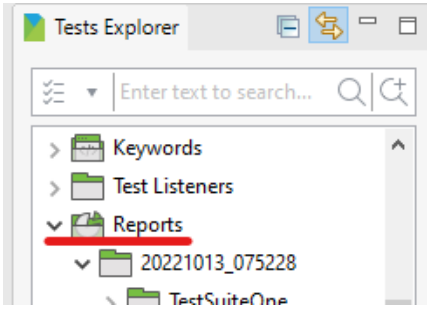
5.1.2. Report

1. Dalam katalon studio kita dapat mengatur format report yang digunakan untuk menyajikan report melalui menu project > setting > plugins > report

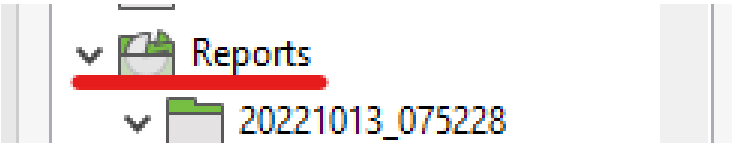
Ada tiga jenis format report yang disajikan yaitu html, csv, dan pdf.



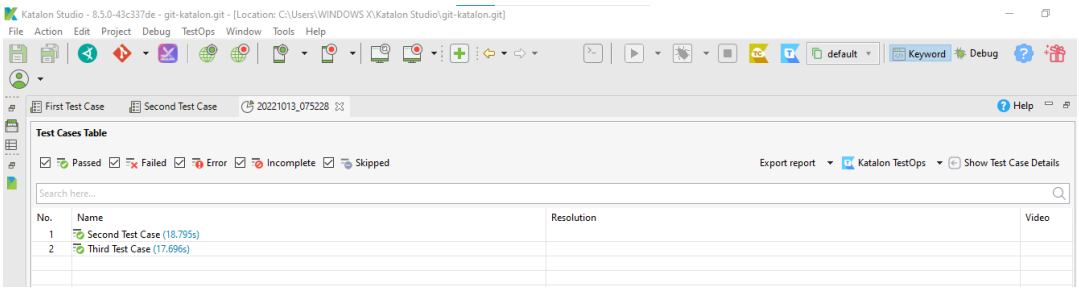
2. Untuk melihat report kita dapat menuju jendela explorer dan memilih menu report



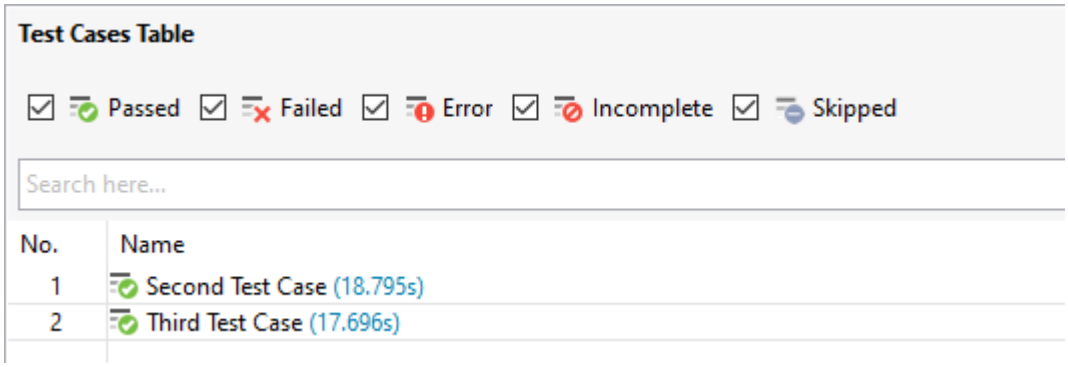
3. Report yang disajikan sesuai dengan jumlah eksekusi yang dilakukan terhadap test suite ataupun test suite collection. Penamaan report secara default diawali oleh tahun, bulan, dan tanggal atau sesuai dengan waktu eksekusi



4. Berikut ini adalah tampilan jendela report secara keseluruhan.

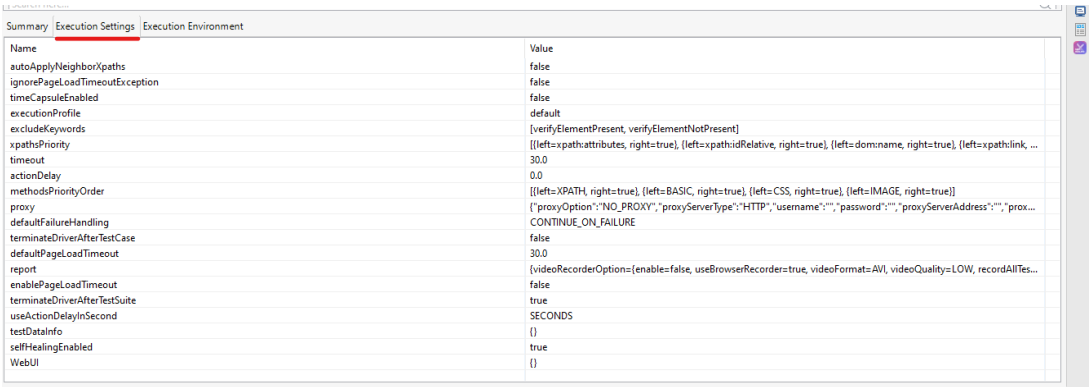


5. Terdapat test case table yang akan menampilkan test case yang diuji beserta dengan status kelulusan, kita juga dapat memfilter tampilan dengan menggunakan checkbox di bagian atas jendela test case table.

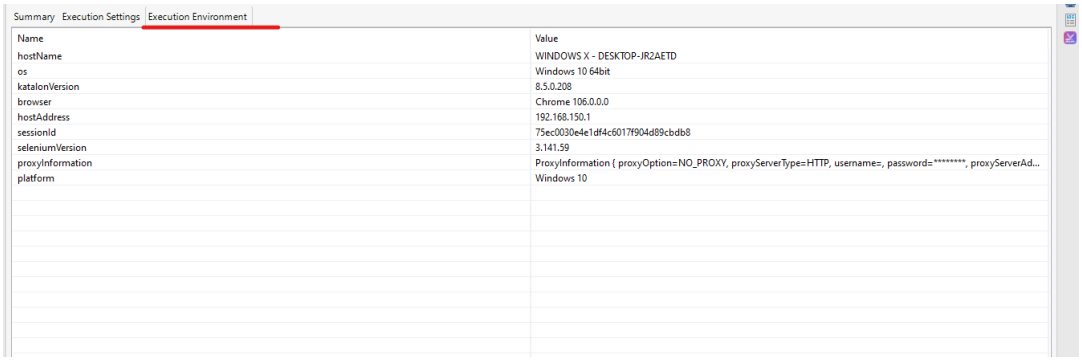


6. Di bagian bawah jendela report juga terdapat ringkasan pengujian yang terletak pada tab summary.

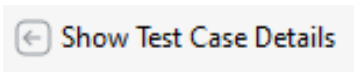
7. Kita juga dapat melihat seluruh setting saat eksekusi melalui tab Execution setting



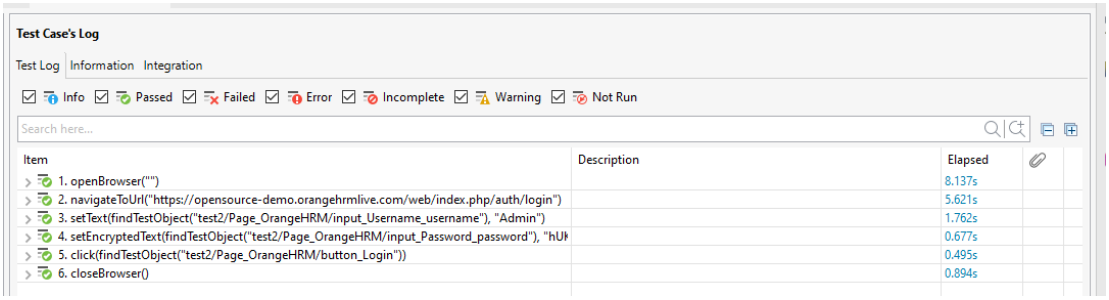
8. Kita juga dapat melihat lingkungan pengujian melalui tab Execution Environtment



9. Untuk melihat tampilan detail dari pengujian kita dapat menggunakan menu test case detail

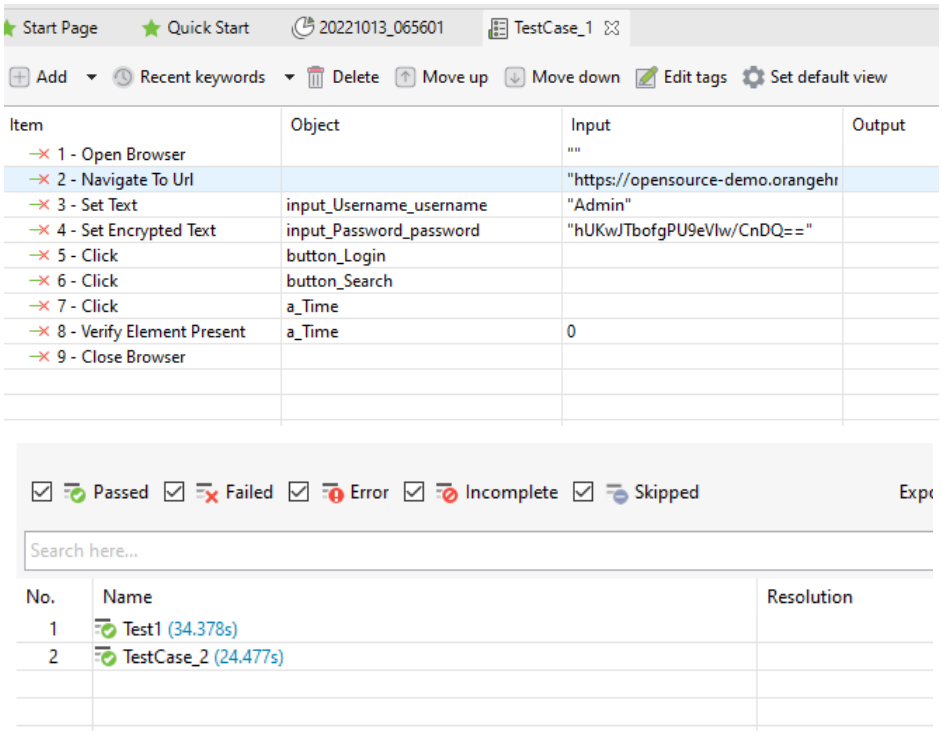


10. Berikut adalah tampilan dari detail setiap pengujian, yang didalamnya terdapat tahapan pengujian, deskripsi, dan waktu pengujian.

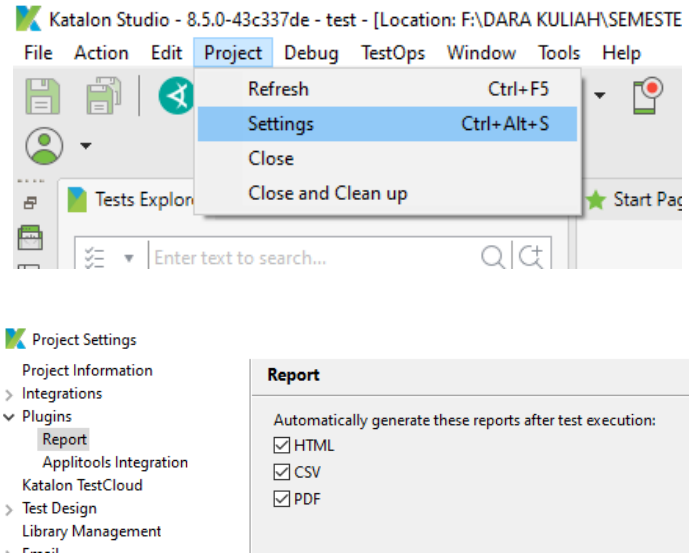


5.2 BASIC REPORTS PLUGINS

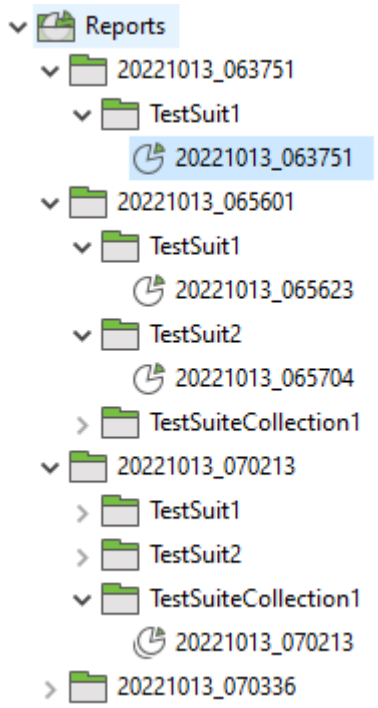
1. Pada tahapan sebelumnya kita sudah banyak membuat test case, test suit, maupun test suit collection pada katalon. Salah satunya test case seperti yang terlihat pada gambar di bawah ini.



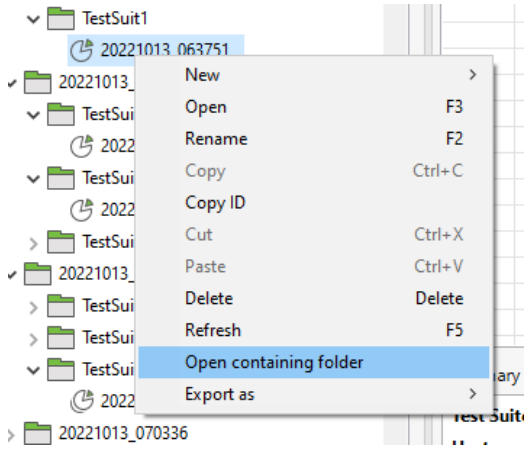
2. Sebelum kita mulai mengeksport laporan hasil pengujian dalam format HTML, CSV, atay PDF, terlebih dahulu kita melakukan setting pada menu projek, dimana untuk semua plugins report ceklis untuk semua format pilihan yang ada.



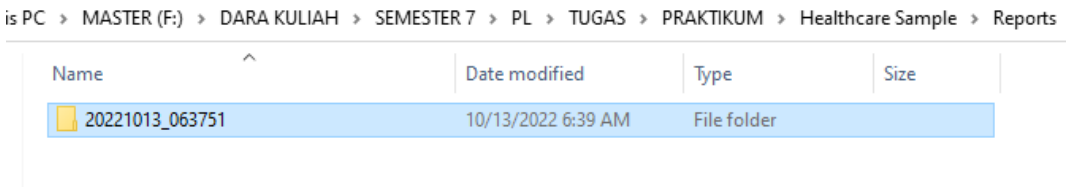
3. Setelah itu pada menu reports, dapat kita lihat berbagai pengujian yang telah kita buat. Untuk kasus ini kita melakukan export terhadap pengujian test case yang sudah dilakukan penambahan pada test suit1.



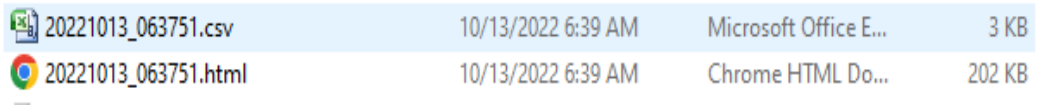
4. Untuk melihat hasil report dalam format HTML, klik kanan pada report kemudian pilih **open containing folder**.



5. Selanjutnya user akan di arahkan ke halaman lokasi tempat penyimpanan folder.



6. Ketika folder dibuka, user dapat melihat hasil report pengujian sudah ada dalam format HTML dan CSV, ini dapat terjadi karena sebelumnya kita telah melakukan setting terhadap report plugin untuk mengaktifkan ceklis terhadap berbagai pilihan format plugin yang tersedia.



7. Berikut report hasil pengujian dalam format HTML dan CSV.

TestSuit1 Test Log

Execution Environment

Host name: ACER - DARA-MELISA-PC
Local OS: Windows 10 64bit
Katalon version: 8.5.0.208
Browser: Chrome 106.0.0.0

Test Execution Log

TEST SUITE: **TestSuit1**

Full Name: TestSuit1
Start / End / Elapsed: 2022-10-13 06:38:26.583 / 2022-10-13 06:39:26.409 / 00:00:59.826
Status: 2 test total, 2 passed, **0 failed**, **0 error**, 0 incomplete, 0 skipped

TEST CASE: **Test Cases/Test1**

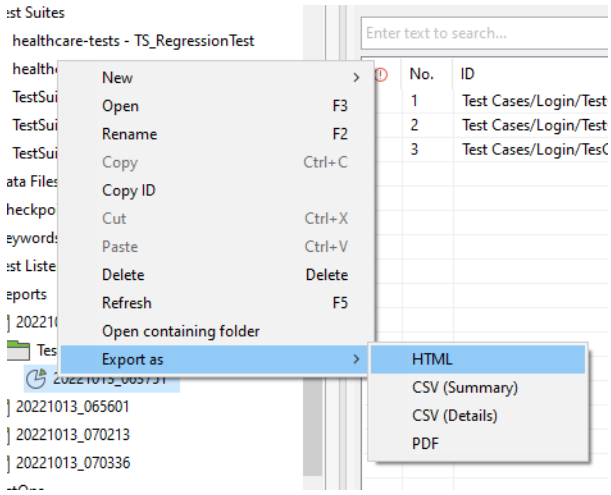
Full Name: TestSuit1/Test Cases/Test1
Start / End / Elapsed: 2022-10-13 06:38:27.534 / 2022-10-13 06:39:01.912 / 00:00:34.378
Status: PASSED

TEST STEP: *openBrowser("")*
TEST STEP: *navigateToUrl("https://opensource-demo.orangehrmlive.com/web/index.php/auth/login")*
TEST STEP: *setText(findTestObject("Object Repository/Test2/Page_OrangeHRM/input_Username_username"), "Admin")*
TEST STEP: *setEncryptedText(findTestObject("Object Repository/Test2/Page_OrangeHRM/input_Password_password"), "hUKwJTbofgPU9eVlw/CnDQ==")*
TEST STEP: *click(findTestObject("Object Repository/Test2/Page_OrangeHRM/button_Login"))*
TEST STEP: *click(findTestObject("Object Repository/Test2/Page_OrangeHRM/button_Search"))*
TEST STEP: *click(findTestObject("Object Repository/Test2/Page_OrangeHRM/a_Time"))*
TEST STEP: *verifyElementPresent(findTestObject("Object Repository/Test2/Page_OrangeHRM/a_Time"), 0)*
TEST STEP: *closeBrowser()*

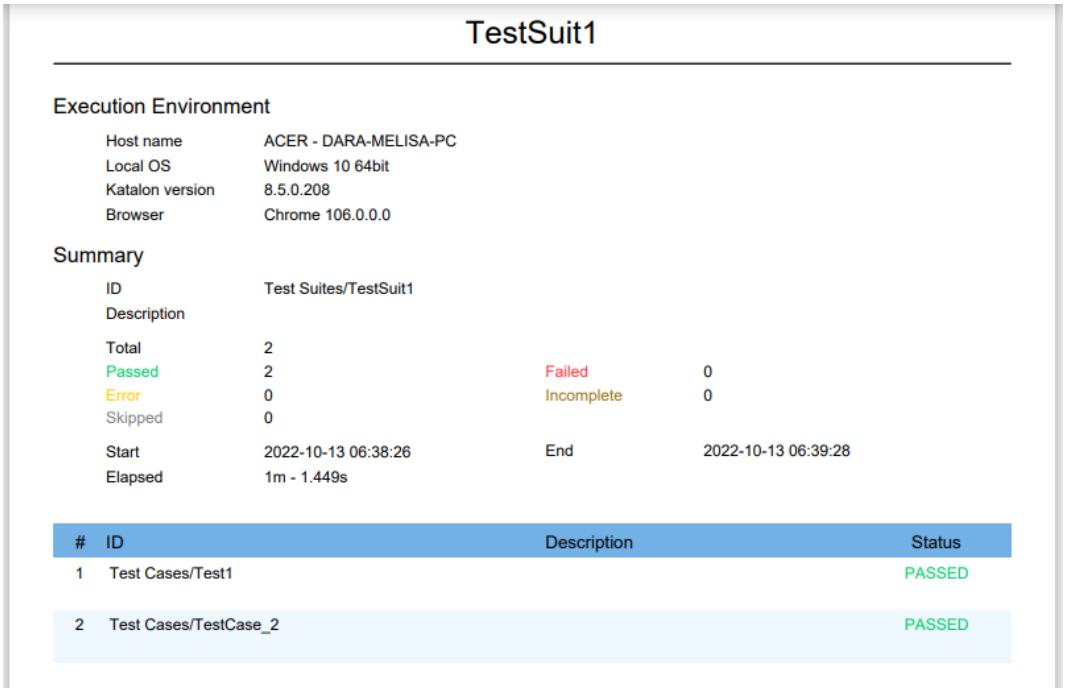
A1 — Suite/Test/Step Name

	A	B	C	D	E	F	G	H	I	J	K
1	Suite/Tes	Browser	Descriptic	Tag	Start time	End time	Duration	Status			
2	TestSuit1	Chrome 106.0.0.0			########	########	59.826s	PASSED			
3											
4	Test Case	Chrome 106.0.0.0			########	########	34.378s	PASSED			
5	openBrow	Chrome 106.0.0.0			########	########	10.253s	PASSED			
6	navigateT	Chrome 106.0.0.0			########	########	6.128s	PASSED			
7	setText(fi	Chrome 106.0.0.0			########	########	2.376s	PASSED			
8	setEncryp	Chrome 106.0.0.0			########	########	0.860s	PASSED			
9	click(findT	Chrome 106.0.0.0			########	########	0.730s	PASSED			
10	click(findT	Chrome 106.0.0.0			########	########	5.336s	PASSED			
11	click(findT	Chrome 106.0.0.0			########	########	3.779s	PASSED			
12	verifyElen	Chrome 106.0.0.0			########	########	0.434s	PASSED			
13	closeBrow	Chrome 106.0.0.0			########	########	0.947s	PASSED			
14	Video	Chrome 106.0.0.0			########	########	0.003s	PASSED			
15											
16	Test Case	Chrome 106.0.0.0			########	########	24.477s	PASSED			
17	openBrow	Chrome 106.0.0.0			########	########	8.099s	PASSED			
18	setText(fi	Chrome 106.0.0.0			########	########	2.614s	PASSED			
19	setText(fi	Chrome 106.0.0.0			########	########	0.984s	PASSED			
20	click(findT	Chrome 106.0.0.0			########	########	0.517s	PASSED			
21	click(findT	Chrome 106.0.0.0			########	########	5.388s	PASSED			
22	click(findT	Chrome 106.0.0.0			########	########	2.760s	PASSED			
23	closeBrow	Chrome 106.0.0.0			########	########	0.940s	PASSED			
24	Video	Chrome 106.0.0.0			########	########	0.003s	PASSED			
25											

8. Untuk melihat report hasil pengujian dalam format PDF, anda bisa klik kanan pada report, kemudian pilih **exports as**, lalu pilih format pdf.



Pilih tempat penyimpanan file, kemudian save. Berikut reprt hasil pengujian dalam format PDF.



TestSuit1

Execution Environment

Host name	ACER - DARA-MELISA-PC
Local OS	Windows 10 64bit
Katalon version	8.5.0.208
Browser	Chrome 106.0.0.0

Summary

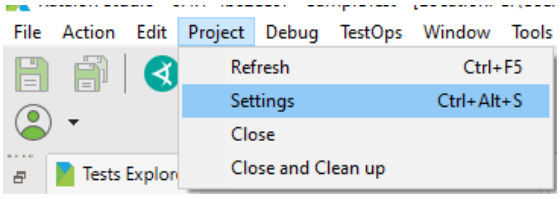
ID	Test Suites/TestSuit1		
Description			
Total	2		
Passed	2	Failed	0
Error	0	Incomplete	0
Skipped	0		
Start	2022-10-13 06:38:26	End	2022-10-13 06:39:28
Elapsed	1m - 1.449s		

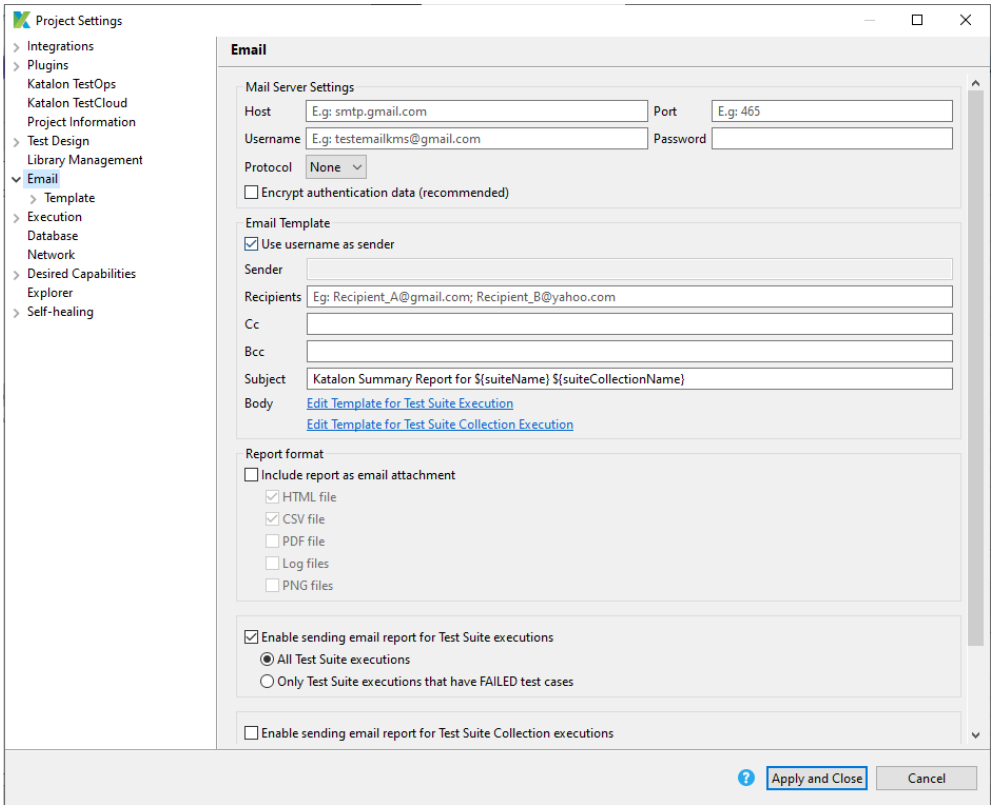
#	ID	Description	Status
1	Test Cases/Test1		PASSED
2	Test Cases/TestCase_2		PASSED

5.3 HOW TO EMAIL RESULTS

Setelah melakukan pengujian perangkat lunak, terkadang seorang dari tim ingin hasil pengujian tersebut dikirim ke email secara otomatis agar menghemat waktu dalam kolaborasi sesama tim. Adapun langkah-langkah yang diperlukan untuk mengirim hasil pengujian ke email sebagai berikut :

1. Bukan bagian Project>Settings>Email

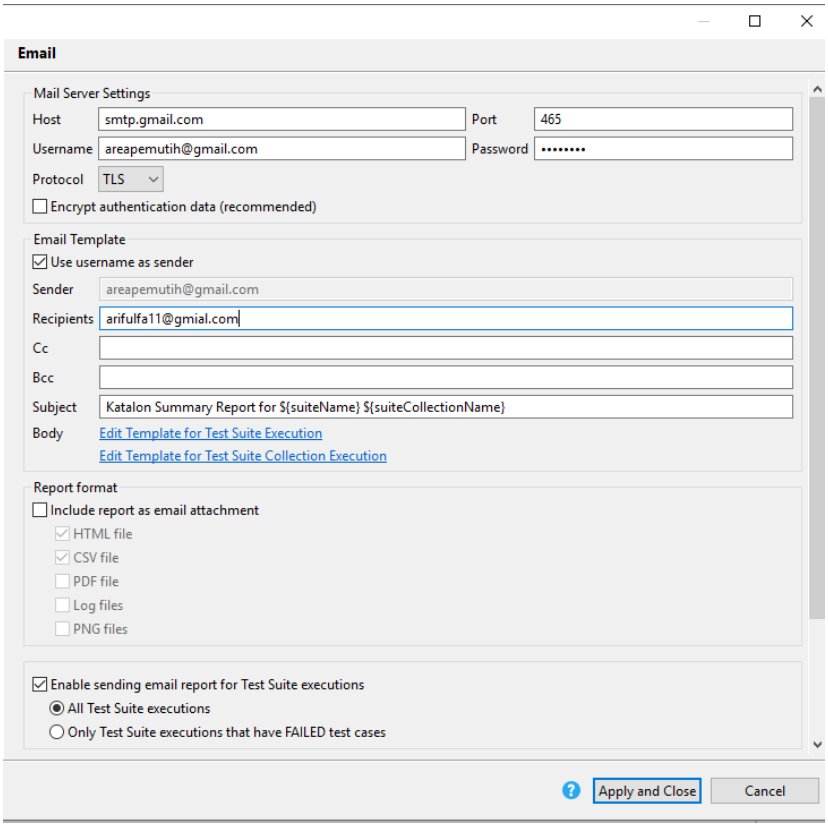


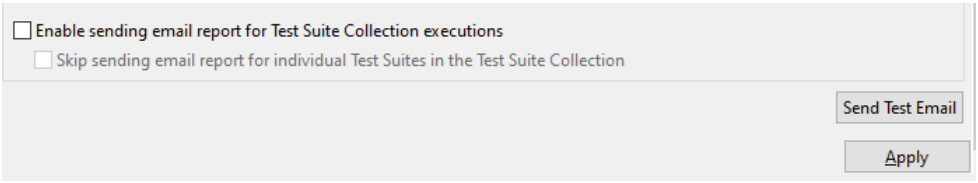


2. Sebelum mengisi kolom email, pahami dulu yang dimaksud dengan *Mail Server Settings*

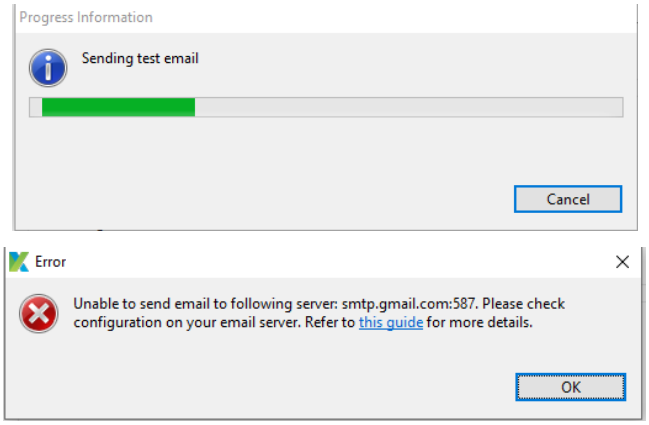
Email sever	Host	Port	Reference
Gmail	`smtp.gmail.com`	465 or 587	Check Gmail through other email platforms
Outlook	`smtp.office365.com`	587 or 25	How to set up a multifunction device or application to send email using Microsoft 365 or Office 365
Yahoo! Mail	`smtp.mail.yahoo.com`	465	POP access settings and instructions for Yahoo Mail

3. Setelah memahami isi dari *Mail Server Settings.* Selanjutnya isikan pada *email* seperti berikut ini, sesuaikan dengan email anda. Kemudian tekan *Send Test Email*, *email* akan dikirim kepada *email* yang terdapat pada kolom *Recipient*.



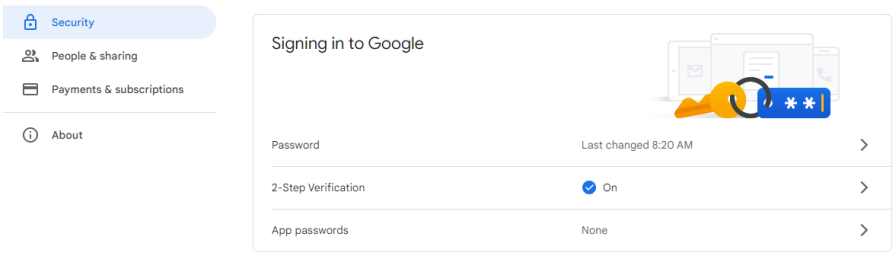


4. Tunggu proses berjalan, dan jika terjadi *error*, ikuti langkah selanjutnya

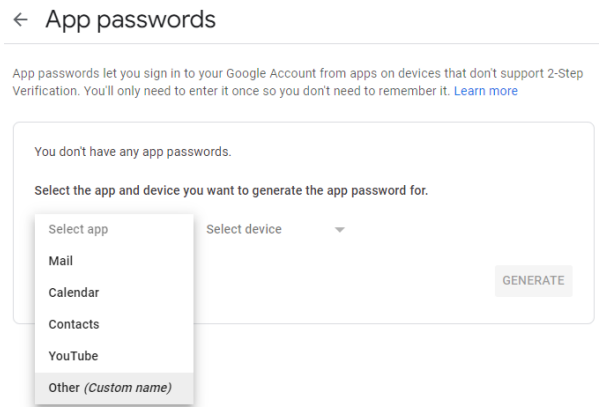


5. Adapun *error* terjadi dikarenakan kita menggunakan *protocol smtp* untuk mengirim *email*, dan *protocol* ini adalah sebuah *protocol* pihak ketiga. Oleh karena itu, untuk membuat *email* dapat terkirim dengan berhasil, lakukan langkah berikut untuk memberikan akses *email* kepada pihak ketiga dengan cara pilih akun gmail anda, kemudian pilih *Manage your Google Account*

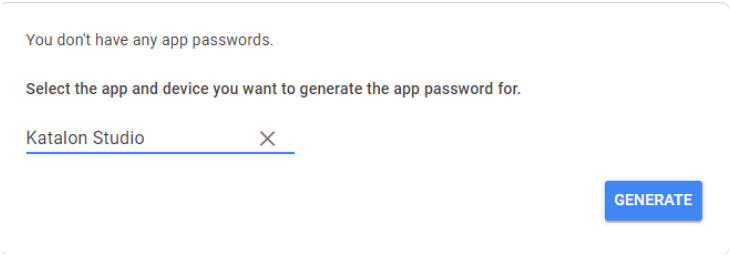
6. Masuk ke bagian menu *Security*, kemudian *scroll* pada bagian *Signin in to Google*, pada bagian tersebut aktifkan *2-Step Verification* dengan cara menginput nomor *handphone* yang *valid*, setelah berhasil, maka akan terdapat menu *App passwords*, buka menu tersebut



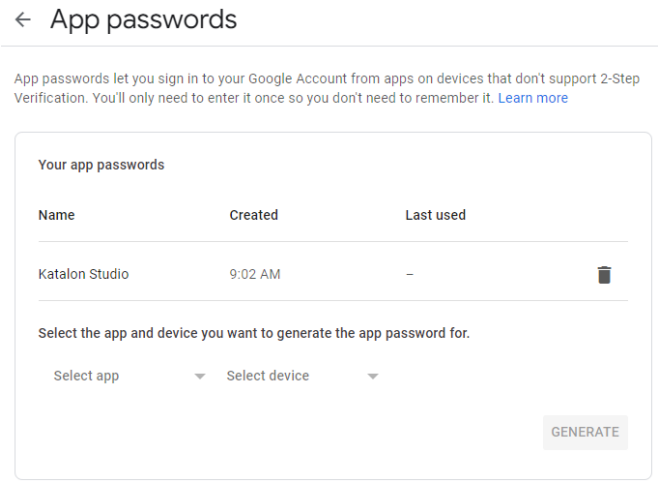
7. Pada bagian *Select App*, pilih *Other (Custom name)*



8. Isikan dengan Katalon Studio atau nama yang anda inginkan. Kemudian klik *Generate*



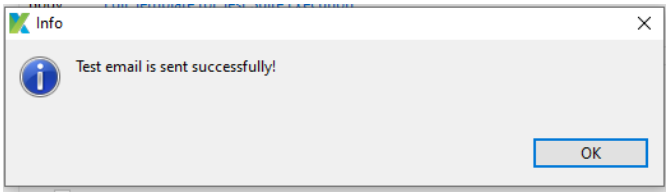
9. Kemudian anda akan diberikan *password* otomatis, kemudian klik *DONE*
10. Secara otomatis muncul daftar *App Password*



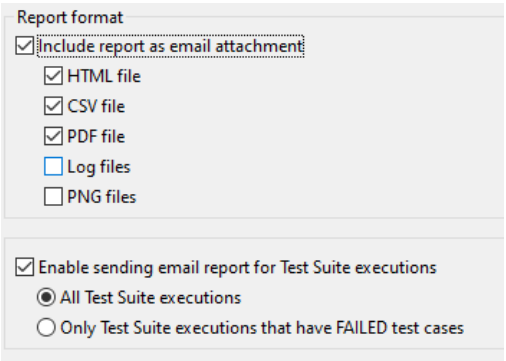
11. *Copy*-kan *password* sebelumnya, dan simpan pada bagian kolom *Password*, kemudian lakukan *Send Test Email* ulang.

    Jika gagal menggunakan *port* dan *protocol* di atas, coba ganti ke *port* dan *protocol* seperti gambar di bawah

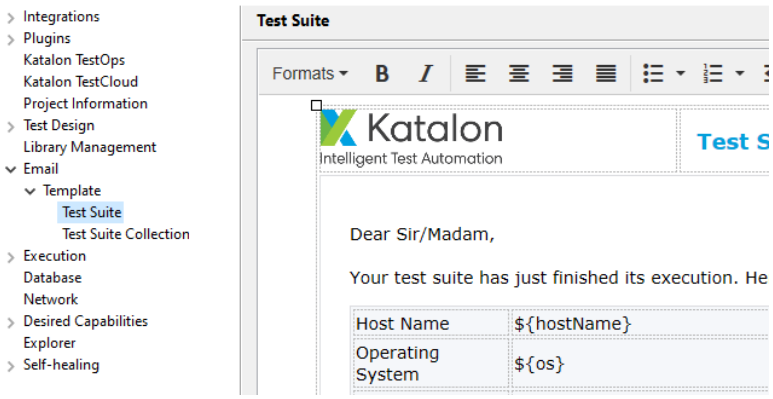
12. Maka *email* akan berhasil dikirimkan. Jika gagal, ganti *Port* yang lainnya



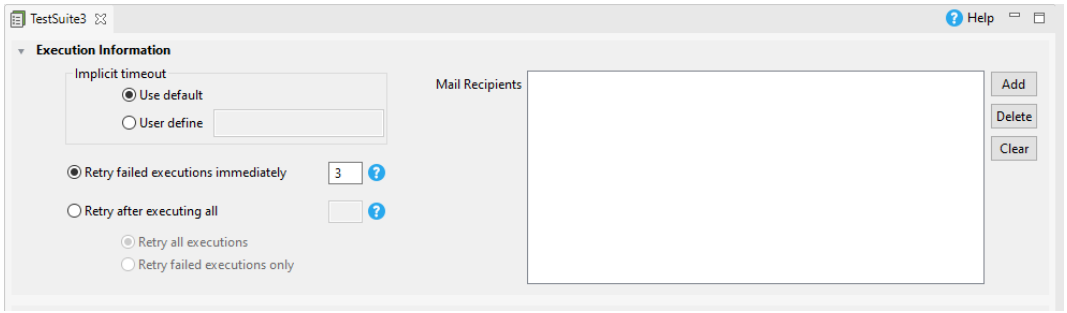
13. Jika tidak terdapat pada Kotak Masuk, kemungkinan akan dianggap sebagai *spam*
14. Langkah berikutnya, atur pada bagian *Report format* dengan berikut, dan klik *Apply* setelah melakukan konfigurasi



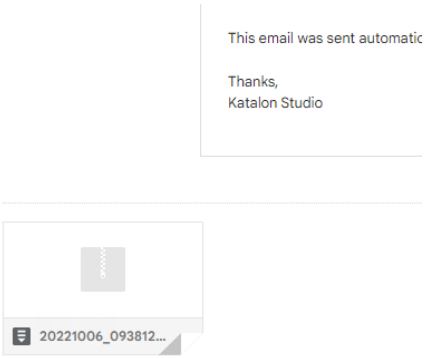
15. Adapun untuk mengubah *template* dari *email*, anda dapat melakukannya pada bagian Email>Template>Test Suite atau Test Suite Collection.



16. Setelah melakkan konfigurasi *email*, selanjutnya buka file *Test Suite* anda, lakukan *expand* terhadap *Execution Information*. Pada jendela tersebut terdapat informasi mengenail *Mail Recipients*. Masukkan daftar nama *email* yang akan menerima pemberitahuan mengenai tes yang dilakukan



Jika sebelumnya anda mencentang bagian *include report as email attachment*, maka pada notifikasi *email* anda akan mendapat file berupa seperti .csv, .html, .pdf, dan lain sebagainya.

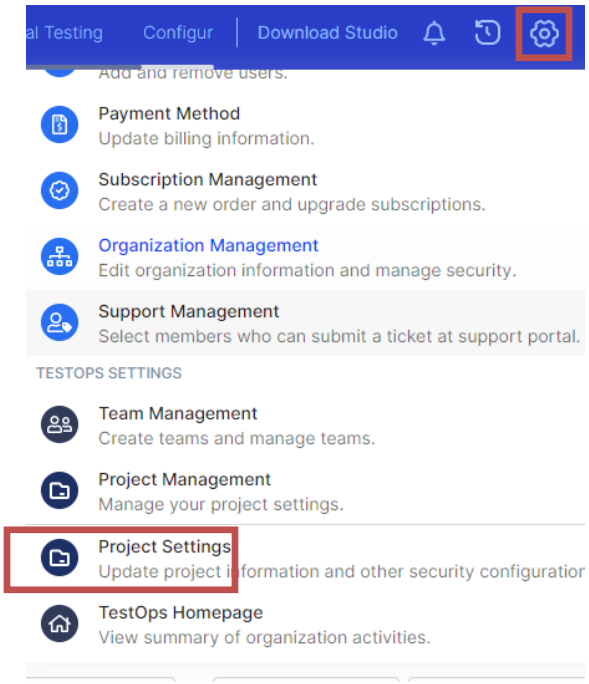


Setelah mempelajari modul ini, mahasiswa diharapkan sudah dapat melakukan pengujian beserta mengirim *email* hasil pengujian dengan berhasil.

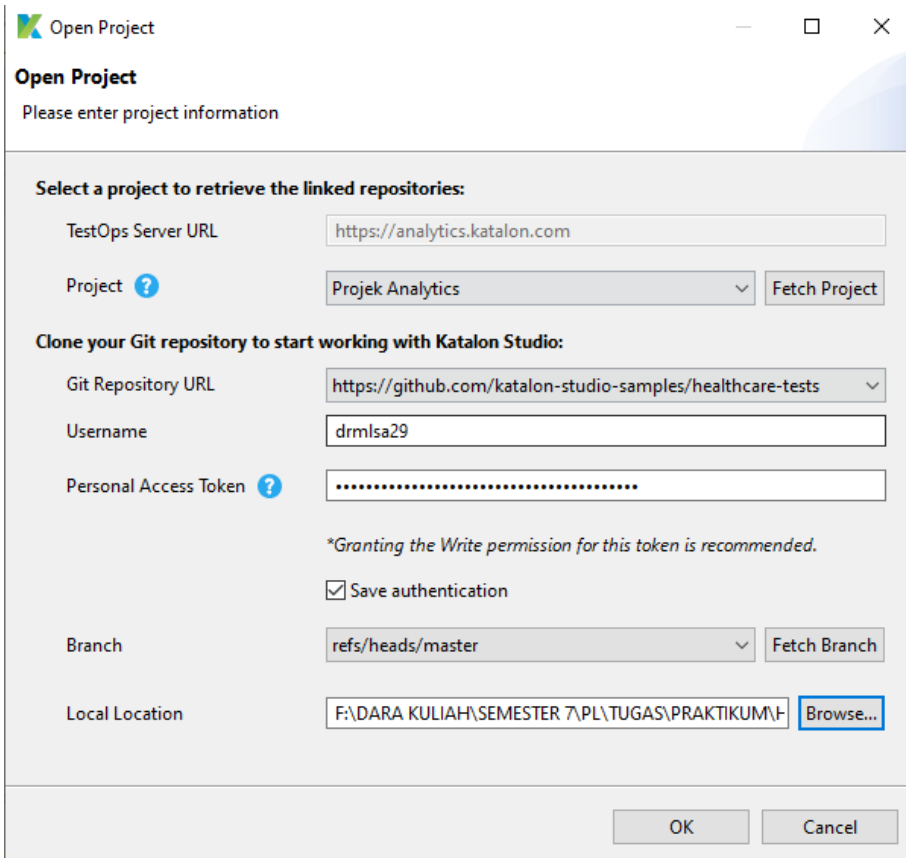
5.4 HOW TO USE KATALON ANALYTICS STEP BY STEP

Pada modul ini kita akan mempelajari bagaimana cara menggunakan katalon analytics. Katalon analytic merupakan suatu proses yang berfungsi untuk melihat, menyimpan, dan menganalisis hasil atau laporan dari pengujian yang dilakukan pada katalon. Berikut ini merupakan tahapan menggunakan katalon analytics.

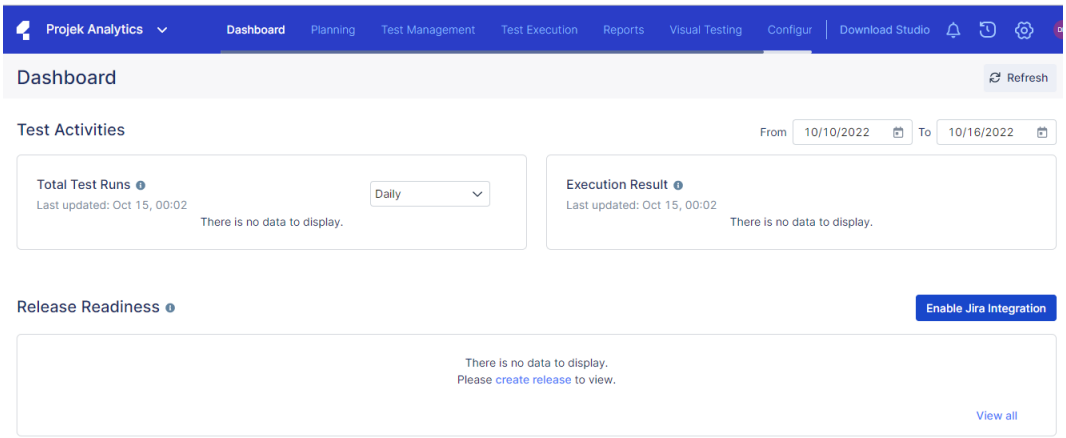
1. Langkah pertama yang kita lakukan adalah, kita bisa membuat projek baru untuk katalon. Projek baru dapat dibuat dengan masuk ke web katalon, kemudian klik setting, lalu pilih new projek, dan projek baru pun telah dibuat.



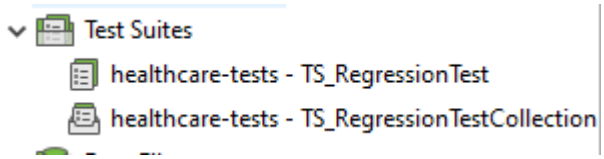
2. Selanjutnya, buka aplikasi katalon kemudian klik file lalu open projek. Masukkan username dan token dari akun github anda, lalu pada bagian projek pilih nama projek yang sudah dibuat pada web katalon analytics sebelumnya, pada kasus ini saya membuat projek dengan nama projek analytics. Pada proses ini kita sudah berhasil menghubungkan langsung test projek yang kita lakukan di aplikasi katalon ke web katalon analytics.



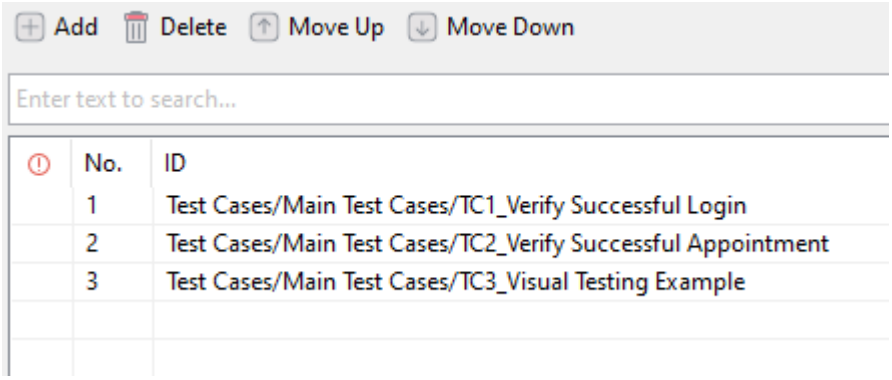
3. Jika kita lihat pada halaman test activities pada web katalon analytics, belum ada test activities apapun yang terekam karena memang belum ada pengujian yang dilakukan.



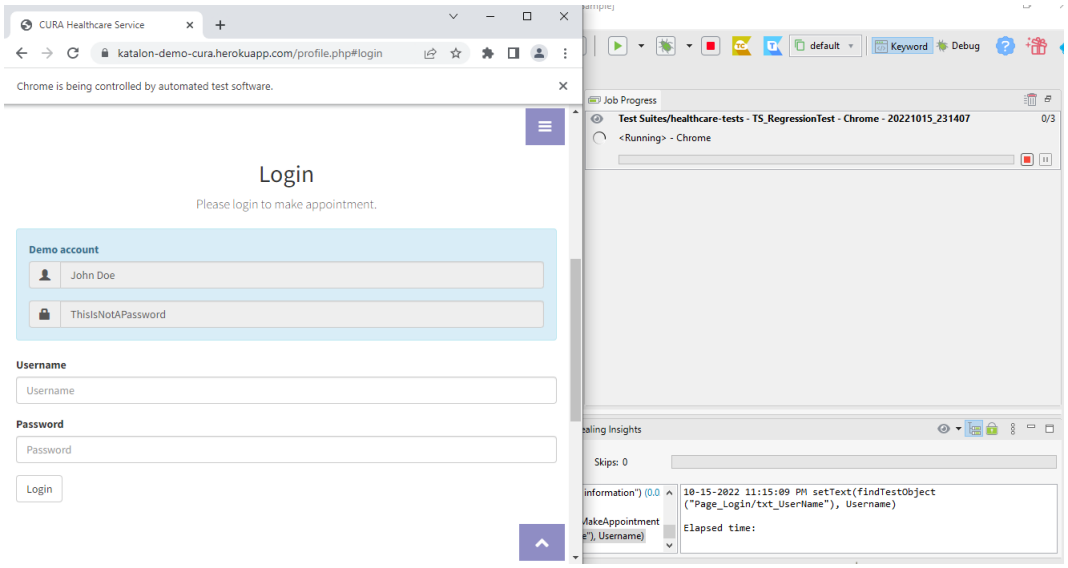
4. Selanjutnya kita akan menjalankan test suit sebagai bentuk percobaannya. Test suit ini merupakan test case yang sudah disedikan oleh katalon sebagi sampel pengujian. Untuk kasus ini saya memilih test case TR_RegressioonTest.

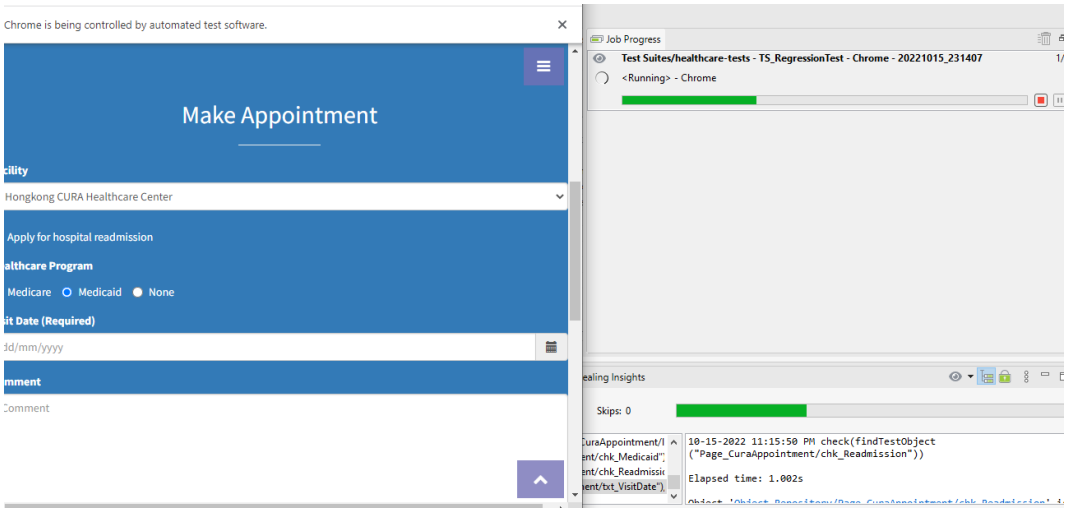


Seperti yang dapat dilihat pada gambar di bawah ini, pada test suit tersebut sudah ditambahkan beberapa test case.

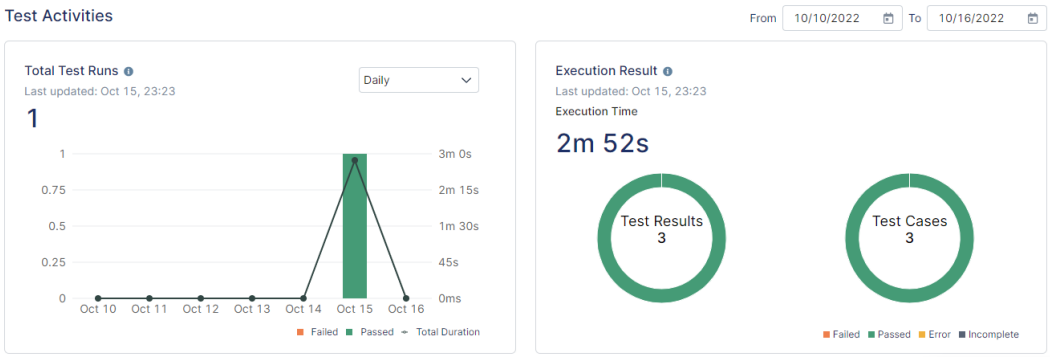


5. Jalankan test suit dengan bwowser pilihan anda, dan tunggu sampai proses eksekusi selesai. Berikut ini gambar tampilan test suit yang sedang dieksekusi.





6. Setelah proses eksekusi selesai, kita bisa melihat hasil report pengujian pada web alytics katalon. Terlihat pada test activities sudah ada 1 proses pengujian yang sudah silakukan.



Dapat dilihat seperti pada gambar di atas, hasil detail report menunjukkan 1 passed, dan 0 error untuk pengujian yang dilakukan.

MODUL 9
API TESTING

A. Tujuan

1. Mengetahui apa itu API
2. Mengetahui bagaimana cara API Testing pada katalon
3. Mengetahui bagaimana cara mengirim nilai dari satu api ke api lainnya dengan xml.
4. Mengetahui bagaimana cara mengirim nilai dari satu api ke api lainnya dengan json.

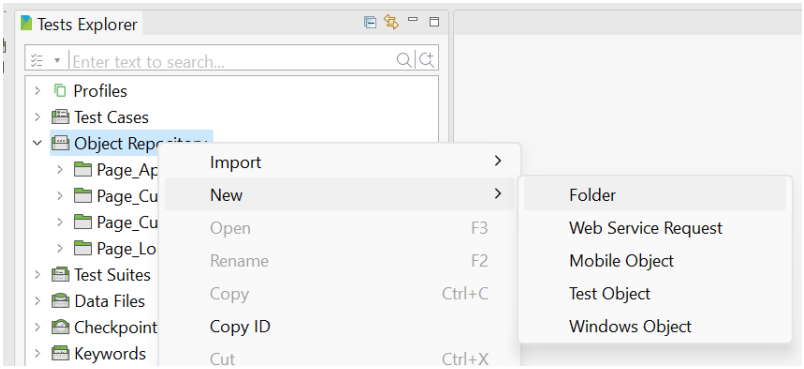
B. Dasar Teori

Api merupakan aplikasi yang memungkinkan para developer untuk mengintegrasikan bagian aplikasi dengan bagian aplikasi lainnya.

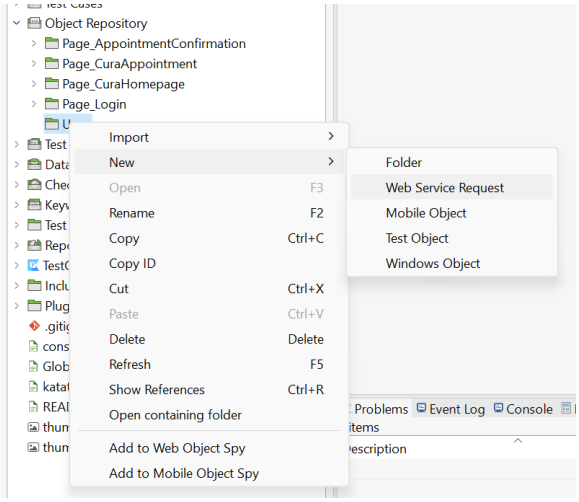
9.1. HOW TO DO API TESTING WITH KATALON

Katalon juga dapat digunakan sebagai aplikasi untuk melakukan pengujian terhadap API. Berikut langkah-langkah yang dapat dilakukan dalam menguji API pada katalon.

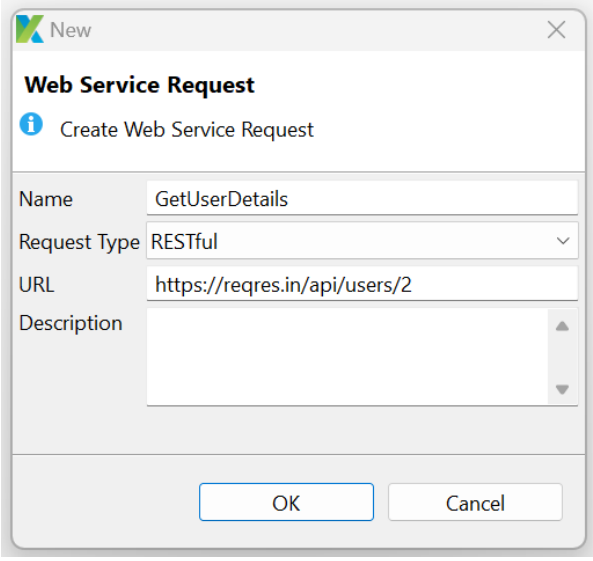
1. Buka project pada katalon, pada bagian Test Explorer klik kanan pada folder Object Respository untuk membuat folder baru.



2. Setelah folder berhasil dibuat klik kanan, pilih web service Request

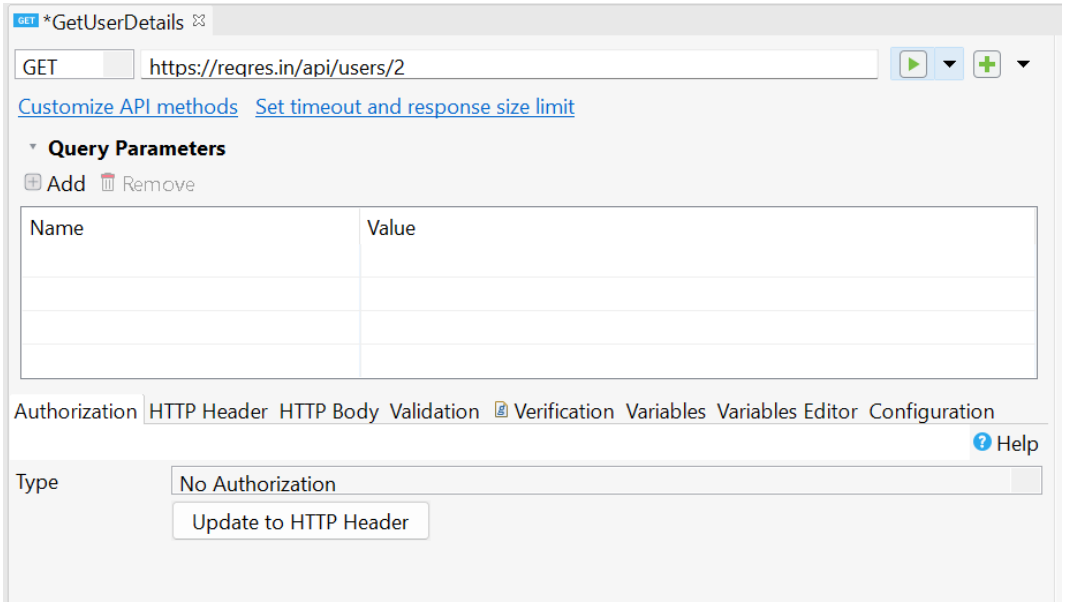


3. Kemudian isikan name untuk file yang akan dibuat, pilih RESTful sebagai request type, dan masukkan URL API, klik Ok.

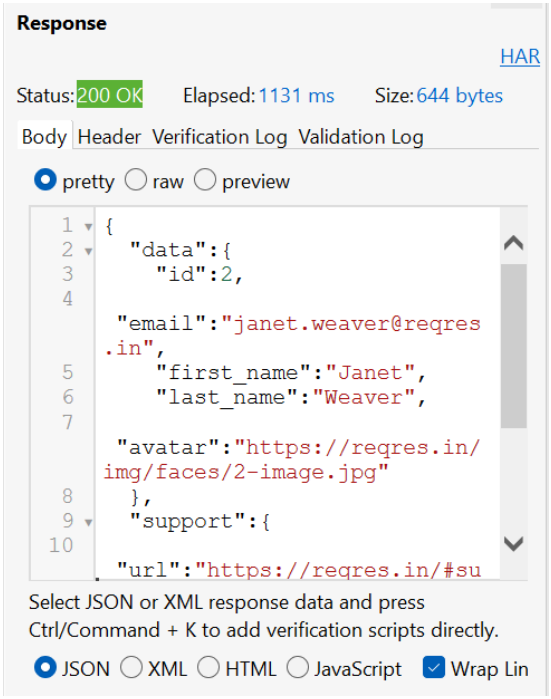


4. URL API tersebut jika dibuka melalui web akan tampil seperti berikut

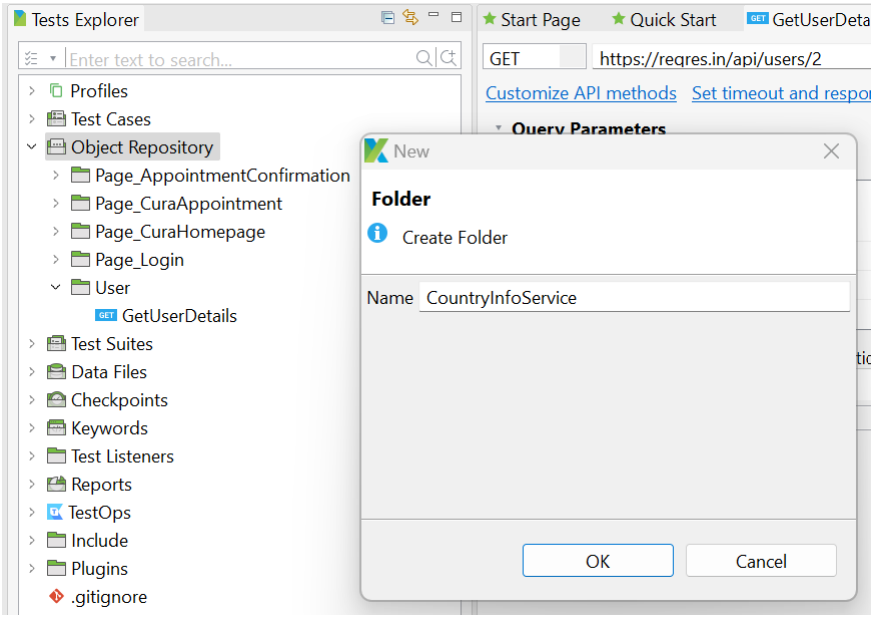
5. Berikut tampilan file yang dibuat, selanjutnya klik icon play.



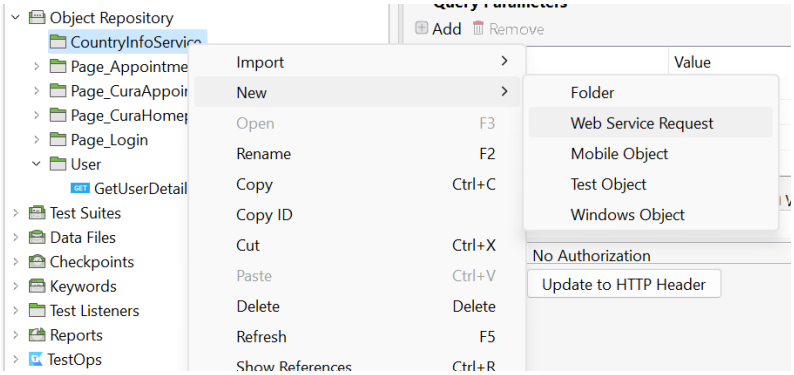
6. Berikut tampilaan hasilnya, yaitu GET user dengan id 2 berhasil dilakukan, ditandai dari Responsenya yaitu status 200 OK



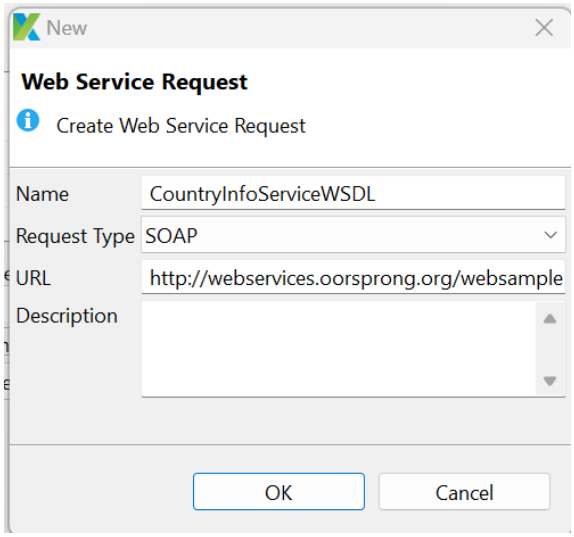
7. Selanjutnya buat folder baru dengan nama “CountryInfoService” pada Object Repository



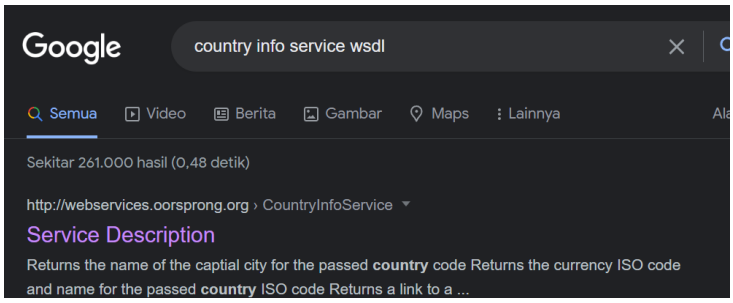
8. Klik kanan pada folder “CountryInfoService”, pilih new dan klik Web Service Request



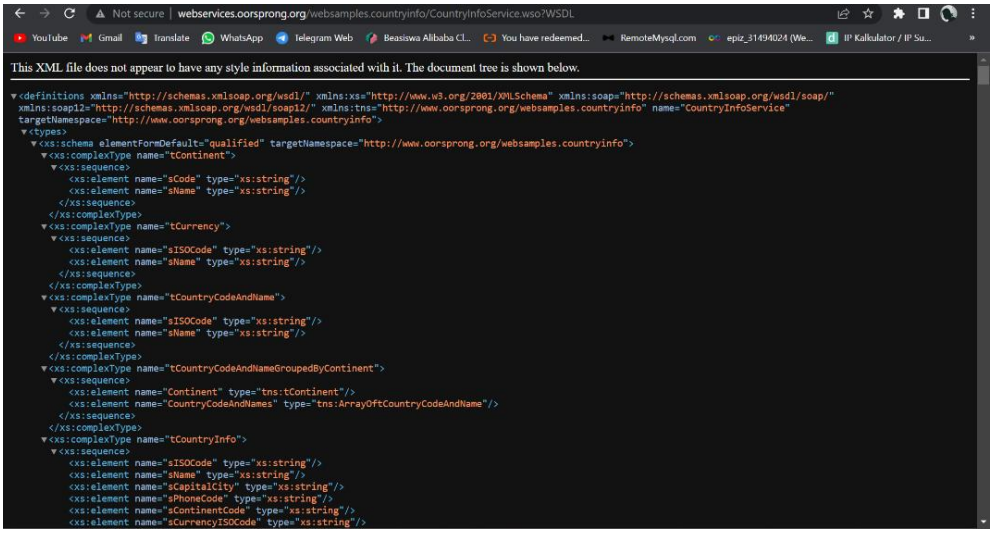
9. Kemudian isikan name sebagai berikut, pilih SOAP sebagai Request Type dan tempelkan url berikut.



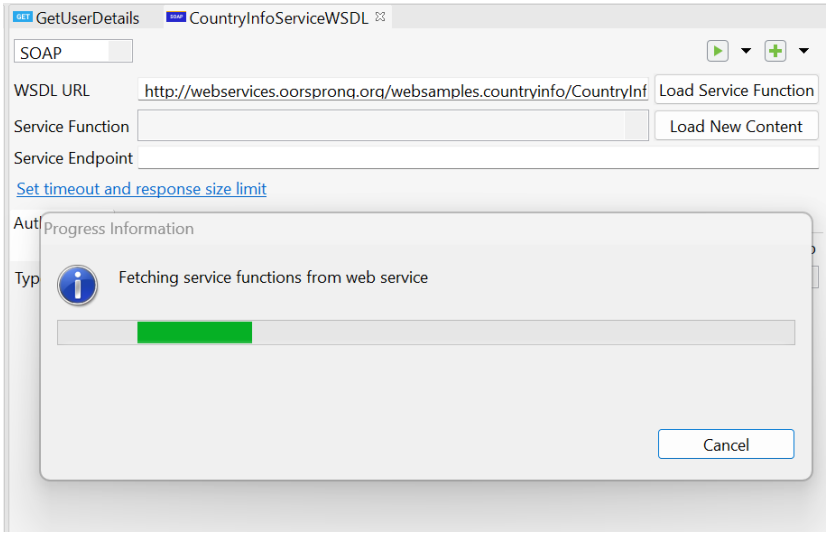
10. Untuk mendapatkan URL diatas dapat di searching di google, buka website berikut.



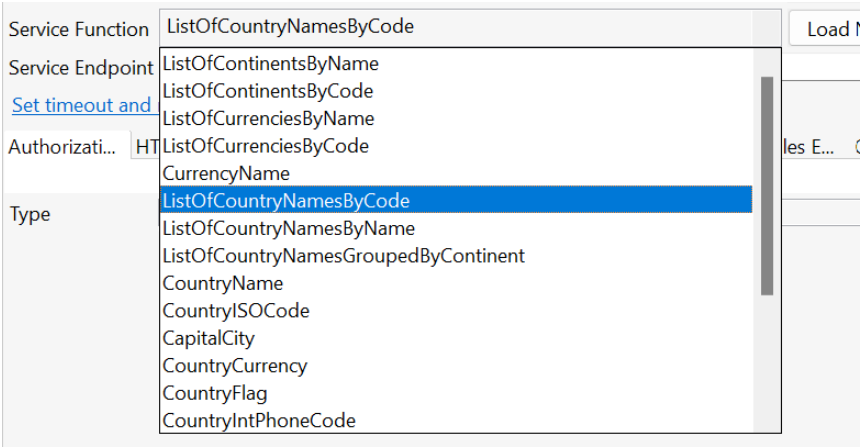
11. Berikut tampilan website, lalu copy link addressnya.



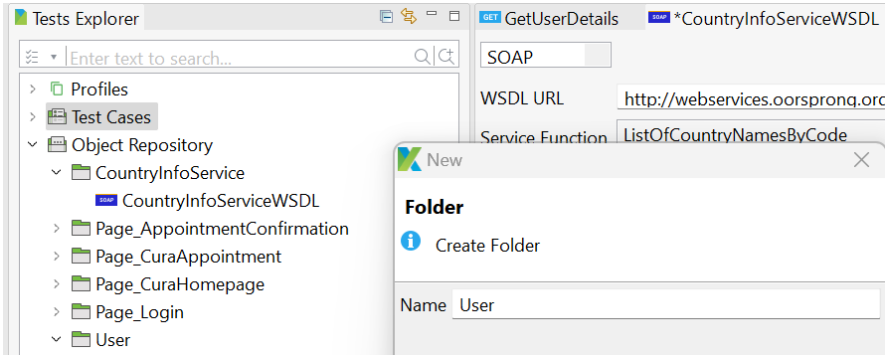
12. Berikut Tampilan File yang telah kita buat, klik “Load Service Function”.



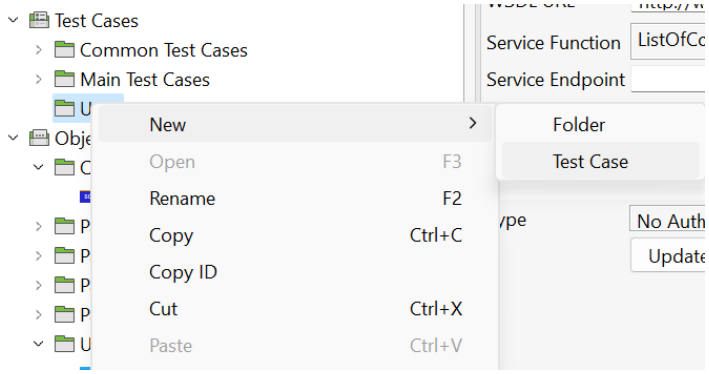
13. Pada Service Function pilih “ListOfCountryNamesByCode”



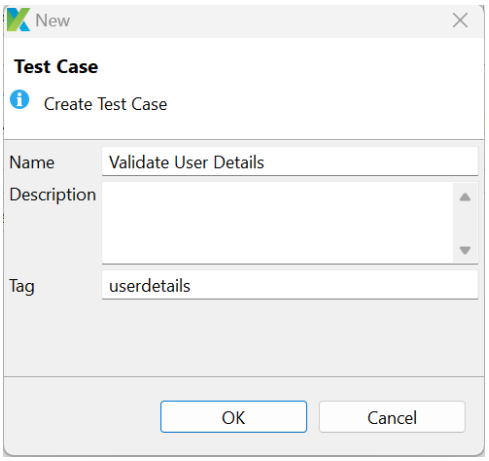
14. Langkah selanjutnya pada Test Explorer, buat folder “User” didalam Test Case



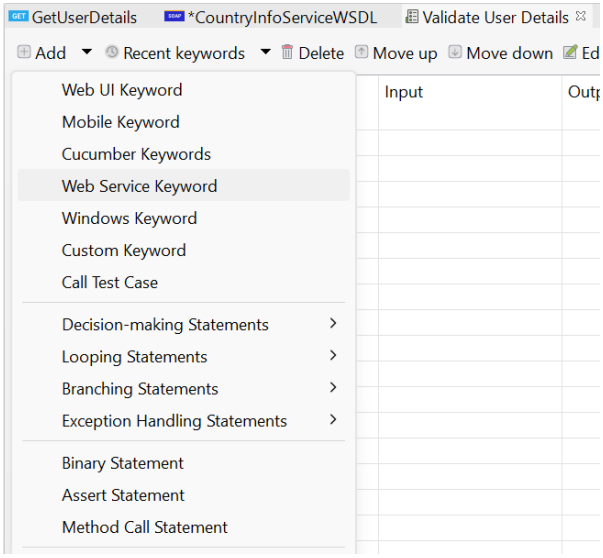
15. Pada folder “User” klik kanan pilih New dan klik Test Case



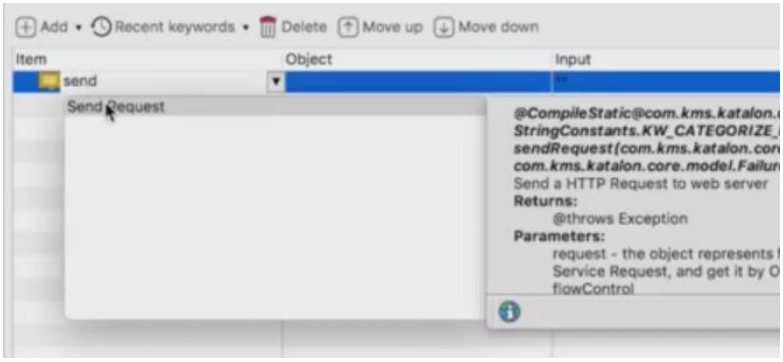
16. Isikan form seperti berikut, klik Ok



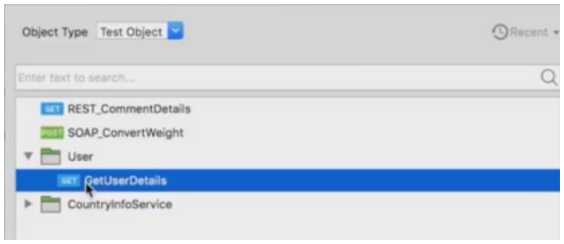
17. Selanjutnya klik dropdown Add pilih Web Service Keyword



18. Gantikan Namanya menjadi “Send Request”



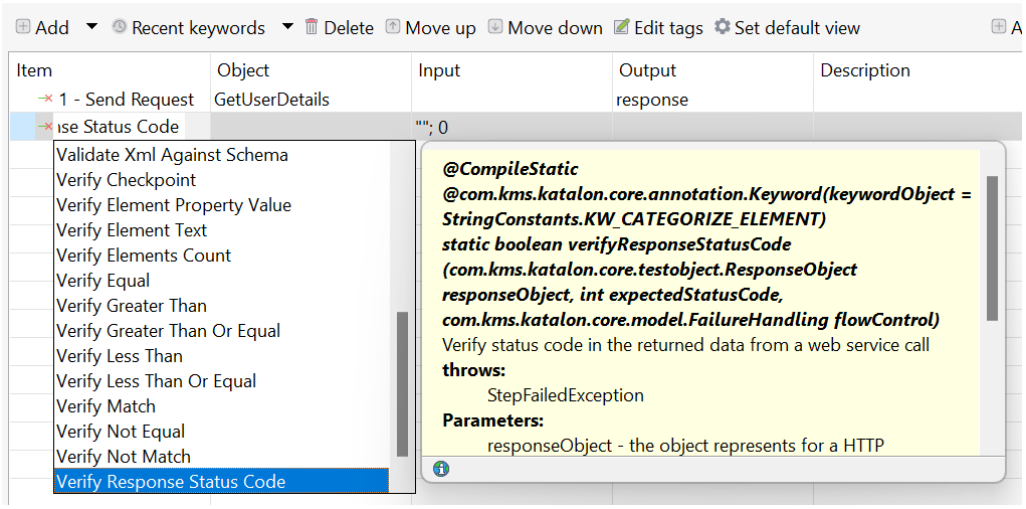
19. Pada Object yang berisi null, klik lalu pilih “GetUserDetails”



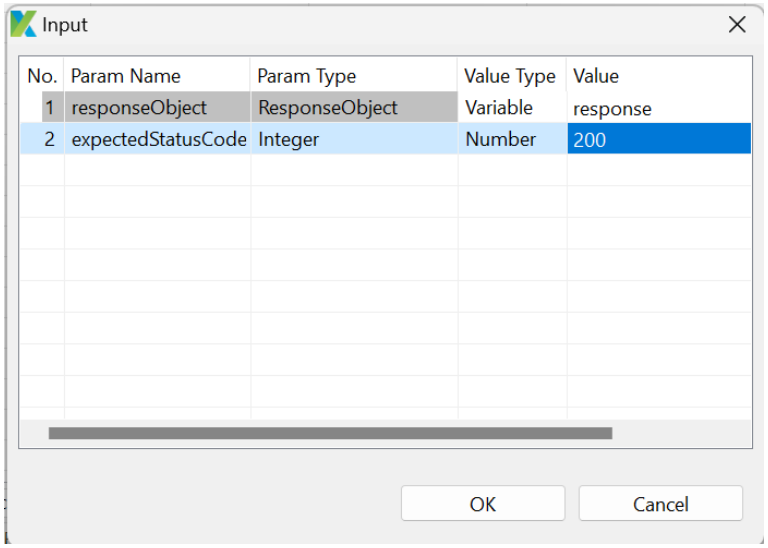
20. Pada Output isikan “response”

Item	Object	Input	Output	Description
1 - Send Request	GetUserDetails		response	

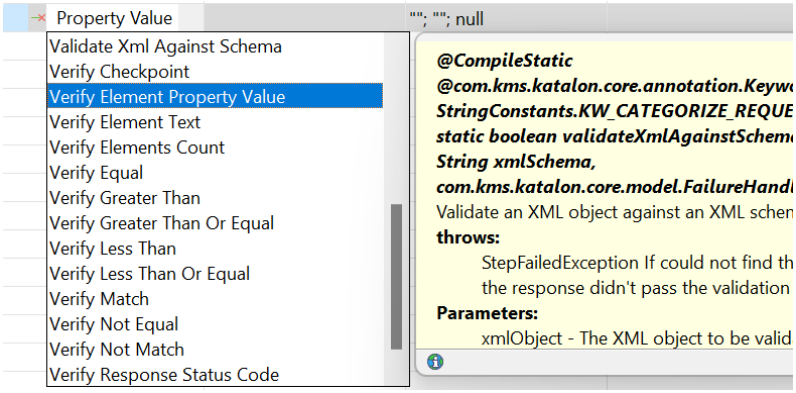
21. Tambahkan “Verify Response Status Code”



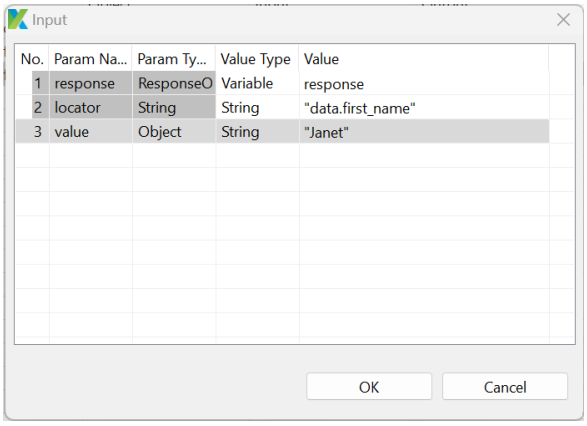
22. Ubah input dari “Verify Response Status Code” seperti berikut



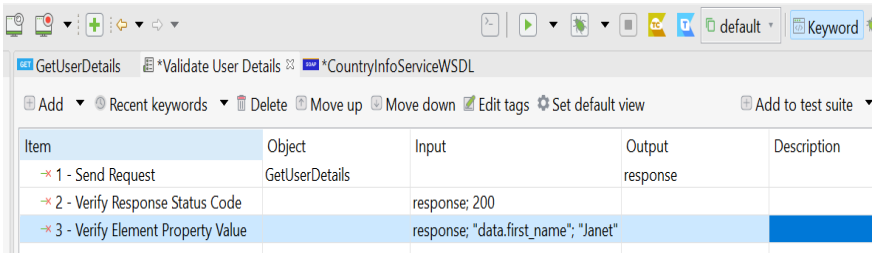
23. Kemudian tambahkan “Verify Element Property Value”



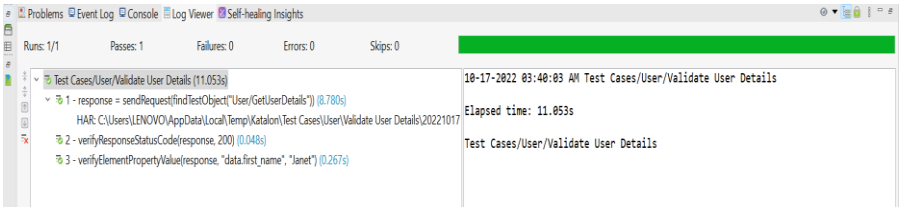
24. Ubah input dari “Verify Element Property Value” seperti berikut



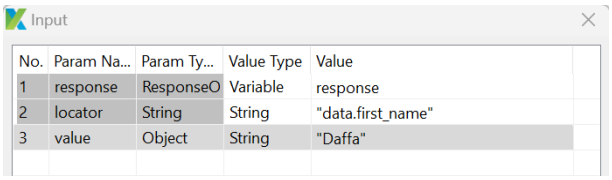
25. Sehingga menjadi seperti berikut, selanjutnya klik icon play



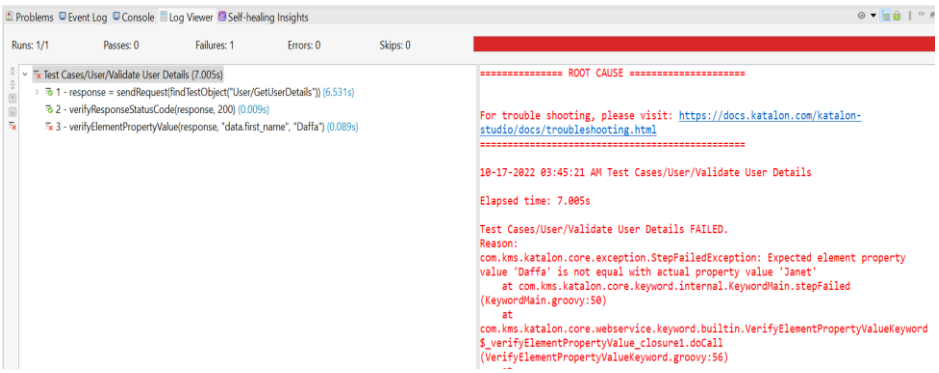
26. Berikut hasilnya, dapat dilihat lewat Log Viewer, hasilnya tidak ada errornya



27. Namu Ketika kita ganti inputan pada “Verify Element Property Value” seperti berikut

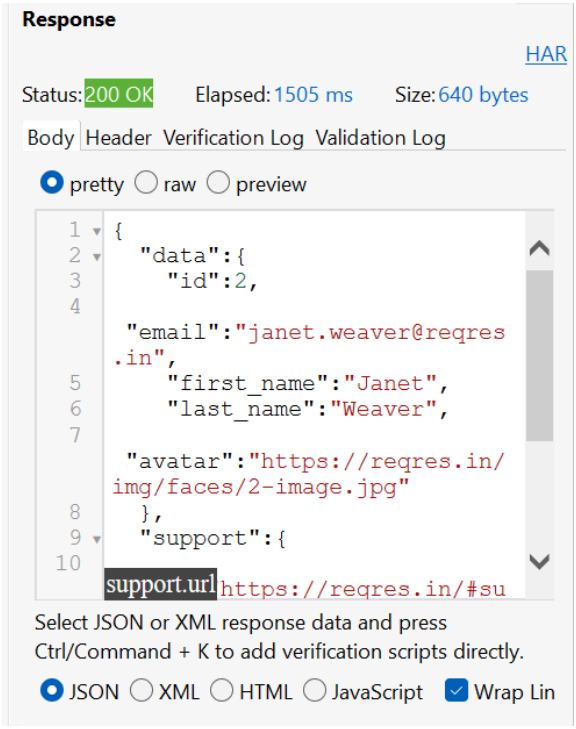


28. Maka hasilnya akan error, seperti berikut

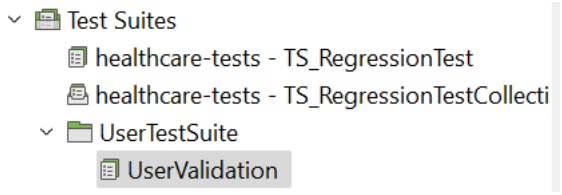


29. Penyebab Error dikarenakan pada Object Respository terdapat folder User yang didalamnya memiliki file GetUserDetails yang berisi sebagai berikut. Oleh

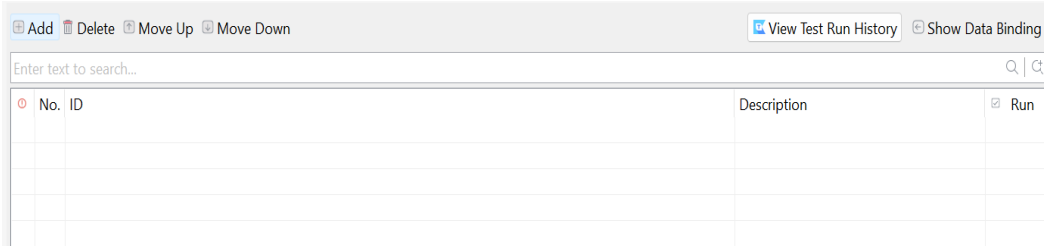
karena itu inputan pada “Verify Element Property Value” hanya menerima “Janet” sebagai string selain itu akan error.



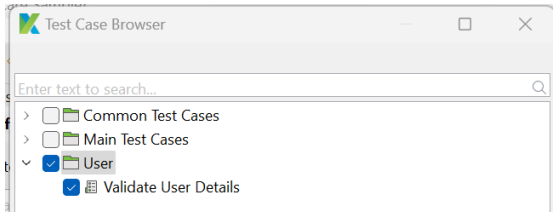
30. Buatkan test suite didalam folder UserTestSuite berikan nam “UserValidation”



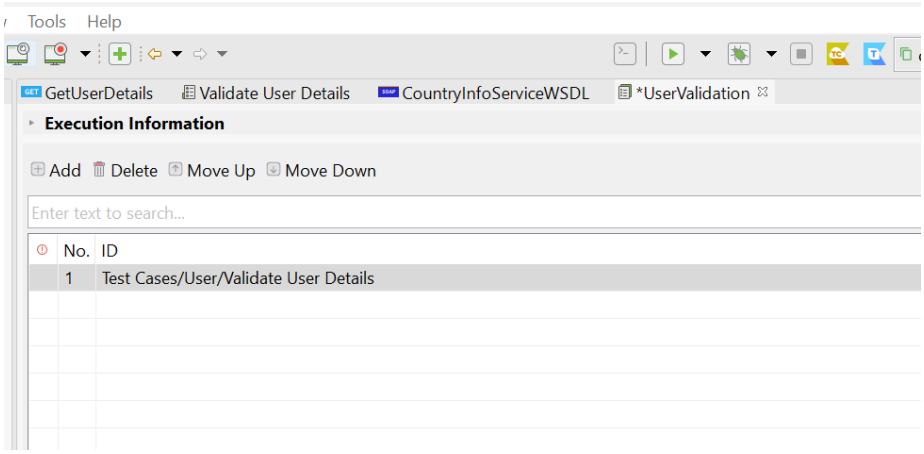
31. Berikut tampilannya, klik add



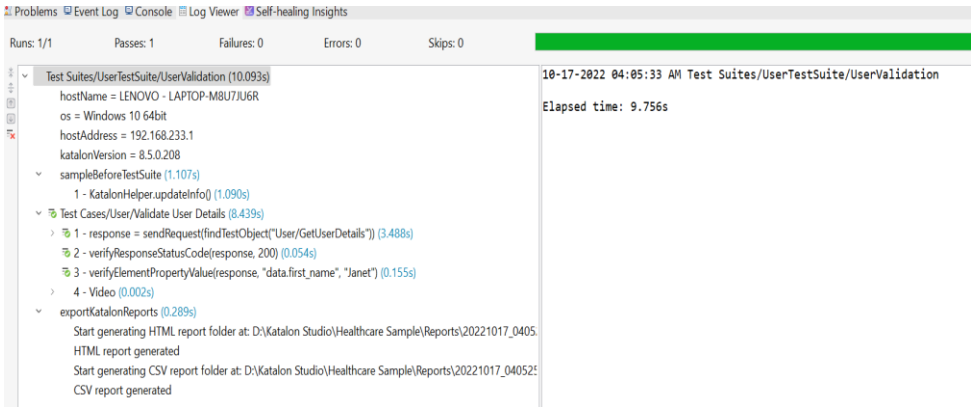
32. Centang Validate User Details



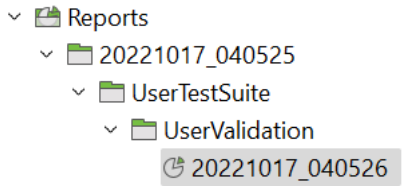
33. Berikut setelah di add, klik icon play



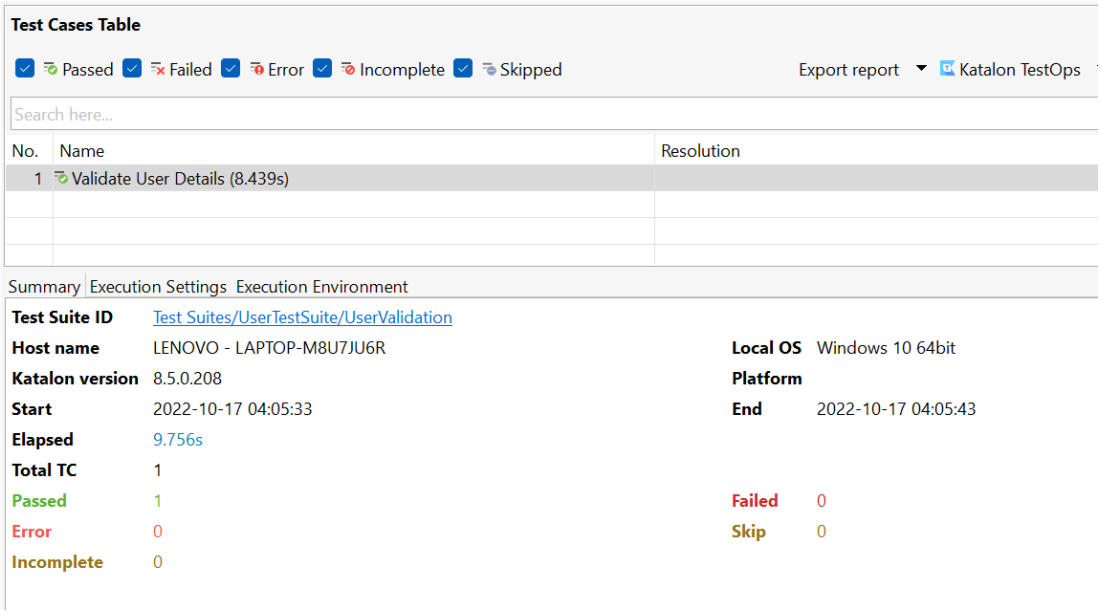
34. Berikut hasil dari Log Viewer



35. Selanjutnya kita buka Reports untuk melihat hasil pengujian



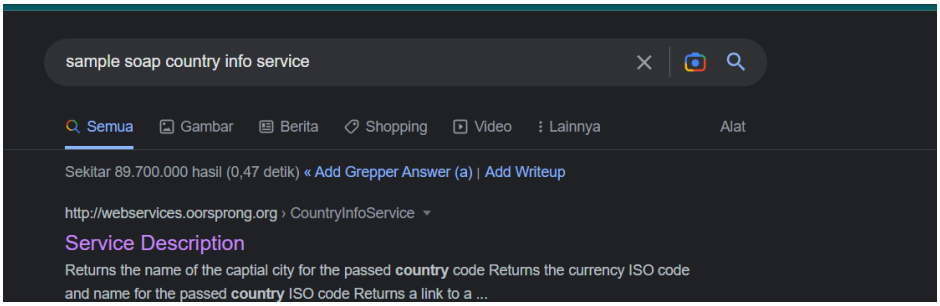
36. Berikut hasil report dari pengujian file json yang telah dilakukan



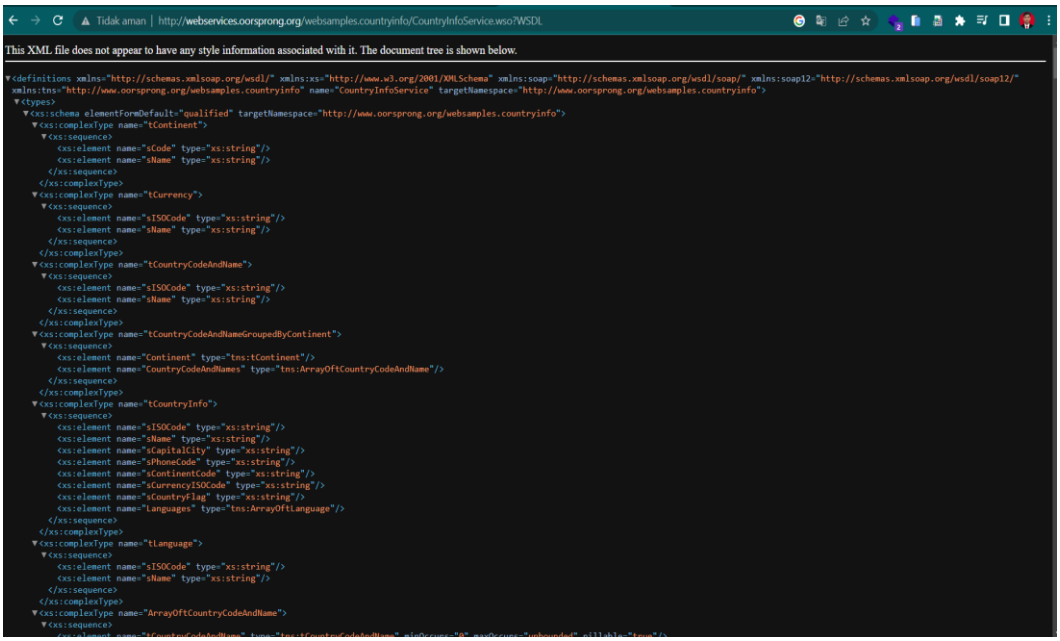
9.2 API CHAINING | SOAP-XML | HOW TO SEND VALUE FROM ONE API TO OTHER

Katalon juga dapat melakukan chaining API. Chaining API adalah suatu proses untuk mengekstrak nilai dari respon suatu API dan memberikan nilai ke permintaan API berikutnya. Pada tahapan ini kita akan melakukan chaining API pada API SOAP yang memiliki respon XML. Berikut tahapannya.

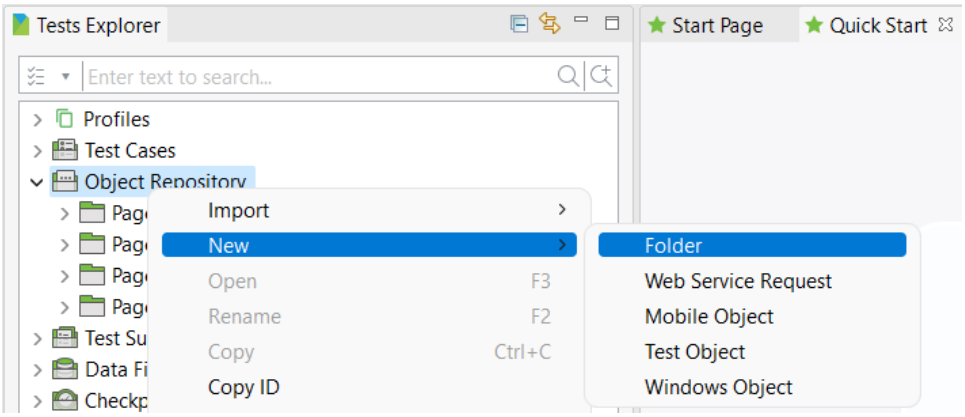
1. Cari satu contoh SOAP API, seperti gambar di bawah ini.



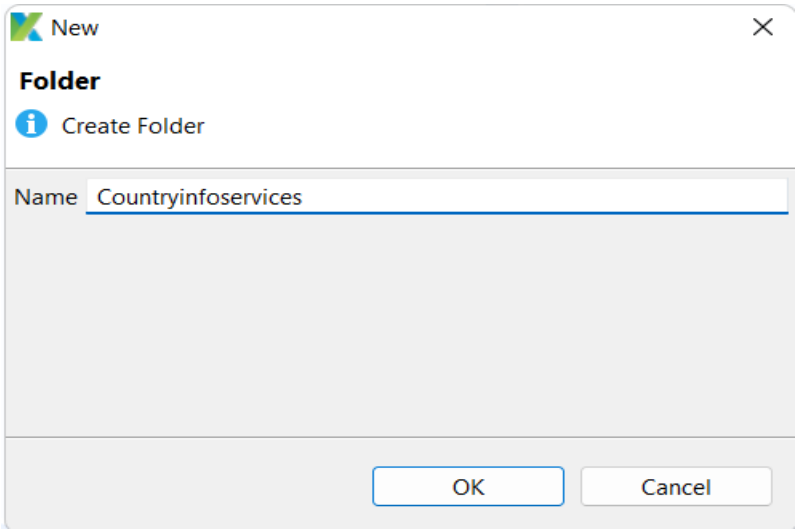
2. Berikut merupakan contoh script soap api dalam format xml.



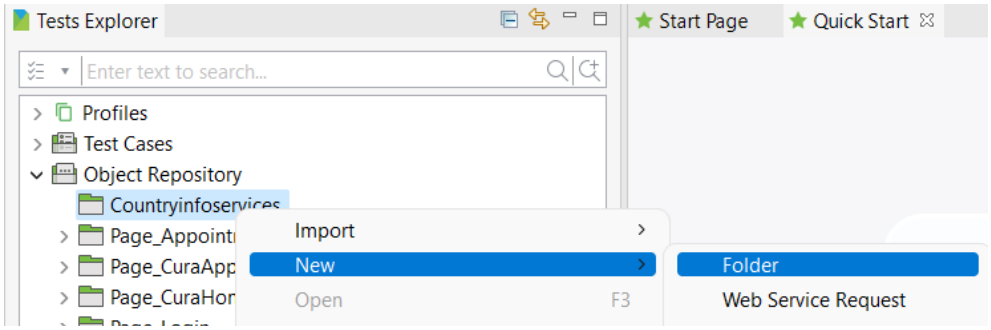
3. Selanjutnya buat 1 folder baru.



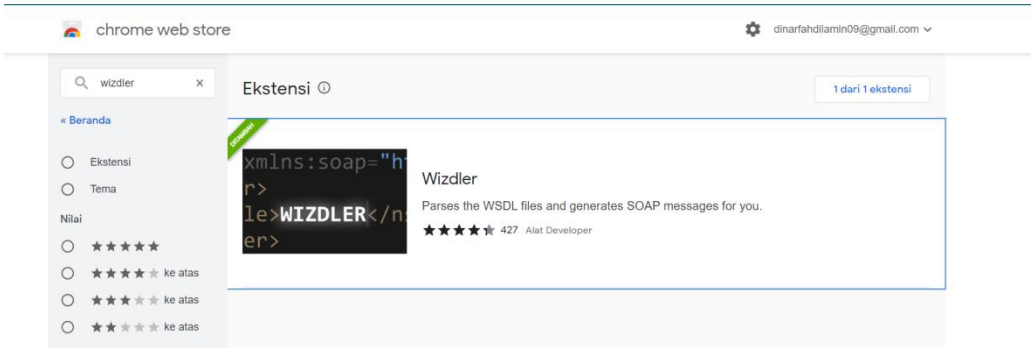
4. Kemudian, beri nama Countrinfoservices, lalu klik ok.



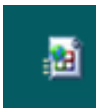
5. Pada folder Countrinfoservice, buat 1 web service request.



6. Karena kita akan mengambil beberapa permintaan dari script xml tadi, maka tambahkan extansi wizdler pada chrome. Extansi ini berguna untuk melewati dokumen vestal seperti forma, sehingga permintaan yang di request tadi dapat diterima.



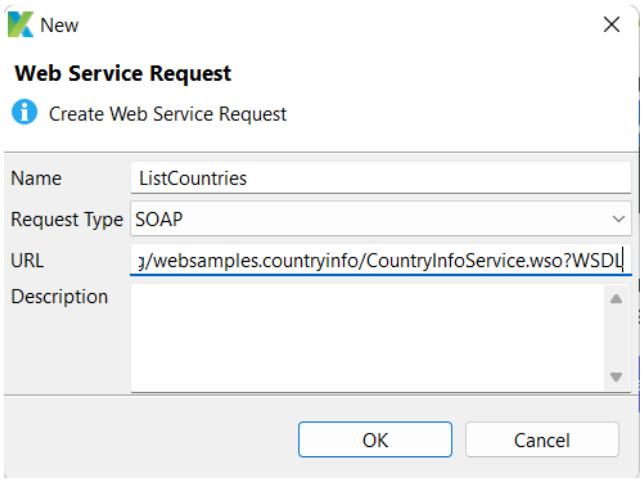
7. Jika extensi sudah ditambahkan, maka akan muncul ikon seperti gambar dibawah ini.



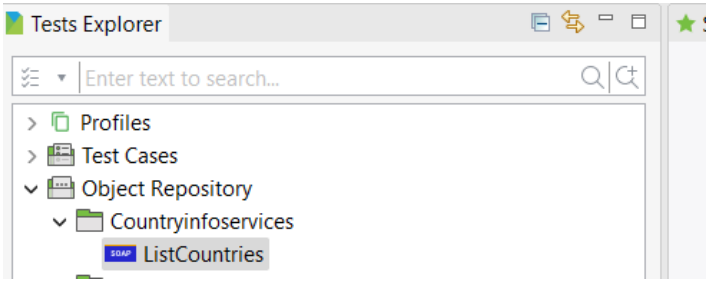
8. Berikut ini merupakan list request yang tersedia.

9. Selanjutnya kita akan mengambil request dari ListOfCountriNamesByName, berikut tampilannya.

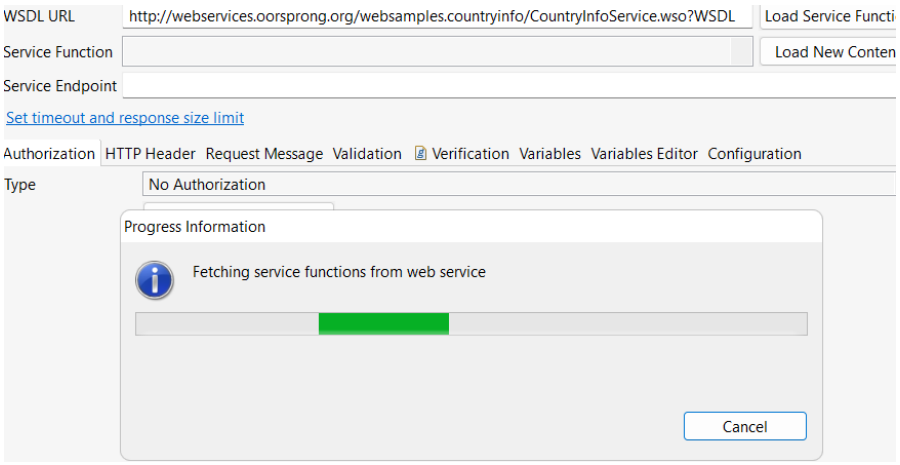
10. Buat web service request dengan nama ListCountries, dengan pilihan requestType adalah SOAP, dan masukan URL dari sample SOAP API tadi.



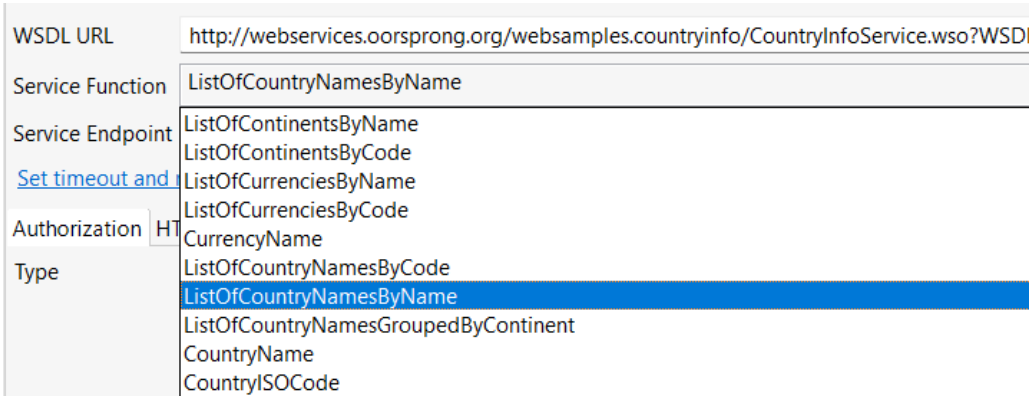
11. Dapat kita lihat web service request sudah terbentuk.



12. Masukkan service function, sebelumnya klik load service function terlebih dahulu untuk melakukan pencarian terhadap service function yang tersedia.

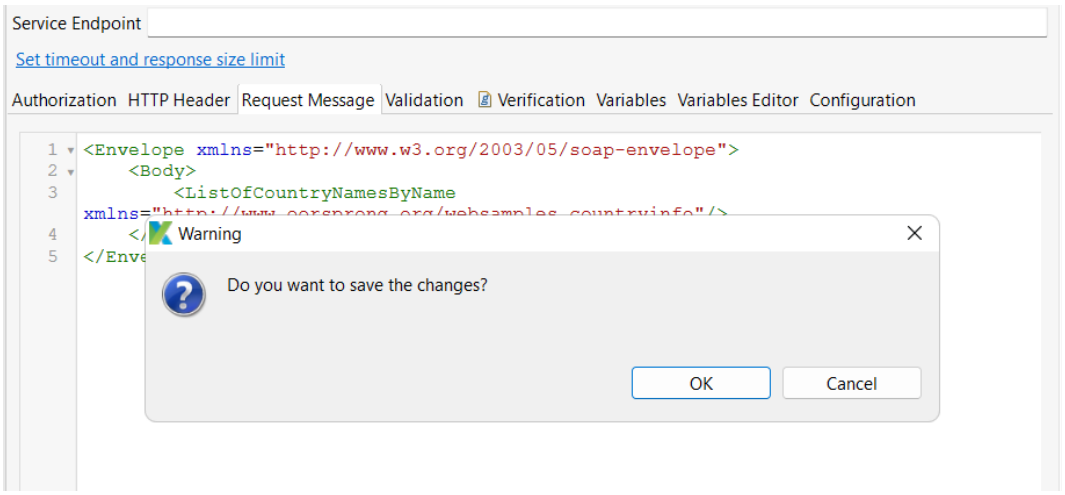


13. Pilih ListOfCountriNamesByName untuk service functionnya.

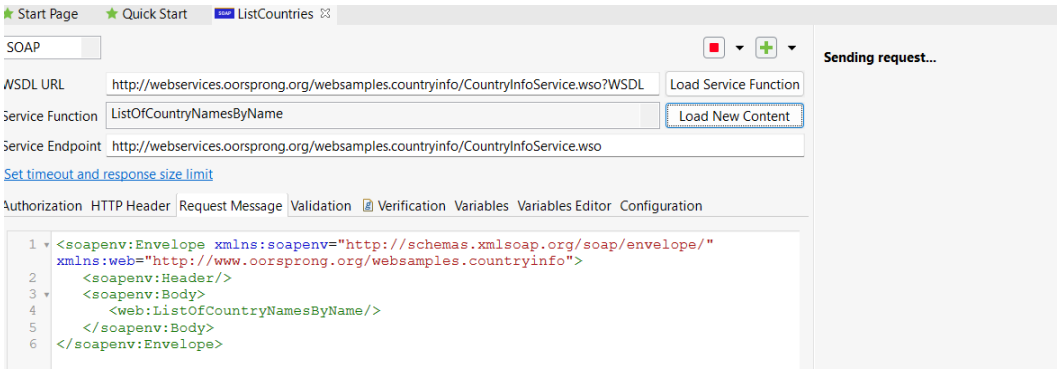


14. Kemudian pada request message copi script yang ada pada url ListOfCountriNamesByName tadi.

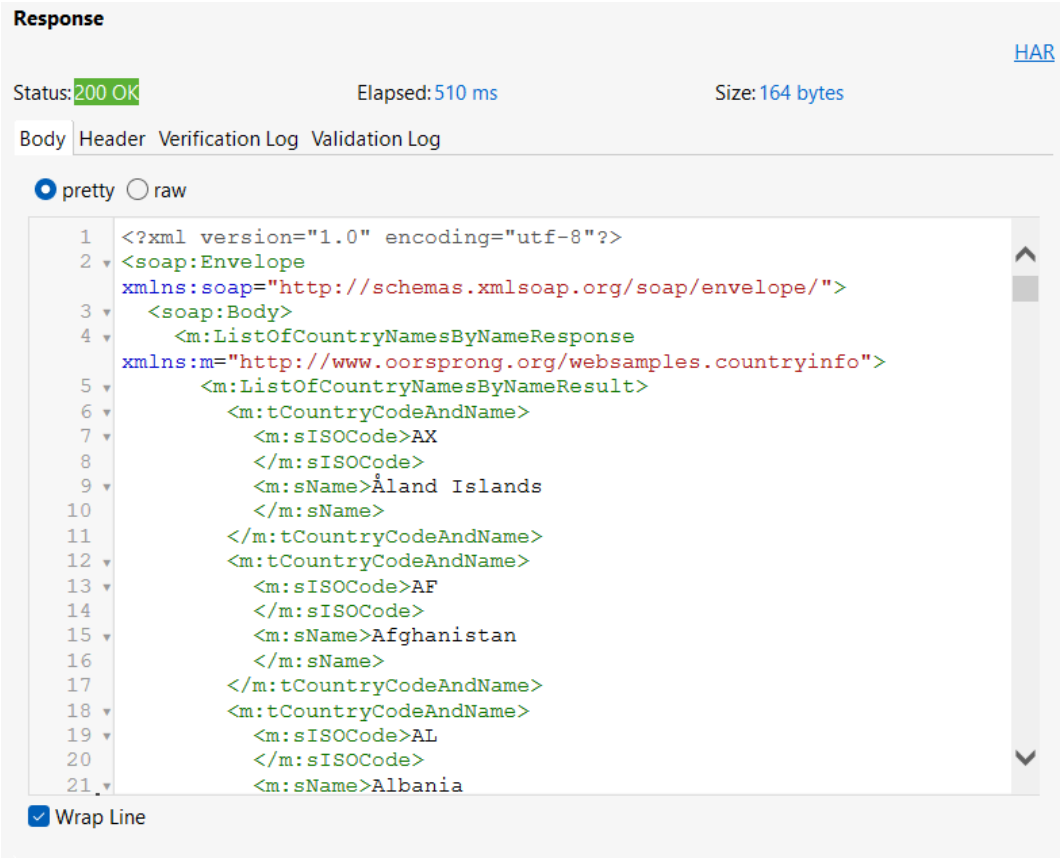
15. Kemudian save script.



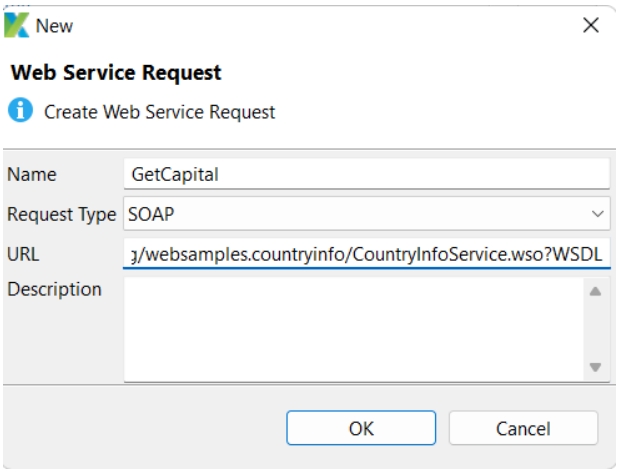
16. Selanjutnya jalankan script untuk melakukan request/permintaan.



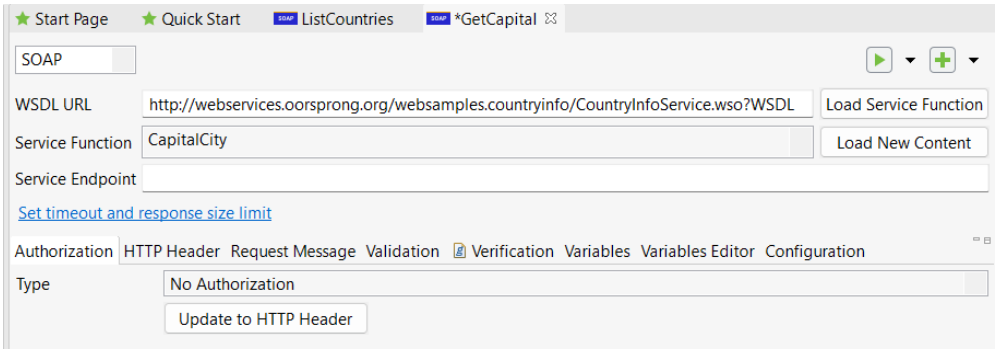
17. Berikut ini hasil dari request yang diberikan, dimana berisi list nama-nama dari berbagai negara.



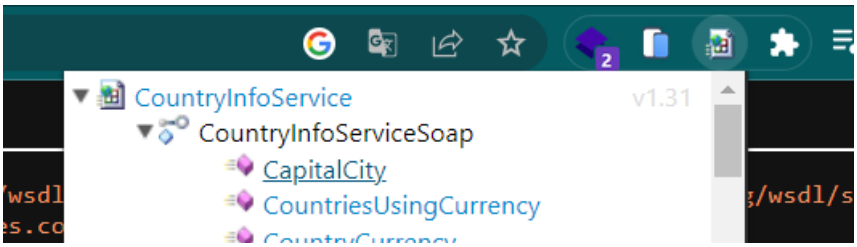
18. Buat web service request lainnya.



19. Pilih server functio nya CapitalCity.

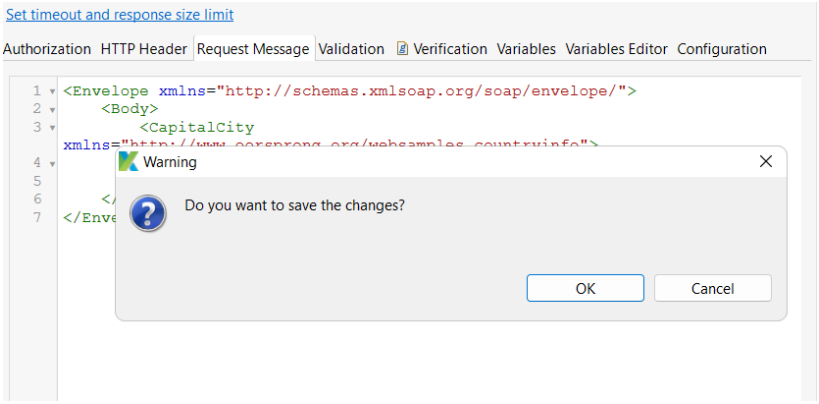


20. Pada extansi wizdler, pilih CapitalCity.

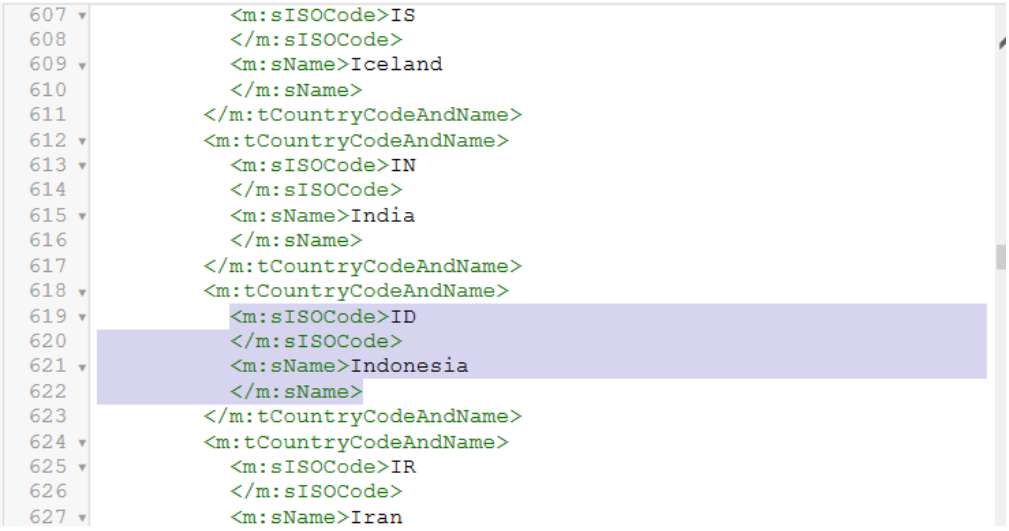


21. Berikut script dari CapitalCity.

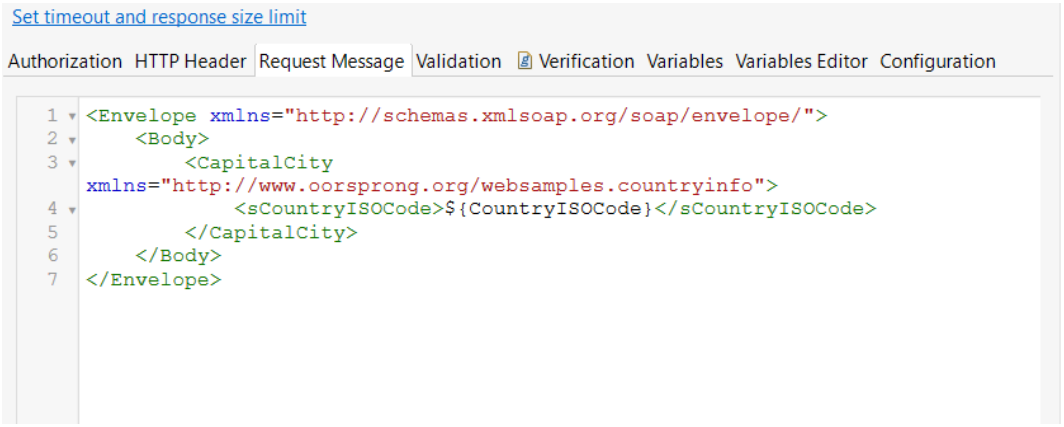
22. Save script.



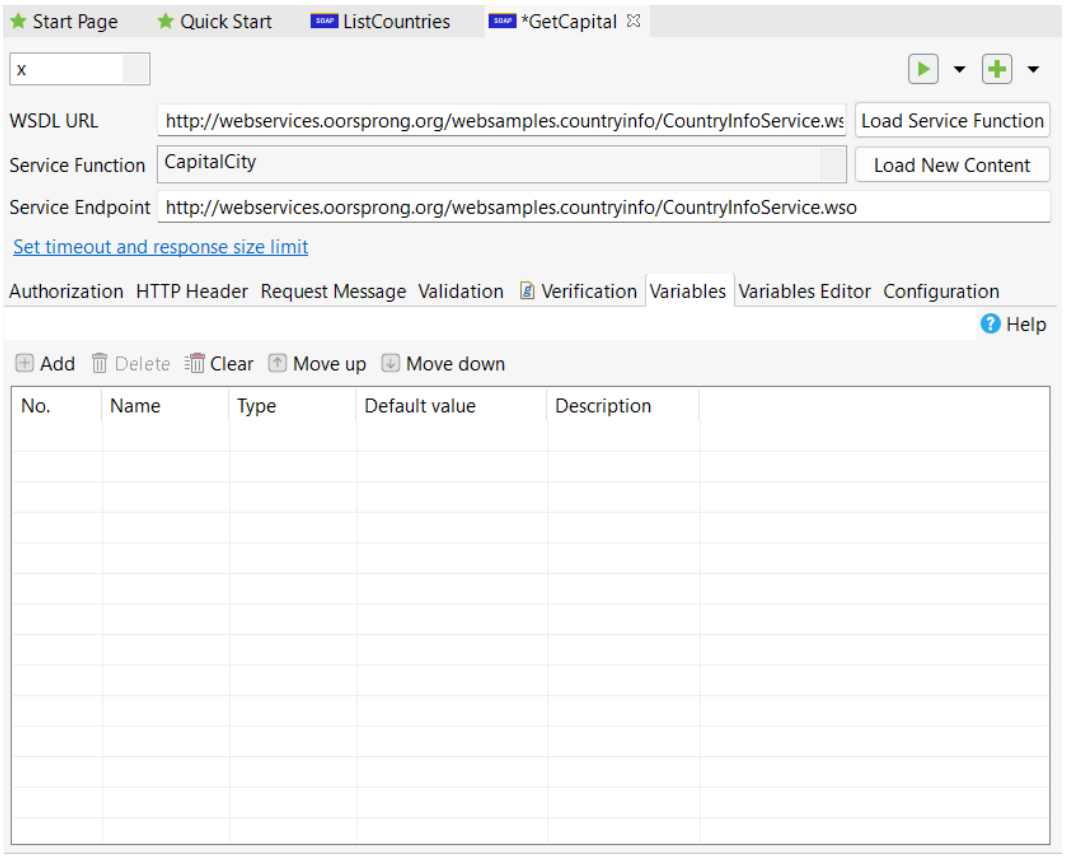
23. Berikut ini hasil dari request yang diberikan, dengan menampilkan nama-nama ibukota dari tiap negara.



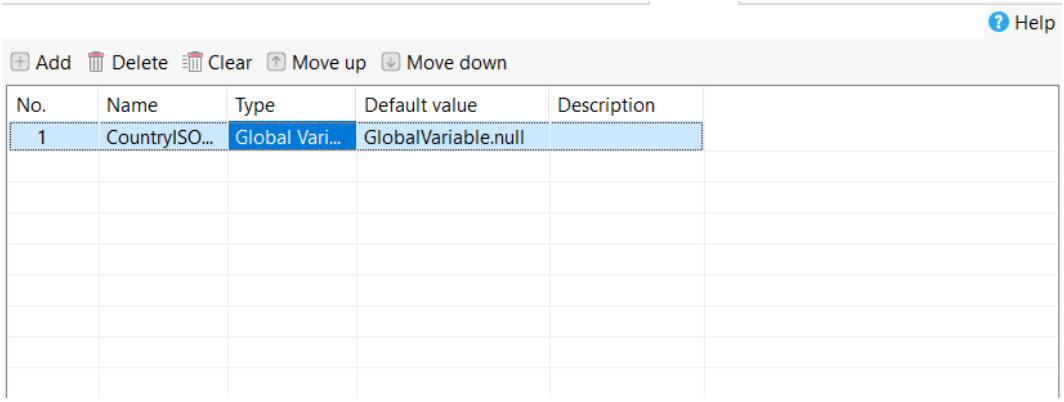
24. Kita juga dapat mengambil request berdsarkan variabel tertentu, seperti yang terlihat pada gambar di bawah ini terjadi perubahan script pada syntaks pemanggila ISO code.



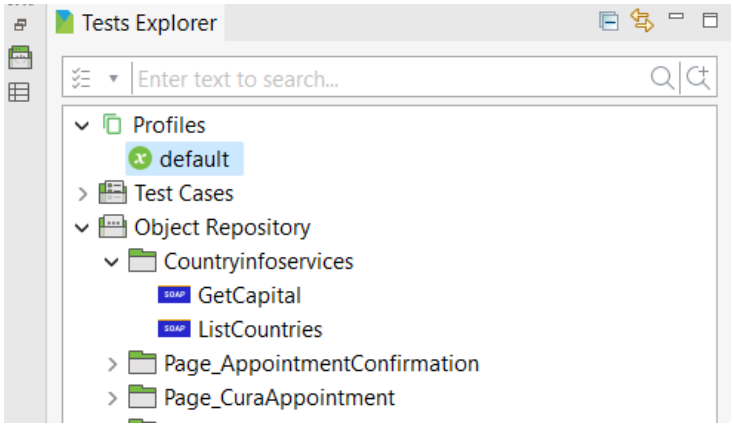
25. Kemudian pilih menu variabel dan klik add.

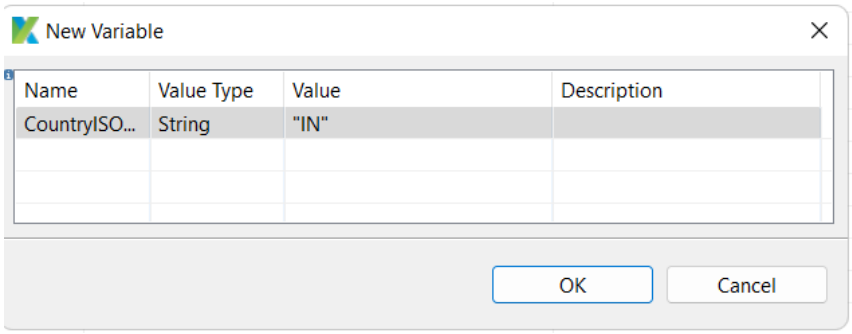


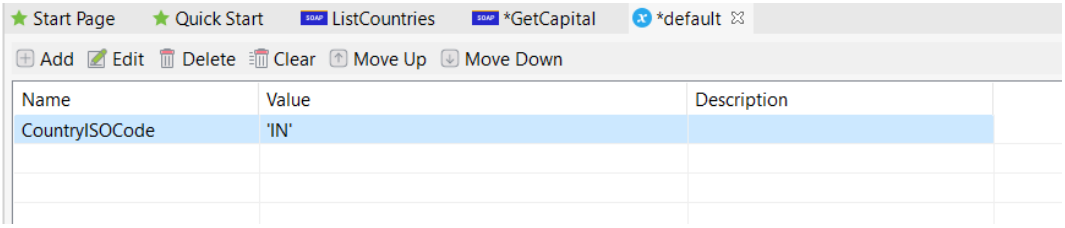
26. Selanjutnya, pilih name yaitu CountryISOCODE, pilih type sebagai Global Variable.



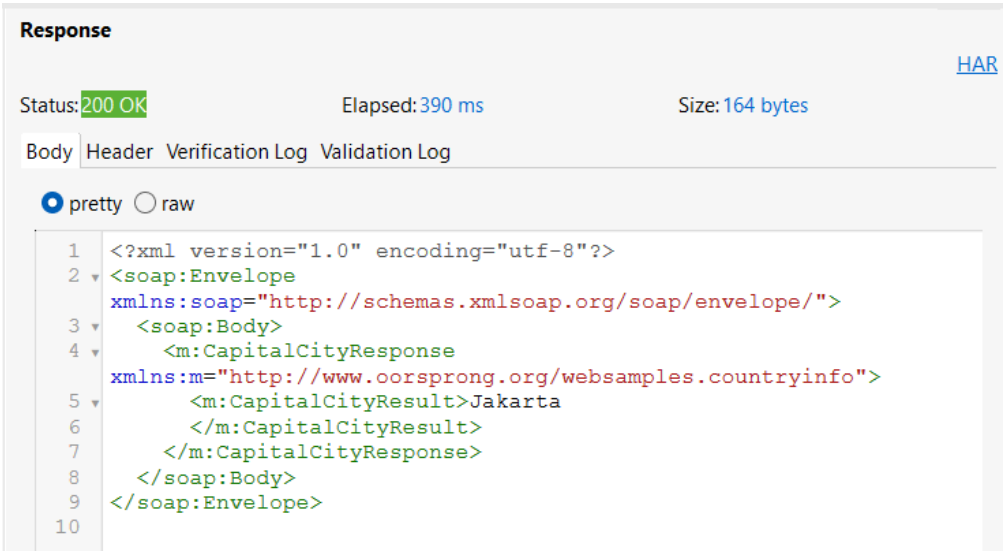
27. Buka profiles, kemudian pilih default, isi value dengan ISO CODE yang ingin anda tampilkan, untuk kasus ini saya isi value dengan inisial ID.



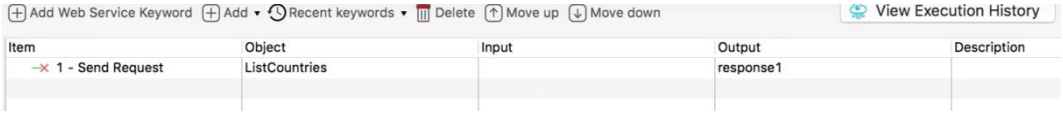




28. Berikut hasil request yang diberikan dari modifikasi sesuai dengan variabel ISO code dengan inisial ID.



29. Selanjutnya kita akan membuat pengujian test case/kasus uji, terhadap kedua web service request. Anda dapat klik icon tambah, kemudian pilih add to new test case, kemudian beri dengan nama TestOne, lalu klik ok.

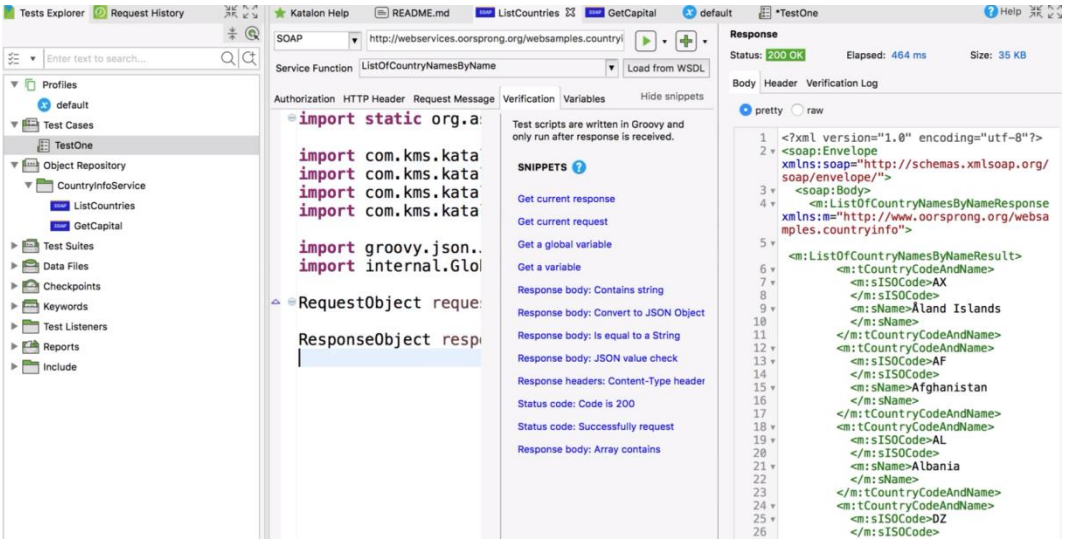


30. Selanjutnya kita akan menyimpan output dari API ListCountries pada variabel yang disebut response1.

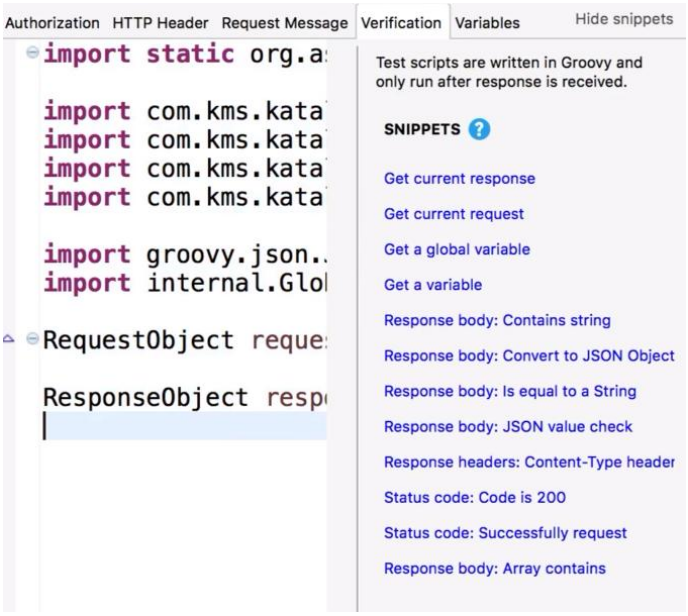
31. Kemudian pergi ke script, dan anda dapat melihat script dari variabel response1 yang telah dibuat.

```
import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint
import static com.kms.katalon.core.testcase.TestCaseFactory.findTestCase
import static com.kms.katalon.core.testdata.TestDataFactory.findTestData
import static com.kms.katalon.core.testobject.ObjectRepository.findTestObject
import com.kms.katalon.core.checkpoint.Checkpoint as Checkpoint
import com.kms.katalon.core.cucumber.keyword.CucumberBuiltinKeywords as CucumberKW
import com.kms.katalon.core.mobile.keyword.MobileBuiltInKeywords as Mobile
import com.kms.katalon.core.model.FailureHandling as FailureHandling
import com.kms.katalon.core.testcase.TestCase as TestCase
import com.kms.katalon.core.testdata.TestData as TestData
import com.kms.katalon.core.testobject.TestObject as TestObject
import com.kms.katalon.core.webservice.keyword.WSBuiltInKeywords as WS
import com.kms.katalon.core.webui.keyword.WebUiBuiltInKeywords as WebUI
import internal.GlobalVariable as GlobalVariable

response1 = WS.sendRequest(findTestObject('CountryInfoService/ListCountries'))
```

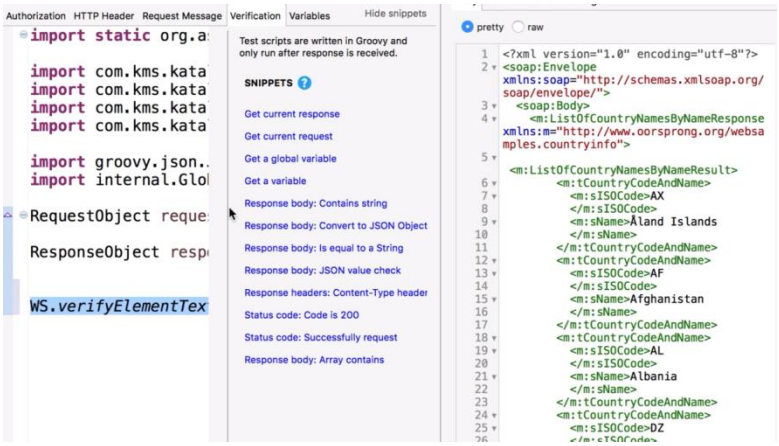
32. Sekarang kita akan melakukan fetch pada respon, dan mengambil nilai tertentu dari respon tersebut.



33. Pada jendela verifikasi kita dapat melihat list cuplikan yang tersedia, dan dapat digunakan untuk cuplikaan verifikasi.



34. Untuk melihat pemeriksaan atau verifikasi nilai XML kita dapat melakukannya dengan mengarahkan kursor ke syntaks Iso Code, kemudian tekan ctrl + case pada keyboard, sehingga kita dapat melihat pada jendela show snipped , telah ada verifikasi khusus yang sudah dibuat.



35. Selanjutnya kita akan menuliskan kode untuk menyimpan nilai respon yang ada pada variabel response1 tadi, dimana nilainya kita tampung pada variabel xml1.

```
import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint

response1 = WS.sendRequest(findTestObject('CountryInfoService/ListCountries'))
String xml1 = response1.responseBodyContent
```

36. Kemudian parsing nilai respon tersebut ke dalam bentuk xml, dengan kode berikut, dimana hasil akan disimpan pada variabel dataValue.

```
import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint

response1 = WS.sendRequest(findTestObject('CountryInfoService/ListCountries'))
String xml1 = response1.responseBodyContent
def dataValue= new XmlSlurper().parseText(xml1)
```

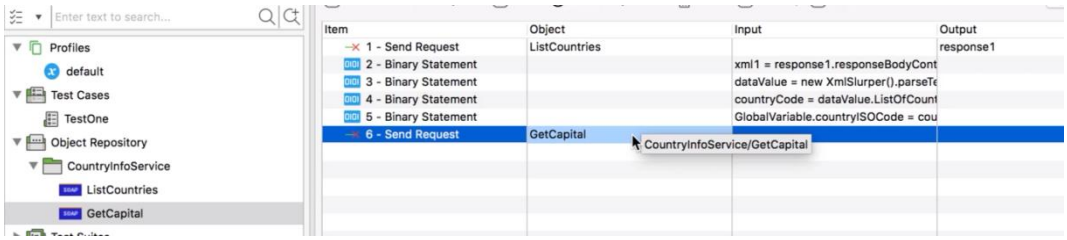
37. Untuk mendapatkan nilai atau simpul tertentu, hasil paring yang ada pada variabel dataValue tadi, kita simpulkan berdasarkan list country name by name, lalu simpan hasil pada variabel countryCode, seperti yang terlihat pada kode berikut.

```
import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint

response1 = WS.sendRequest(findTestObject('CountryInfoService/ListCountries'))
String xml1 = response1.responseBodyContent
def dataValue= new XmlSlurper().parseText(xml1)
def countryCode = dataValue.ListOfCountryNamesByNameResult.tCountryCodeAndName[2
```

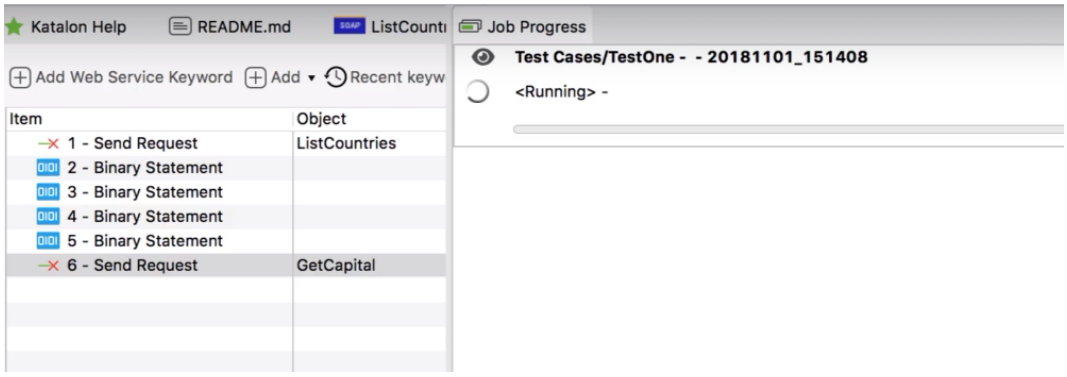
38. Selanjutnya hasil pada variabel countryCode tadi kita simpan pada global variabel, dengan perintah seperti berikut.

```
import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint

response1 = WS.sendRequest(findTestObject('CountryInfoService/ListCountries'))
String xml1 = response1.responseBodyContent
def dataValue= new XmlSlurper().parseText(xml1)
def countryCode = dataValue.ListOfCountryNamesByNameResult.tCountryCodeAndName[2
GlobalVariable.countryISOCode = countryCode
```

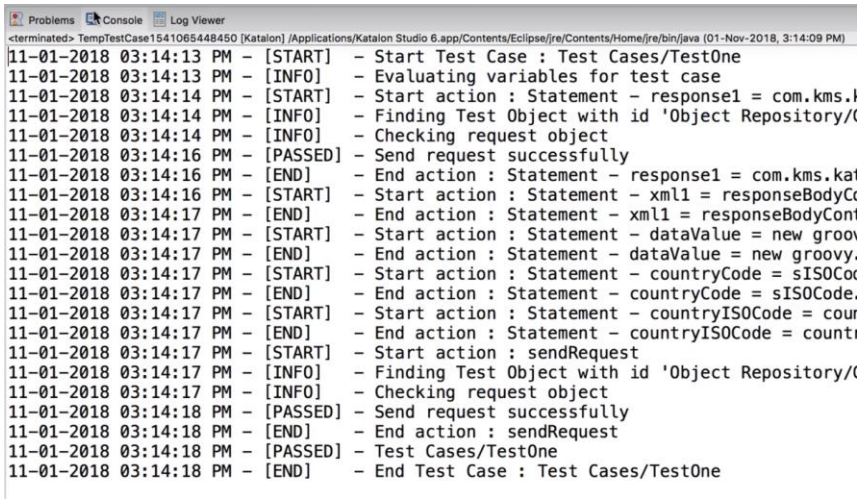
39. Setelah itu pergi ke jenda test case, dan panggil API GetCapital yang telah dibuat sebelumnya.



40. Lalu save dan jalankan, dan berikut hasil yang diberikan.



41. Buka jendela console, dan memperlihatkan hasil verifikasi yang telah kita lakukan.



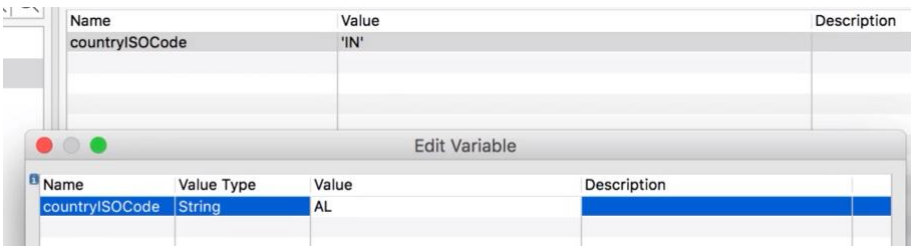
42. Untuk melihat country code yang di ekstrak, dapat tambahkan code berikut.

```
import com.kms.katalon.core.testobject.TestObject as TestObject
import com.kms.katalon.core.webservice.keyword.WSBuiltInKeywords as
import com.kms.katalon.core.webui.keyword.WebUiBuiltInKeywords as We
import internal.GlobalVariable as GlobalVariable

response1 = WS.sendRequest(findTestObject('CountryInfoService/ListCo

String xml1 = response1.responseBodyContent

def dataValue = new XmlSlurper().parseText(xml1)

def countryCode = dataValue.ListOfCountryNamesByNameResult.tCountryC

println("The extracted country code is : "+countryCode)

GlobalVariable.countryISOCode = countryCode

WS.sendRequest(findTestObject('CountryInfoService/GetCapital'))
```

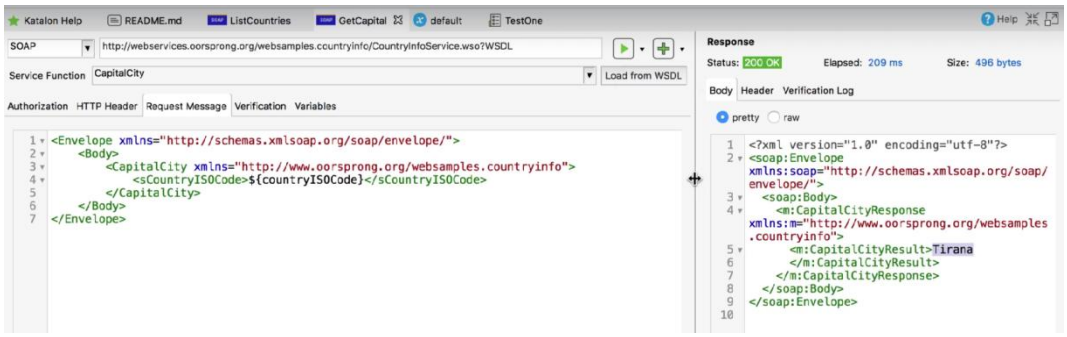
43. Dapat dilihat berikut hasil dari country code yang di ekstrak, yaitu country code dengan kode negara AL.

```
Problems  Console  Log Viewer
<terminated> TempTestCase1541065503872 [Katalon] /Applications/Katalon Studio 6.app/Contents/Eclipse/jre/Contents/Home/jre/bin/java (01-Nov-2018, 3:15:04 PM)
11-01-2018 03:15:08 PM - [START]  - Start Test Case : Test Cases/TestOne
11-01-2018 03:15:09 PM - [INFO]   - Evaluating variables for test case
11-01-2018 03:15:10 PM - [START]  - Start action : Statement - response1 = com.kms.katalon.core.webservice.ke
11-01-2018 03:15:10 PM - [INFO]   - Finding Test Object with id 'Object Repository/CountryInfoService/ListCou
11-01-2018 03:15:10 PM - [INFO]   - Checking request object
11-01-2018 03:15:12 PM - [PASSED] - Send request successfully
11-01-2018 03:15:12 PM - [END]    - End action : Statement - response1 = com.kms.katalon.core.webservice.keyw
11-01-2018 03:15:12 PM - [START]  - Start action : Statement - xml1 = responseBodyContent
11-01-2018 03:15:13 PM - [END]    - End action : Statement - xml1 = responseBodyContent
11-01-2018 03:15:13 PM - [START]  - Start action : Statement - dataValue = new groovy.util.XmlSlurper().parse
11-01-2018 03:15:13 PM - [END]    - End action : Statement - dataValue = new groovy.util.XmlSlurper().parseTe
11-01-2018 03:15:13 PM - [START]  - Start action : Statement - countryCode = sISOCode.text()
11-01-2018 03:15:13 PM - [END]    - End action : Statement - countryCode = sISOCode.text()
11-01-2018 03:15:13 PM - [START]  - Start action : Statement - println("The extracted country code is : " + c
The extracted country code is : AL
11-01-2018 03:15:13 PM - [END]    - End action : Statement - println("The extracted country code is : " + cou
11-01-2018 03:15:13 PM - [START]  - Start action : Statement - countryISOCode = countryCode
11-01-2018 03:15:13 PM - [END]    - End action : Statement - countryISOCode = countryCode
11-01-2018 03:15:13 PM - [START]  - Start action : sendRequest
11-01-2018 03:15:13 PM - [INFO]   - Finding Test Object with id 'Object Repository/CountryInfoService/GetCapi
11-01-2018 03:15:13 PM - [INFO]   - Checking request object
11-01-2018 03:15:14 PM - [PASSED] - Send request successfully
11-01-2018 03:15:14 PM - [END]    - End action : sendRequest
11-01-2018 03:15:14 PM - [PASSED] - Test Cases/TestOne
11-01-2018 03:15:14 PM - [END]    - End Test Case : Test Cases/TestOne
```

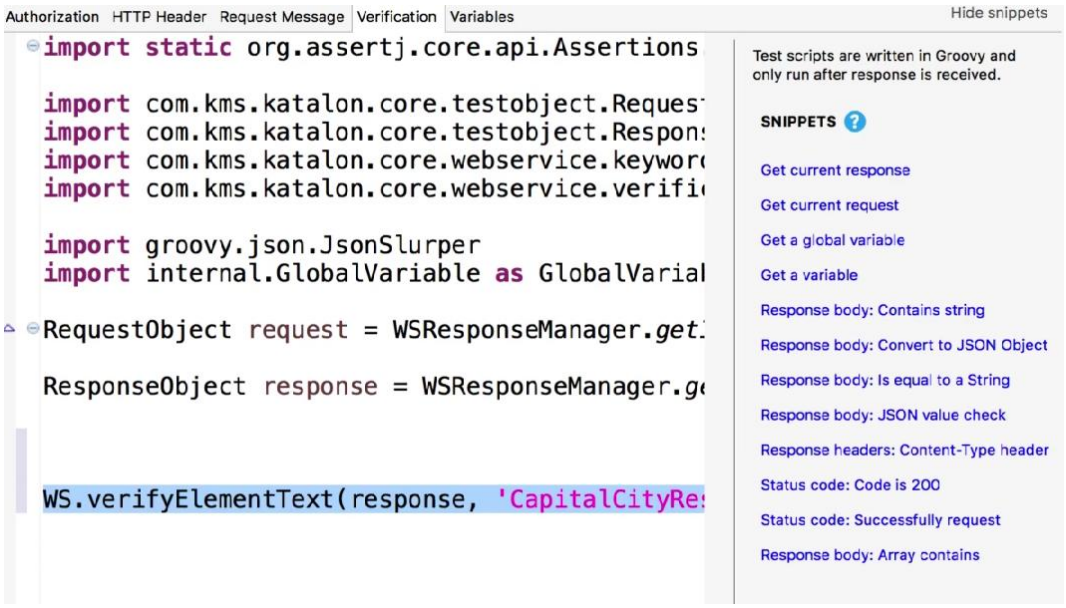
44. Untuk memvalidasi country code tersebut, pergi ke halaman default dan ubah value dari country code menjadi AL.



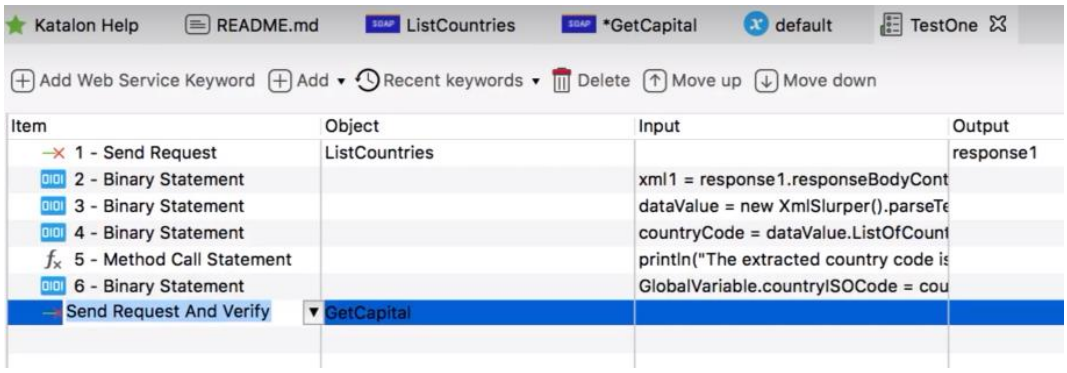
45. Lalu save dan jalankan, hasil memperlihatkan country coude dengan inisial AL adalah negara Tirana.



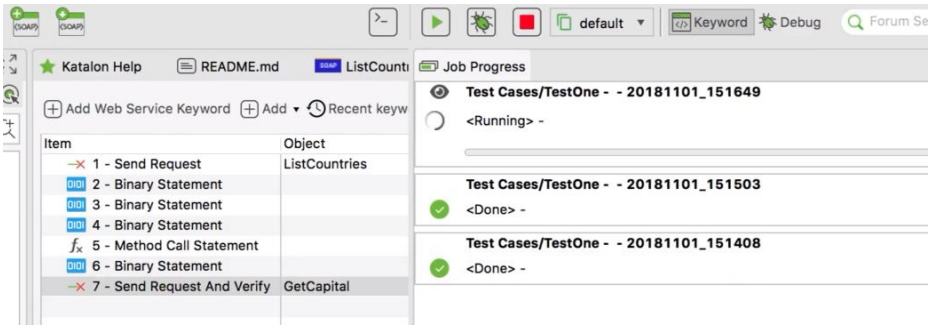
46. Kembali ke halaman verifikasi, arahkan kursor ke “Tirana” lalu tekan ctrl + case pada keyboard, dan dapat dilihat code verifikasi telah terbentu.



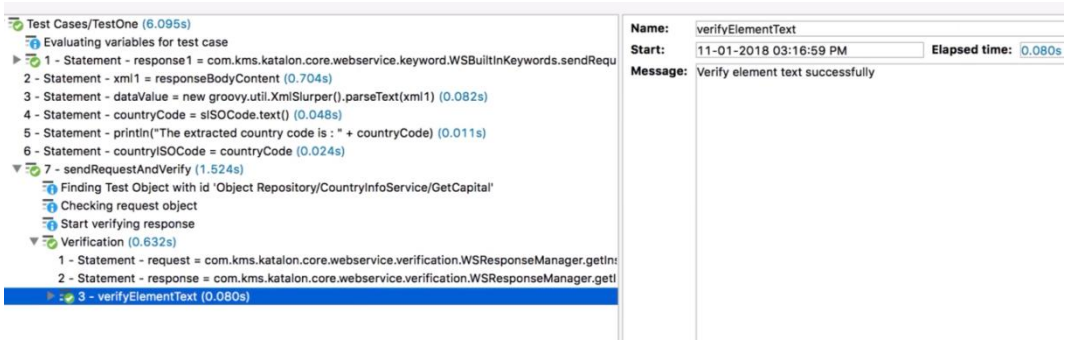
47. Selanjutnya masuk ke halaman test case, lalu update item dengan “send request and verify”. Hal ini dilakukan untuk memverifikasi permintaan/request dari API yang dilakukan.



48. Kemudian save script, dan jalankan tet case.



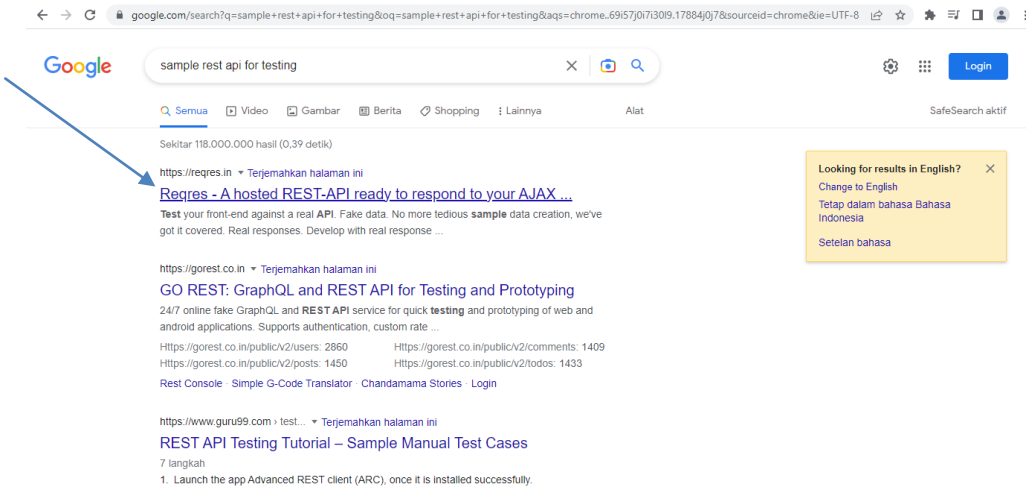
49. Setelah proses running selesai dan berhasil, pergi ke halaman log viewer dan kita dapat meliihat verifikasi yang sudah berhasil dan berfungsi dengan baik.



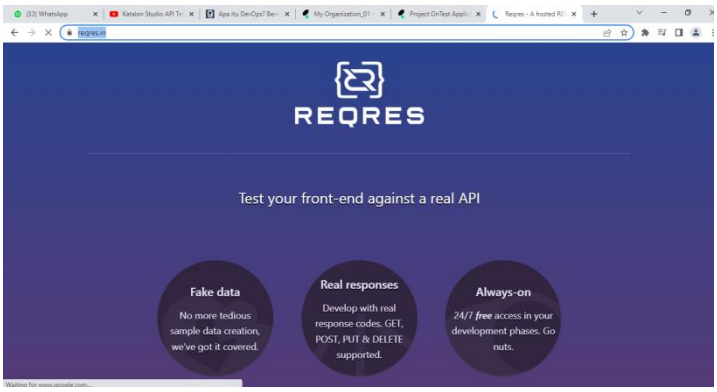
9.3. API CHAINING | REST - JSON | HOW TO SEND VALUE FROM ONE API TO OTHER

Setelah rangkaian pengujian atau kumpulan rangkaian pengujian, sesi terakhir kita telah melihat cara mengek nilai kita dari respon XML kita dan kemudian memberikan nilai itu dalam permintaan API, selanjutnya kita akan menggunakan API Rest dan melihat bagaimana melakukannya dengan respons Json. Berikut tahapan-tahapannya.

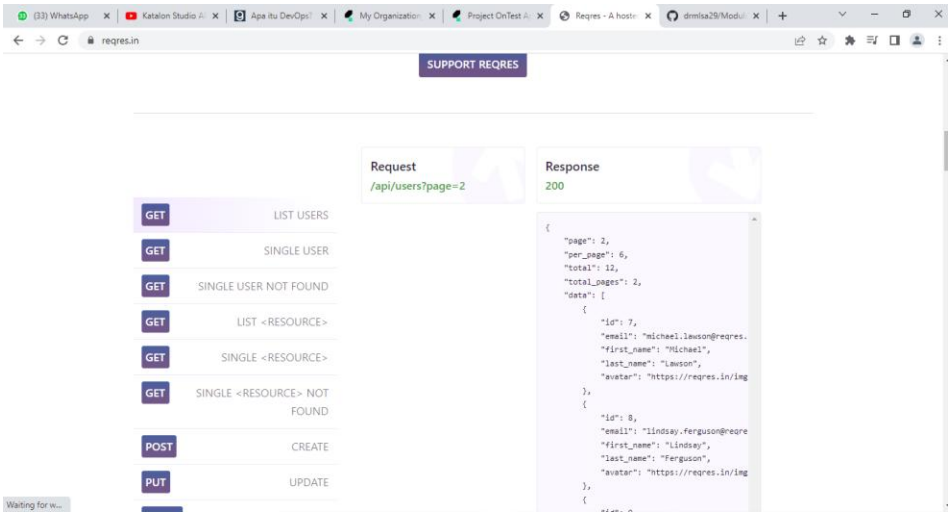
1. Setelah membuka katalon studio. Selanjutnya kita akan mencari sampel sisa untuk pengujian dan disini kita memiliki sampel eqres di web



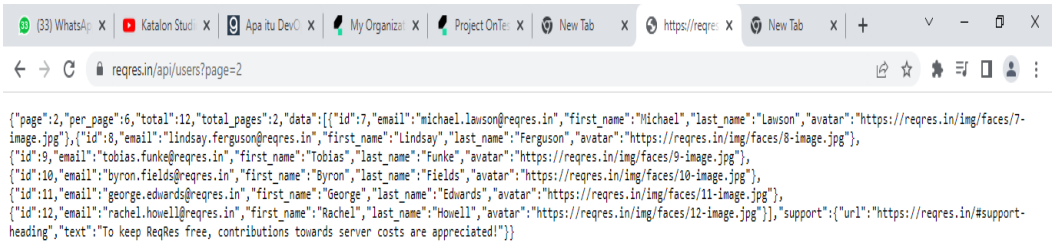
2. Situs web ini memiliki api istirahat yang baik untuk pengujian dan disini kita dapat melihat ini memiliki api palsu sehingga kita dapat menggunakan untuk pengujian dan yang paling pasti berlaku kita akan membuat url ini tersedia sehingga kita dapat merujuk mereka menjadi



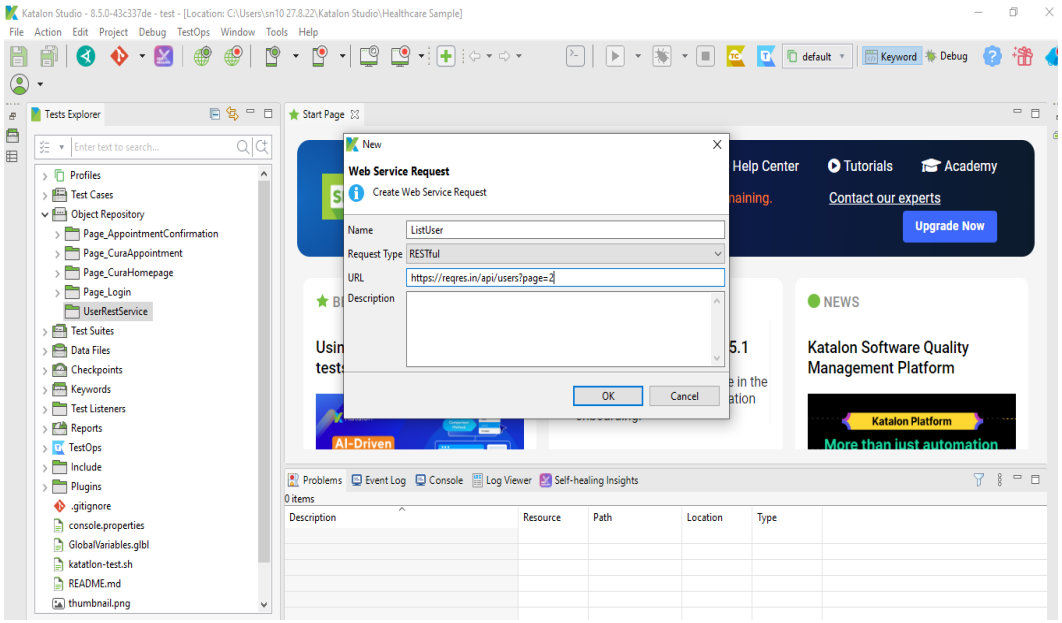
3. Disini bisa kita lihat jika kita turun kita akan memiliki banyak aplikasi dan kita telah mendapatkan posting yang di hapus batch dan seterusnya kita akan menggunakan daftar ini, pengguna mendapatkan API, jadi kita akan menyalin url ini dan meletakkan nya di tab baru.



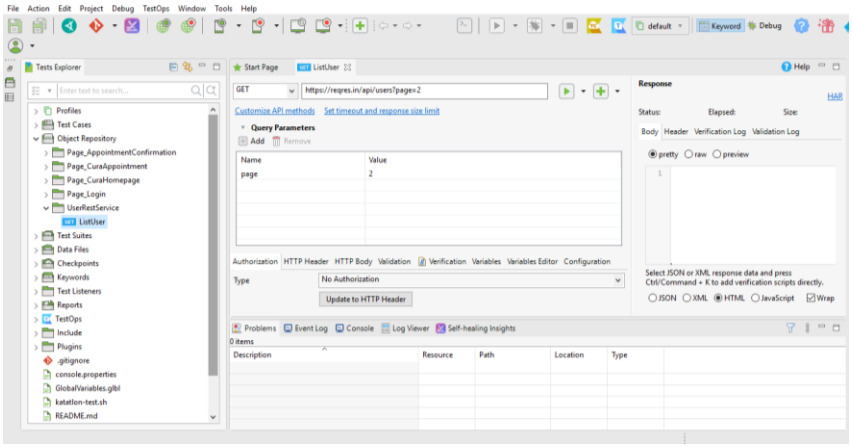
4. Menyalin teks url reqres.in dan menyalin sisa titik akhir dari sini dan menambahkan nya lagi, dan ini adalah titik terakhir API jika kita menjalankan nya kita dapat melihat respon. Jadi saya akan menyalin url ini



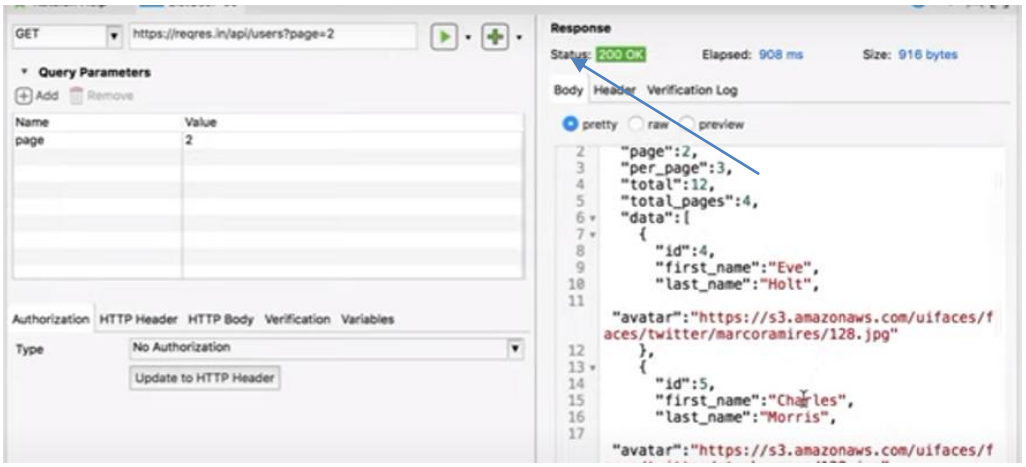
5. Setelah menyalin url, kemudian buka katalon studio dan buat folder baru pada object repository dengan nama UserRestService dan memasukkan link url ke web service request



6. Dan ini hasil setelah kita masukkan link url dan tidak ada otorsasi yang diperlukan untuk contoh API khusus ini, jika kita memerlukan otorisasi kita dapat mengedit di verifikasi dan dalam verifikasi kita dapat menambahkan beberapa cuplikan dan variable



7. Dan kita jalankan apakah sesuai dan kita akan melihat apa yang harus dilakukan kemudian kita memeriksa apakah kita mendapatkan respons yang valid. Dan hasil nya seperti pada gambar dibawah dengan status 200 ok. Dan kita juga mendapatkan daftar pengguna dengan API ini dan pengujian ini berfungsi dengan baik.



MODUL 12
CUSTOM KEYWORS

A. Tujuan

1. Mengetahui apa itu custom keywords pada katalon studio.
2. Mengetahui bagaimana cara membuat dan cara merujuk costum keywords pada katalon studio.

B. Dasar Teori

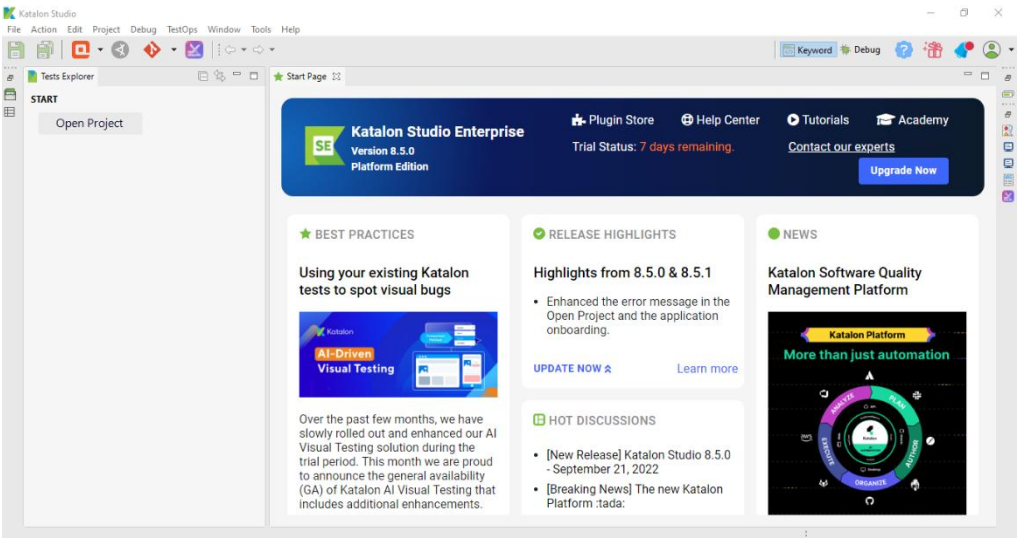
Custom keywords merupakan kata kunci khusus yang dapat anda gunakan untuk memperluas kemampuan Katalon Studio. Kata kunci ini berguna saat kita ingin menerapkan kasus pengujian, sama seperti kata kunci bawaan lainnya.

12.1. HOW TO CREATE CUSTOM KEYWORDS

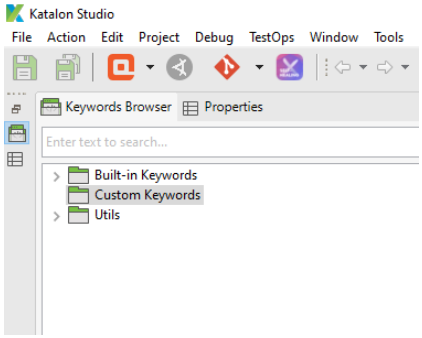
Bagaimana cara membuat custom keywords pada katalon studio? Selain kata kunci bawaan, pada katalon kita juga dapat menentukan kata kunci khusus, berikut ini langkah-langkah dalam pembuatan costum keywords pada katalon studio.

12.1.1. Bagaimana cara membuat kata kunci khusus

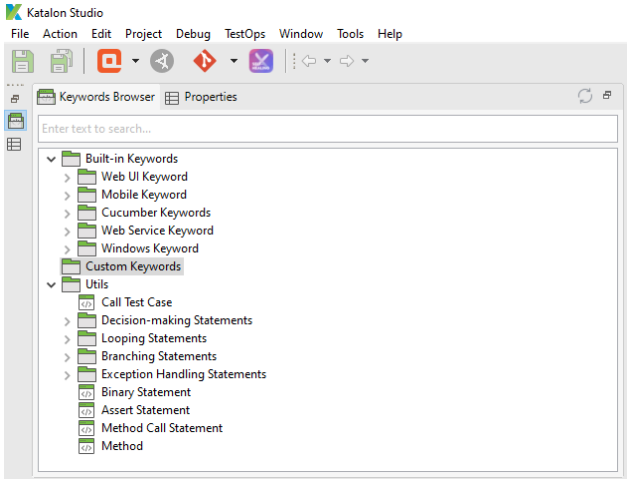
1. Untuk membuat kata kunci khusus pertama bukan aplikasi katalon studio.



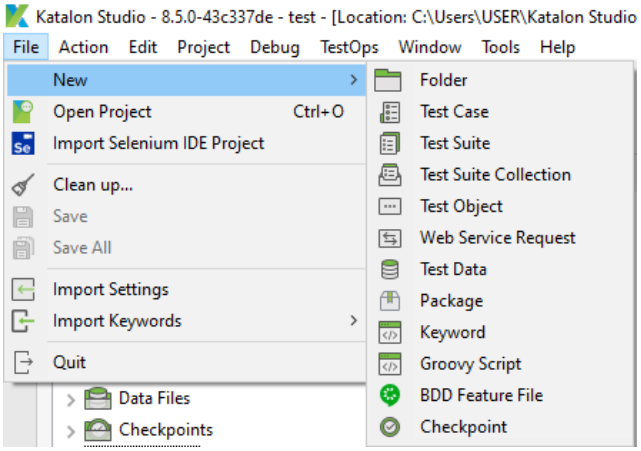
2. Kemudian pilih costume keywords.



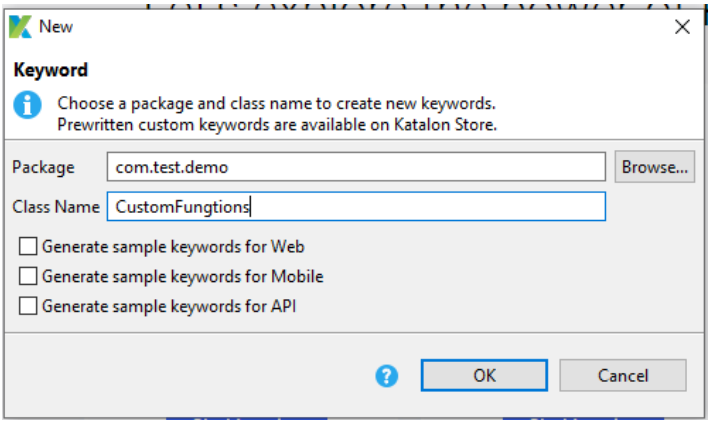
3. Berikut adalah fitur-fitur yang terdapat pada costume keyword dan semua kata kunci yang terdapat pada keyword dibwah ini semua bias dijalankan.



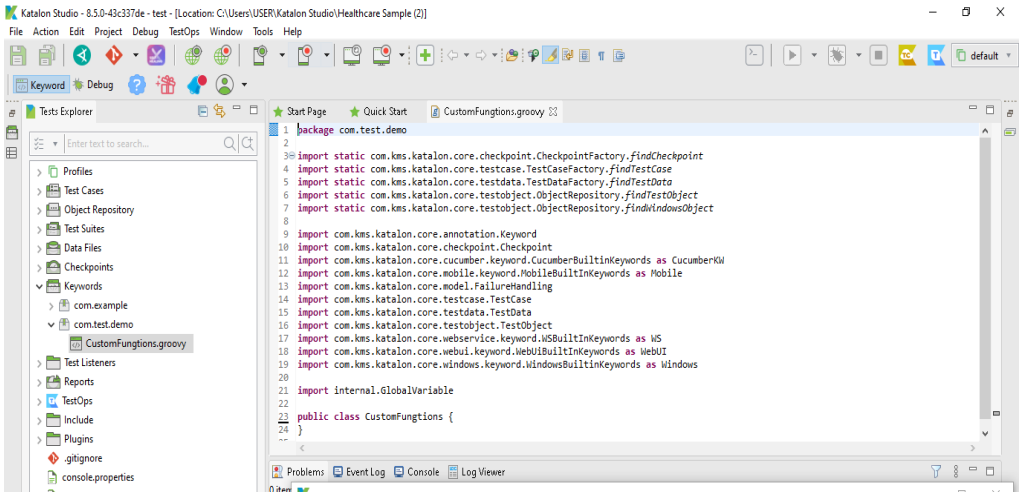
4. Langkah pertama membuat kata kunci khusus adalah klik file, lalu pilih keyword.



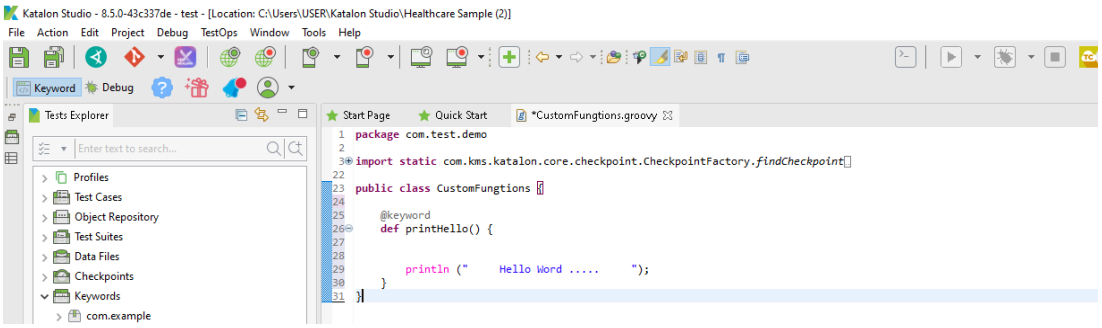
5. Kemudian inputkan package dengan nama *com.test.demo* lalu pada class name inputkan dengan nama CustomFungsions.



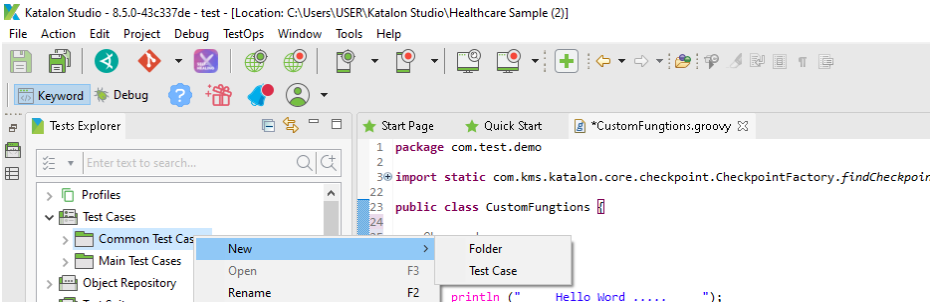
6. Setelah mengklik OK maka akan muncul tampilan seperti dibawah ini lalu Kata kunci baru dibuat di bawah paket yang ditentukan.



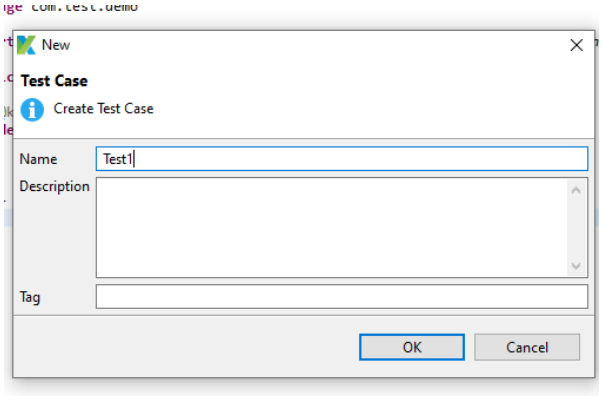
7. Masukkan kode berikut di kelas yang telah ditentukan untuk menentukan kata kunci khusus:



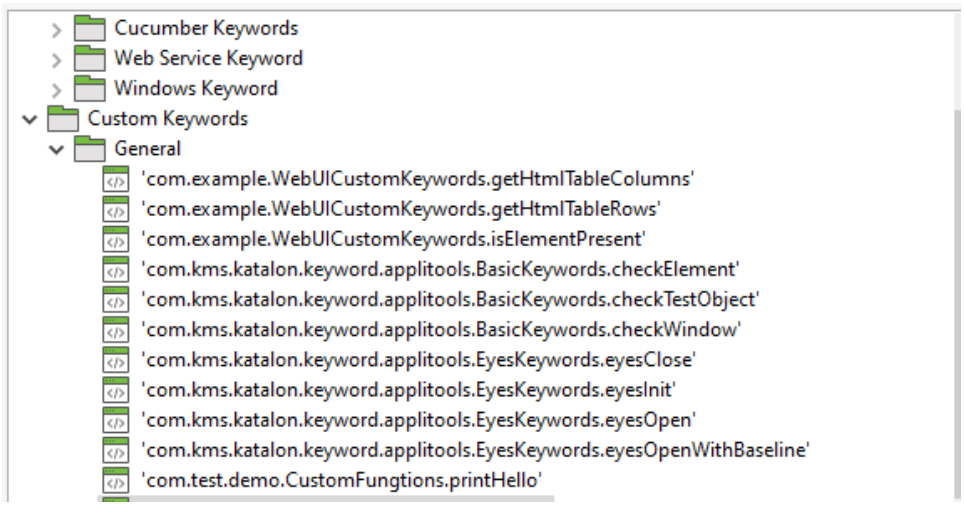
8. Kemudian buka kasus uji dalam tampilan Manual dan pilih Kata Kunci Khusus dari toolbar perintah`, pilih “NEW KLIK TEST CASE”.



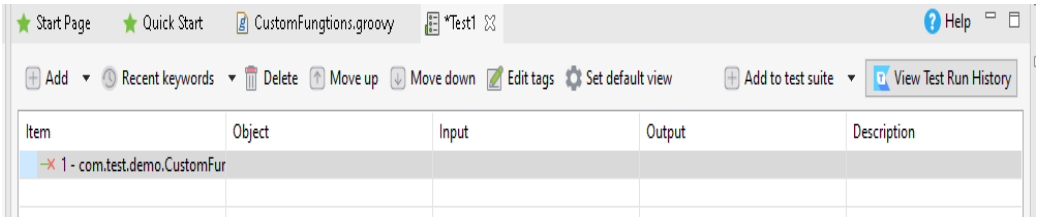
9. Inputkan name case sebagai Test1, lalu kil ok.



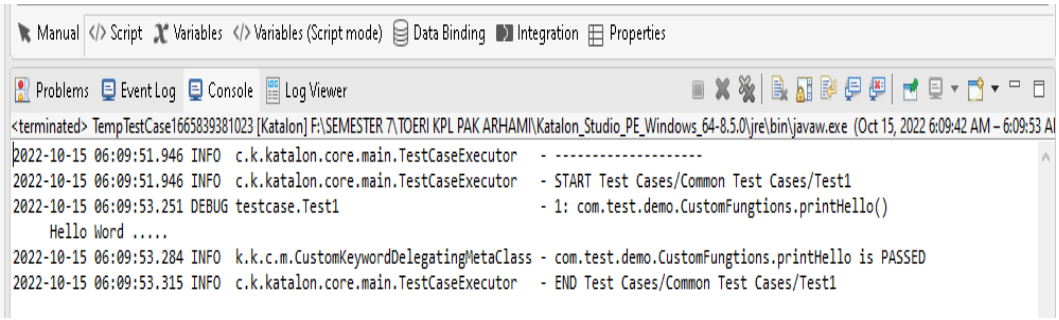
10. Selanjutnya Langkah pengujian baru ditambahkan. Pilih salah satu kata kunci khusus.



11. Setelah ditambahkan maka akan muncul tampilan seperti gambar di bawah.

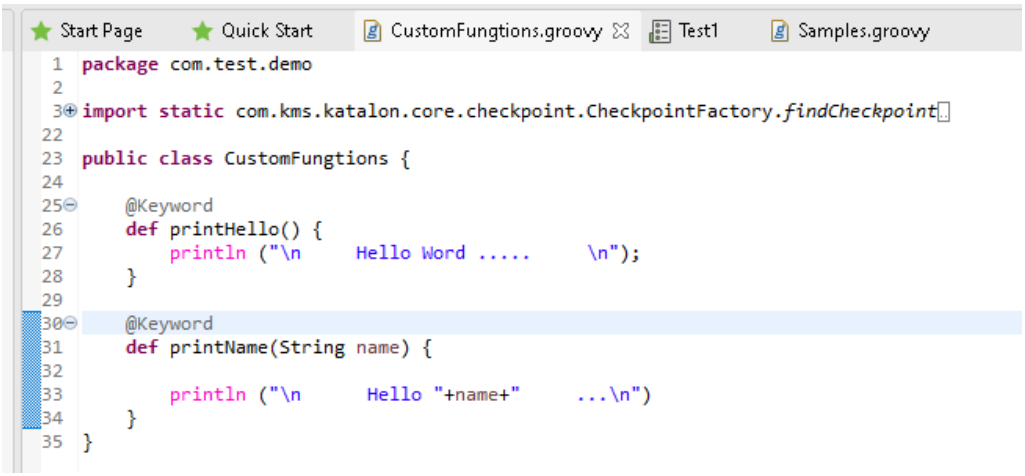


12. Selanjutnya klik running dan tunggu beberapa saat jika telah selesai klik Consule maka muncul inputan Hello word

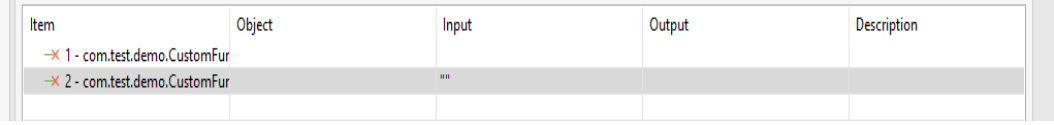


12.1.2. Bagaimana cara merujuk kata kunci khusus

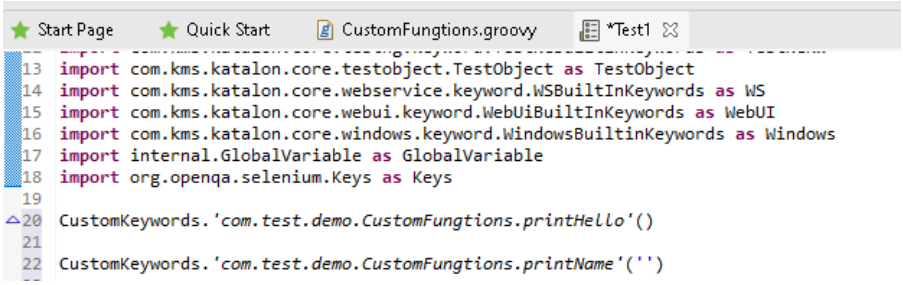
13. Pertama buat terlebih dahulu kata kunci lain, seperti gambara di bawah ini.



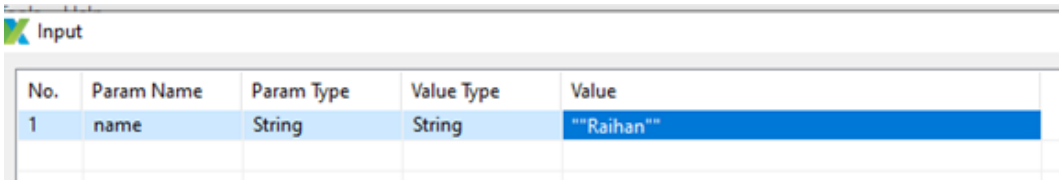
14. Untuk menginputkan file dibawahnya langkah yang dilakukan sama seperti langkah sebelumnya.



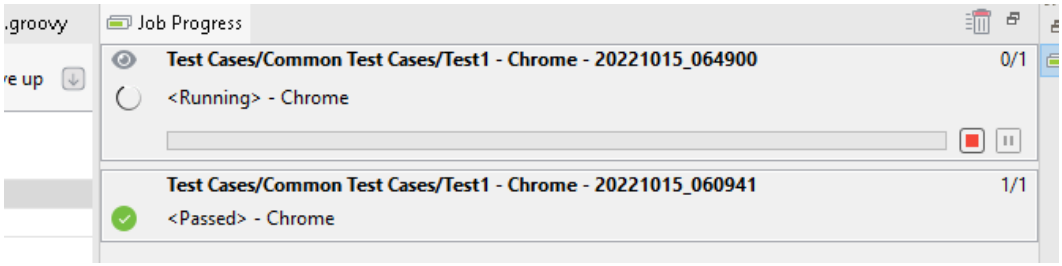
15. Dan saat menu skrip dibuka, maka kita dapat melihat tampilan seperti gambar ini.



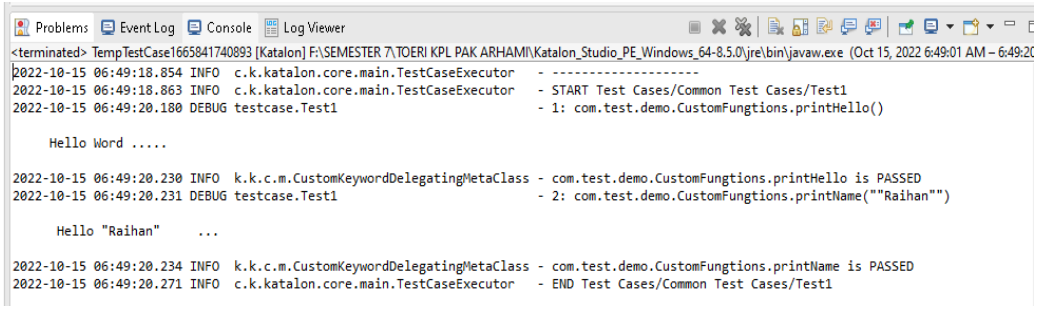
16. Kemudian kembali lagi ke halaman Test1 klik tanda petik “” lalu inputkan kata kunci contohnya “Raihan” lalu klik ok.



17. Setelah itu klik running, dan tunggu hingga proses running selesai.



18. Jika proses running telah selesai, klik Consule dan akan muncul kata kunci tambahan.



MODUL 17
WINDOWS DEKSTOP APPLICATION TESTING

A. Tujuan

1. Mempelajari tentang pengujian terhadap dekstop.
2. Mengetahui bagaimana cara melakukan pengujian terhadap dekstop dengan menggunakan katalon studio.

B. Dasar Teori

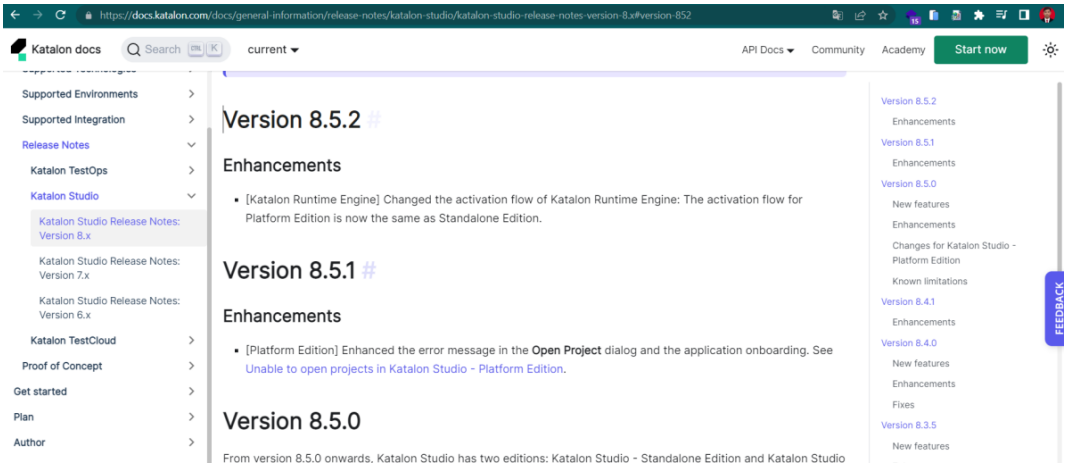
Aplikasi desktop adalah program yang berjalan secara independen di sistem operasi desktop. Tidak seperti aplikasi web, aplikasi desktop memerlukan sumber daya perangkat keras yang cukup untuk berfungsi.

Pengujian aplikasi desktop adalah praktik pengujian perangkat lunak yang memeriksa fungsionalitas, keamanan, kegunaan, dan stabilitas aplikasi setelah diterapkan. Dalam pengujian aplikasi desktop, kita perlu memperhatikan pemasangan serta pengujian penghapusan instalasi untuk sepenuhnya memenuhi persyaratan pengujian aplikasi. Apa yang bisa diuji pada dekstop dengan menggunakan katalon studio? Katalon studio dapat melakukan beberapa pengujian diantaranya :

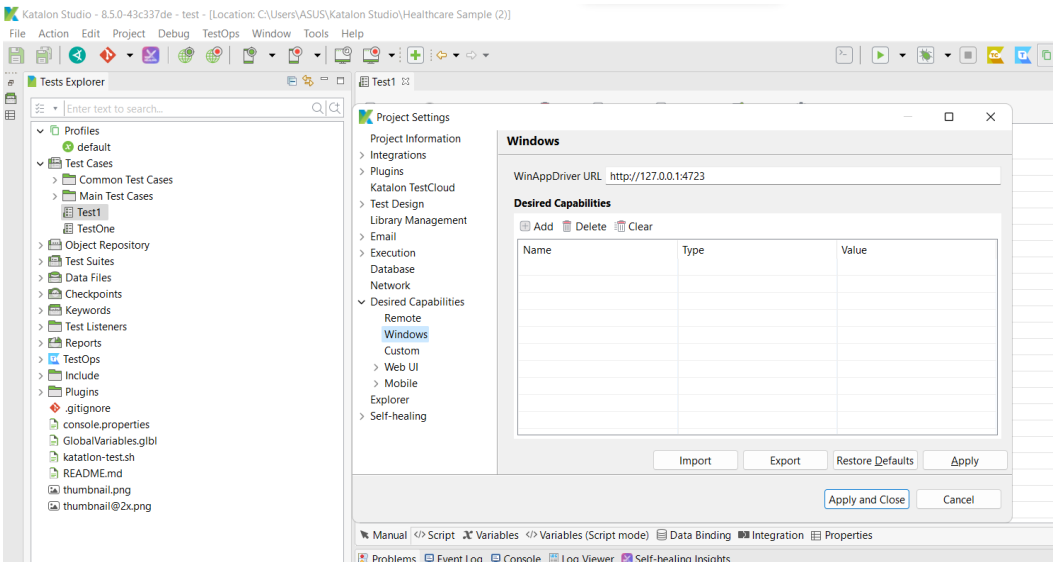
- Melakukan pengaturan dan konfigurasi yang mudah – baik secara lokal maupun jarak jauh
- Mendeteksi dan memata-matai objek Windows
- Merekam tindakan Windows
- Menguji lokasi elemen cerdas
- Menguji kata kunci bawaan dan khusus Windows Perawatan minimal

Berikut ini merupakan langkah-langkah untuk melakukan pengujian dekstop dengan menggunakan Katalon Studio.

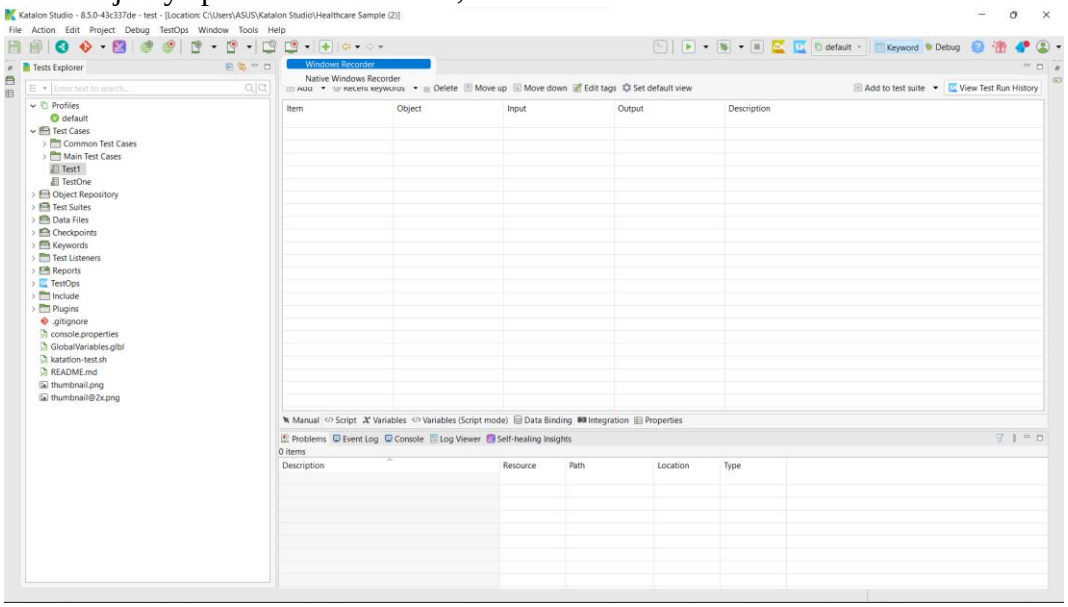
1. Sebelum melakukan pengujian terhadap dekstop pada katalon studio, pastikan katalon studio yang digunakan sudah dalam versi terbaru. Jika anda masih menggunakan katalon studio versi lama, anda dapat mendonwload versi terbaru katalon studio pada web resmi katalon studio.com. Hal ini dilakukan untuk memastikan bahwa menu atau tools pengujian terhadap dekstop telah tersedia.



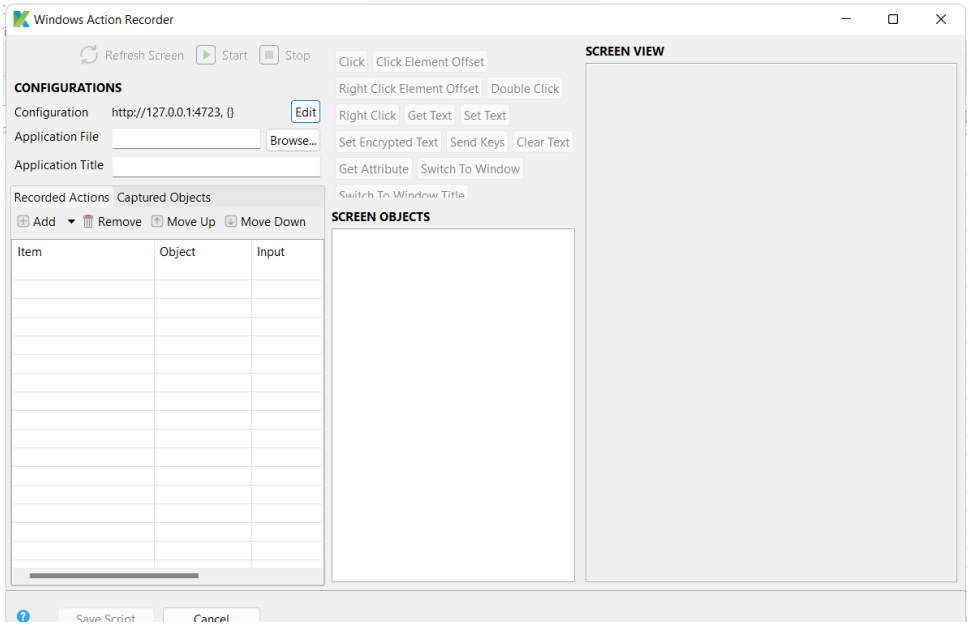
2. Setelah katalon studio berhasil di download, buka katalon studio kemudian untuk memastikan apakah katalon studio yang dipakai bisa melakukan pengujian terhadap dekstop, pada menu utama buka projek, lalu pilih Desired Capabilities, pada menu tersebut bisa dilihat bahwa pilihan untuk windows telah tersedia.



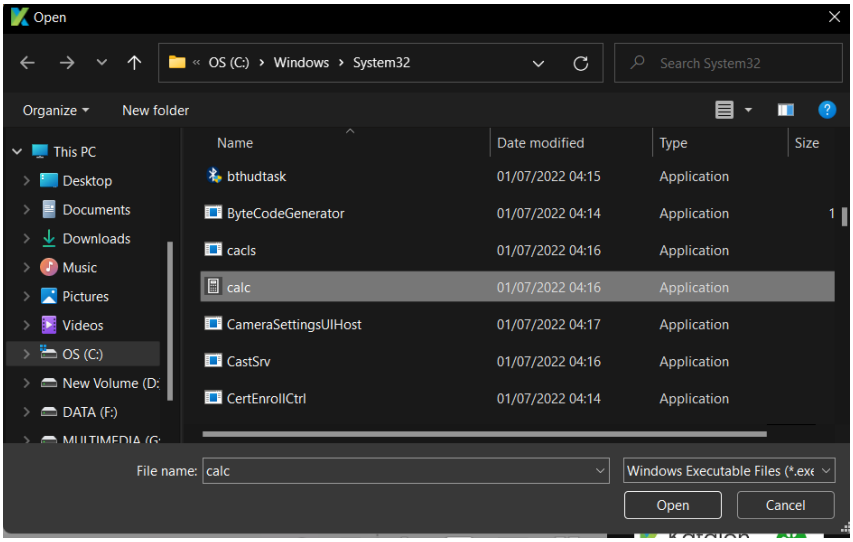
3. Selanjutnya pada menu test case, buat satu test case baru.



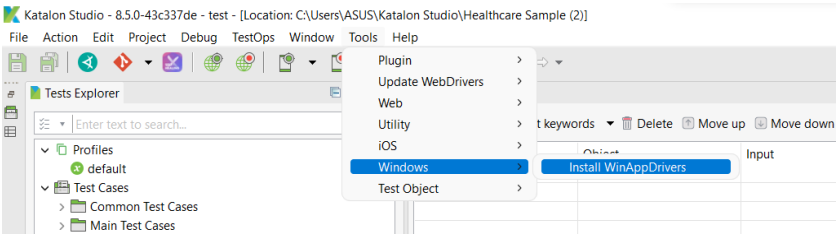
4. Klik tombol windows recorder. Setelah tombol di klik, maka user akan disuguhkan dengan tampilan jendela yang berisi rekaman tindakan windows.



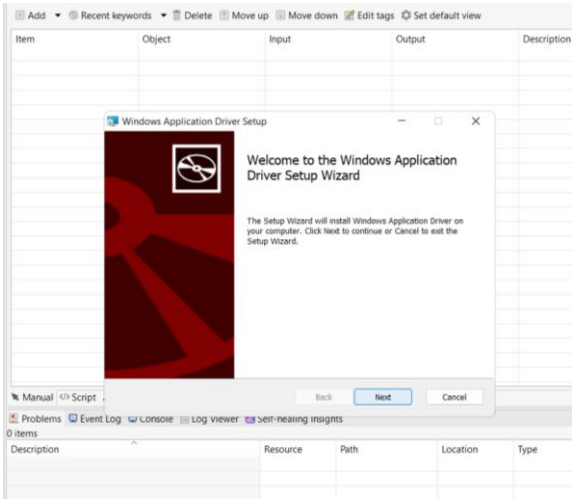
5. Pada application file, pilih file yang ingin diuji, dan biasa file yang diuji berformat .exe. Pada kasus ini saya menggunakan file calc.exe untuk pengujiannya. File calc.exe bisa di dapat pada folder AdvancedInstallers yang berada di dalam folder System32 pada localdisc C. Setelah file dipilih, lalu klik ok.



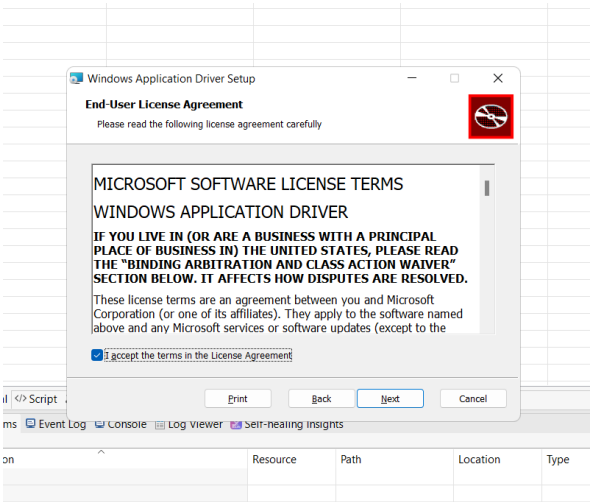
6. Sebelum melakukan proses recording, ada beberapa hal yang harus dilakukan. Terlebih dahulu kita harus menginstall winAppDriver, winAppDriver ini digunakan sebagai libarary agar bisa berinteraksi dengan aplikasi windows. Proses penginstallan dapat dilakukan dengan tools pada menu utama, pilih windows, lalu klik Install WinAppDrivers.



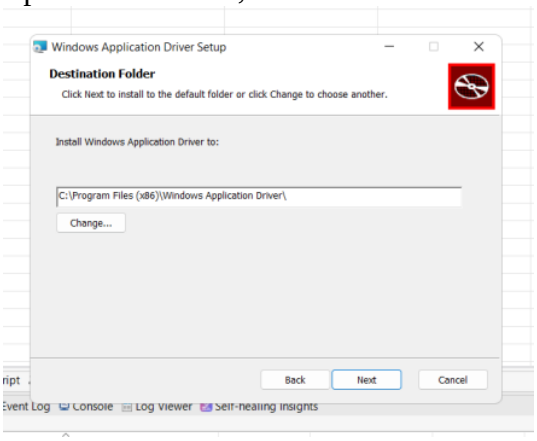
7. Pada instalasi wizard klik next saja.



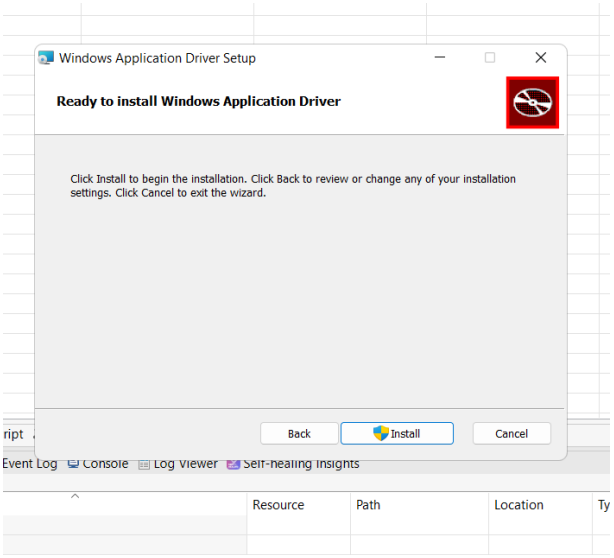
8. Pilih I accept untuk license agreement, kemudian next.



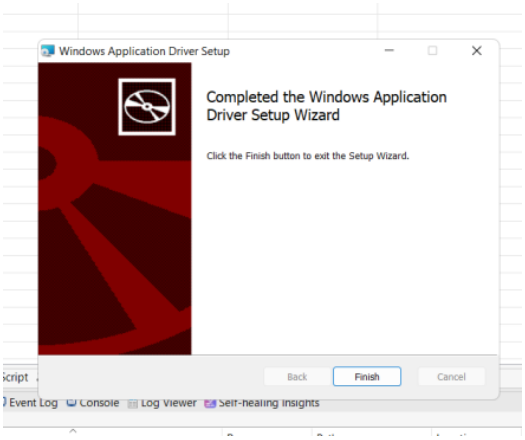
9. Pilih lokasi penyimpanan file install, lalu klik next.



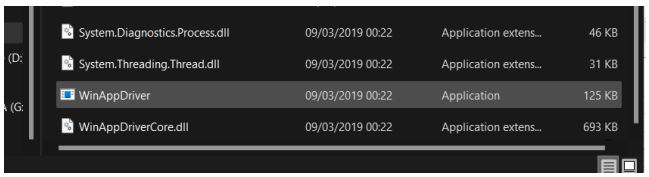
10. Kemudian klik install.



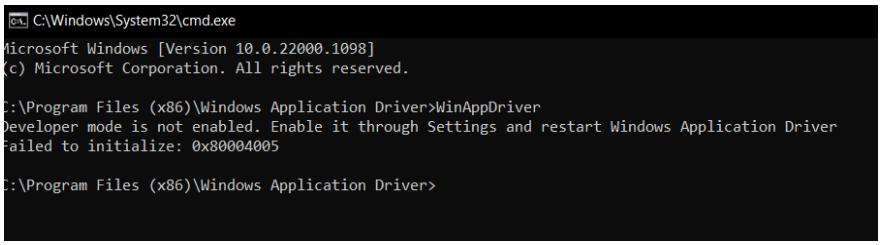
11. Tunggu sampai proses installasi selesai, lalu klik finish.



12. Berikut merupakan tampilan WinAppDriver yang sudah di install, dan tersimpan pada directory komputer. Langkah selanjutnya adalah melakukan start terhadap WinAppDriver.

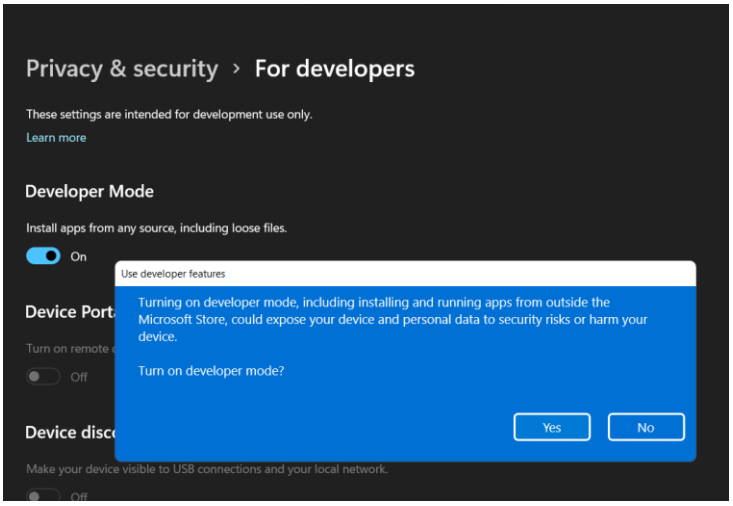


13. Start WinAppDriver bisa dilakukan dengan menggunakan CMD. Copy file path WinAppDriver, lalu paste ke CMD, kemudian klik enter.

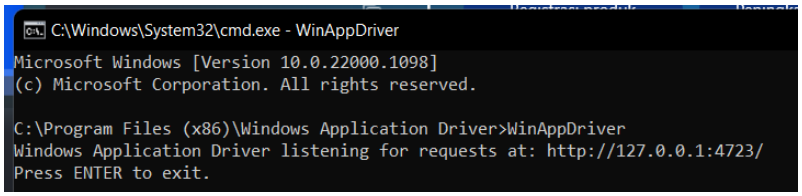


Hasil di atas menunjukkan bahwa WinAppDriver belum berhasil di start, anda dapat mengatasinya dengan mengaktifkan terlebih dahulu developer mode yang ada pada windows anda.

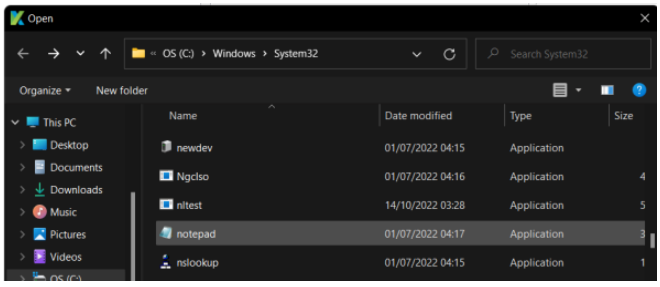
14. Anda dapat mengaktifkan developer mode pada developer setting, seperti yang terlihat pada gambar di bawah ini.



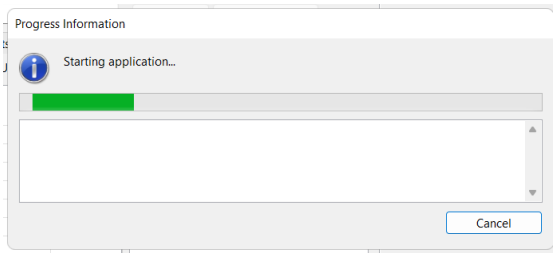
15. Setelah developer mode berhasil di aktifkan, kembali ke halaman cmd kemudian lakukan kembali proses start pada WinAppDriver. Pada kasus ini WinAppDriver sudah berhasil di start, dengan hasil yang ditampilkan seperti gambar di bawah ini.



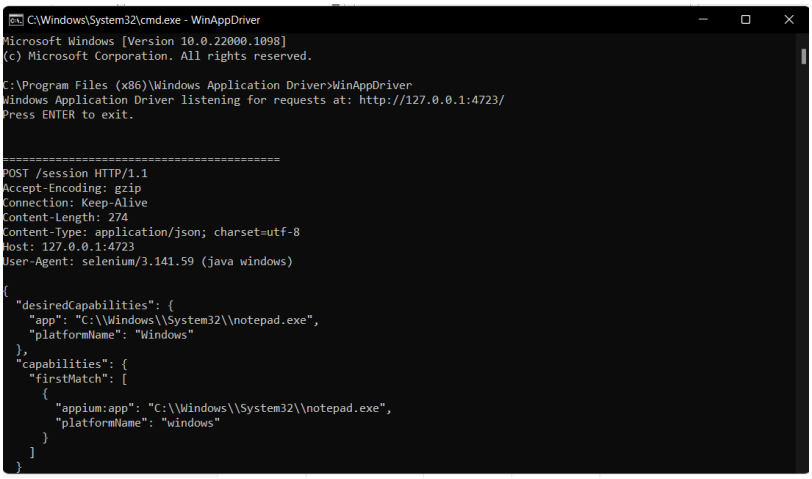
16. Selanjutnya pada katalon studio pilih application dengan notepad.exe, lalu klik open.



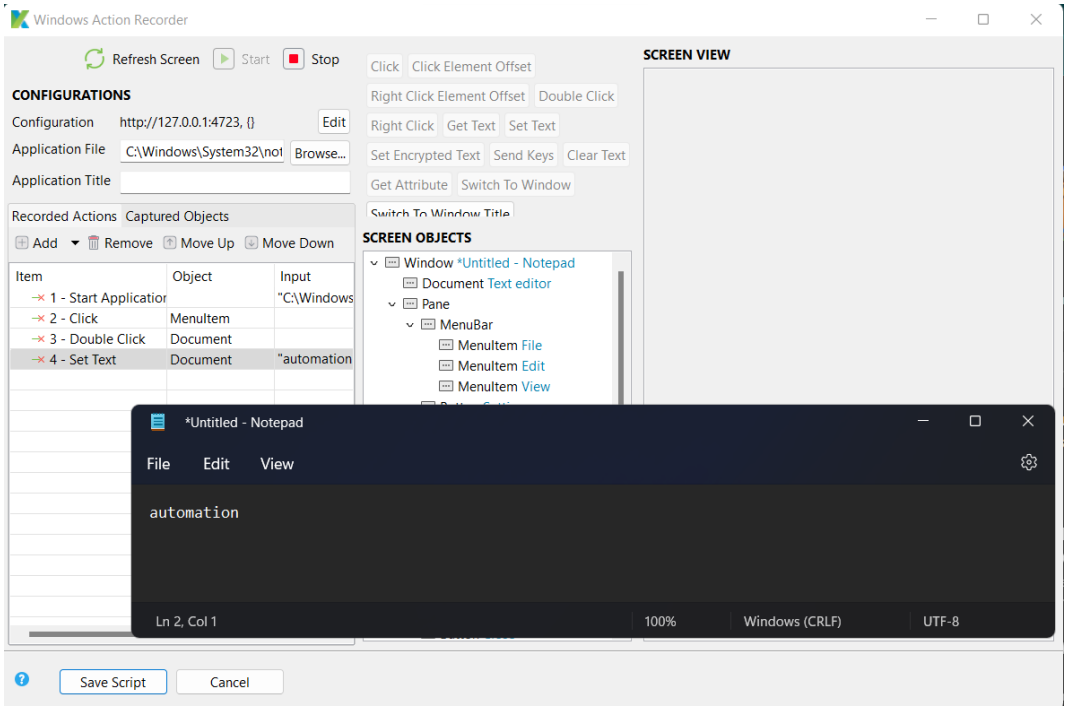
17. Setelah itu klik tombol start, dan tunggu sampai proses start selesai.



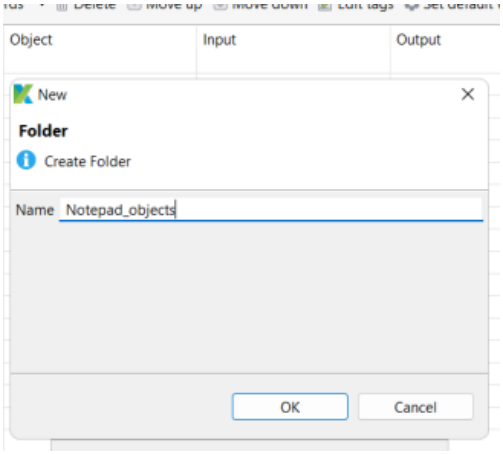
18. Kembali ke halaman cmd, kemudian klik enter. Setelah tombol enter di klik, maka user akan disuguhkan dengan tampilan jendela notepad yang ada pada sistem kita.



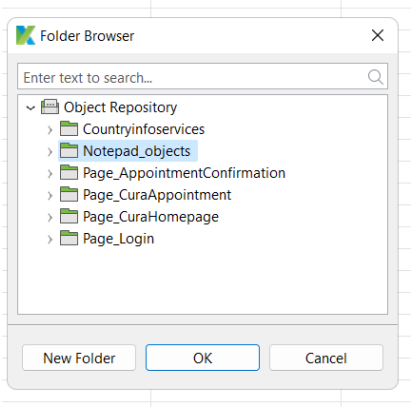
19. Selanjutnya, anda sudah dapat melakukan pengujian terhadap dekstop. Pada menu possible action, user bebas untuk melakukan proses yang ada didalamnya. Untuk kasus ini user melakukan proses click, double click, dan proses untuk menambahkan text dengan kata “automation”. Jika proses pengujian selesai dilakukan, anda dapat klik tombol stop.



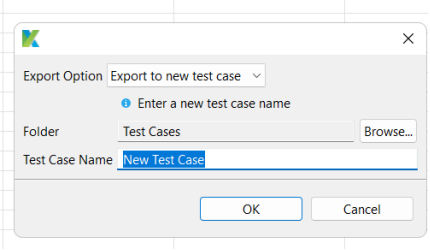
20. Kemudian, pada objek repository buat satu folder baru dengan nama Notepad_objects, lalu klik ok.



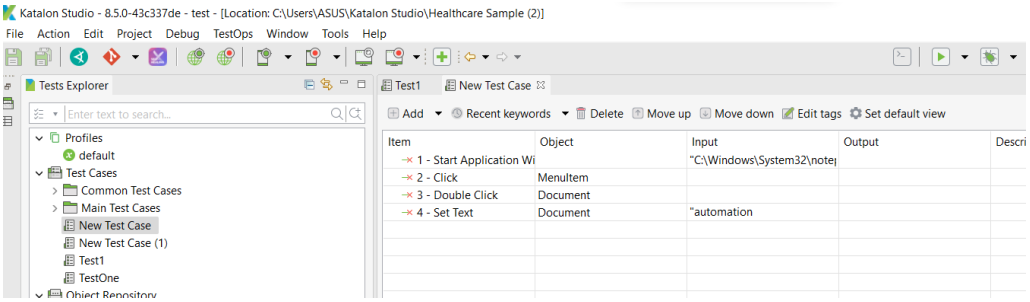
21. Folder Notepad_objects berhasil dibuat.



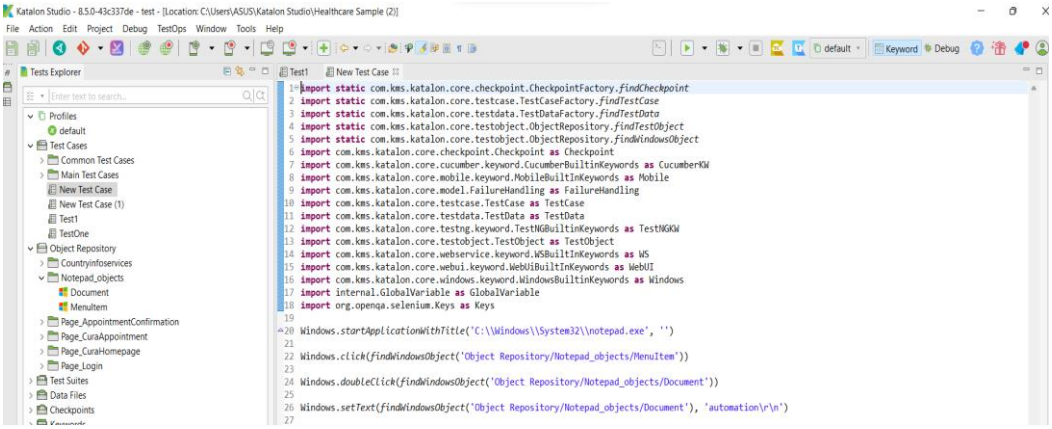
22. Pada folder Notepad_object dapat dilihat objek dekstop yang telah dibuat, kemudian, buat satu test case baru, lalu klik ok.



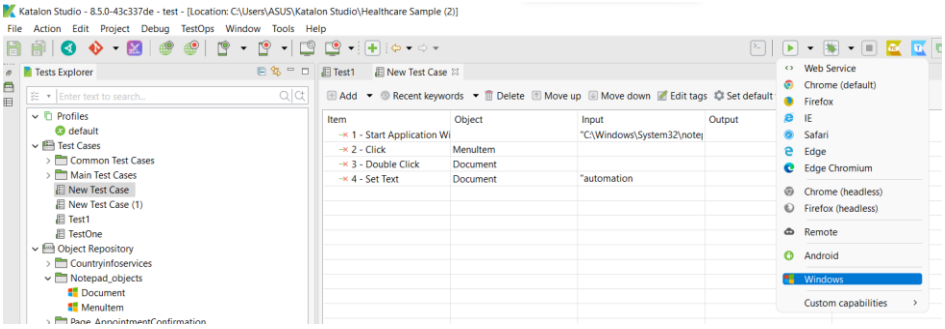
23. Pada test case tersebut dapat dilihat kasus uji yang telah direkam dan telah dilakukan sebelumnya. Seperti yang terlihat pada gambar di bawah ini.



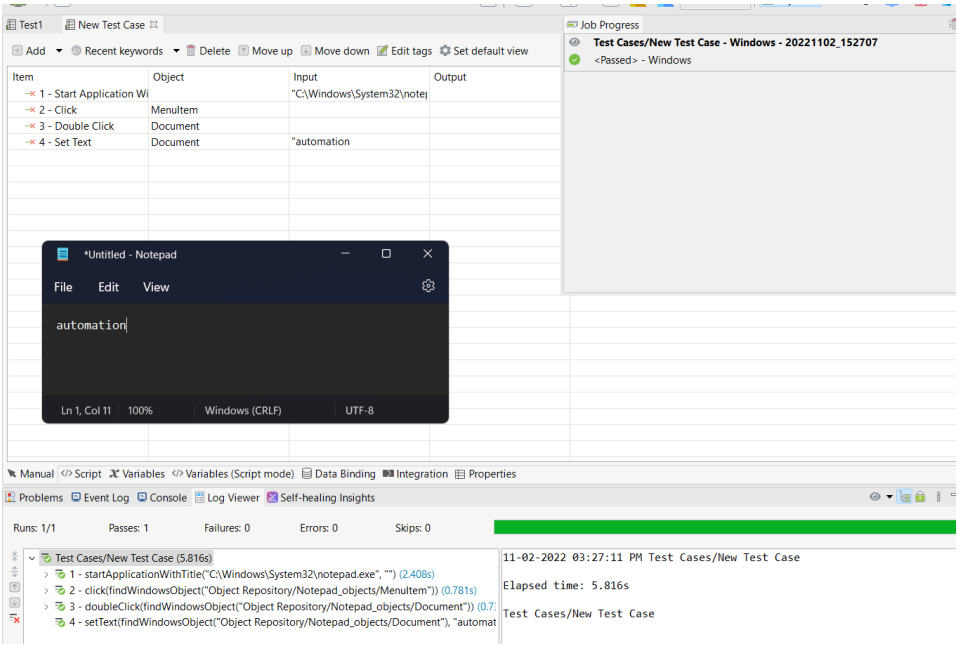
24. Kemudian jika kita lihat pada jendela script juga tertera proses-proses apa saja yang telah dilakukan saat melakukan pengujian.



25. Selanjutnya running test case untuk melihat proses eksekusinya.



26. Berikut merupakan hasil running yang telah berhasil dilakukan, dan tidak terjadi error.


Test1
New Test Case
Add
Recent keywords
Delete
Move up
Move down
Edit tags
Set default view
Item
Object
Input
Output
1 - Start Application Wi
"C:\Windows\System32\note
2 - Click
MenuItem
3 - Double Click
Document
4 - Set Text
Document
"automation
Job Progress
Test Cases/New Test Case - Windows - 20221102_152707
<Passed> - Windows
*Untitled - Notepad
File
Edit
View
automation
Ln 1, Col 11
100%
Windows (CRLF)
UTF-8
Manual
Script
Variables
Variables (Script mode)
Data Binding
Integration
Properties
Problems
Event Log
Console
Log Viewer
Self-healing Insights
Runs: 1/1
Passes: 1
Failures: 0
Errors: 0
Skips: 0
Test Cases/New Test Case (5.816s)
1 - startApplicationWithTitle("C:\Windows\System32\notepad.exe", "") (2.408s)
2 - click(findWindowsObject("Object Repository/Notepad_objects/MenuItem")) (0.781s)
3 - doubleClick(findWindowsObject("Object Repository/Notepad_objects/Document")) (0.7
4 - setText(findWindowsObject("Object Repository/Notepad_objects/Document"), "automat
11-02-2022 03:27:11 PM Test Cases/New Test Case
Elapsed time: 5.816s
Test Cases/New Test Case