



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

The screenshot shows the Katalon Studio interface with the 'Log Viewer' tab selected. A sidebar on the left contains a list of log levels: All, Info, Passed, Failed, Error, Warning, and Not Run. The 'Passed' item is highlighted with a red border. The main area displays a table of log entries:

| Level  | Time                    |
|--------|-------------------------|
| START  | 2022-10-13 07:58:54.106 |
| PASSED | 2022-10-13 07:59:00.118 |
| END    | 2022-10-13 07:59:00.120 |
| START  | 2022-10-13 07:59:00.120 |
| PASSED | 2022-10-13 07:59:03.628 |
| END    | 2022-10-13 07:59:03.630 |
| START  | 2022-10-13 07:59:03.630 |
| PASSED | 2022-10-13 07:59:04.303 |
| END    | 2022-10-13 07:59:04.305 |
| START  | 2022-10-13 07:59:04.305 |

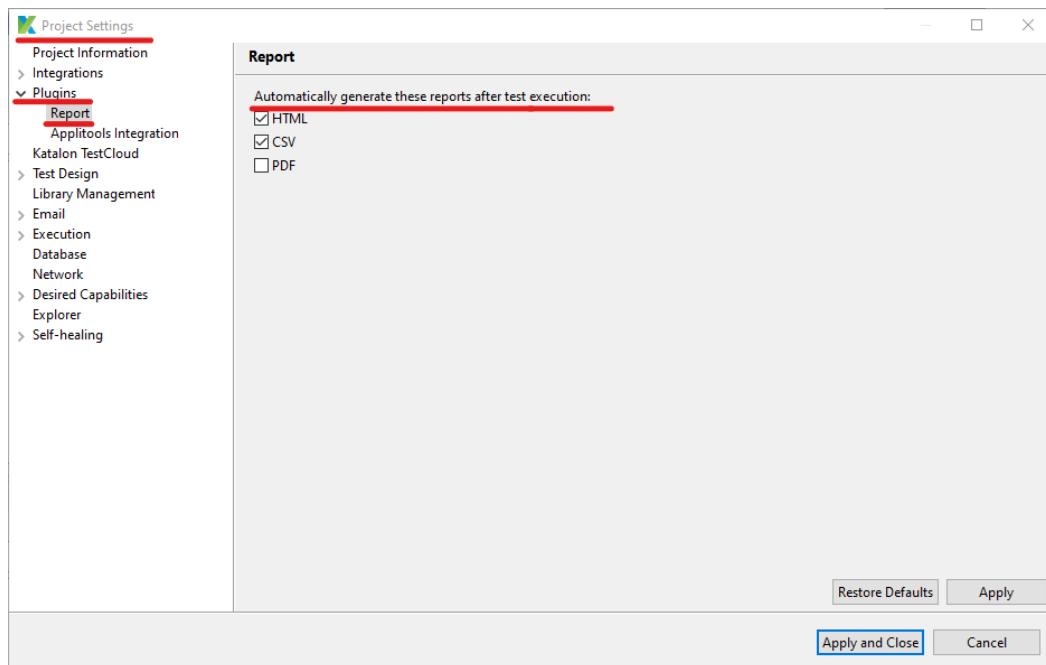
Katalon juga menyediakan tampilan hirarki yang didalamnya terdapat jendela log dan jendela detail dari setiap tahapan pengujian

The screenshot shows the Katalon Studio interface with the 'Log Viewer' tab selected. On the left, there is a tree view of test cases: 'Test Suites/TestSuiteOne (45.301s)' and 'Test Cases/Second Test Case (27.913s)'. The 'Second Test Case' node is expanded, showing its steps: '1: openBrowser()', '2: navigateToUrl("https://opensource-demo.orangehrmlive.com/web/index.php/auth/login")', '3: select(findTestObject("test2/Page\_OrangeHRM/input\_Username\_username"), "Admin")', '4: select(findTestObject("test2/Page\_OrangeHRM/input\_Password\_password"), "hUmKwJbOfgF'), '5: click(findTestObject("test2/Page\_OrangeHRM/button\_Login"))', and '6: closeBrowser()'. To the right of the tree view is a large text area displaying the log output for the second test case, including timestamped logs and elapsed times for each step.

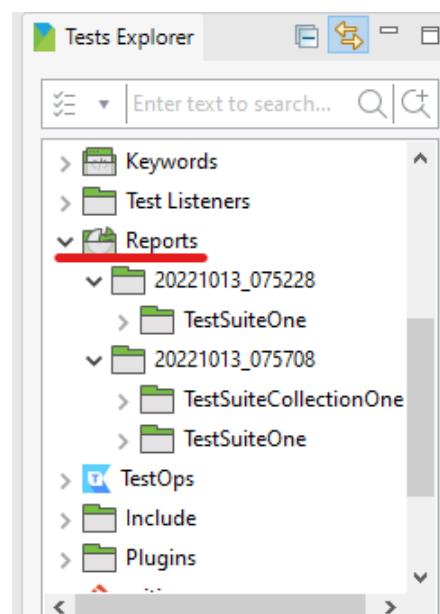
## 2. Report

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.



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



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



Berikut ini adalah tampilan jendela report secara keseluruhan

A screenshot of the Katalon Studio interface. The main window shows the 'Test Cases Table' with two entries: 'Second Test Case (18.795s)' and 'Third Test Case (17.696s)'. Above the table, there are several checkboxes for filtering results: Passed, Failed, Error, Incomplete, and Skipped. Below the table, there is a summary section with execution details: Test Suite ID (Test Suites/TestSuiteOne), Host name (WINDOWS X - DESKTOP-JR2AETD), Katalon version (8.5.0.208), Start (2022-10-13 07:52:59), Elapsed (37.150s), Local OS (Windows 10 64bit), Platform (Chrome 106.0.0.0), and End (2022-10-13 07:53:36). The interface has a clean, modern design with a light blue header and various toolbars and panels on the sides.

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.

A detailed screenshot of the 'Test Cases Table' from Katalon Studio. At the top, there is a row of checkboxes for filtering: Passed (checked), Failed, Error, Incomplete, and Skipped. Below this is a search bar labeled 'Search here...'. The main table has columns for 'No.' and 'Name'. It contains two rows of data: '1 Second Test Case (18.795s)' and '2 Third Test Case (17.696s)'. The 'Name' column for each row includes a small green checkmark icon followed by the test case name and duration.



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

| Item   | Description | Elapsed |
|--|-------------|---------|
| > 1. openBrowser("")   |             | 8.137s  |
| > 2. navigateToUrl("https://opensource-demo.orangehrmlive.com/web/index.php/auth/login")                         |             | 5.621s  |
| > 3. setText(findTestObject("test2/Page_OrangeHRM/input_Username_username"), "Admin")                            |             | 1.762s  |
| > 4. setEncryptedText(findTestObject("test2/Page_OrangeHRM/input_Password_password"), "hUKwJTbogPU9eVlw/CnDQ==") |             | 0.677s  |
| > 5. click(findTestObject("test2/Page_OrangeHRM/button_Login"))  |             | 0.495s  |
| > 6. closeBrowser()  |             | 0.894s  |

## 5.2 BASIC REPORTS PLUGINS

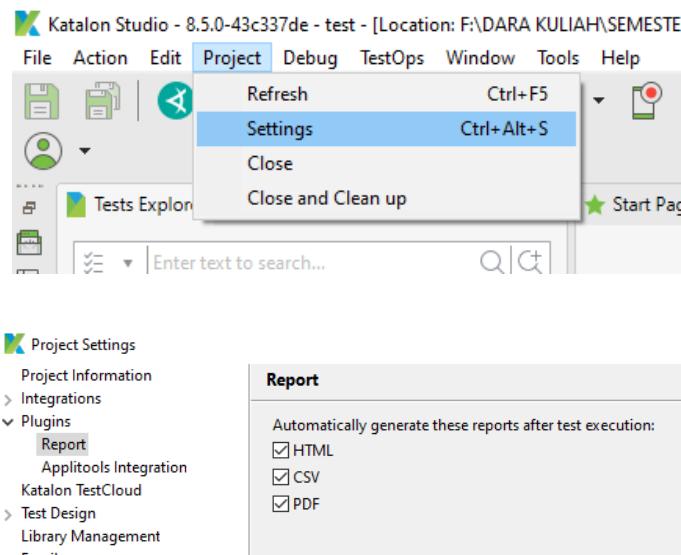
- Pada tahapan sebelumnya kita sudah banyak membuat test case, test suit, maupun test suit collection pada katalon. Salah satunya test case seperti yang terlihat pada gambar di bawah ini.

| Item                       | Object                  | Input  | Output |
|----------------------------|-------------------------|--|--------|
| 1 - Open Browser           |                         | ""   |        |
| 2 - Navigate To Url        |                         | "https://opensource-demo.orangehrmlive.com/web/index.php/auth/login" |        |
| 3 - Set Text               | input_Username_username | "Admin"  |        |
| 4 - Set Encrypted Text     | input_Password_password | "hUKwJTbogPU9eVlw/CnDQ=="  |        |
| 5 - Click                  | button_Login            |  |        |
| 6 - Click                  | button_Search           |  |        |
| 7 - Click                  | a_Time                  |  |        |
| 8 - Verify Element Present | a_Time                  | 0  |        |
| 9 - Close Browser          |                         |  |        |

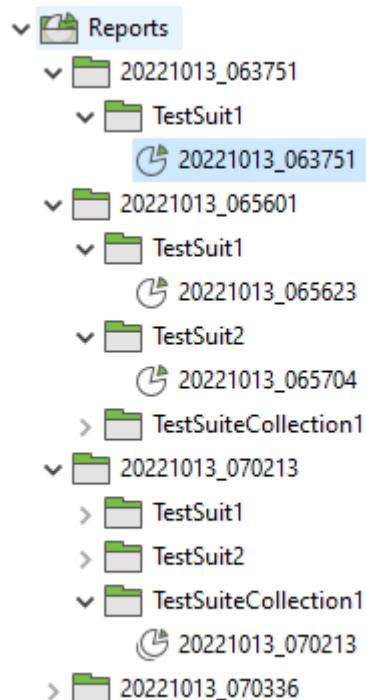
  

| No. | Name                 | Resolution |
|-----|----------------------|------------|
| 1   | Test1 (34.378s)      |            |
| 2   | TestCase_2 (24.477s) |            |

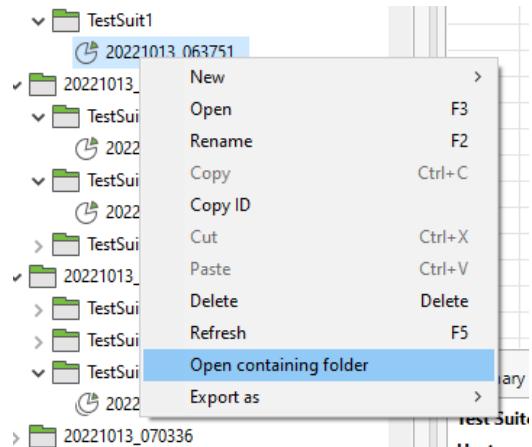
- Sebelum kita mulai mengeksport laporan hasil pengujian dalam format HTML, CSV, atay PDF, terlebih dahulu kita melakukan setting pada menu projek, dimana untuk semua plugins report ceklis untuk semua format pilihan yang ada.



- 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.



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



- Selanjutnya user akan diarahkan ke halaman lokasi tempat penyimpanan folder.

is PC > MASTER (F:) > DARA KULIAH > SEMESTER 7 > PL > TUGAS > PRAKTIKUM > Healthcare Sample > Reports

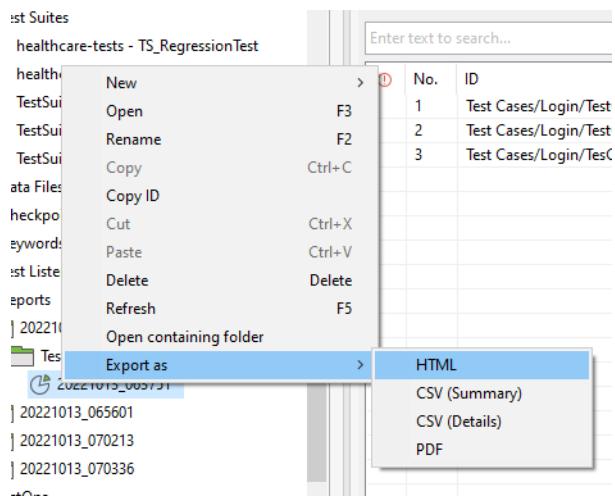
| Name            | Date modified      | Type        | Size |
|-----------------|--------------------|-------------|------|
| 20221013_063751 | 10/13/2022 6:39 AM | File folder |      |

- 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.

|                      |                    |                       |        |
|----------------------|--------------------|-----------------------|--------|
| 20221013_063751.csv  | 10/13/2022 6:39 AM | Microsoft Office E... | 3 KB   |
| 20221013_063751.html | 10/13/2022 6:39 AM | Chrome HTML Do...     | 202 KB |



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**

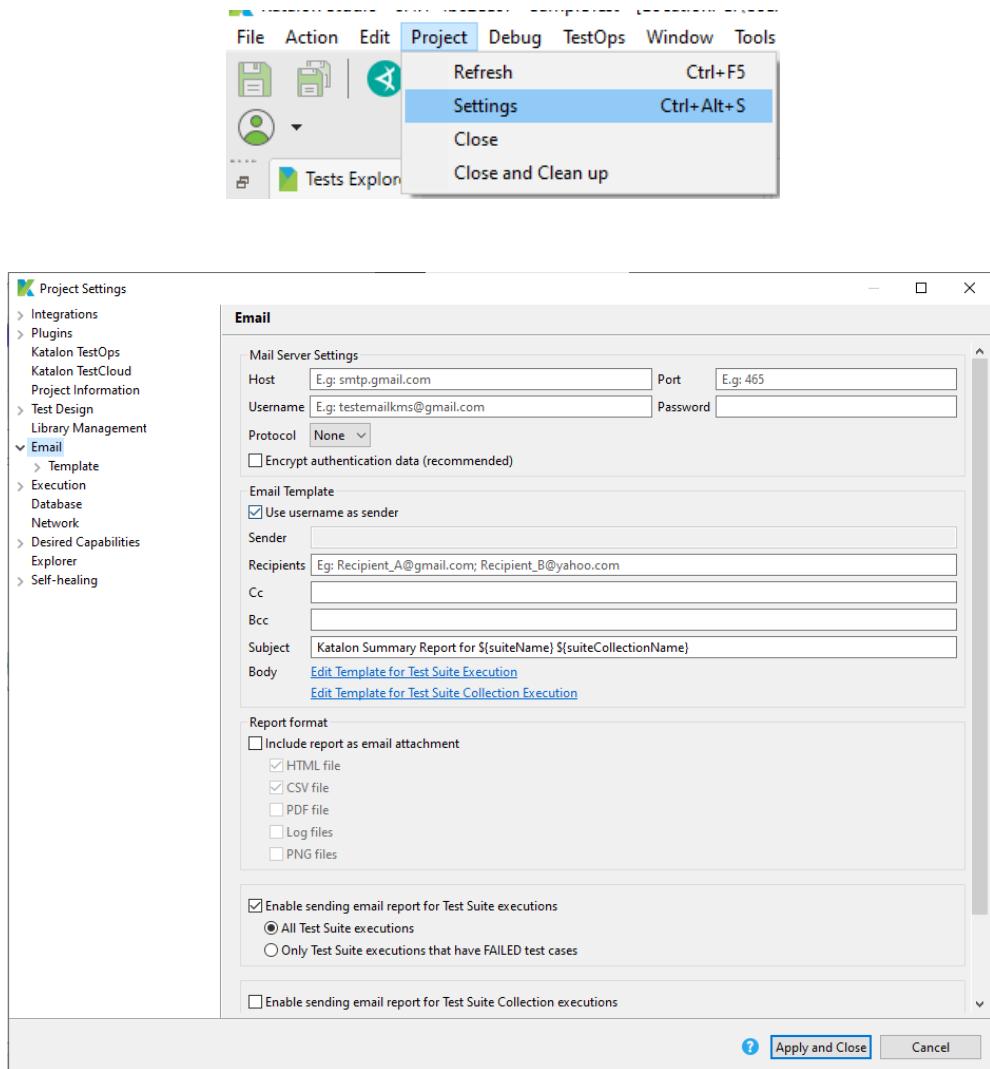
| ID      | Description         | Status              |
|---------|---------------------|---------------------|
| Total   | 2                   |                     |
| Passed  | 2                   | Passed              |
| Error   | 0                   | Incomplete          |
| Skipped | 0                   |                     |
| Start   | 2022-10-13 06:38:26 | End                 |
| Elapsed | 1m - 1.449s         | 2022-10-13 06:39:28 |

| # | 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

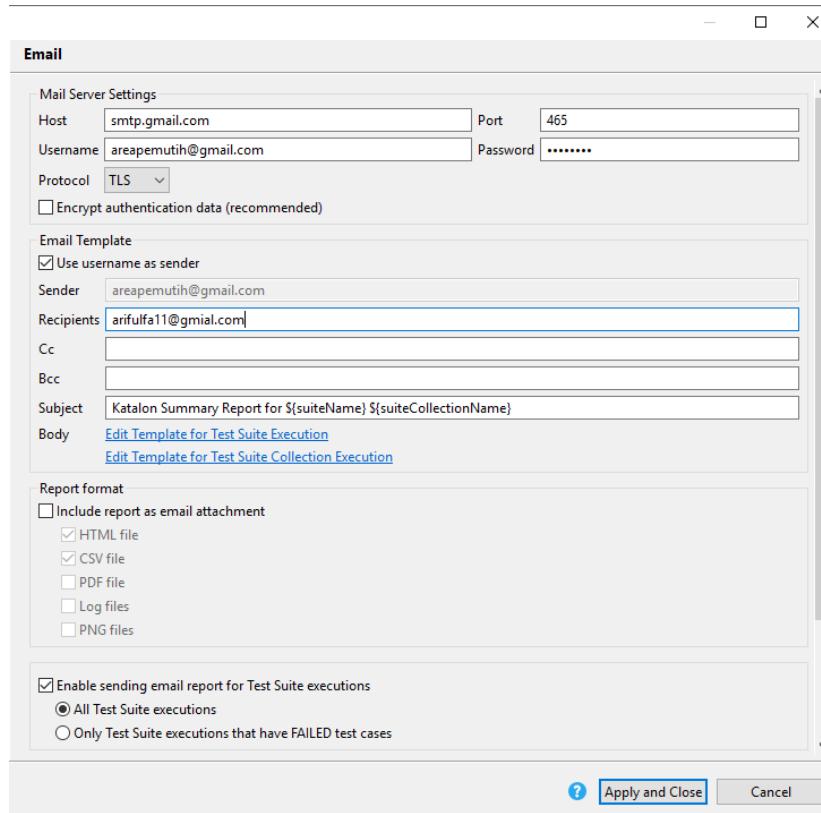


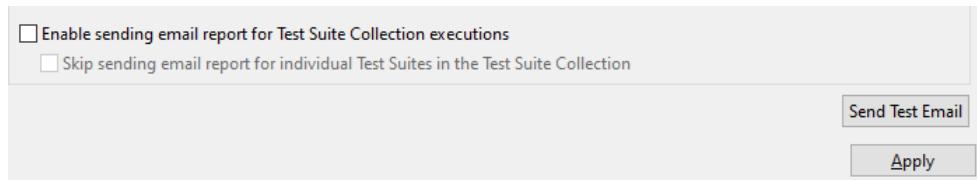
2. Sebelum mengisi kolom email, pahami dulu yang dimaksud dengan *Mail Server Settings*

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

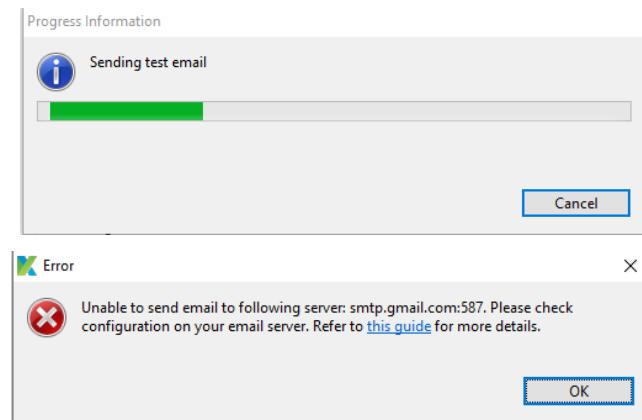
Sumber: <https://docs.katalon.com/docs/analyze/reports/manage-reports/share-test-reports-via-email-in-katalon-studio#ariaid-title1>

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*.

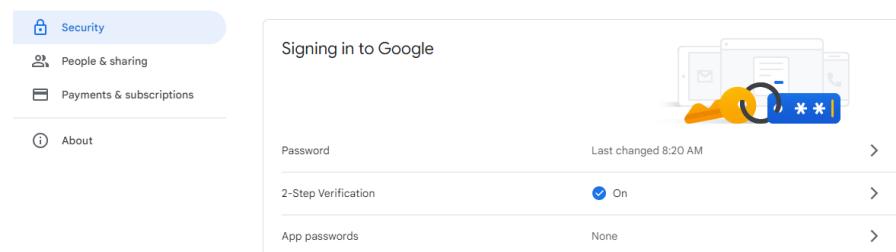




4. Tunggu proses berjalan, dan jika terjadi *error*, ikuti langkah selanjutnya



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)*

← App passwords

App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. [Learn more](#)

You don't have any app passwords.

Select the app and device you want to generate the app password for.

|                            |               |
|----------------------------|---------------|
| Select app                 | Select device |
| Mail                       | ▼             |
| Calendar                   |               |
| Contacts                   |               |
| YouTube                    |               |
| <b>Other (Custom name)</b> |               |

**GENERATE**

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

You don't have any app passwords.

Select the app and device you want to generate the app password for.

Katalon Studio X

**GENERATE**

9. Kemudian anda akan diberikan *password* otomatis, kemudian klik *DONE*

10. Secara otomatis muncul daftar *App Password*

← App passwords

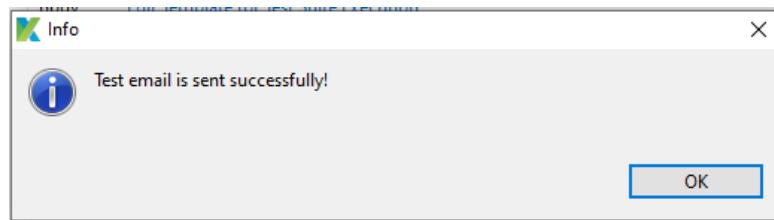
App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. [Learn more](#)

| Your app passwords   |         |               |   |
|--|---------|---------------|---|
| Name   | Created | Last used     |   |
| Katalon Studio   | 9:02 AM | -             |   |
| Select the app and device you want to generate the app password for. |         |               |   |
| Select app   | ▼       | Select device | ▼ |
| <b>GENERATE</b>  |         |               |   |

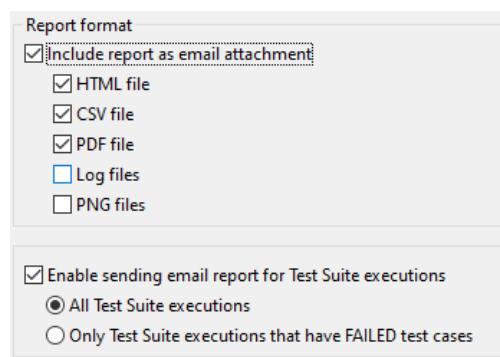
11. Copy-kan *password* sebelumnya, dan simpan pada bagian kolom *Password*, kemudian lakukan *Send Test Email* ulang.

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



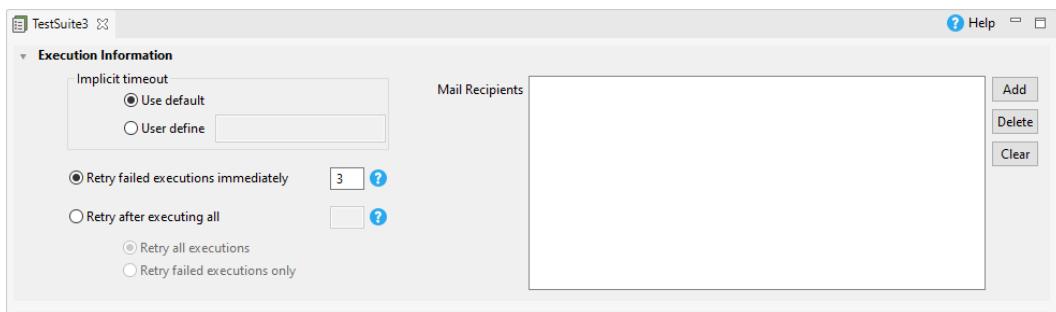
13. Jika tidak terdapat pada Kotak Masuk, kemungkinan akan dianggap sebagai *spam*
14. Langkah berikutnya, atur pada bagian *Report format* dengan berikut, dan klik *Apply* setelah melakukan konfigurasi



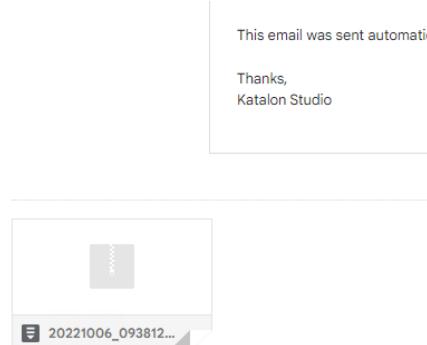
15. Adapun untuk mengubah *template* dari *email*, anda dapat melakukannya pada bagian Email>Template>Test Suite atau Test Suite Collection.

A screenshot of the Katalon Studio interface. On the left is a sidebar with navigation items like Integrations, Plugins, Katalon TestOps, Katalon TestCloud, Project Information, Test Design, Library Management, Email, Template, Execution, Database, Network, Desired Capabilities, Explorer, and Self-healing. Under 'Email' &gt; 'Template', 'Test Suite' is selected. The main area is titled 'Test Suite' and shows a rich-text editor toolbar. Below it is a preview window with the Katalon logo and the text 'Test S'. The preview content includes a greeting 'Dear Sir/Madam,' and a message 'Your test suite has just finished its execution. He'. At the bottom is a table with two rows: 'Host Name' and '\${hostName}' in the first column, and 'Operating System' and '\${os}' in the second column.

16. Setelah melakukan konfigurasi *email*, selanjutnya buka file *Test Suite* anda, lakukan *expand* terhadap *Execution Information*. Pada jendela tersebut terdapat informasi mengenai *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.

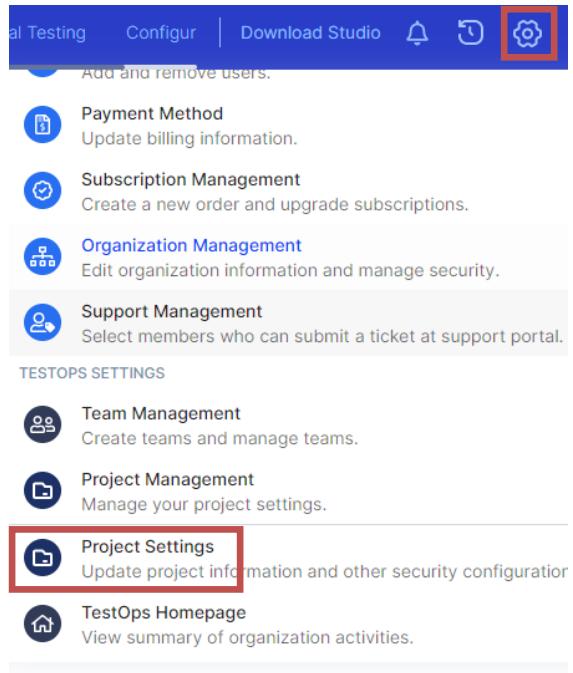


Setelah mempelajari modul ini, mahasiswa diharapkan sudah dapat melakukan pengujian beserta mengirim *email* hasil pengujian dengan berhasil.

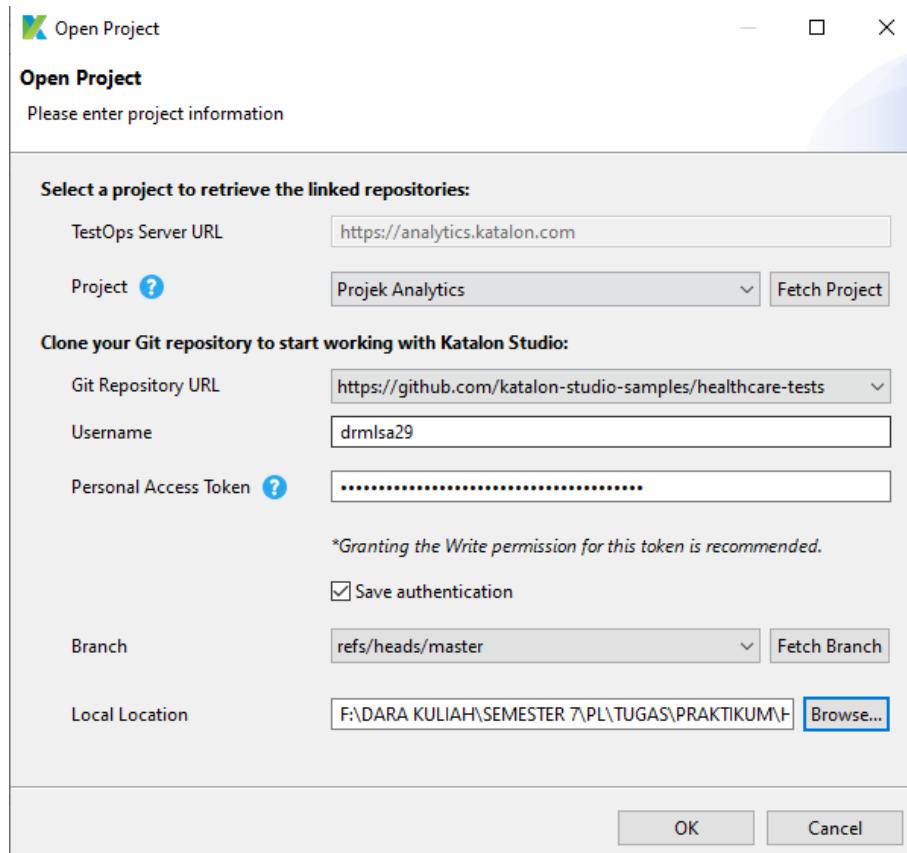
## 5.4 HOW TO USE KATALON ANALYTICS STEP BY STEP

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.

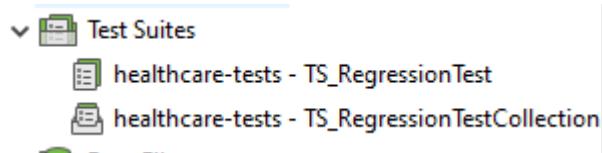


2. Selanjutnya, buka aplikasi katalon kemudian klik file lalu open projek. Masukkan username dan token dari akun github anda, lalu pada bagian projek pilih nama projek yang sudah dibuat pada web katalon analytics sebelumnya, pada kasus ini saya membuat projek dengan nama projek analytics. Pada proses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

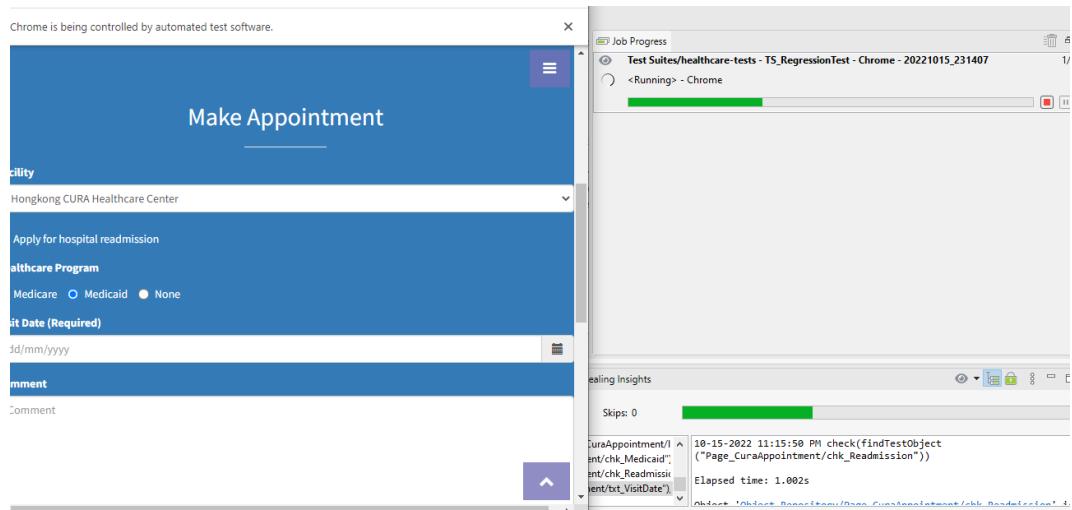
4. Selanjutnya kita akan menjalankan test suit sebagai bentuk percobaannya. Test suit ini merupakan test case yang sudah disediakan oleh katalon sebagai sampel pengujian. Untuk kasus ini saya memilih test case TR\_RegressioonTest.



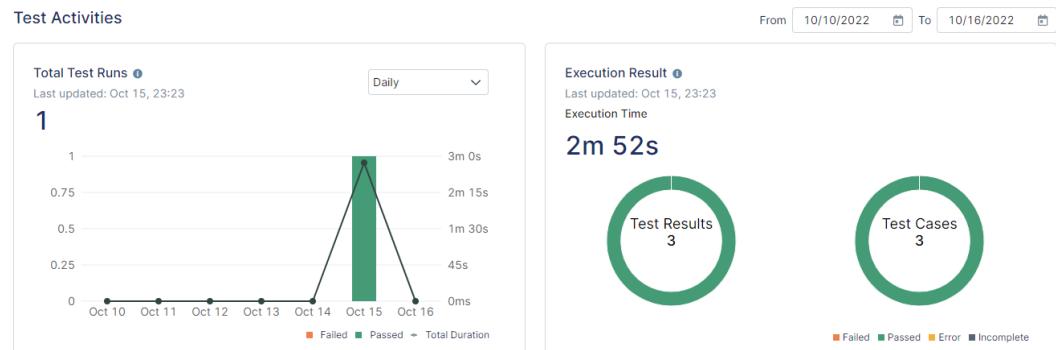
Seperti yang dapat dilihat pada gambar di bawah ini, pada test suit tersebut sudah ditambahkan beberapa test case.

| Enter text to search... |     |  |
|-------------------------|-----|--|
|                         | No. | ID   |
|                         | 1   | Test Cases/Main Test Cases/TC1_Verify Successful Login       |
|                         | 2   | Test Cases/Main Test Cases/TC2_Verify Successful Appointment |
|                         | 3   | Test Cases/Main Test Cases/TC3_Visual Testing Example        |

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



- 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 dilakukan.



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

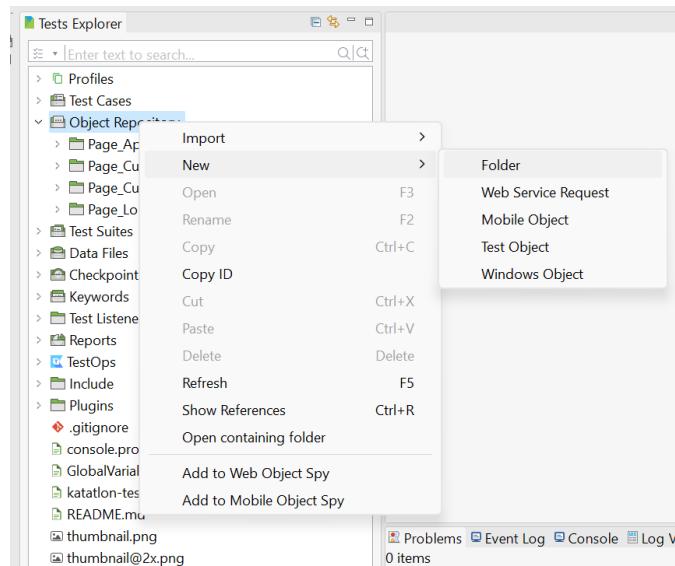
## MODUL 9

### API TESTING

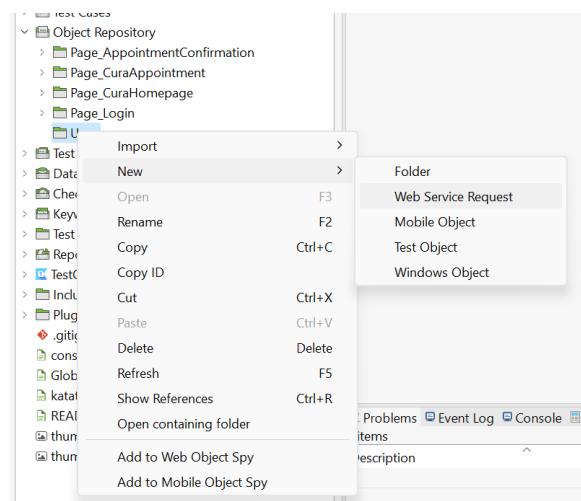
#### 9.1. HOW TO DO API TESTING WITH KATALON

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

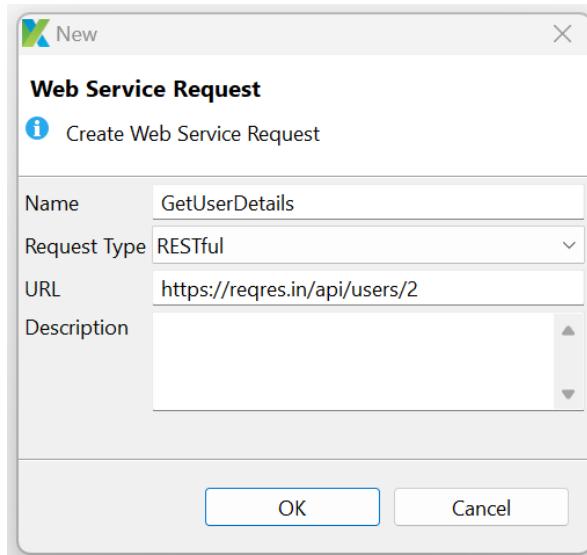
1. Buka project pada katalon, pada bagian Test Explorer klik kanan pada folder Object Repository untuk membuat folder baru.



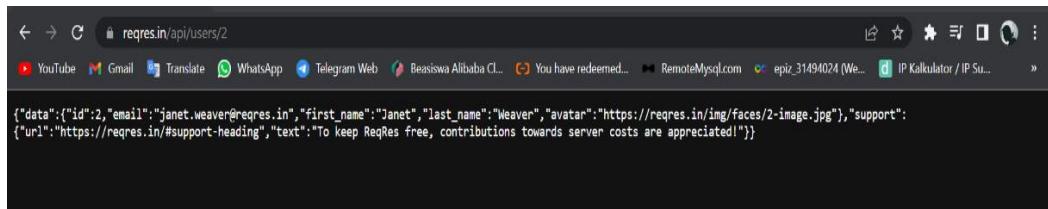
2. Setelah folder berhasil dibuat klik kanan, pilih web service Request



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 tampilan hasilnya, yaitu GET user dengan id 2 berhasil dilakukan, ditandai dari Responsenya yaitu status 200 OK

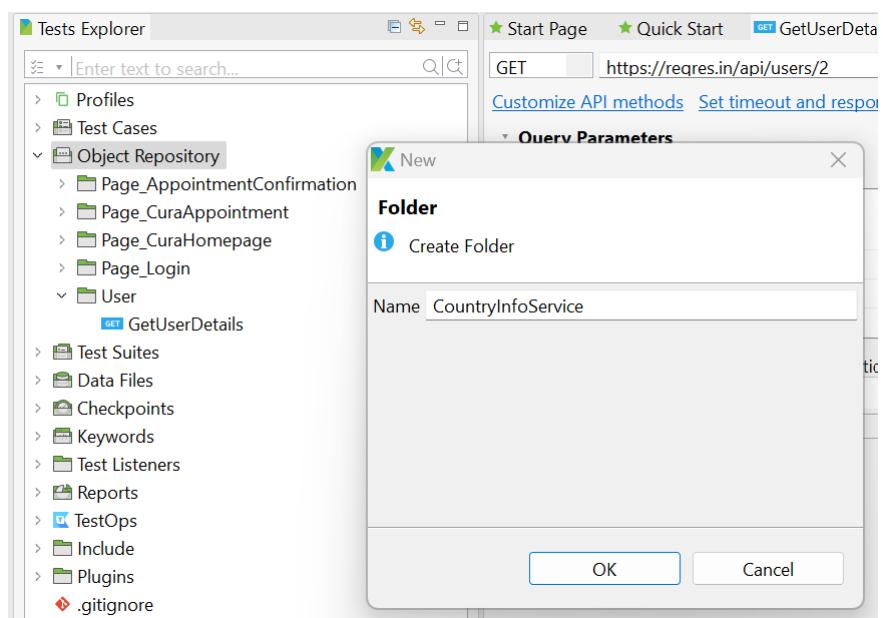
```

1  {
2    "data": {
3      "id": 2,
4
5      "email": "janet.weaver@reqres.in",
6      "first_name": "Janet",
7      "last_name": "Weaver",
8
9      "avatar": "https://reqres.in/img/faces/2-image.jpg"
10     },
11     "support": {
12       "url": "https://reqres.in/#support-link"
13     }
14   }
  
```

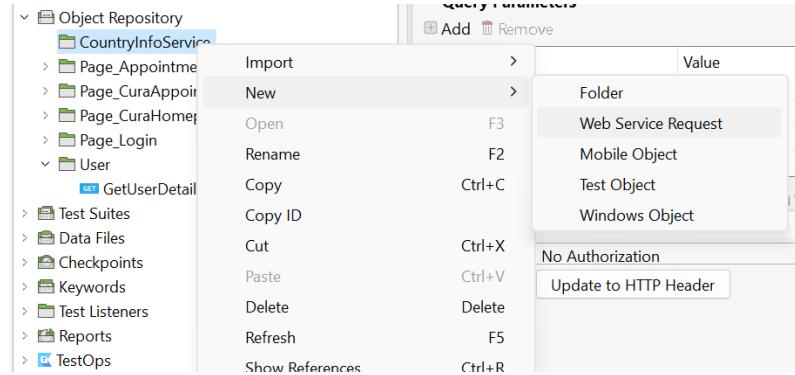
Select JSON or XML response data and press Ctrl/Command + K to add verification scripts directly.

JSON  XML  HTML  JavaScript  Wrap Lin

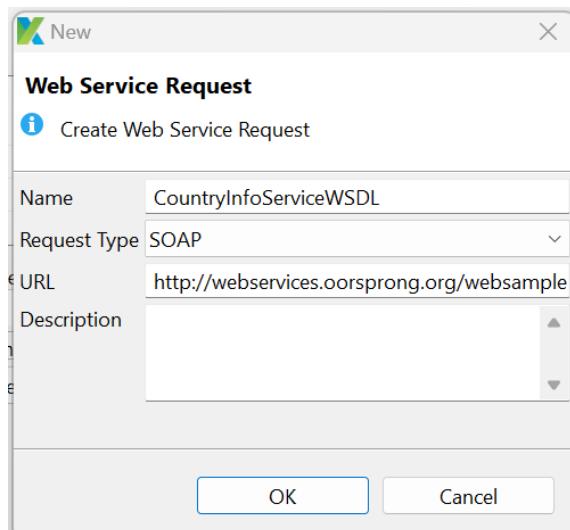
7. Selanjutnya buat folder baru dengan nama “CountryInfoService” pada Object Repository



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.

Google

country info service wsdl

Sekitar 261.000 hasil (0,48 detik)

<http://webservices.oorsprong.org> > CountryInfoService

**Service Description**

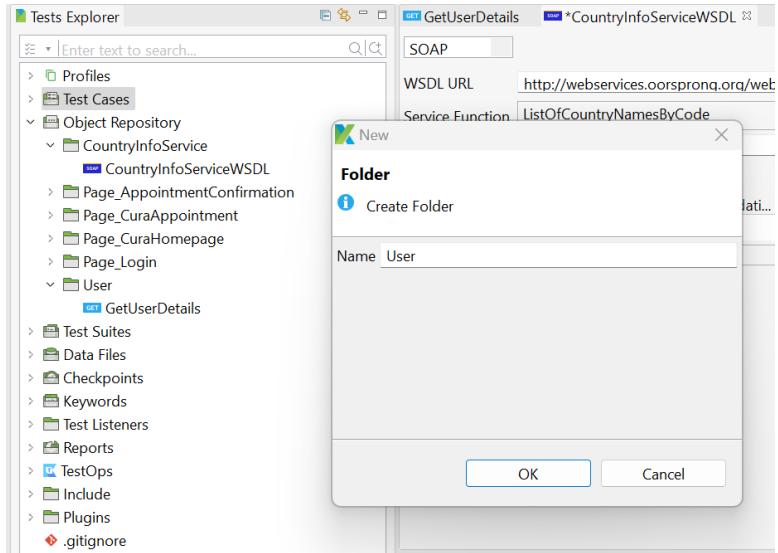
Returns the name of the capital city for the passed **country** code Returns the currency ISO code and name for the passed **country** ISO code Returns a link to a ...

Orang lain juga menelusuri

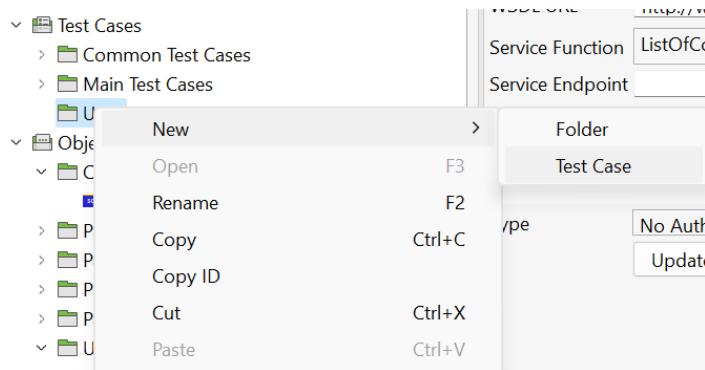
- calculator web service wsdl, soap
- get cities by country wsdl country list web service
- wsdl example url wsdl service url



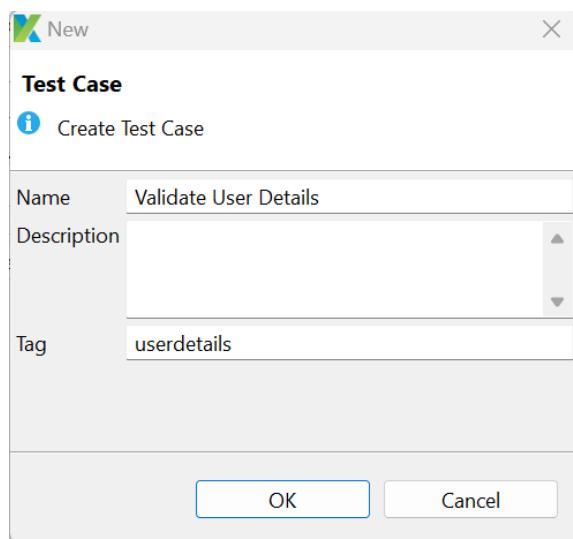
14. Langkah selanjutnya pada Test Explorer, buat folder “User” didalam Test Case



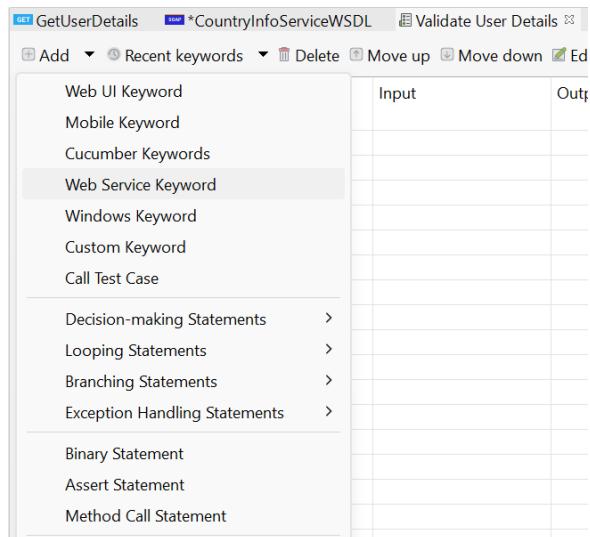
15. Pada folder “User” klik kanan pilih New dan klik Test Case



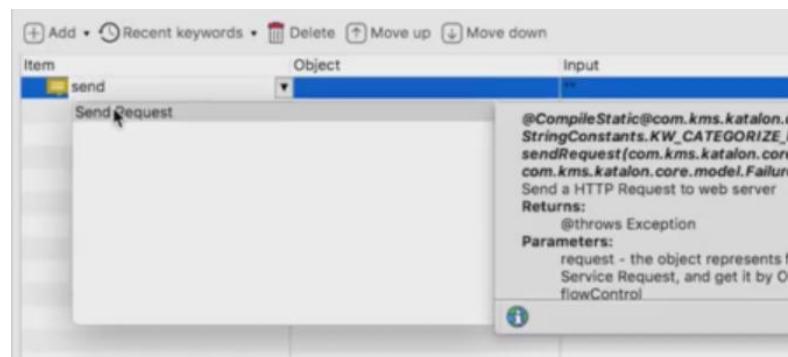
16. Isikan form seperti berikut, klik Ok



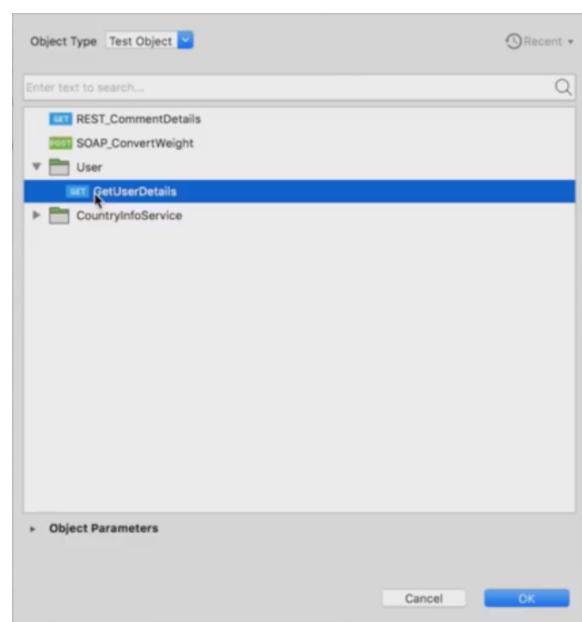
17. Selanjutnya klik dropdown Add pilih Web Service Keyword



18. Gantikan Namanya menjadi “Send Request”



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



20. Pada Output isikan “response”

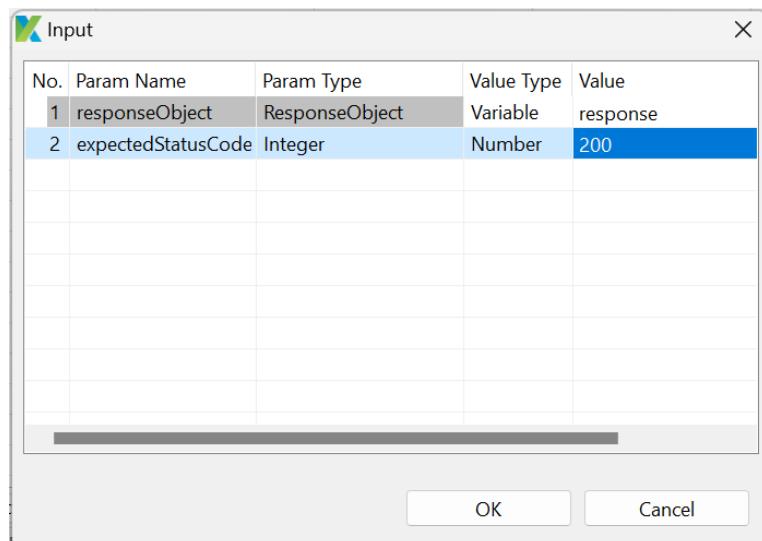
| Item                | Object         | Input | Output   | Description |
|---------------------|----------------|-------|----------|-------------|
| -> 1 - Send Request | GetUserDetails |       | response |             |

21. Tambahkan “Verify Response Status Code”

The screenshot shows the Katalon Studio Keyword Catalog interface. A tooltip is displayed over the 'Verify Response Status Code' keyword, providing its documentation. The tooltip includes the following text:

```
@CompileStatic  
@com.kms.katalon.core.annotation.Keyword(keywordObject =  
StringConstants.KW_CATEGORIZE_ELEMENT)  
static boolean verifyResponseStatusCode  
(com.kms.katalon.core.testobject.ResponseObject  
responseObject, int expectedStatusCode,  
com.kms.katalon.core.model.FailureHandling flowControl)  
Verify status code in the returned data from a web service call  
throws:  
StepFailedException  
Parameters:  
responseObject - the object represents for a HTTP
```

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



23. Kemudian tambahkan “Verify Element Property Value”

The screenshot shows the Katalon Studio Keyword Catalog interface. A tooltip is displayed over the 'Verify Element Property Value' keyword, providing its documentation. The tooltip includes the following text:

```
@CompileStatic  
@com.kms.katalon.core.annotation.Keyword(keywordObject =  
StringConstants.KW_CATEGORIZE_REQUEST)  
static boolean validateXmlAgainstSchema  
(String xmlSchema,  
com.kms.katalon.core.model.FailureHandler  
failureHandler)  
Validate an XML object against an XML schema  
throws:  
StepFailedException If could not find the  
the response didn't pass the validation  
Parameters:  
xmlObject - The XML object to be valid
```



28. Maka hasilnya akan error, seperti berikut

The screenshot shows the Katalon Studio interface with the 'Test Cases/User/Validate User Details' test case selected. The log panel displays the following details:

- Runs: 1/1
- Passes: 0
- Failures: 1
- Errors: 0
- Skips: 0

**ROOT CAUSE**

```
For trouble shooting, please visit: https://docs.katalon.com/katalon-studio/docs/troubleshooting.html
```

18-17-2022 03:45:21 AM Test Cases/User/Validate User Details

Elapsed time: 7.005s

Test Cases/User/Validate User Details FAILED.

Reason:  
com.kms.katalon.core.exception.StepFailedException: Expected element property value 'Daffa' is not equal with actual property value 'Janet'  
at com.kms.katalon.core.keyword.internal.KeywordMain.stepFailed  
(KeywordMain.groovy:58)  
at  
com.kms.katalon.core.keyword.builtin.VerifyElementPropertyValueKeyword  
\$\_verifyElementPropertyValue\_closure1.doCall  
(VerifyElementPropertyValueKeyword.groovy:56)

29. Penyebab Error dikarenakan pada Object Repository 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.

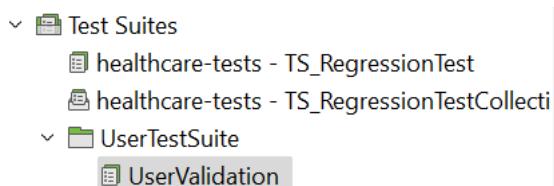
The screenshot shows the Postman application displaying a successful JSON response. The response body is as follows:

```
1 {  
2   "data": {  
3     "id": 2,  
4     "email": "janet.weaver@reqres.in",  
5     "first_name": "Janet",  
6     "last_name": "Weaver",  
7     "avatar": "https://reqres.in/img/faces/2-image.jpg"  
8   },  
9   "support": {  
10     "url": "https://reqres.in/#support-link"  
11   }  
12 }
```

Below the response, there is a note: "Select JSON or XML response data and press Ctrl/Command + K to add verification scripts directly."

Response tab settings: pretty (radio button selected), Body, HAR, Status: 200 OK, Elapsed: 1505 ms, Size: 640 bytes.

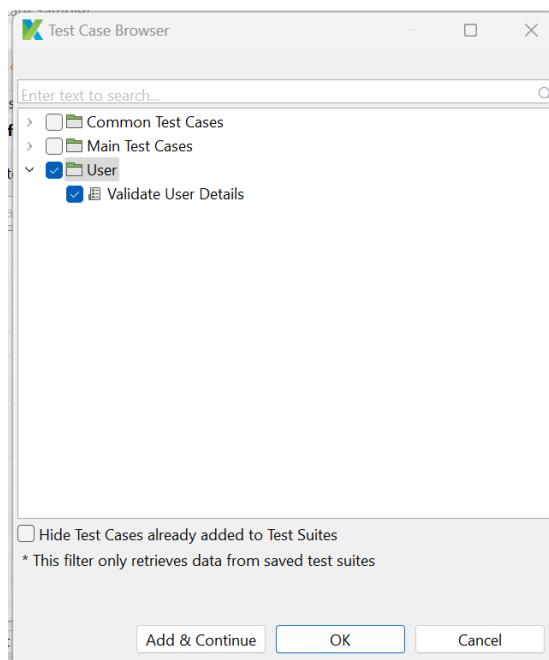
30. Buatkan test suite didalam folder UserTestSuite berikan nam “UserValidation”



31. Berikut tampilannya, klik add

| No. | ID                                    | Description | Run |
|-----|---------------------------------------|-------------|-----|
| 1   | Test Cases/User/Validate User Details |             |     |

32. Centang Validate User Details



33. Berikut setelah di add, klik icon play

| No. | ID | Description                           |
|-----|----|---------------------------------------|
| 1   |    | Test Cases/User/Validate User Details |

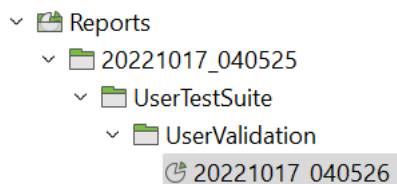
### 34. Berikut hasil dari Log Viewer

The screenshot shows the Katalon Studio interface with the 'Log Viewer' tab selected. The log details the execution of a test suite named 'UserValidation'. It includes system information (hostName: LENOVO - LAPTOP-M8U7JU6R, os: Windows 10 64bit, hostAddress: 192.168.233.1, katalonVersion: 8.5.0.208), test steps (sampleBeforeTestSuite, Test Cases/User/Validate User Details), and reporting (HTML and CSV reports generated). The total elapsed time is 9.756s.

```
Test Suites/UserTestSuite/UserValidation (10.093s)
  hostName = LENOVO - LAPTOP-M8U7JU6R
  os = Windows 10 64bit
  hostAddress = 192.168.233.1
  katalonVersion = 8.5.0.208
  sampleBeforeTestSuite (1.107s)
    1 - KatalonHelper.updateInfo() (1.090s)
  > 2 - response = sendRequest(findTestObject("User/GetUserDetails")) (3.488s)
    2 - verifyResponseStatus(response, 200) (0.054s)
    3 - verifyElementPropertyValue(response, "data.first_name", "Janet") (0.155s)
    4 - Video (0.002s)
  > 3 - exportKatalonReports (0.289s)
    Start generating HTML report folder at: D:\Katalon Studio\Healthcare Sample\Reports\20221017_04052
    HTML report generated
    Start generating CSV report folder at: D:\Katalon Studio\Healthcare Sample\Reports\20221017_04052
    CSV report generated
```

Elapsed time: 9.756s

### 35. Selanjutnya kita buka Reports untuk melihat hasil pengujian



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

The screenshot shows the 'Test Cases Table' and 'Test Suite Summary' sections of the Katalon Studio interface.

**Test Cases Table:**

| No. | Name                           | Resolution |
|-----|--------------------------------|------------|
| 1   | Validate User Details (8.439s) |            |

**Test Suite Summary:**

| Test Suite ID   | Test Suites/UserTestSuite/UserValidation |
|-----------------|--|
| Host name       | LENOVO - LAPTOP-M8U7JU6R                 |
| Katalon version | 8.5.0.208                                |
| Start           | 2022-10-17 04:05:33                      |
| Elapsed         | 9.756s                                   |
| Total TC        | 1  |
| Passed          | 1  |
| Error           | 0  |
| Incomplete      | 0  |
| Failed          | 0  |
| Skip            | 0  |

## 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.

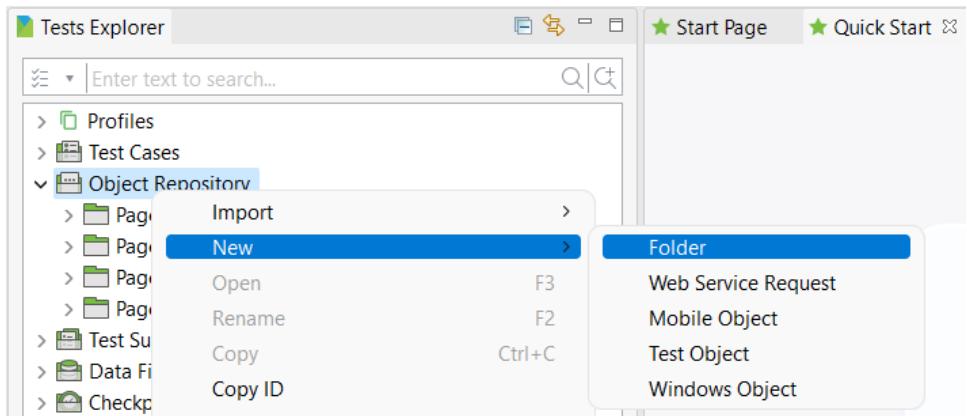
1. Cari satu contoh SOAP API, seperti gambar di bawah ini.

A screenshot of a search results page. The search query is "sample soap country info service". The results show a link to "http://webservices.orsprong.org/ CountryInfoService". The description for this link states: "Returns the name of the capital city for the passed country code Returns the currency ISO code and name for the passed country ISO code Returns a link to a ...". Below the link, there is a sample SOAP request and response.

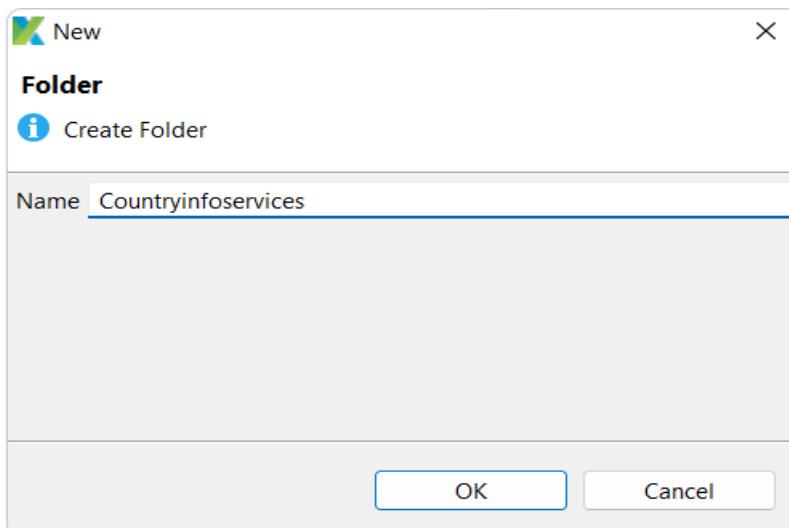
2. Berikut merupakan contoh script soap api dalam format xml.

A screenshot of a browser displaying the WSDL (Web Services Description Language) XML for the CountryInfoService. The XML defines various complex types like tContinent, tCurrency, tCountryCodeAndName, etc., with their respective elements and types. The XML is well-structured and includes definitions for ISO codes, names, and continent codes.

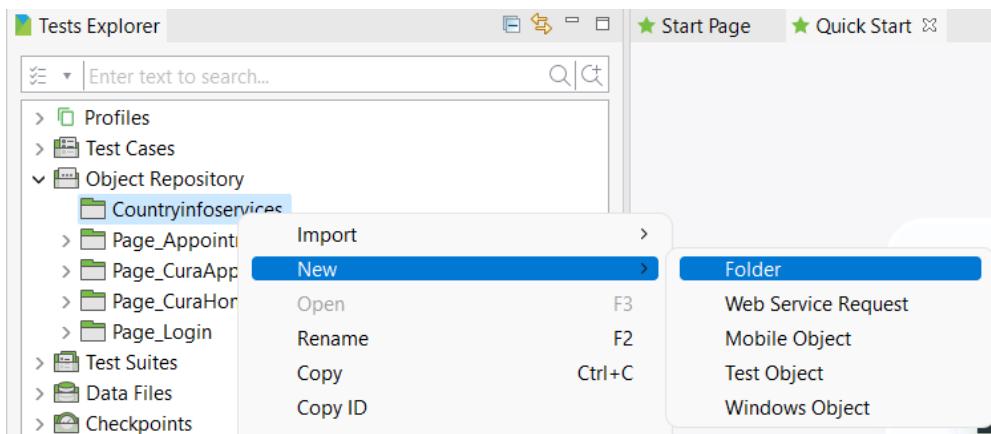
3. Selanjutnya buat 1 folder baru.



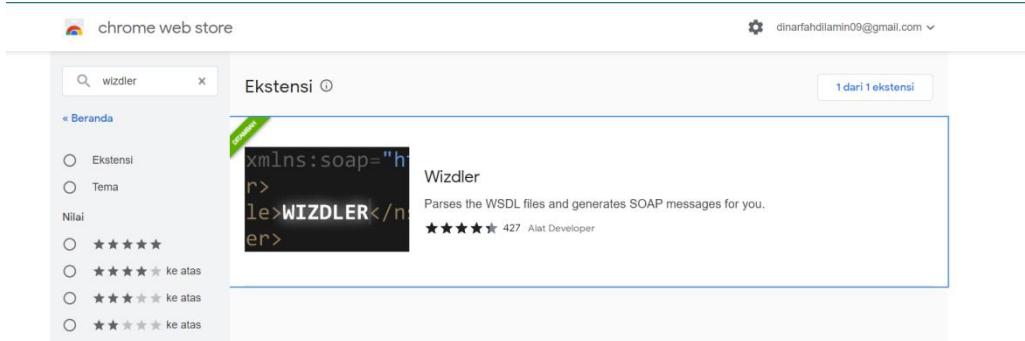
4. Kemudian, beri nama Countrinfoservices, lalu klik ok.



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



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



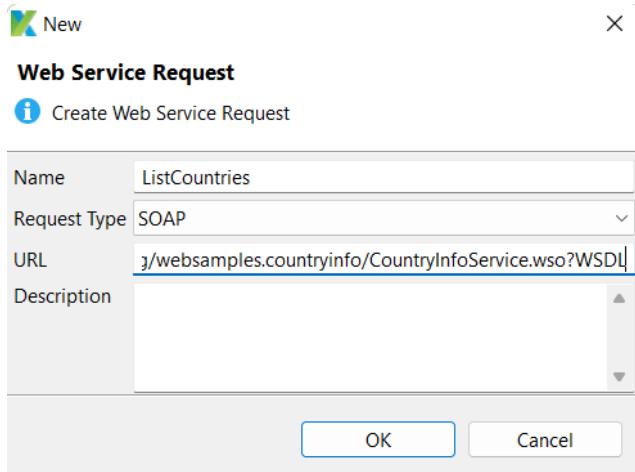
7. Jika extensi sudah ditambahkan, maka akan muncul ikon seperti gambar dibawah ini.



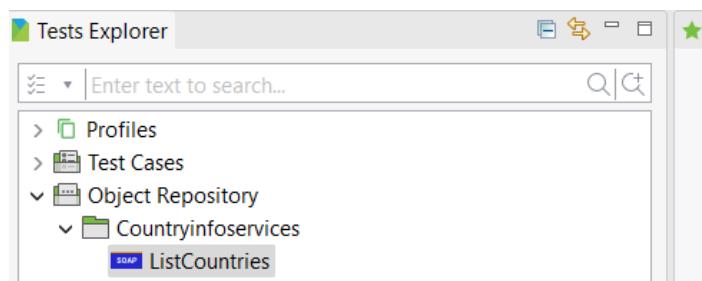
8. Berikut ini merupakan list request yang tersedia.

9. Selanjutnya kita akan mengambil request dari `ListOfCountryNamesByName`, berikut tampilannya.

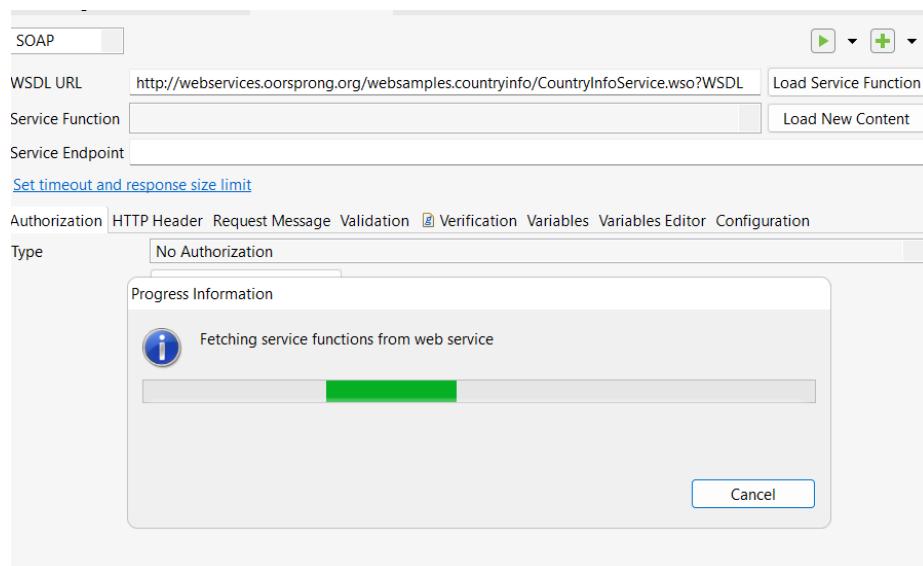
10. Buat web service request dengan nama ListCountries, dengan pilihan requestType adalah SOAP, dan masukan URL dari sample SOAP API tadi.



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 ListOfCountryNamesByName untuk service functionnya.

|                  |   |
|------------------|---|
| WSDL URL         | http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL   |
| Service Function | ListOfCountryNamesByName  |
| Service Endpoint | ListOfContinentsByName<br>ListOfContinentsByCode<br>ListOfCurrenciesByName<br>ListOfCurrenciesByCode<br>CurrencyName<br>ListOfCountryNamesByCode<br><b>ListOfCountryNamesByName</b>   |
| Set timeout and  |   |
| Authorization    | HT  |
| Type             | ListOfCountryNamesGroupedByContinent<br>CountryName<br>CountryISOCode<br>CapitalCity<br>CountryCurrency<br>CountryFlag<br>CountryIntlPhoneCode<br>FullCountryInfo<br>FullCountryInfoAllCountries<br>CountriesUsingCurrency<br>ListOfLanguagesByName |

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

Request Message

```
<Envelope xmlns="http://www.w3.org/2003/05/soap-envelope">
  <Body>
    <ListOfCountryNamesByName
      xmlns="http://www.oorsprong.org/websamples.countryinfo"/>
  </Body>
</Envelope>
```

15. Kemudian save script.

Service Endpoint

Set timeout and response size limit

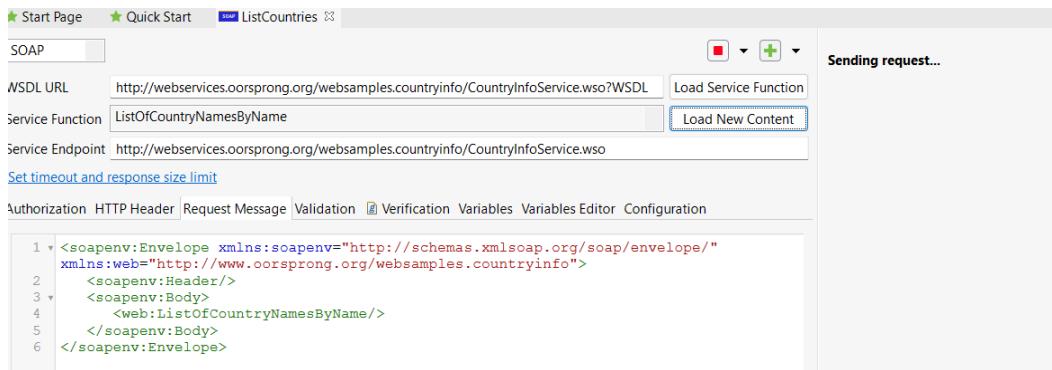
Request Message

```
<Envelope xmlns="http://www.w3.org/2003/05/soap-envelope">
  <Body>
    <ListOfCountryNamesByName
      xmlns="http://www.oorsprong.org/websamples.countryinfo"/>
  </Body>
</Envelope>
```

Do you want to save the changes?

OK Cancel

16. Selanjutnya jalankan script untuk melakukan request/permintaan.

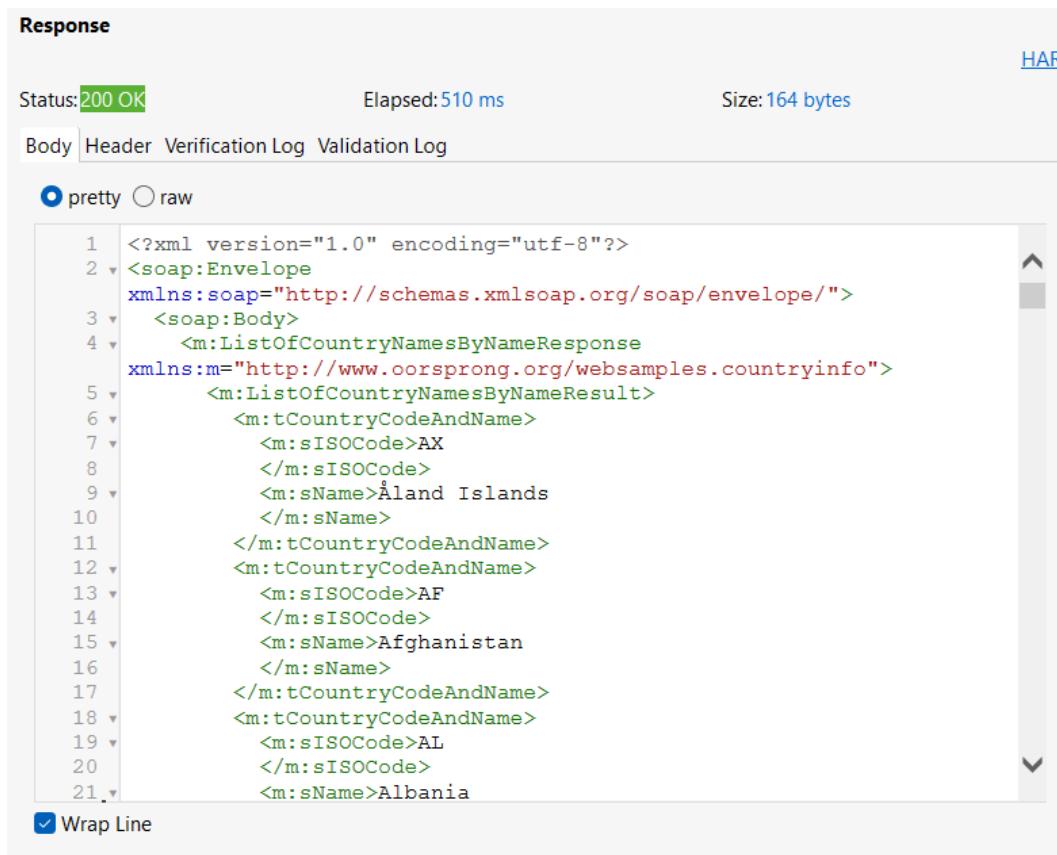


The screenshot shows the SoapUI interface with the following details:

- Toolbar: Start Page, Quick Start, ListCountries
- Request Type: SOAP
- WSL URL: <http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wsdl>
- Service Function: ListOfCountryNamesByName
- Service Endpoint: <http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wsdl>
- Buttons: Load Service Function, Load New Content, Set timeout and response size limit
- Message Editor tab: Request Message (selected)
- Request XML content:

```
1 <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:web="http://www.oorsprong.org/websamples.countryinfo">
2   <soapenv:Header>
3   <soapenv:Body>
4     <web:ListOfCountryNamesByName/>
5   </soapenv:Body>
6 </soapenv:Envelope>
```

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



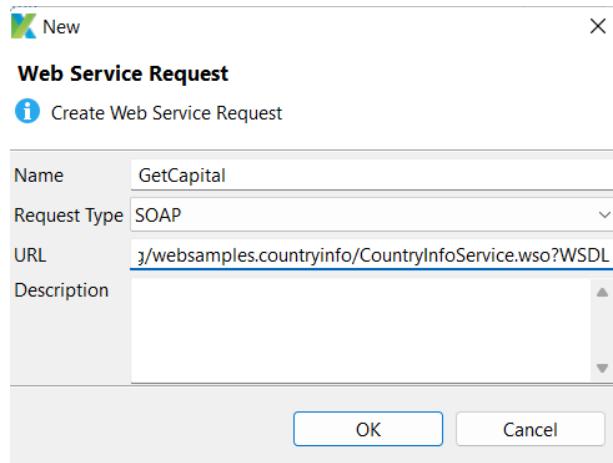
The screenshot shows the SoapUI interface with the following details:

- Toolbar: Response (selected), HAR
- Status: 200 OK
- Elapsed: 510 ms
- Size: 164 bytes
- Tab: Body (selected), Header, Verification Log, Validation Log
- Pretty Print: pretty (selected)
- Response XML content:

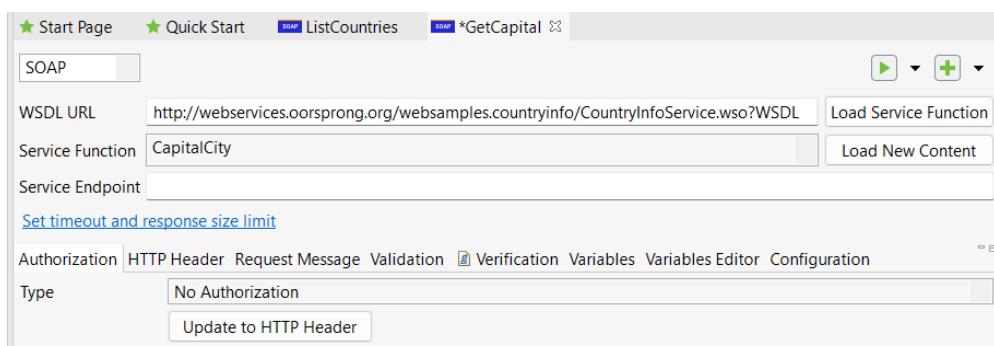
```
1 <?xml version="1.0" encoding="utf-8"?>
2 <soap:Envelope
3   xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
4   <soap:Body>
5     <m:ListOfCountryNamesByNameResponse
6       xmlns:m="http://www.oorsprong.org/websamples.countryinfo">
7       <m:ListOfCountryNamesByNameResult>
8         <m:tCountryCodeAndName>
9           <m:sISOCode>AX
10          </m:sISOCode>
11          <m:sName>Åland Islands
12          </m:sName>
13        </m:tCountryCodeAndName>
14        <m:tCountryCodeAndName>
15          <m:sISOCode>AF
16          </m:sISOCode>
17          <m:sName>Afghanistan
18          </m:sName>
19        </m:tCountryCodeAndName>
20        <m:tCountryCodeAndName>
21          <m:sISOCode>AL
          </m:sISOCode>
          <m:sName>Albania
        </m:tCountryCodeAndName>
      </m:ListOfCountryNamesByNameResult>
    </m:ListOfCountryNamesByNameResponse>
  </soap:Body>
</soap:Envelope>
```

- Checkboxes: Wrap Line

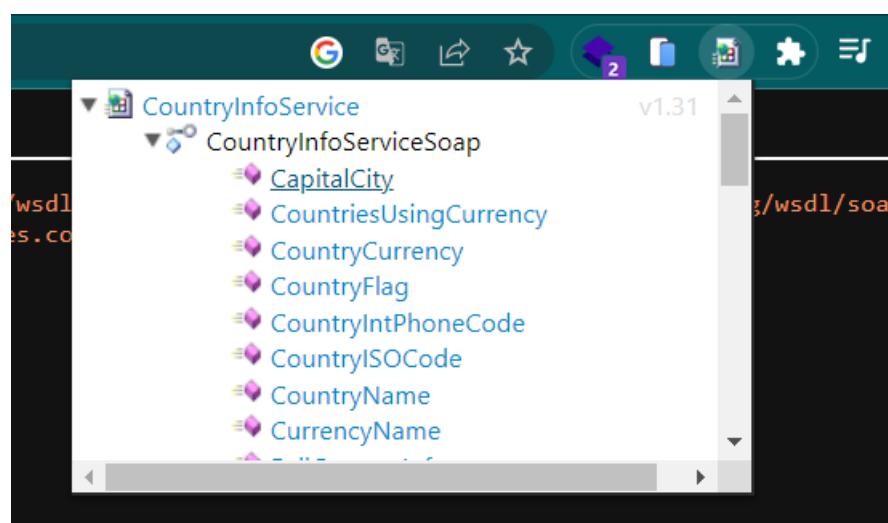
18. Buat web service request lainnya.



19. Pilih server function nya CapitalCity.



20. Pada extensi wizdler, pilih CapitalCity.

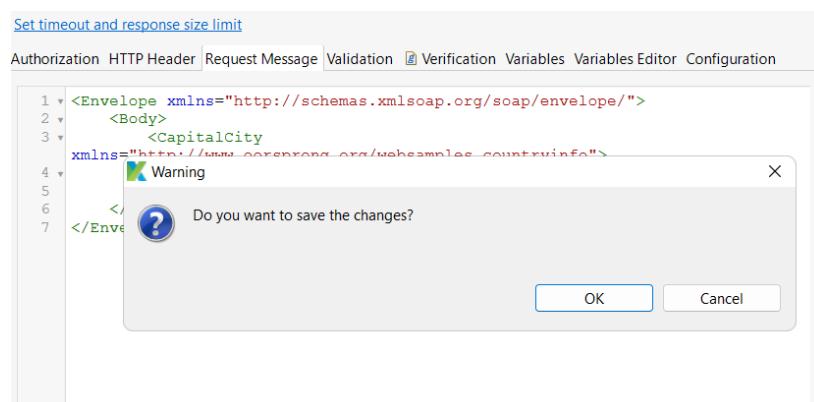


21. Berikut script dari CapitalCity.

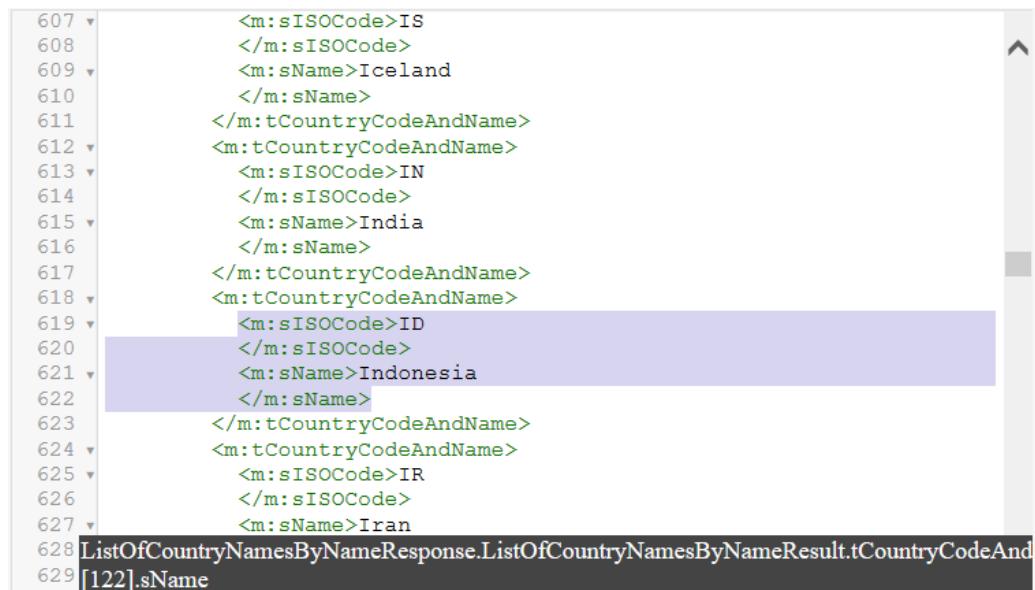


```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <CapitalCity xmlns="http://www.oorsprong.org/websamples.countryinfo">
      <sCountryISOCode>[string]</sCountryISOCode>
    </CapitalCity>
  </Body>
</Envelope>
```

22. Save script.

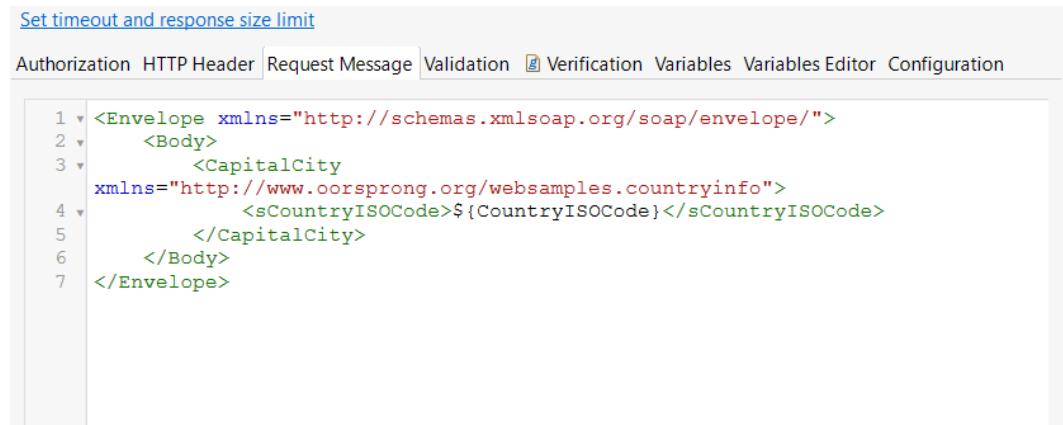


23. Berikut ini hasil dari request yang diberikan, dengan menampilkan nama-nama ibukota dari tiap negara.



```
607  <m:sISOCode>IS
608  </m:sISOCode>
609  <m:sName>Iceland
610  </m:sName>
611  </m:tCountryCodeAndName>
612  <m:tCountryCodeAndName>
613  <m:sISOCode>IN
614  </m:sISOCode>
615  <m:sName>India
616  </m:sName>
617  </m:tCountryCodeAndName>
618  <m:tCountryCodeAndName>
619  <m:sISOCode>ID
620  </m:sISOCode>
621  <m:sName>Indonesia
622  </m:sName>
623  </m:tCountryCodeAndName>
624  <m:tCountryCodeAndName>
625  <m:sISOCode>IR
626  </m:sISOCode>
627  <m:sName>Iran
628 ListOfCountryNamesByNameResponse.ListOfCountryNamesByNameResult.tCountryCodeAnd
629 [122].sName
```

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



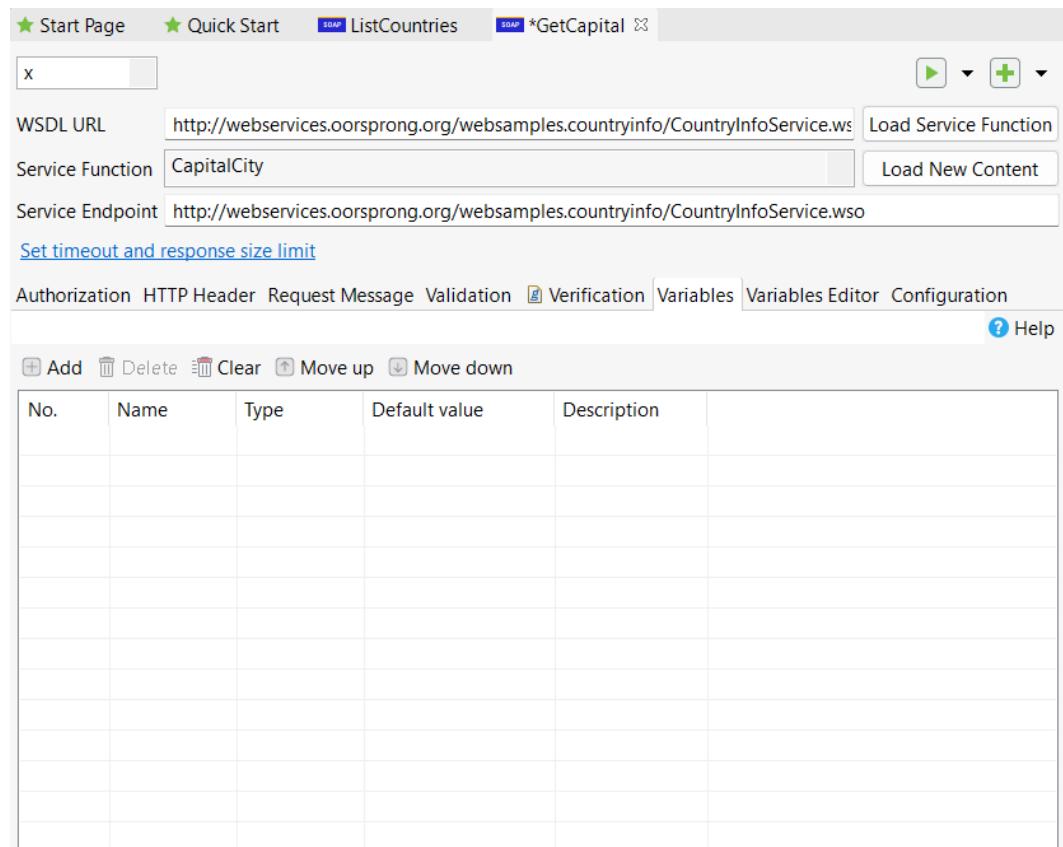
The screenshot shows a software interface for managing web services. At the top, there are tabs: Authorization, HTTP Header, Request Message, Validation, Verification, Variables, Variables Editor, and Configuration. The "Request Message" tab is selected. Below it, a code editor displays the following XML code:

```

1 <Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
2   <Body>
3     <CapitalCity
4       xmlns="http://www.oorsprong.org/websamples.countryinfo">
5       <sCountryISOCode>${CountryISOCode}</sCountryISOCode>
6     </CapitalCity>
7   </Body>
8 </Envelope>

```

25. Kemudian pilih menu variabel dan klik add.



The screenshot shows the same software interface with the "Variables" tab selected. This tab contains several configuration fields:

- WSDL URL: http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wsdl
- Service Function: CapitalCity
- Service Endpoint: http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso

Below these fields is a link labeled "Set timeout and response size limit".

At the bottom of the window, there is a toolbar with icons for Add, Delete, Clear, Move up, and Move down. A table below the toolbar has columns for No., Name, Type, Default value, and Description. The table is currently empty.



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

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <soap:Envelope
3   xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
4     <soap:Body>
5       <m:CapitalCityResponse
6         xmlns:m="http://www.oorsprong.org/websamples.countryinfo">
7           <m:CapitalCityResult>Jakarta
8           </m:CapitalCityResult>
9         </m:CapitalCityResponse>
10      </soap:Body>
11    </soap:Envelope>

```

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.

| Test Cases          |               |       |           |             |  |
|---------------------|---------------|-------|-----------|-------------|--|
| Item                | Object        | Input | Output    | Description |  |
| -> 1 - Send Request | ListCountries |       | response1 |             |  |

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

Output  
response1

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

```

1 import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint
2 import static com.kms.katalon.core.testcase.TestCaseFactory.findTestCase
3 import static com.kms.katalon.coretestdata.TestDataFactory.findTestData
4 import static com.kms.katalon.core.testobject.ObjectRepository.findTestObject
5 import com.kms.katalon.core.checkpoint.Checkpoint as Checkpoint
6 import com.kms.katalon.core.cucumber.keyword.CucumberBuiltInKeywords as CucumberKW
7 import com.kms.katalon.core.mobile.keyword.MobileBuiltInKeywords as Mobile
8 import com.kms.katalon.core.model.FailureHandling as FailureHandling
9 import com.kms.katalon.core.testcase.TestCase as TestCase
10 import com.kms.katalon.coretestdata.TestData as TestData
11 import com.kms.katalon.core.testobject.TestObject as TestObject
12 import com.kms.katalon.core.webservice.keyword.WSBuiltInKeywords as WS
13 import com.kms.katalon.core.webui.keyword.WebUiBuiltInKeywords as WebUI
14 import internal.GlobalVariable as GlobalVariable
15
16 response1 = WS.sendRequest(findTestObject('CountryInfoService/ListCountries'))

```

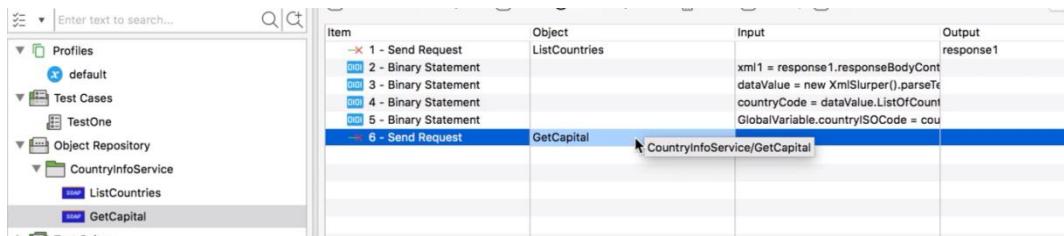




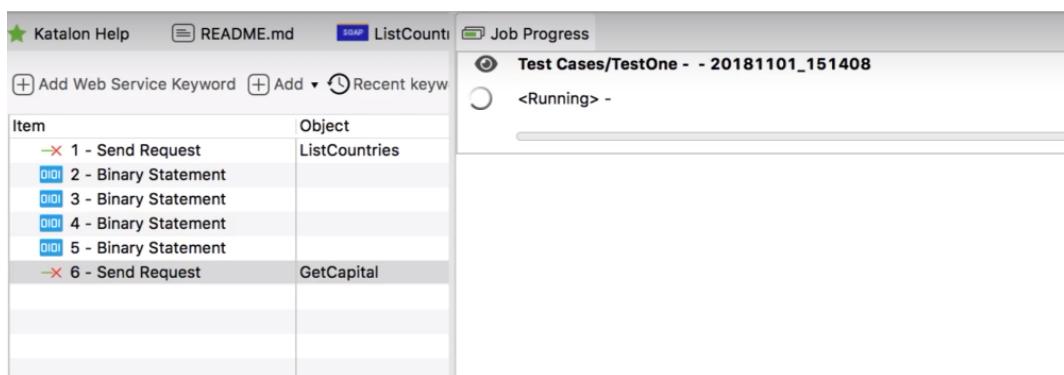
38. Selanjutnya hasil pada variabel countryCode tadi kita simpan pada global variabel, dengan perintah seperti berikut.

```
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.



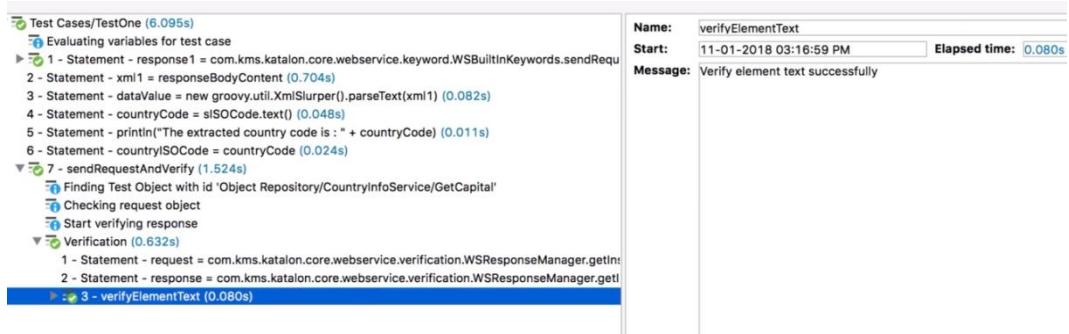
41. Buka jendela console, dan memperlihatkan hasil verifikasi yang telah kita lakukan.







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



The screenshot shows the Katalon Studio Log Viewer interface. On the left, there is a tree view of the test case structure:

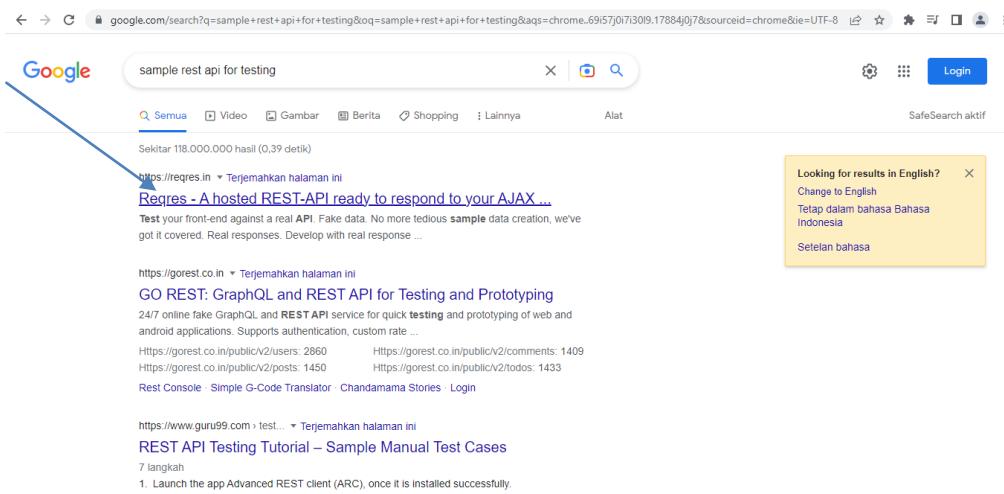
- Test Cases/TestOne (6.095s)
  - Evaluating variables for test case
  - 1 - Statement - response1 = com.kms.katalon.core.webservice.keyword.WSBuiltinKeywords.sendRequest(  
2 - Statement - xml1 = responseBodyContent (0.704s)  
3 - Statement - dataValue = new groovy.util.XmlSlurper().parseText(xml1) (0.082s)  
4 - Statement - countryCode = sISOCODE.text() (0.048s)  
5 - Statement - println("The extracted country code is: " + countryCode) (0.011s)  
6 - Statement - countryISOCode = countryCode (0.024s)
  - 7 - sendRequestAndVerify (1.524s)
    - Finding Test Object with id 'Object Repository/CountryInfoService/GetCapital'
    - Checking request object
    - Start verifying response
    - Verification (0.632s)
      - 1 - Statement - request = com.kms.katalon.core.webserviceverification.WSResponseManager.getInitialRequest(  
2 - Statement - response = com.kms.katalon.core.webserviceverification.WSResponseManager.getInitialResponse(  
3 - verifyElementText (0.080s)

Name: verifyElementText  
Start: 11-01-2018 03:16:59 PM Elapsed time: 0.080s  
Message: Verify element text successfully

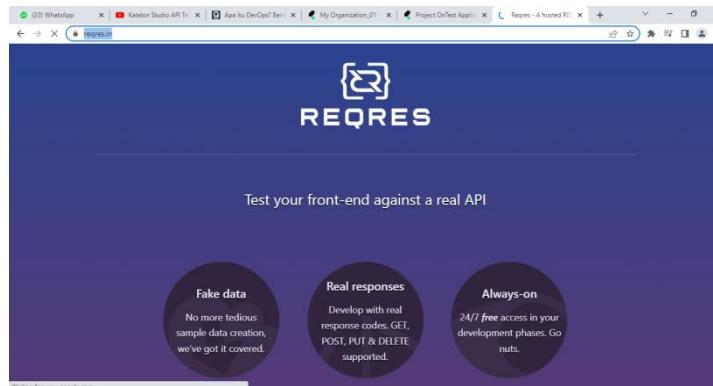
### 9.3. API CHAINING | REST - JSON | HOW TO SEND VALUE FROM ONE API TO OTHER

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



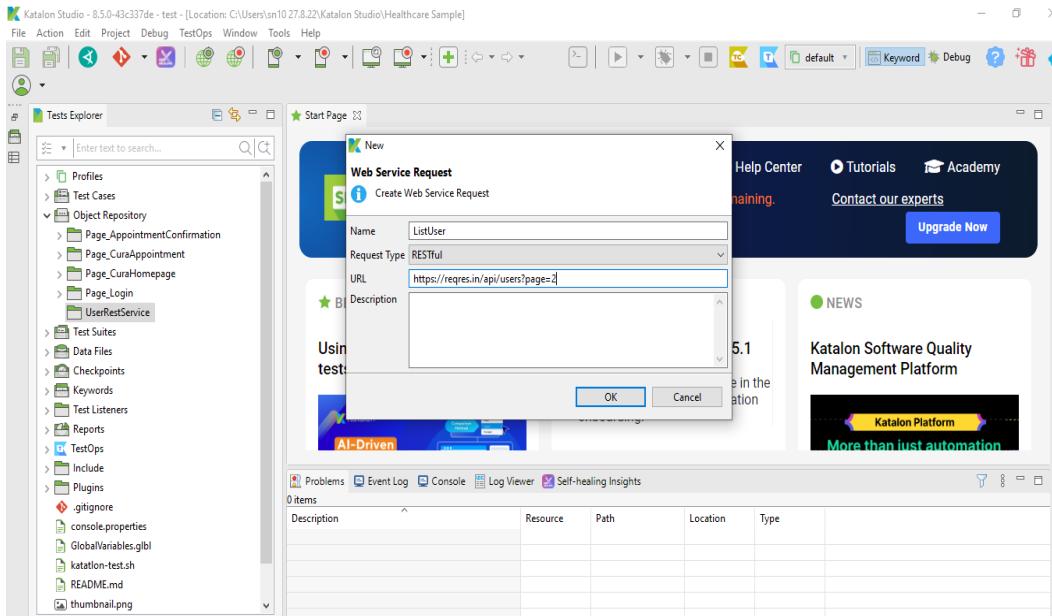
3. Disini bisa kita lihat jika kita turun kita akan memiliki banyak aplikasi dan kita telah mendapatkan posting yang di hapus batch dan seterusnya kita akan menggunakan daftar ini, pengguna mendapatkan API, jadi kita akan menyalin url ini dan meletakkan nya di tab baru.

This screenshot shows the Reqres API documentation page. It features a sidebar on the left with various API endpoints listed under 'Request /api/users?page=2': GET LIST USERS, GET SINGLE USER, GET SINGLE USER NOT FOUND, GET LIST &lt;RESOURCE&gt;, GET SINGLE &lt;RESOURCE&gt;, GET SINGLE &lt;RESOURCE&gt; NOT FOUND, POST CREATE, and PUT UPDATE. The main area is split into two panes: 'Request' on the left showing the URL /api/users?page=2, and 'Response' on the right showing a JSON response. The response body includes fields like page, per\_page, total, total\_pages, and data, which lists multiple user objects with properties such as id, email, first\_name, last\_name, and avatar. A progress bar at the bottom indicates 'Waiting for w...'. A support button is located at the top right.

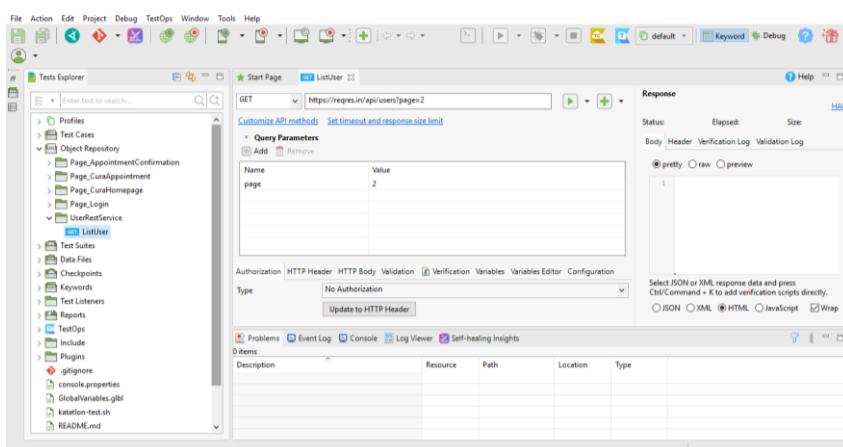
4. Menyalin teks url reqres.in dan menyalin sisanya titik akhir dari sini dan menambahkan nya lagi, dan ini adalah titik terakhir API jika kita menjalankannya kita dapat melihat respon. Jadi saya akan menyalin url ini

A screenshot of a browser window displaying the URL https://reqres.in/api/users?page=2. The page content is identical to the one shown in the previous screenshot, displaying the list of users with their details and a JSON response body. The browser's address bar shows the full URL, and the status bar indicates the page is loading.

- 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



- 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



- 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.

The screenshot shows a Postman interface. The request method is GET, and the URL is <https://reqres.in/api/users?page=2>. In the 'Query Parameters' section, there is one entry: 'page' with a value of '2'. Under 'Authorization', it says 'No Authorization'. The 'Response' tab is selected, showing the following JSON output:

```
200 OK
Elapsed: 908 ms
Size: 916 bytes

Status: 200 OK
Body Header Verification Log
pretty raw preview

{
  "page": 2,
  "per_page": 3,
  "total": 12,
  "total_pages": 4,
  "data": [
    {
      "id": 4,
      "first_name": "Eve",
      "last_name": "Holt",
      "avatar": "https://s3.amazonaws.com/uifaces/faces/twitter/marcoramires/128.jpg"
    },
    {
      "id": 5,
      "first_name": "Charles",
      "last_name": "Morris",
      "avatar": "https://s3.amazonaws.com/uifaces/faces/twitter/marcoramires/128.jpg"
    }
  ]
}
```