

# QA Documentation

## Katalon Studio - Spellbound Novice (Mudah)



# Daftar Isi

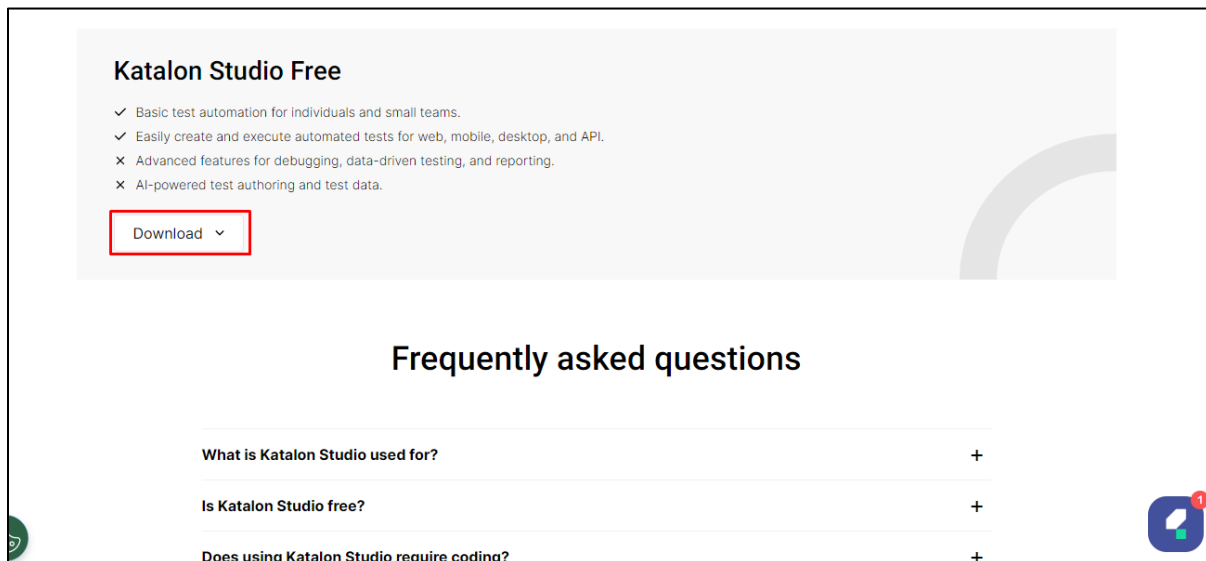
<b>Instalasi Aplikasi Tools QA.....</b>	<b>3</b>
• Katalon Studio .....	3
• Apache JMeter .....	4
• Java JDK.....	4
• GIT.....	5
• VSCode .....	5
<b>Buat Projek Baru .....</b>	<b>6</b>
• Halaman Utama Katalon Studio .....	6
• Membuat Projek Baru .....	7
• Halaman Ketika sudah dibuka projeknya .....	8
<b>Struktur Projek .....</b>	<b>9</b>
<b>Object Repository .....</b>	<b>10</b>
<b>Folder &amp; Naming File Rules.....</b>	<b>11</b>
• Root Folder .....	11
• Folder Page .....	11
• Folder Halaman Situs .....	12
• Folder Komponen dan Penamaan File .....	13
<b>Create Folder &amp; Object.....</b>	<b>14</b>
• Create Folder .....	14
• API Object .....	15
• Test Object (Web, Mobile & Desktop) .....	17
<b>Capture Object.....</b>	<b>19</b>
<b>Capture Object – Spy Web.....</b>	<b>20</b>
<b>Capture Object – Record Web .....</b>	<b>27</b>
<b>Capture Object – Alternative Capture .....</b>	<b>34</b>
<b>Test Case .....</b>	<b>37</b>
<b>Membuat Test Case .....</b>	<b>39</b>
<b>Membuat Perintah Automation pada Test Case .....</b>	<b>40</b>
<b>Failure Handling pada Test Case (Test Script) .....</b>	<b>41</b>
<b>Drag &amp; Drop di Test Case.....</b>	<b>42</b>
<b>Call Test Case.....</b>	<b>43</b>
<b>Variable.....</b>	<b>46</b>

• Global Variable.....	46
• Local Variable.....	48
• Local Variable Binding.....	49
<b>Encrypt Text.....</b>	<b>51</b>
<b>Keywords.....</b>	<b>52</b>
• Accept Alert .....	52
• Click.....	53
• Delay .....	53
• Dismiss Alert .....	53
• Double Click .....	54
• Focus .....	54
• Get Text.....	54
• Get Attribute .....	55
• Navigate to Url .....	55
• Open Browser .....	55
• Close Browser .....	56
• Maximize Window .....	56
• Refresh .....	56
• Select Option by Index .....	57
• Select Option by Value .....	57
• Select Option by Label .....	57
• Set Text .....	58
• Set Encrypt Text .....	58
• Verify.....	58
• Wait .....	59

## Instalasi Aplikasi Tools QA

<b>Katalon Studio</b>	<a href="https://katalon.com/download">https://katalon.com/download</a>
<b>Apache JMeter</b>	<a href="https://jmeter.apache.org/download_jmeter.cgi">https://jmeter.apache.org/download_jmeter.cgi</a>
<b>Java JDK 17+ (kalo sudah abaikan)</b>	<a href="https://www.oracle.com/java/technologies/downloads/">https://www.oracle.com/java/technologies/downloads/</a>
<b>GIT</b>	<a href="https://git-scm.com/downloads">https://git-scm.com/downloads</a>
<b>VSCode (Opsional)</b>	<a href="https://code.visualstudio.com/download">https://code.visualstudio.com/download</a>

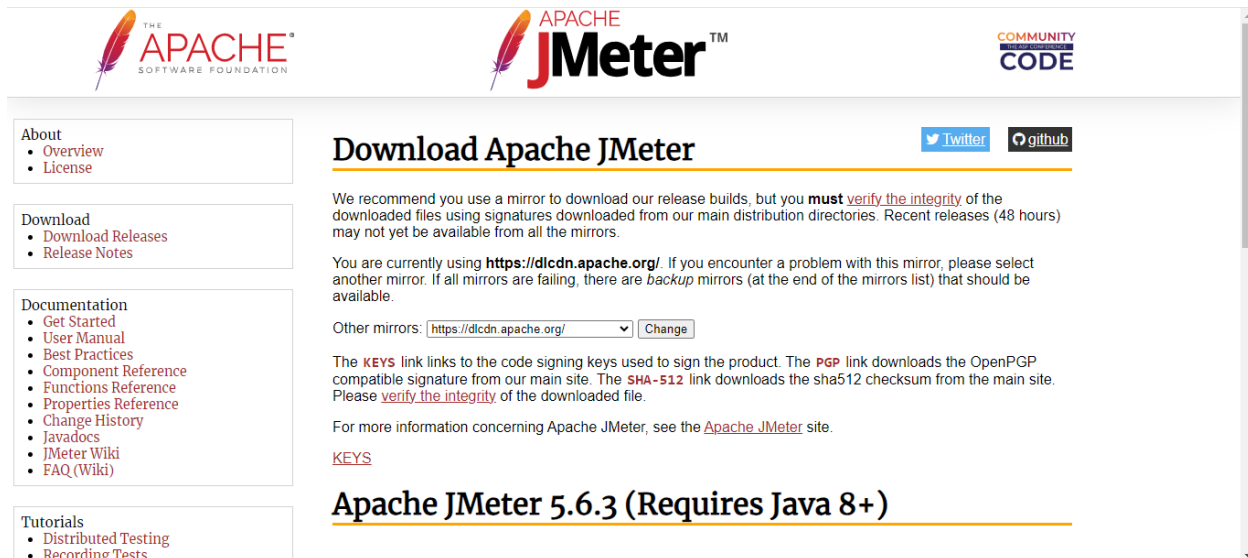
- **Katalon Studio**



Gambar 1 - Website Katalon Studio

Katalon Studio, digunakan untuk menjalankan proses automation, ditulis dengan Bahasa Groovy (Bisa juga java, tapi tidak jauh berbeda dengan Java), pastikan mengunduh versi terbaru. Jika setelah instalasi Katalon, nanti diharuskan untuk melakukan login, untuk login sendiri lebih baik menggunakan Google agar lebih cepat, pastikan untuk mengunduh versi free (gratis) bukan versi enterprise, untuk bagian yang free terdapat setelah bagian enterprise pada halaman unduh katalon.

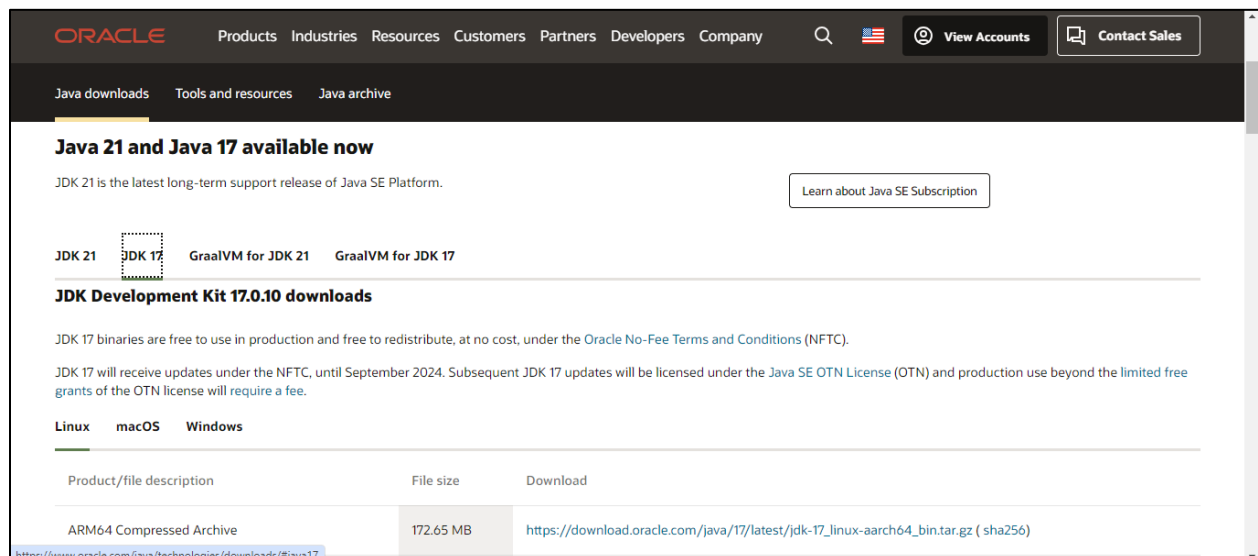
- **Apache JMeter**



Gambar 2 – Apache JMeter

Apache JMeter digunakan untuk melakukan pengujian performa seperti contoh Load Testing, Stress Testing dan lainnya, dan dapat diunduh secara gratis pada website resminya.

- **Java JDK**

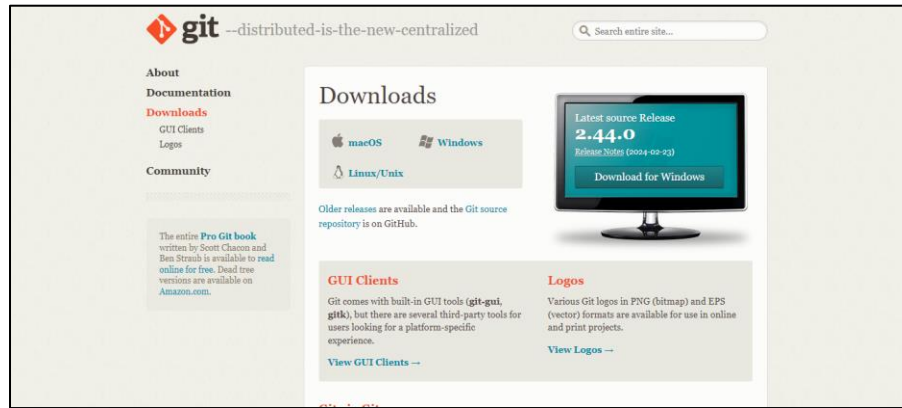


Gambar 3 - Java JDK

Java JDK, digunakan untuk membantu proses si Katalon Studio dalam pemrosesan compile atau penggunaan Java Virtual Machine (JVM) pada saat menjalankan automation di Katalon Studio, jika

sebelumnya sudah terinstall Java JDK, maka tidak perlu diinstall Kembali, tapi untuk menjalankan **Katalon Studio** diperlukan **Java JDK versi 17** keatas. Abaikan jika sudah menginstallnya.

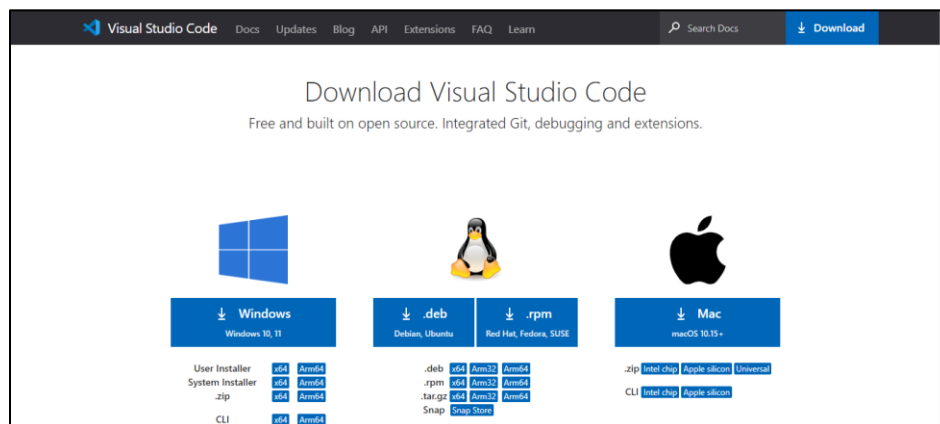
- **GIT**



Gambar 4 - GIT

GIT digunakan untuk membantu proses manajemen kode, mungkin untuk melakukan pembaruan, pengambilan kode dll (Clone, Checkout, Push, Pull, dll) dari Automation Telkom CRA.

- **VSCode**

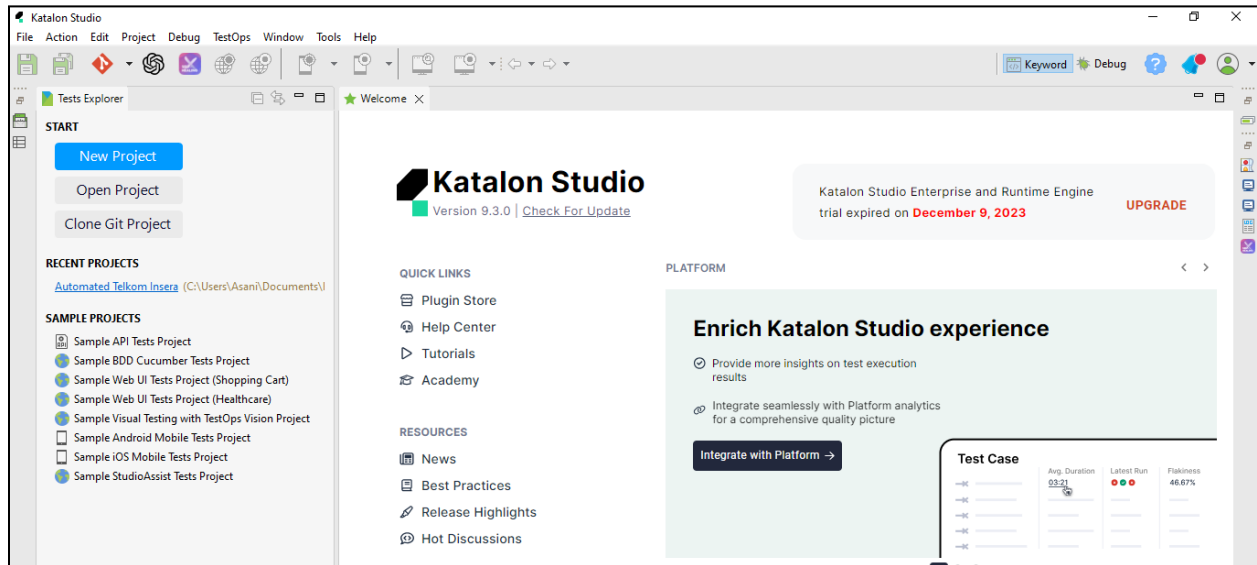


Gambar 5 - VSCode

VSCode, digunakan untuk membantu dalam jika terjadi kendala pada saat menuis kode automation, misalnya kayak tertimpa textnya (overwrite) atau tiba-tiba hilang dan ingin melakukan proses pengembalian (Discard, Revert atau Rollback). Untuk membuka folder proyek Automated Telkom CRA ini tidak dapat dilakukan menggunakan IDE IntelliJ, dan disarankan menggunakan VSCode, jika sudah terinstall bisa diabaikan.

## Buat Projek Baru

- **Halaman Utama Katalon Studio**

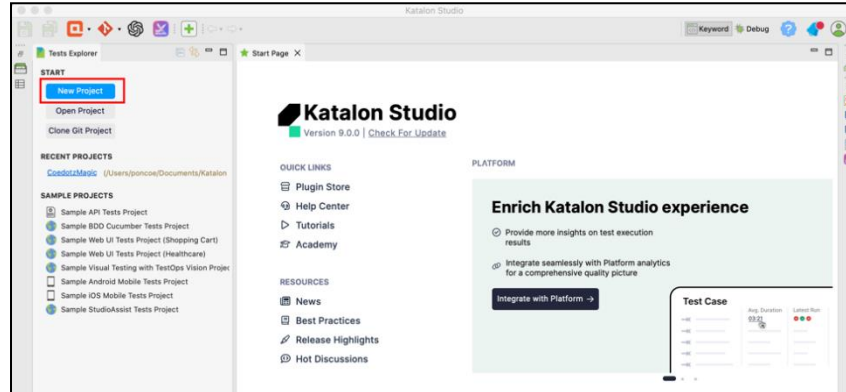


Gambar 6

Disini akan menampilkan halaman utama dari Katalon Studio jika sudah berhasil login, berikut penjelasan beberapa komponen yang terlihat pada halaman utama Katalon Studio :

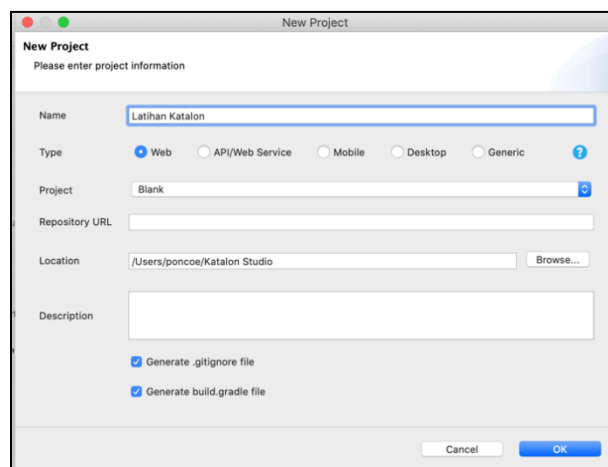
1. **New Project** : Digunakan untuk membuat projek Katalon baru, untuk projek nya bisa berbentuk Web UI, API, Desktop atau Mobile.
2. **Open Project** : Digunakan untuk membuka projek katalon studio.
3. **Git Clone Project** : Digunakan untuk melakukan clone projek, sama seperti yang sudah dilakukan pada bagian “Clone & Checkout Project”
4. **Recent Projects** : Daftar projek yang sudah dibuka dan akan menampilkan history projek tersebut (jika sudah pernah dibuka di katalon studio)
5. **Sample Projects** : Digunakan untuk belajar katalon studio dari sampel projek.

- **Membuat Projek Baru**



*Gambar 7*

Pada gambar diatas, kita akan membuat sebuah projek baru di katalon studio, klik tombol “New Project” untuk membuat projek baru di katalon studio, jika sudah akan menampilkan dialog seperti gambar dibawah.



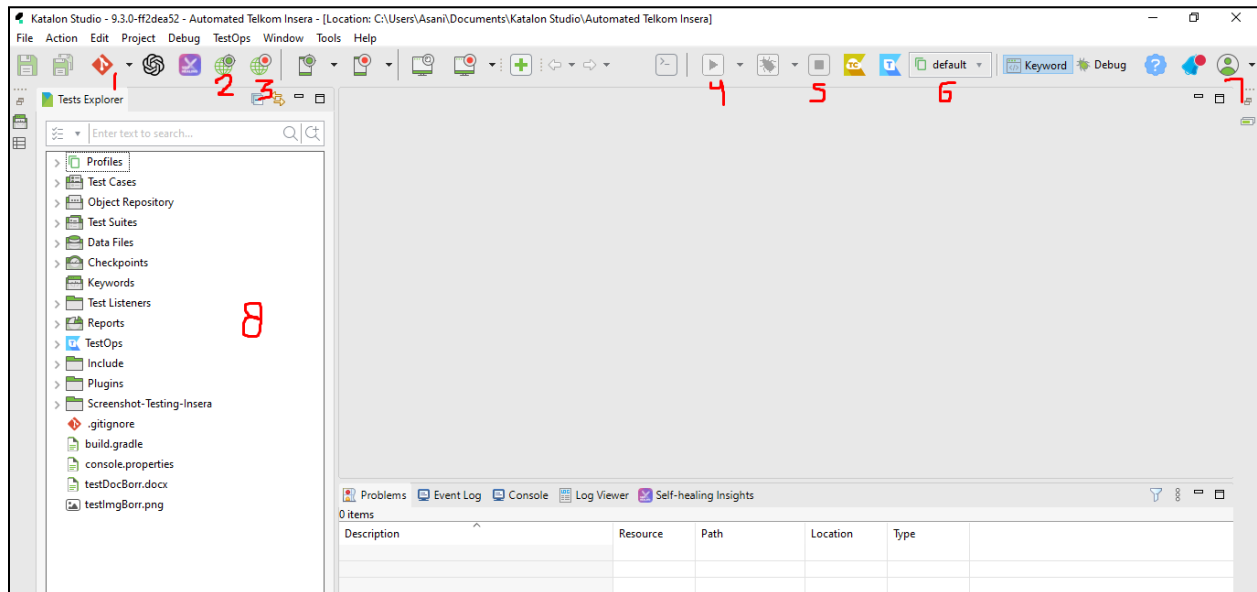
*Gambar 8*

Pada gambar diatas merupakan dialog ketika kita melakukan “New Project” pada katalon studio, untuk type bebas sesuai keinginan kita, tetapi pada kasus gambar diatas, kita akan membuat projek baru dengan tipe “Web”, dimana projek ini akan menguji sebuah aplikasi berbasis web.

Jika ingin melakukan pengujian API atau Web Service, kita bisa menggunakan type API, jika kita ingin membuat projek dari awal dan kosongan bisa menggunakan project “blank”, atau bisa menggunakan template projek, tetapi disarankan untuk menggunakan “blank project”. Pastikan untuk mencentang Generate ignore file dan Generate Build gradle file. Jika sudah terisi dan sesuai, bisa dilanjutkan dengan klik tombol OK.



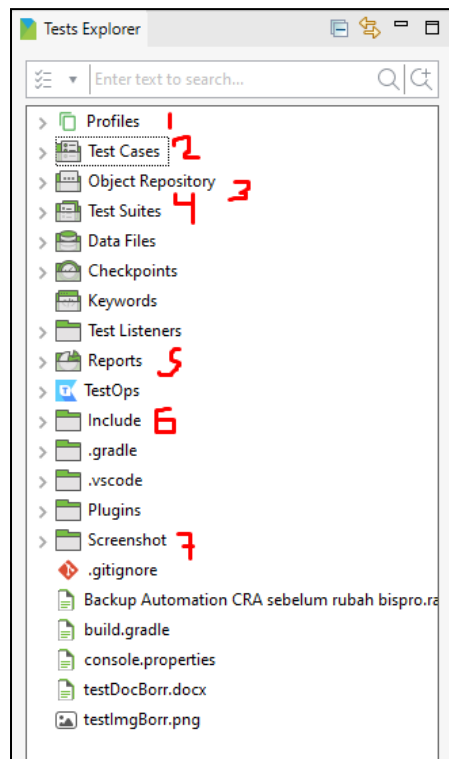
- **Halaman Ketika sudah dibuka proyeknya**



Gambar 9

1. **Git** : Digunakan untuk melakukan perintah yang berkaitan dengan GIT, tapi untuk amannya bisa gunakan Visual Studio Code untuk penggunaan perintah dengan aman.
2. **Spy Object** : Digunakan untuk melakukan Capture / pengambilan objek yang diinginkan, tanpa harus mengambil objek setiap diklik, dan hanya diambil Ketika kita inginkan, dengan cara klik kanan pada objek yang di inginkan dan "Capture Object".
3. **Recorder Web** : Digunakan untuk melakukan Captrue objek setiap kita melakukan klik pengisian field pada objek (**Penggunaan Recorder tidak disarankan jika test objeknya sudah ada, karena akan terjadi duplikasi test object**).
4. **Run Project** : Digunakan untuk menjalankan Automation atau Test Suite (Automation Borongan) pada katalon.
5. **Stop** : Digunakan untuk menghentikan Automation
6. **Profiles** : Digunakan untuk memilih Profile yang berisikan Global Variable, jika ingin berbeda data pada Global Variable, bisa memilih atau membuat Profiles berbeda.
7. **Account** : Digunakan untuk misal, Logout, ke plugin marketplace dll.
8. **Test Explorer** : Digunakan sebagai Hirarki dari proyek.

## Struktur Projek

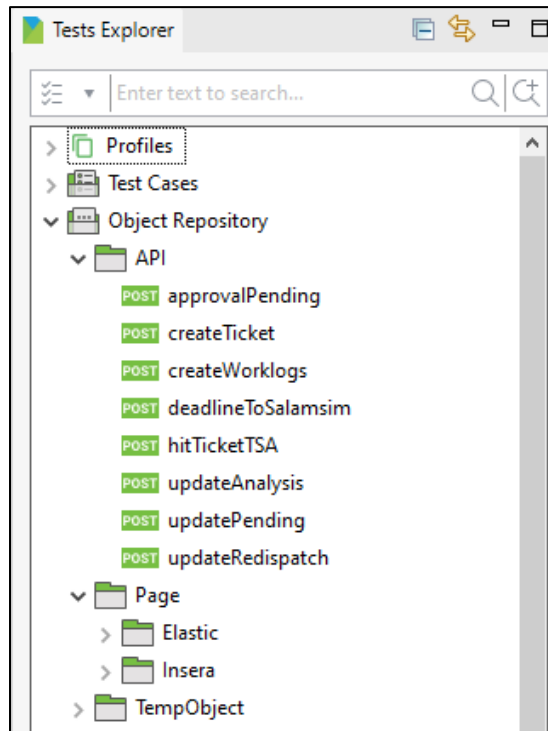


Gambar 10

1. **Profiles** : Digunakan untuk menyimpan GlobalVariable, Dimana variable ini bisa digunakan dikeseluruhan dalam projek, untuk penamaan Global Variable harus Uppercase (Huruf besar semua).
2. **Test Cases** : Tempat menyimpan Test Flow / Skenario Pengujian.
3. **Object Repository** : Tempat menyimpan Object API atau Test Object, yang digunakan untuk pengujian sebagai target komponen pada saat pengujian.
4. **Test Suites** : Digunakan untuk menjalankan pengujian dengan lebih dari 1 testcase sekaligus.
5. **Reports** : Hasil pengujian dari Test Suites, tapi tidak dapat dibuka langsung di aplikasi Katalon karena diperlukan Lisence Enterprise, untuk membuka report tersebut dibuka via Explorer di folder Projeknya, disitu berisikan hasil pengujian menggunakan test suites berupa Screenshot, File PDF, Docs, CSV dll.
6. **Include** : Berisikan tempat untuk menyimpan Kode Script (Groovy, Java), Package, Feature (Behaviour Data Driven menggunakan Gherkin Language) dll.
7. **Screenshoot**: Penyimpanan Gambar jika selesai pengujian, jadi Ketika selesai pengujian akan generated gambar dan disimpan kedalam folder

## Object Repository

---



Gambar 11

Object Repository merupakan tempat untuk menyimpan berupa API Object atau Test Object (Komponen yang di web/mobile/desktop).

**Folder API :** Digunakan untuk menyimpan API Object.

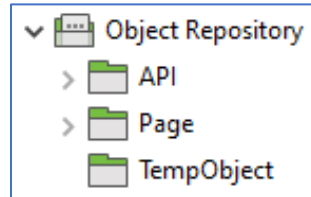
**Page :** Digunakan untuk menyimpan Test Object berupa masing-masing halaman dan situs.

**Temp Object :** Digunakan untuk menyimpan Test Object Sementara, biasanya digunakan untuk pengujian Automation, apakah dia komponen atau objeknya sesuai atau tidak, jika sesuai dan lulus pengujian akan dipindahkan ke folder page.

## Folder & Naming File Rules

---

- **Root Folder**

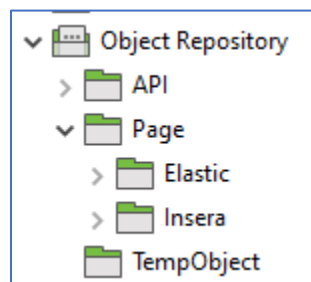


Gambar 12

Pada gambar diatas, di folder Object Repository terdapat 3 folder, yaitu API, Page dan TempObject/Temp. Dimana :

- **API** : Digunakan menyimpan file berupa object API.
- **Page** : Digunakan menyimpan file berupa test object web, mobile atau desktop testing, berdasarkan masing-masing halaman situs yang di uji.
- **TempObject/Temp** : Digunakan untuk menyimpan test object sementara, yang nantinya jika test object tersebut berfungsi atau berjalan dengan baik ketika pada pengujian, akan dipindahkan kedalam folder “Page”.

- **Folder Page**

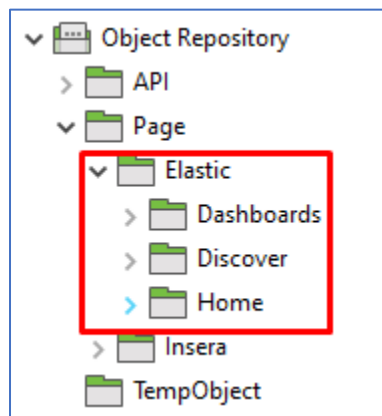


Gambar 13

Pada gambar diatas, di folder page terdapat subfolder yang Bernama **Elastic** dan **Insera**, Dimana subfolder tersebut merupakan folder untuk masing-masing situs. Jika dalam pengujian terdapat perpindahan halaman ke situs lain, kita perlu memisahkan test object pada halaman tersebut

**Contoh Kasus,** Kita ingin menguji halaman situs Itasoft, tetapi pada bagian menu Career terdapat perpindahan situs ke LinkedIn, nah pada saat di Object Repository, kita perlu memisahkan antara file-file object yang ada di Itasoft dan LinkedIn, jadi kita akan membuat 2 folder pada folder Page dengan nama Itasoft dan LinkedIn, Dimana masing-masing folder akan menyimpan test object sesuai masing-masing situs.

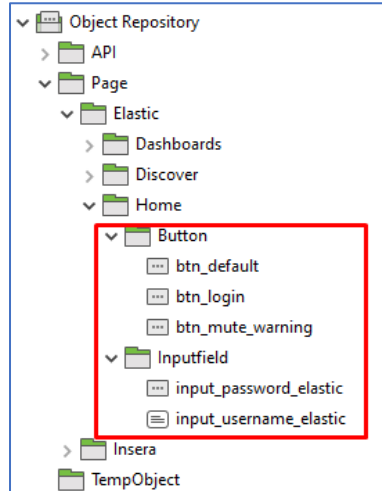
- **Folder Halaman Situs**



Gambar 14

Pada gambar diatas, ketika kita sudah melakukan pemisahan folder page, kita perlu memecah isi dari masing-masing halaman situs, misal pada gambar diatas ada situs **Elastic**, dan didalam situs **Elastic** terdapat halaman **Dashboard**, **Discover** dan **Home**, nah kita perlu memisah masing-masing halaman pada situs yang ingin kita uji, agar memudahkan dalam mengorganisir test object yang telah dibuat.

- **Folder Komponen dan Penamaan File**



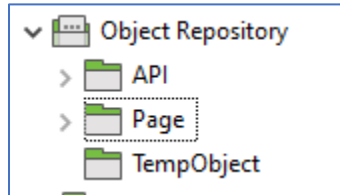
Gambar 15

Pada gambar diatas, ketika kita sudah memisahkan halaman situs, kita perlu memisahkan masing-masing test object berdasarkan komponennya, misal jika komponennya Button maka kita perlu membuat folder button yang nanti isinya adalah test object button semua, dan komponen lainnya juga sama.

Kita tidak bisa mengabungkan semua test object dengan bermacam komponen didalam folder yang sama (tanpa dipisah), karena akan membuat ambigu dan sulit untuk diorganisir ketika test object nya besar atau banyak.

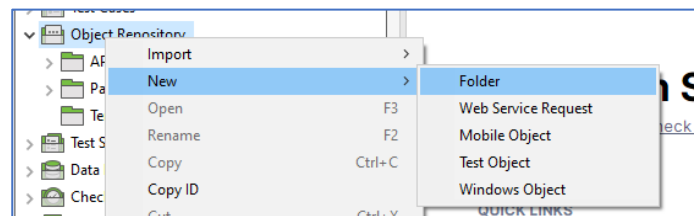
## Create Folder & Object

- **Create Folder**



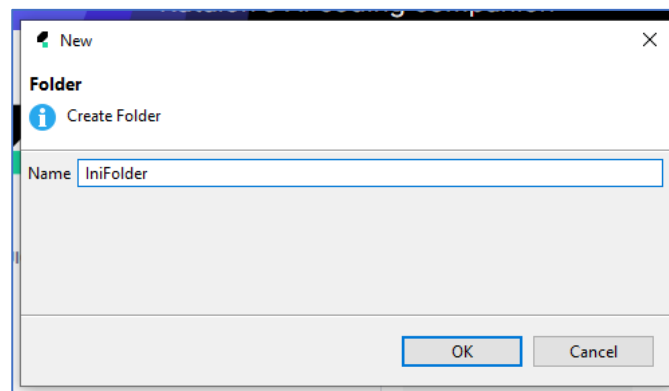
Gambar 16

Berikut cara untuk membuat folder didalam folder Object Repository :



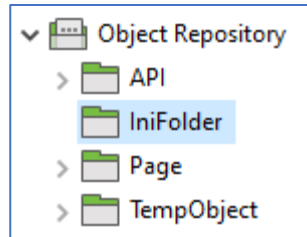
Gambar 17

Pada gambar diatas, klik kanan pada folder “Object Repository” lalu pilih New → Folder.



Gambar 18

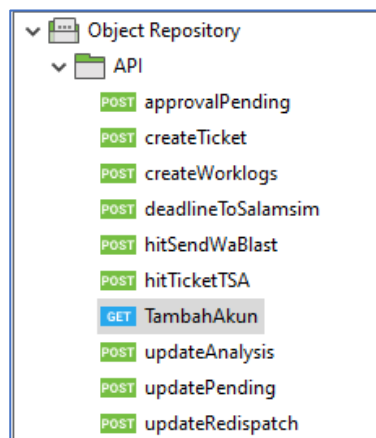
Jika sudah, akan menampilkan dialog new folder pada gambar diatas, isi nama folder yang diinginkan lalu klik tombol OK.



Gambar 19

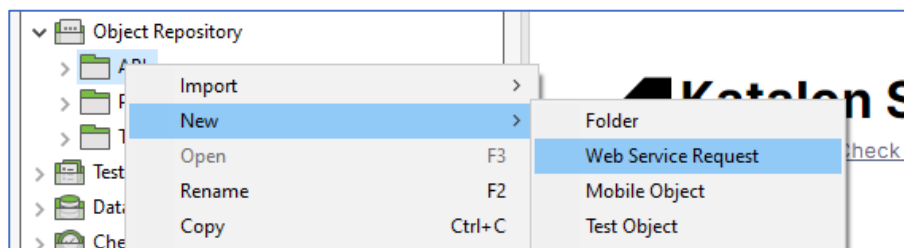
Pada gambar diatas, folder baru didalam Object Repository berhasil dibuat.

- **API Object**



Gambar 20

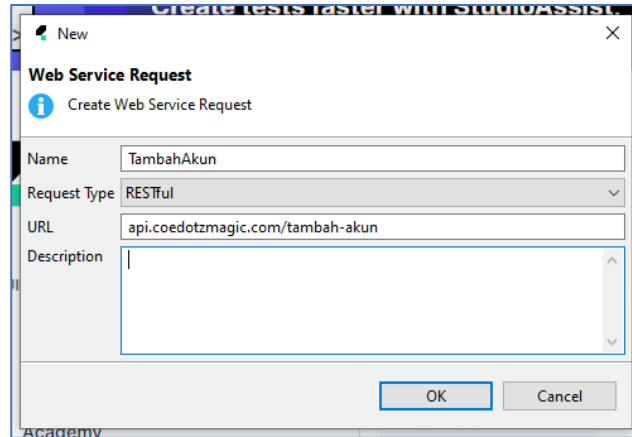
Berikut cara membuat API Object pada Object Repository :



Gambar 21

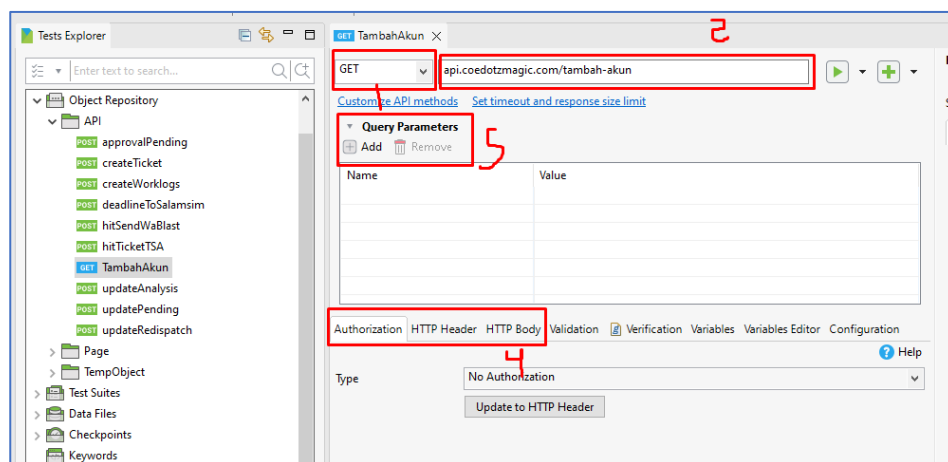
Pada gambar diatas, klik kanan pada folder API lalu pilih New → Web Service Request.





Gambar 22

Pada gambar diatas, akan muncul dialog New, lalu isikan Name Request Type (RESTful atau SOAP) dan URL API nya, jika sudah klik OK.

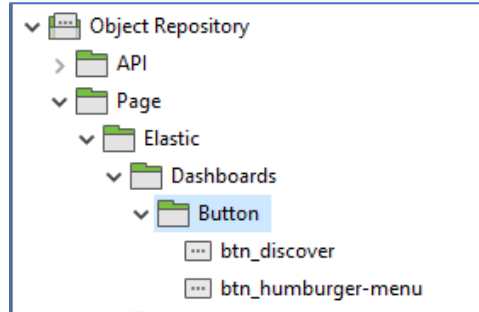


Gambar 23

Pada gambar diatas, kita berhasil membuat API Object.

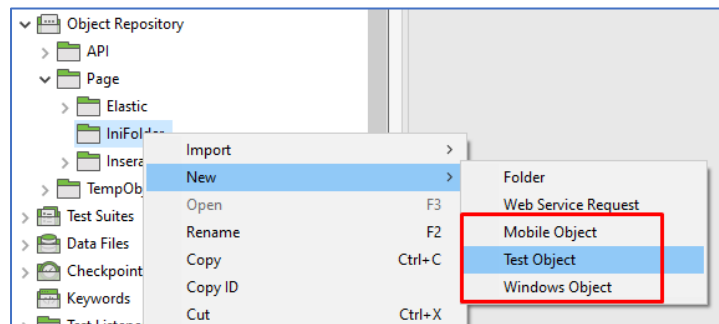
1. **Point 1**, disini kita bisa merubah antara ingin menggunakan GET, POST atau lainnya.
2. **Point 2**, kita bisa merubah url dari API nya, sesuai keinginan.
3. **Point 4**, memiliki fitur tab yang mirip seperti penggunaan Postman, tab untuk mengisi bagian authentication, header dan body.
4. **Point 5**. Kita bisa melakukan penambahan atau pengurangan parameter dengan cara Add atau Remove, sesuai kebutuhan.

- **Test Object (Web, Mobile & Desktop)**



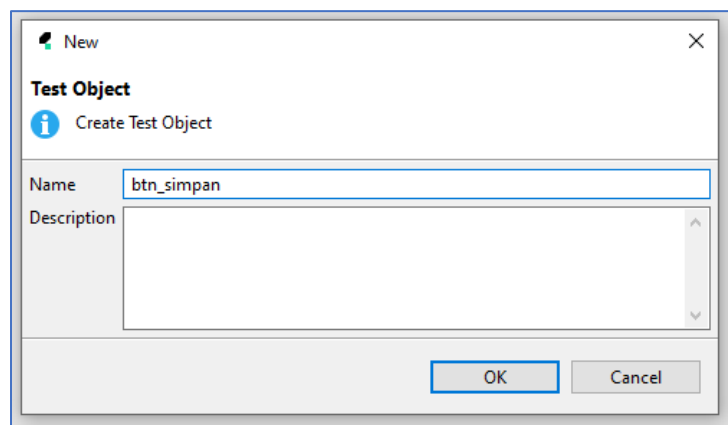
Gambar 24

Berikut cara membuat test object pada Object Repository :



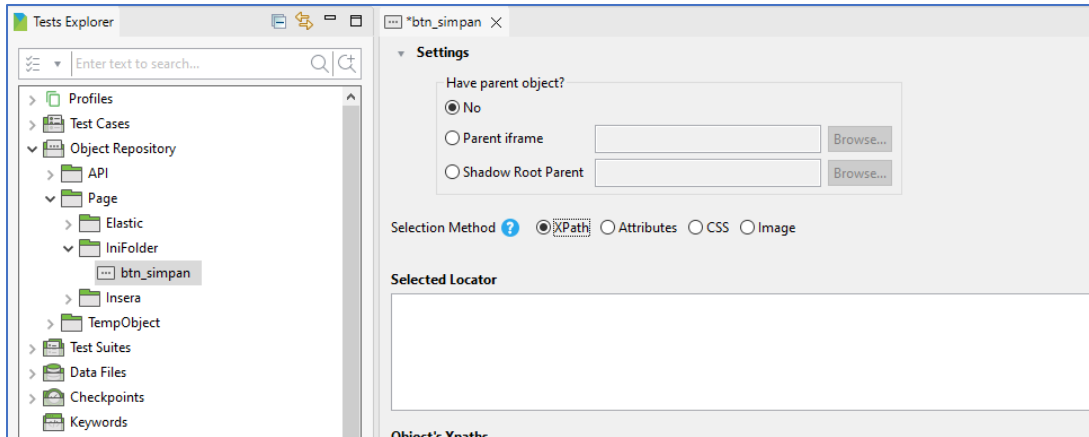
Gambar 25

Pada gambar diatas, lakukan klik kanan pada folder lalu New → Test Object, jika Test Object berupa windows application kita bisa menggunakan Windows Object, dan jika berupa mobile, kita bisa menggunakan Mobile Object.



Gambar 26

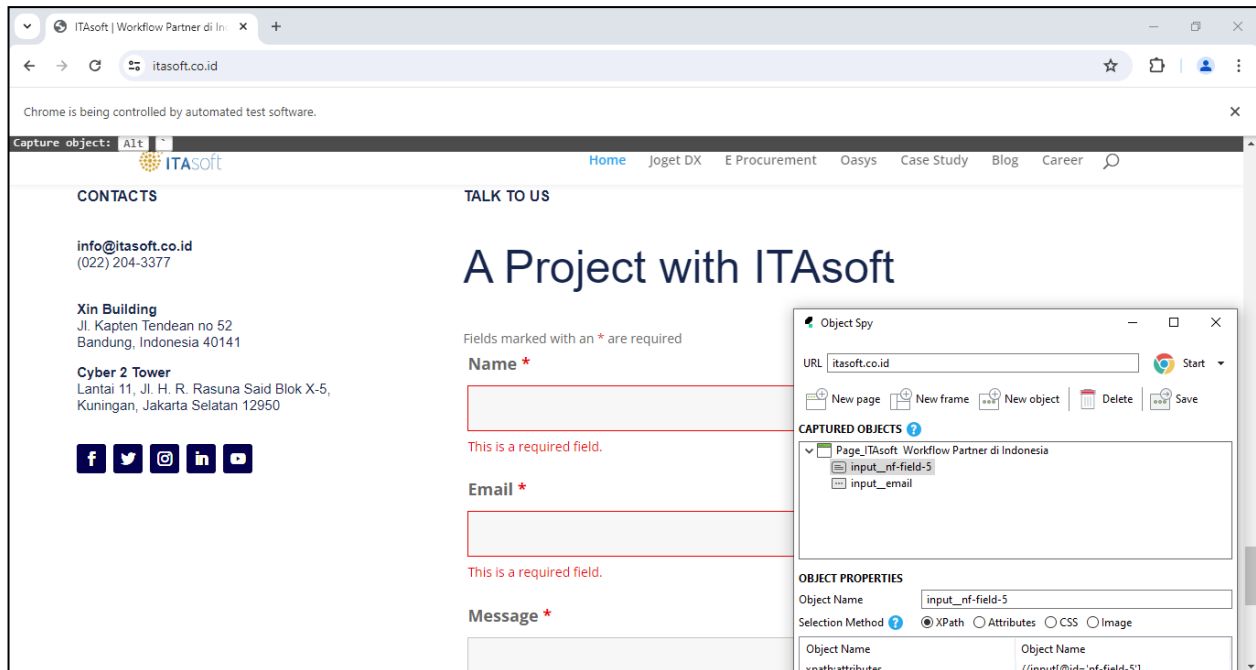
Pada gambar diatas, menampilkan dialog New, isikan nama test object yang diinginkan, lalu klik OK.



Gambar 27

Test Object berhasil terbuat, dan kita bisa mengisi locator xpath didalam text area Selected Locator, dan jangan lupa lakukan Save (CTRL+S / Command+S) untuk menyimpan test object jika melakukan perubahan.

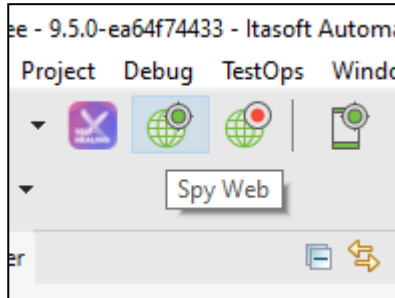
## Capture Object



Gambar 28

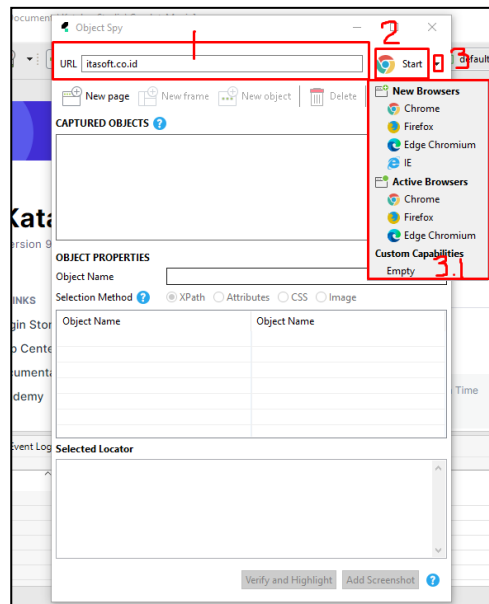
Untuk melakukan pengujian terhadap suatu perangkat lunak, kita memerlukan test object dalam membantu pengujian yang ingin dilakukan, maka dari itu diperlukan pengambilan test object menggunakan metode **Spy Web & Record Web**, 2 metode tersebut dapat membantu dalam membuat atau mengembangkan test object untuk dalam pengujian di Katalon Studio. Kapan waktu yang tepat untuk menggunakan **Spy Web & Record Web** dan apa bedanya?

## Capture Object – Spy Web



Gambar 29

**Spy Web** : Digunakan untuk capture Sebagian test object atau objek yang diinginkan. Metode ini cocok digunakan untuk automation yang telah terbuat Test Object sebelumnya, dan menghindari redundansi atau duplikasi terhadap Test Object.

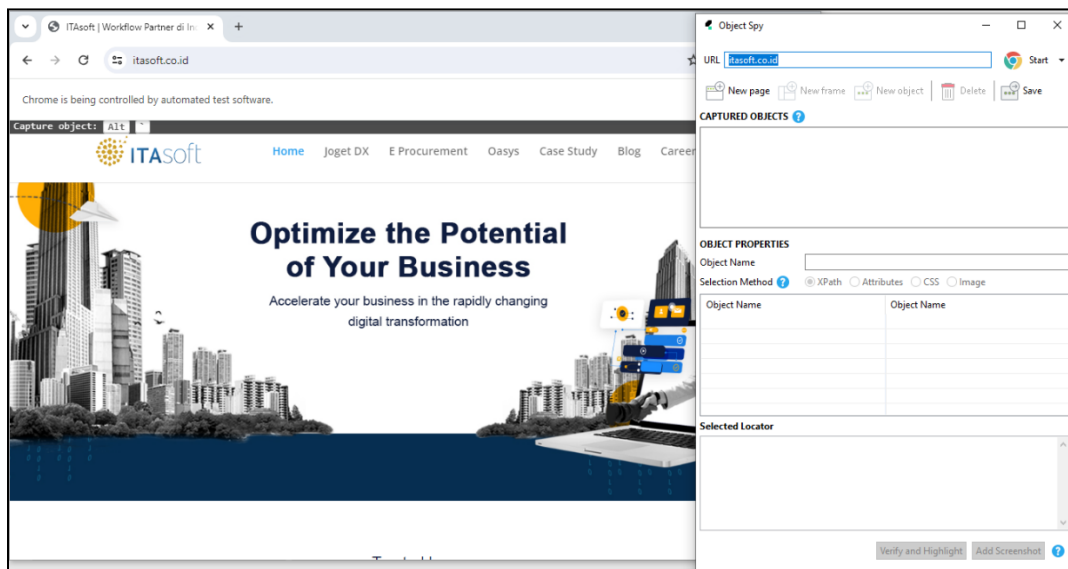


Gambar 30

Pada gambar diatas akan dijelaskan beberapa bagian yang telah ditandai, yaitu

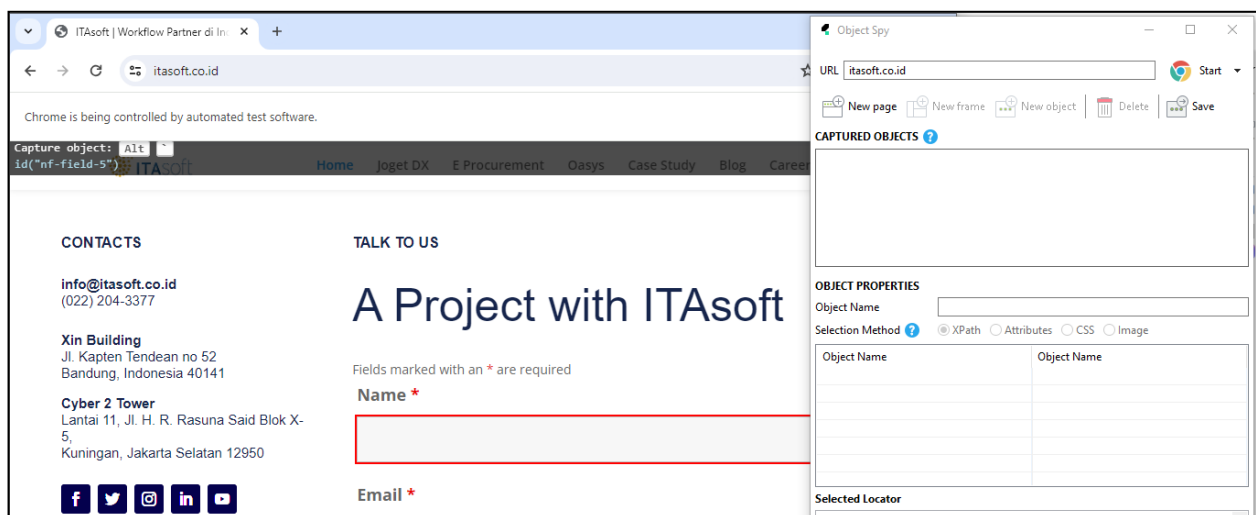
1. **URL**, Dimana url ini digunakan untuk target website yang objeknya akan di capture dengan spy web.
2. **Start**, Jika URL nya sudah diset, maka bisa langsung di start untuk memulai proses capture object menggunakan metode spy web.

3. **Choose Browser**, disini kita bisa memilih browser yang ingin digunakan dalam capture object, biasanya browser chrome paling umum digunakan untuk capture object.

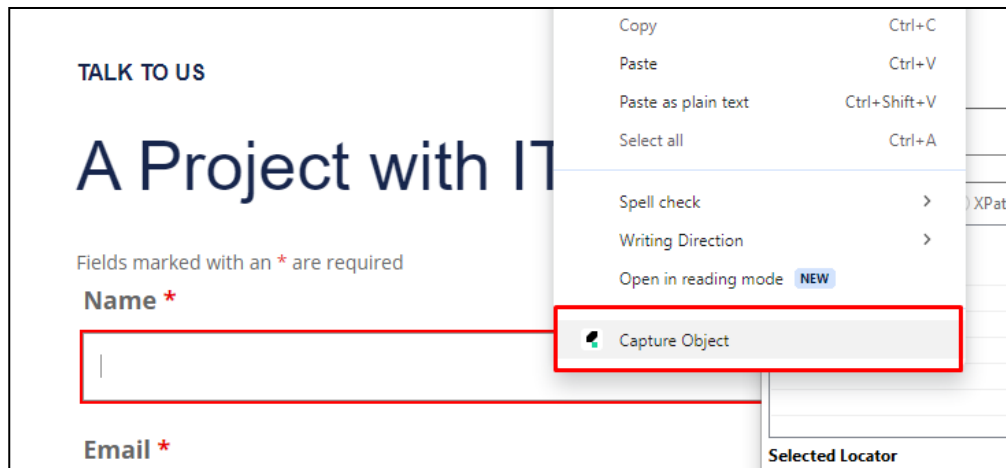


Gambar 31

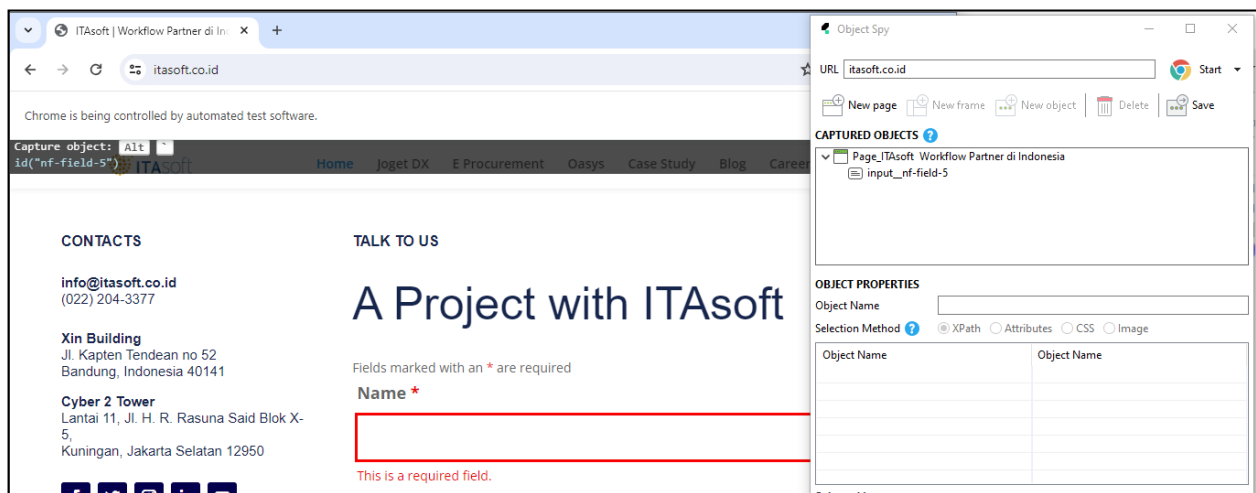
Pada gambar diatas, jika sudah pencet tombol **“Start”**, Dimana akan membuka Web Driver (Browser Automation) yang nantinya akan digunakan untuk melakukan capture objek, mungkin cukup mengganggu karena dialog Object Spy nya menutupi browser, tetapi tidak disarankan untuk mengclose atau minimize dialog Object Spy, karena akan menghambat atau mengakhiri proses capture object.



Gambar 32



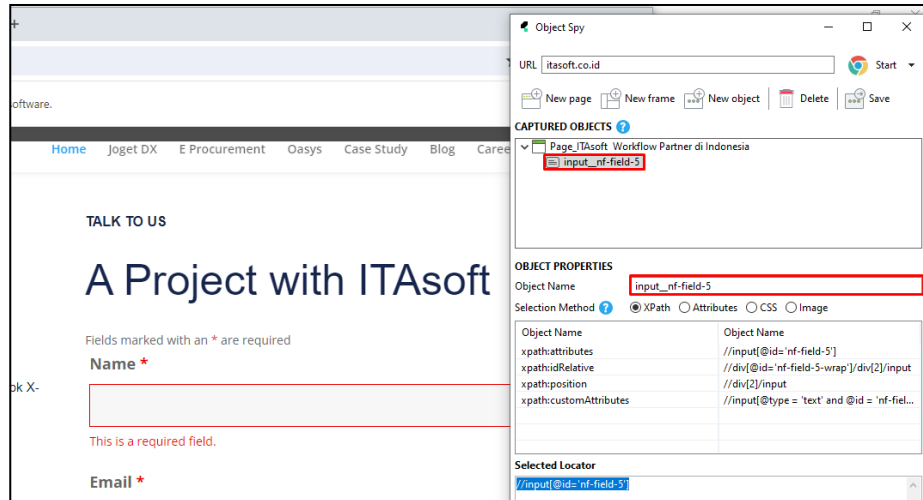
Gambar 33



Gambar 34

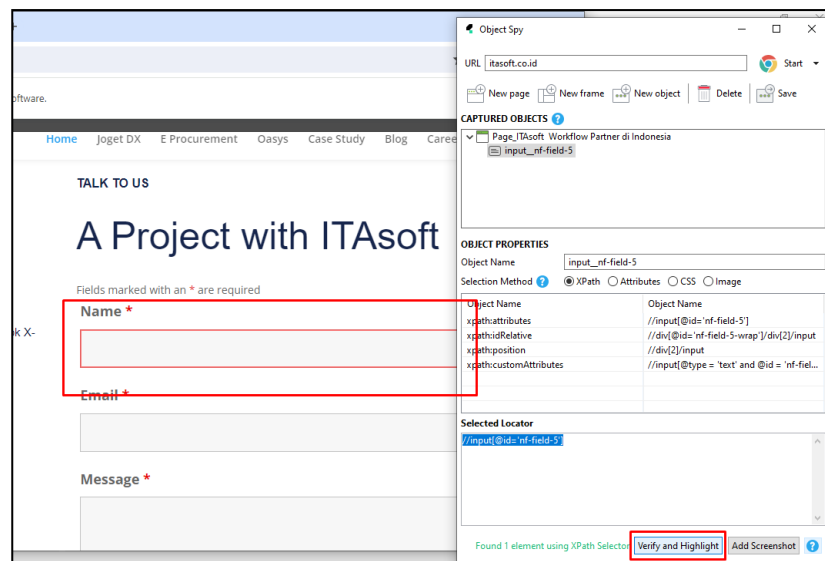
Pada gambar diatas (gambar 33-34) merupakan proses capture object menggunakan metode spy object, Dimana kita memilih dahulu objek atau komponen yang ingin di capture, dengan cara klik kanan pada objek atau komponen yang dicapture lalu klik **“Capture Object”** atau bisa juga menggunakan shortcut **alt+`**, tetapi lebih disarankan untuk menggunakan klik kanan pada komponen atau objek yang ingin di capture lalu pilih Capture Object.

Jika berhasil akan terlihat seperti gambar 34, Dimana objectnya berhasil dicapture (terlihat pada dialog Object Spy).



Gambar 35

Pada gambar diatas, kita bisa melakukan perubahan nama test object yang telah dicapture, dengan cara klik objek yang kita inginkan, dan pada object properties dibagian object name, kita bisa langsung merubah nama test object yang kita inginkan. Untuk perubahan nama ini tidak boleh kosong ataupun Panjang melebihi 255 karakter, jika kosong atau melebihi 255 karakter maka test object tidak bisa disimpan.



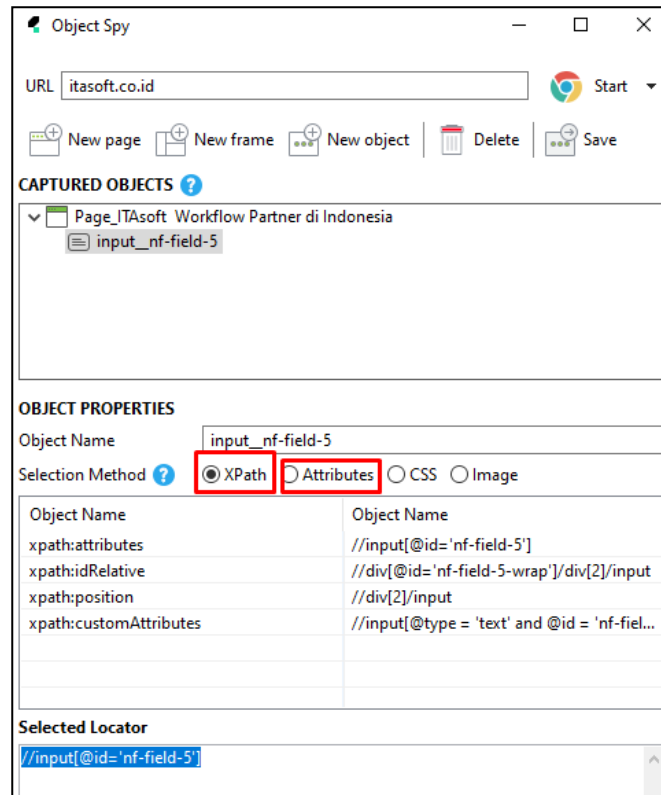
Gambar 36

Pada gambar diatas, ketika kita sudah berhasil melakukan capture object, maka lakukan “**Verify and Highlights**” objek tersebut, apakah objek yang kita capture adalah objek komponen yang benar atau



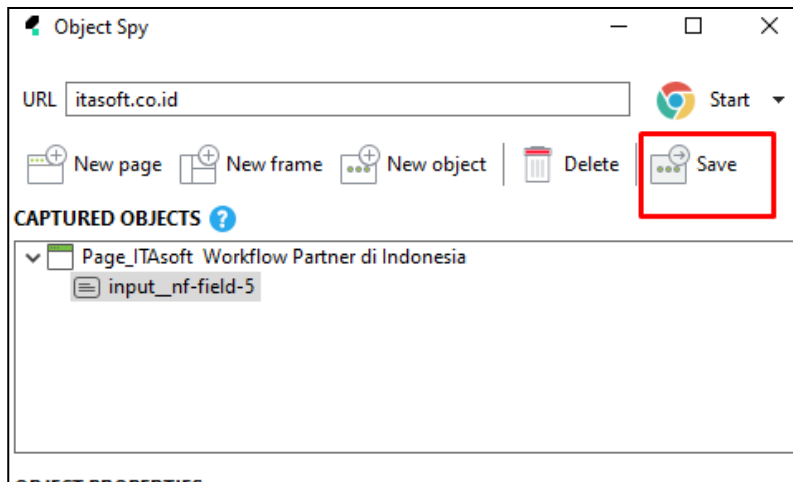
tidak, jika benar nanti pada bagian yang kita capture akan muncul kelap-kelip berwarna merah, jika tidak benar, dia tidak akan menampilkan kelap-kelip merah.

Jika kita tidak melakukan Verify and Highlights sebagai verifikasi komponen yang telah kita capture, maka ada kemungkinan ketika diterapkan kedalam test script di test case akan mengakibatkan error, jika komponen test objek yang telah di capture tidak sesuai dengan komponen yang ada di web atau platform lainnya.

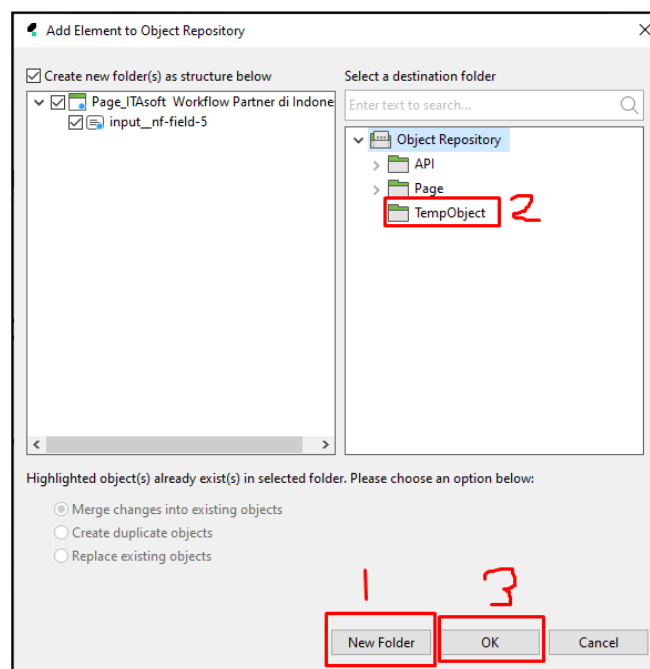


Gambar 37

Pada gambar diatas, jika pada saat melakukan Verify and Highlights dan akhirnya tidak kedip merah, maka bisa lakukan perubahan pada selection method dari XPath ke Attributes, dan lakukan pengecekan pada setiap checkbox yang ada pada atribut hingga pada saat Verify and Highlights komponen test objeknya akan kedip merah, yang menandakan bahwa capture test object sesuai dengan komponen yang ada di web atau platform lainnya.



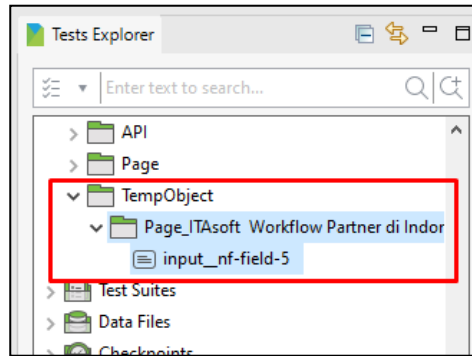
Gambar 38



Gambar 39

Pada gambar diatas (gambar 38-39), kita akan melakukan penyimpanan test object yang telah di capture sebelumnya menggunakan metode object spy. Jika sudah tahap seperti gambar 39, kita perlu lakukan pembuatan folder baru terlebih dahulu di dalam folder Object Repository, dengan nama “TempObject”, gunanya untuk menyimpan test object sementara, yang nantinya test object tersebut akan di test dengan cara di implementasikan ke test script yang ada di test case, diuji apakah test object yang kita capture barusan bisa diproses pada saat testing atau tidak, jika tidak, kita bisa melakukan capture ulang, dan jika berhasil, kita bisa pindahkan kembali test object tersebut kedalam folder yang kita inginkan.

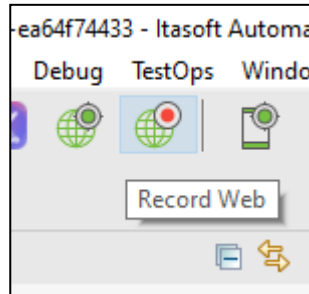
Jika sudah klik OK untuk menyimpan test object yang telah kita capture.



Gambar 40

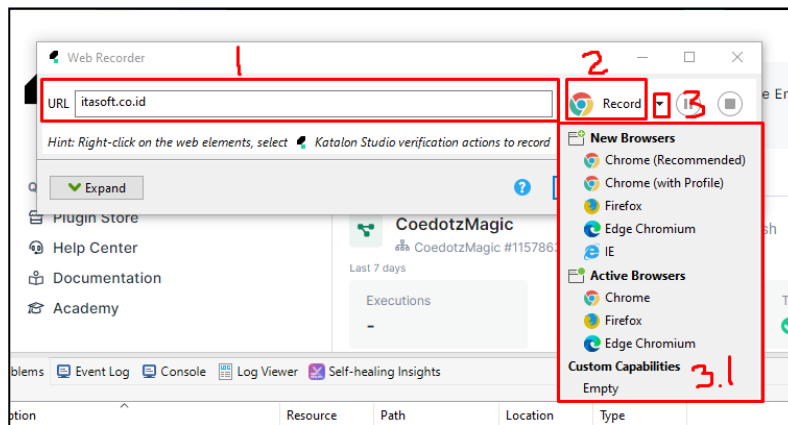
Pada gambar diatas, kita sudah berhasil melakukan penyimpanan test object yang telah kita capture, dan berhasil disimpan ke dalam folder Object Repository.

## Capture Object – Record Web



Gambar 41

**Record Web** : Digunakan untuk capture objek yang telah diklik oleh kursor, selama lakukan record, dia akan mencapture objek yang telah terklik oleh kursor. Metode ini cocok digunakan untuk automation yang belum ada Test Object atau proyek awal pembuatan automation.

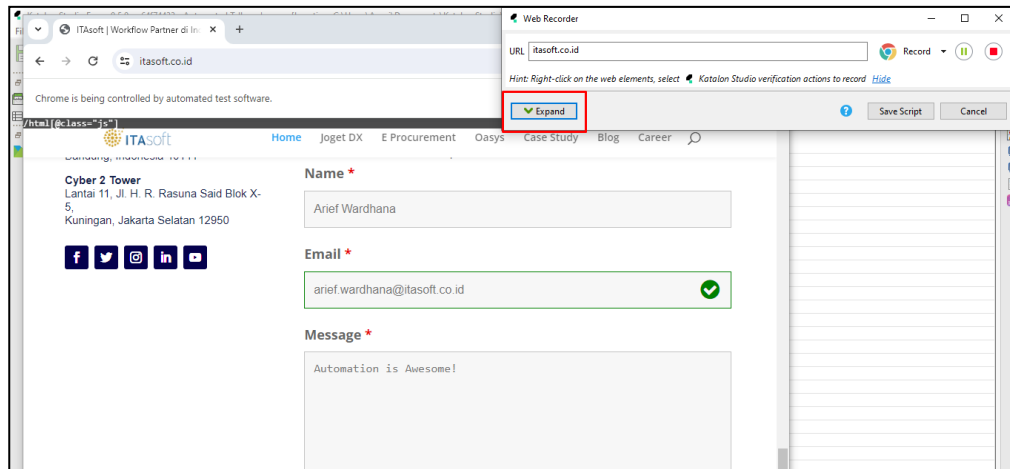


Gambar 42

Pada gambar diatas akan dijelaskan beberapa bagian yang telah ditandai, yaitu

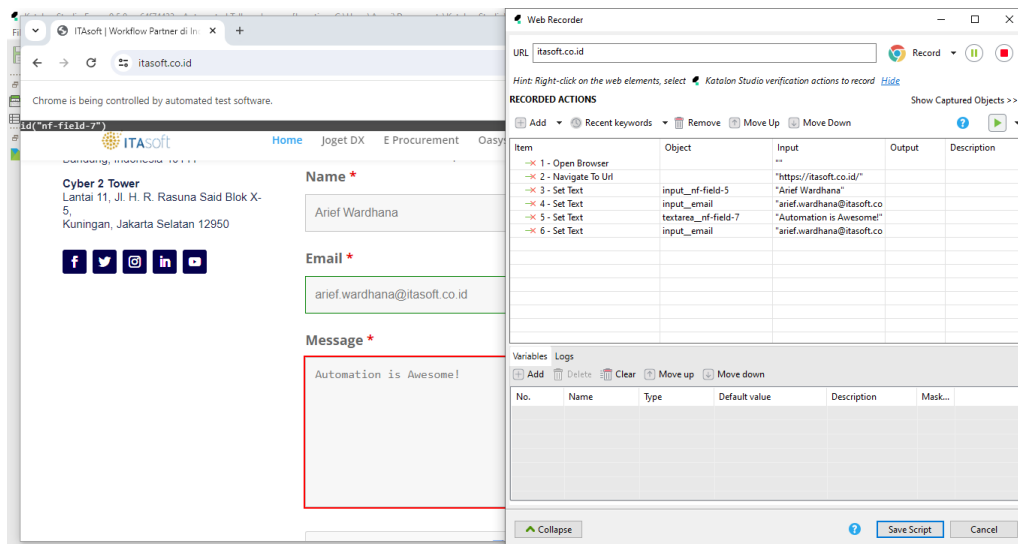
- 1. URL**, Dimana url ini digunakan untuk target website yang objeknya akan di capture dengan Web Recorder.
- 2. Start**, Jika URL nya sudah diset, maka bisa langsung di start untuk memulai proses capture object menggunakan metode Web Recorder.
- 3. Choose Browser**, disini kita bisa memilih browser yang ingin digunakan dalam capture object, biasanya browser chrome paling umum digunakan untuk capture object.

Jika ingin melakukan pengambilan test object menggunakan metode Web Recorder, kita diwajibkan untuk menyiapkan sebuah Test Case untuk menampung keyword yang telah terekam pada saat melakukan Web Recorder, **Disarankan untuk menggunakan Test Case baru agar tidak mengakibatkan rewrite test case yang sudah ada.**



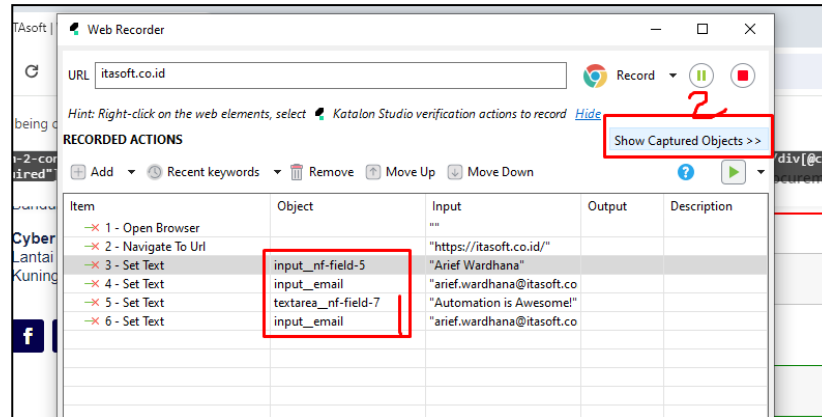
Gambar 43

Pada gambar diatas, ketika kita dalam proses Web Recorder, kita bisa melihat secara detail komponen atau objek yang telah kita klik atau interaksi dengan komponen atau objek tersebut, dengan klik tombol “Expand”.

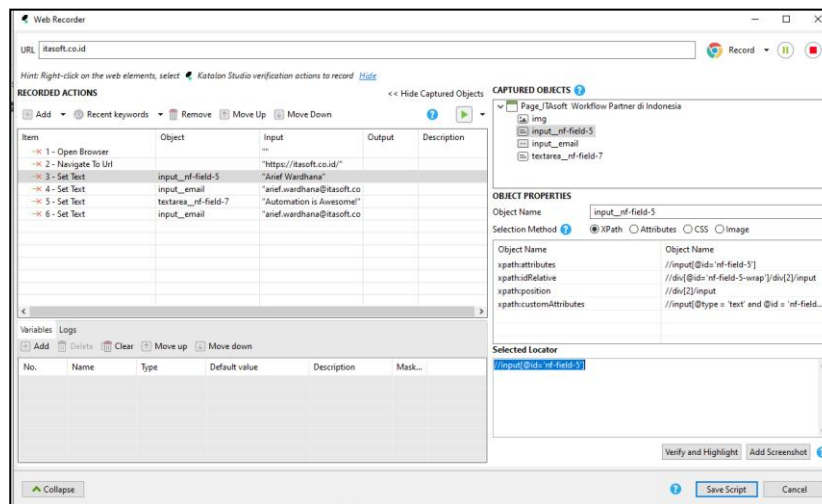


Gambar 44

Pada saat melakukan proses Web Recorder, setiap Gerakan cursor/pointer yang mengarahkan ke komponen (seperti aksi klik dan input) akan langsung terekam secara otomatis (seperti gambar diatas sebelah kanan).

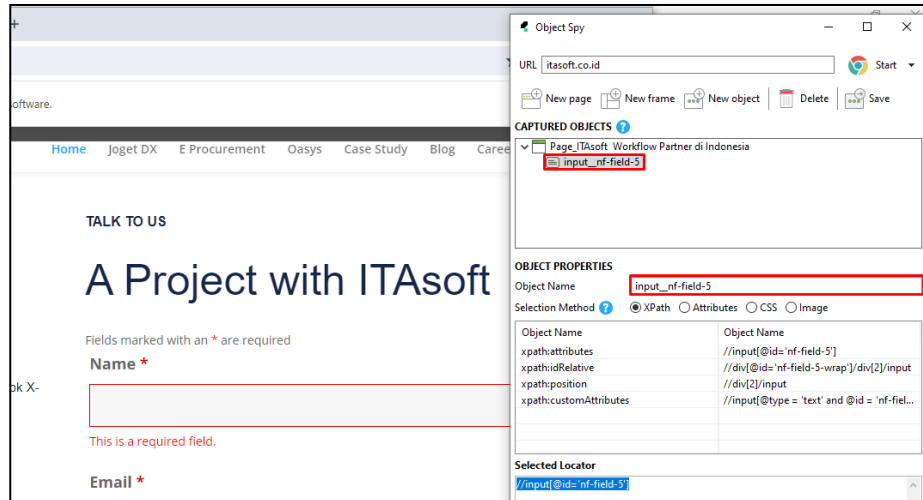


Gambar 45



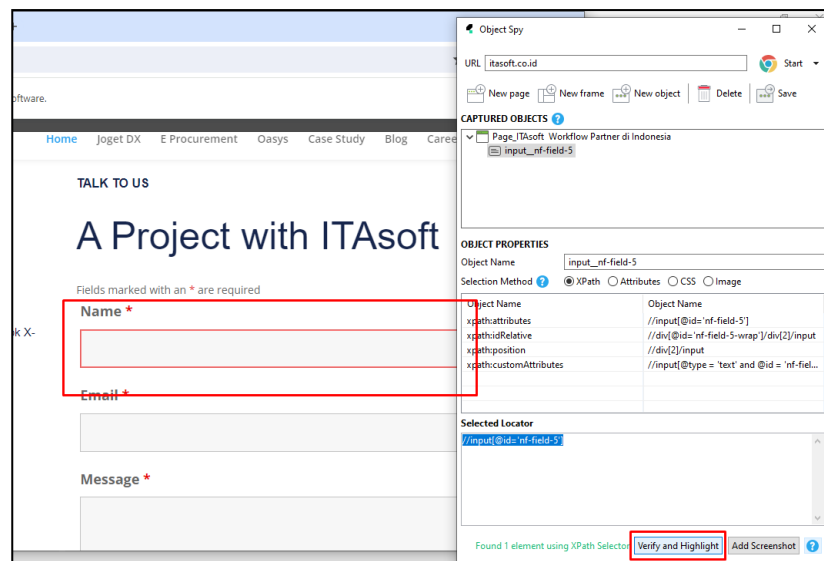
Gambar 46

Pada gambar diatas, kita dapat melihat secara rinci object properties untuk masing-masing test object yang telah dicapture.



Gambar 47

Pada gambar diatas, kita bisa melakukan perubahan nama test object yang telah dicapture, dengan cara klik objek yang kita inginkan, dan pada object properties dibagian object name, kita bisa langsung merubah nama test object yang kita inginkan. Untuk perubahan nama ini tidak boleh kosong ataupun Panjang melebihi 255 karakter, jika kosong atau melebihi 255 karakter maka test object tidak bisa disimpan.

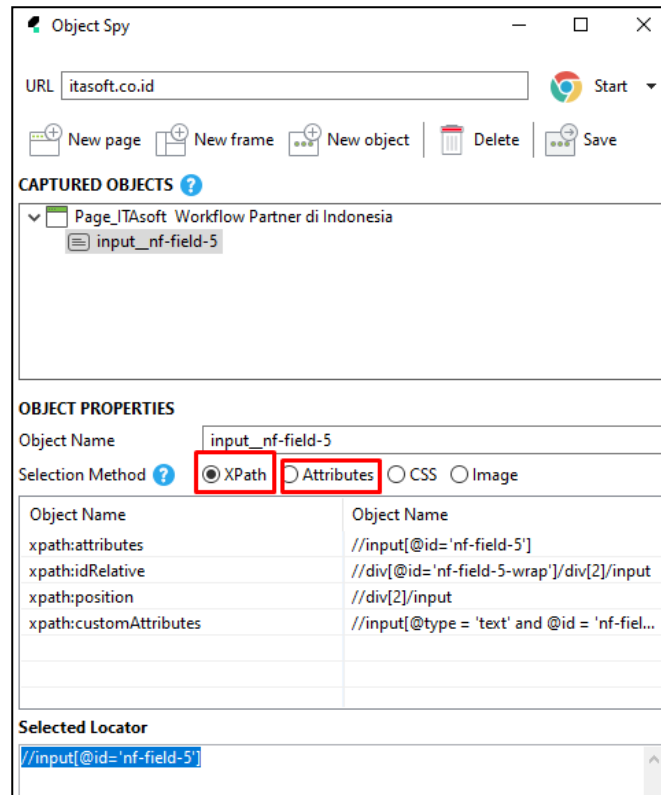


Gambar 48

Pada gambar diatas, ketika kita sudah berhasil melakukan capture object, maka lakukan “**Verify and Highlights**” objek tersebut, apakah objek yang kita capture adalah objek komponen yang benar atau

tidak, jika benar nanti pada bagian yang kita capture akan muncul kelap-kelip berwarna merah, jika tidak benar, dia tidak akan menampilkan kelap-kelip merah.

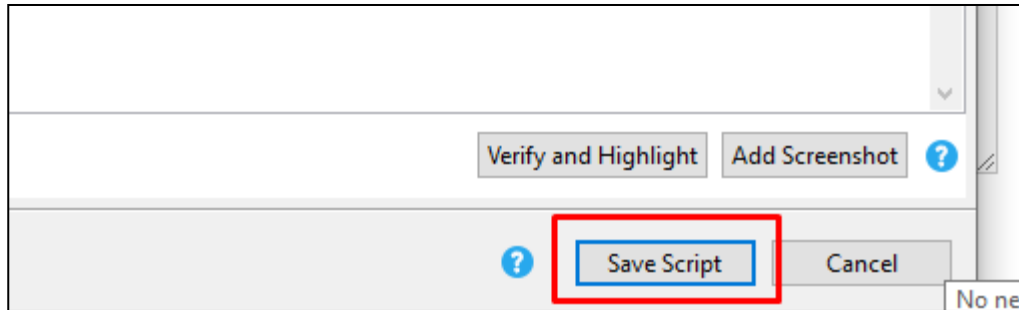
Jika kita tidak melakukan Verify and Highlights sebagai verifikasi komponen yang telah kita capture, maka ada kemungkinan ketika diterapkan kedalam test script di test case akan mengakibatkan error, jika komponen test objek yang telah di capture tidak sesuai dengan komponen yang ada di web atau platform lainnya.



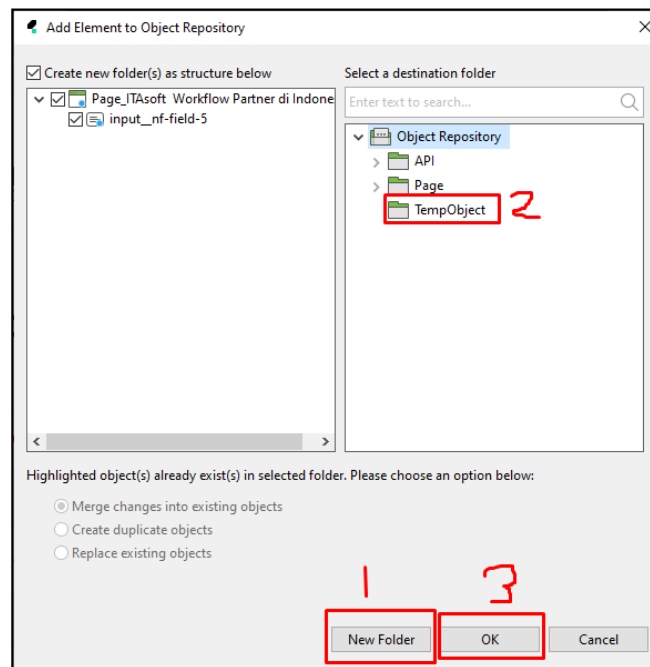
Gambar 49

Pada gambar diatas, jika pada saat melakukan Verify and Highlights dan akhirnya tidak kedip merah, maka bisa lakukan perubahan pada selection method dari XPath ke Attributes, dan lakukan pengecekan pada setiap checkbox yang ada pada atribut hingga pada saat Verify and Highlights komponen test objek nya akan kedip merah, yang menandakan bahwa capture test object sesuai dengan komponen yang ada di web atau platform lainnya.



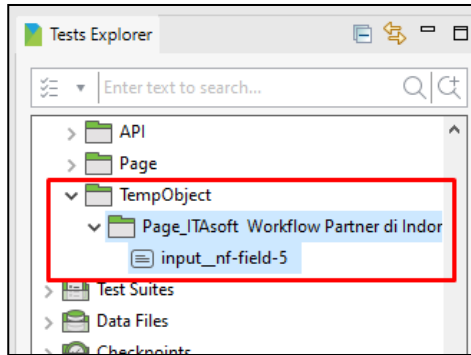


Gambar 50



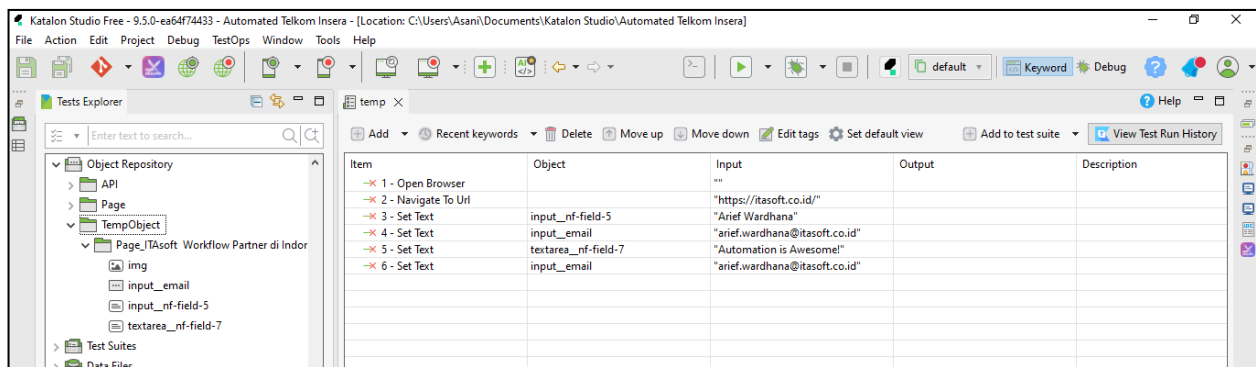
Gambar 51

Pada gambar diatas (gambar 50-51), kita akan melakukan penyimpanan test object yang telah di record sebelumnya menggunakan metode web recorder. Jika sudah tahap seperti gambar 51, kita perlu lakukan pembuatan folder baru terlebih dahulu di dalam folder Object Repository, dengan nama “TempObject”, gunanya untuk menyimpan test object sementara, yang nantinya test object tersebut akan di test dengan cara di implementasikan ke test script yang ada di test case, diuji apakah test object yang kita record/capture barusan bisa diproses pada saat testing atau tidak, jika tidak, kita bisa melakukan record/capture ulang, dan jika berhasil, kita bisa pindahkan kembali test object tersebut kedalam folder yang kita inginkan. Jika sudah klik OK untuk menyimpan test object yang telah kita record/capture.



Gambar 52

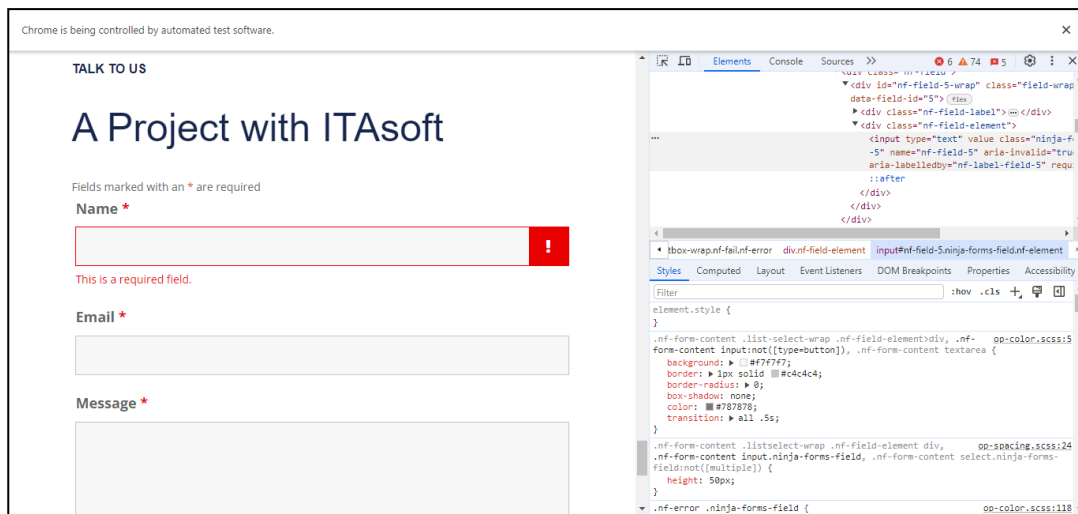
Pada gambar diatas, kita sudah berhasil melakukan penyimpanan test object yang telah kita capture, dan berhasil disimpan ke dalam folder Object Repository.



Gambar 53

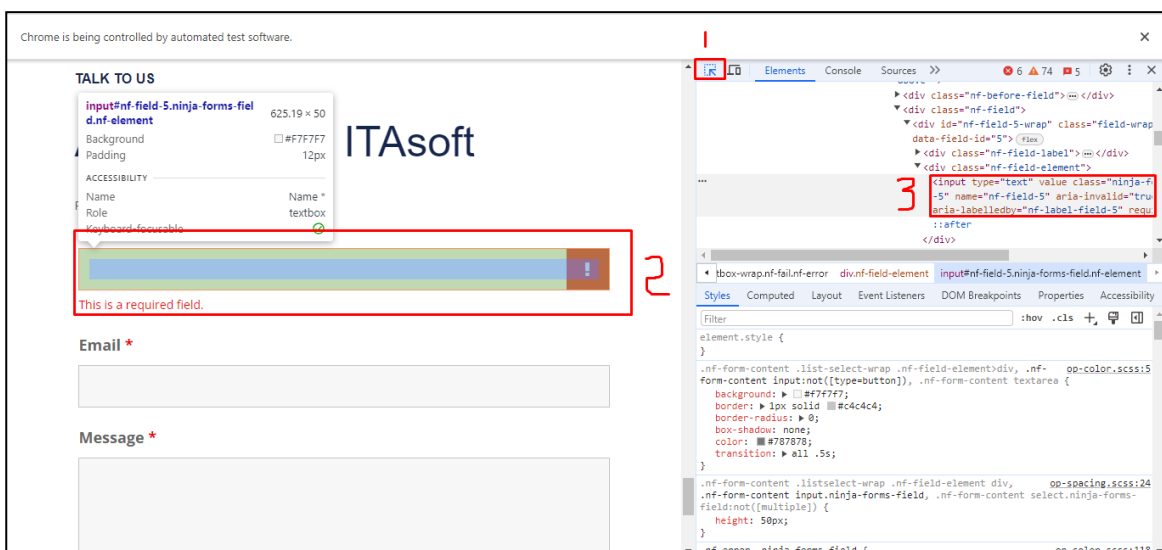
Dan yang terakhir, pada gambar diatas, kita sudah selesai melakukan proses Record Web, dan setelah kita lakukan save script, maka akan menampilkan gambar diatas, dengan maksud, telah menyimpan script tersebut kedalam testcase yang sudah kita siapkan.

## Capture Object – Alternative Capture



Gambar 54

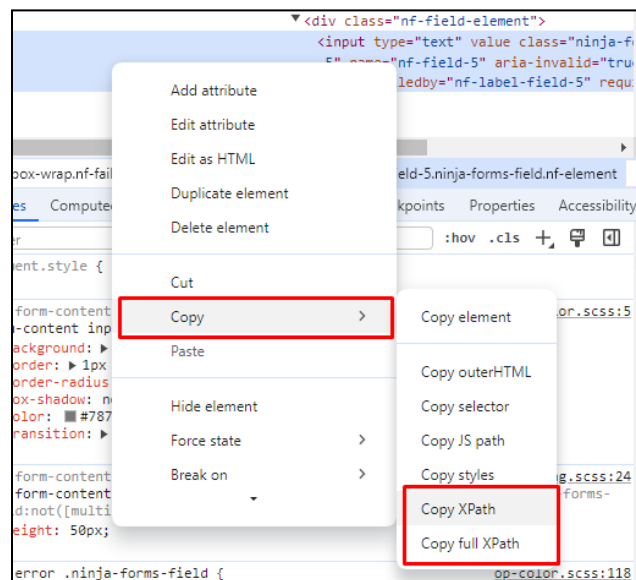
Alternative Capture, metode ini digunakan jika tidak dapat melakukan Verify and Highlights pada saat capture object menggunakan metode Spy Web atau Web Record, Dimana komponen test object yang sudah di capture tidak bisa di verify pada Object Properties baik melalui Selection Method Xpath atau Attributes. Disini kita akan mencoba melakukan pengambilan Xpath secara manual menggunakan Inspect Element yang ada pada browser.



Gambar 55

Pada gambar diatas dijelaskan beberapa Langkah cara untuk mendapatkan Xpath melalui inspect element untuk kebutuhan test object, jangan lupa untuk membuka inspect element pada halaman browser untuk mendapatkan xpath dalam keperluan pembuatan test object.

1. **Select Element**, Dimana kita akan menggunakan select element dalam mengambil element / komponen yang ingin kita ambil xpathnya.
2. **Arahkan ke element yang diinginkan**, Disini kita akan mengarahkan element apa saja yang kita inginkan untuk mendapatkan xpathnya di inspect element.
3. **Lokasi Element di Inspect Element**, jika sudah diarahkan, kita akan mendapatkan lokasi element yang terdapat pada Inspect Element.



Gambar 56

Pada gambar diatas, ketika kita sudah berhasil mendapatkan Lokasi element di inspect element, lakukan klik kanan → Copy → Copy Xpath / Full Xpath. Disini kita bisa mengambil berdasarkan Xpath atau Full Xpath sesuai kebutuhan, dengan perbedaan sebagai berikut :

**Xpath** : Kita akan mendapatkan Xpath berupa id element tanpa harus menspesifik hingga root elemetnya.

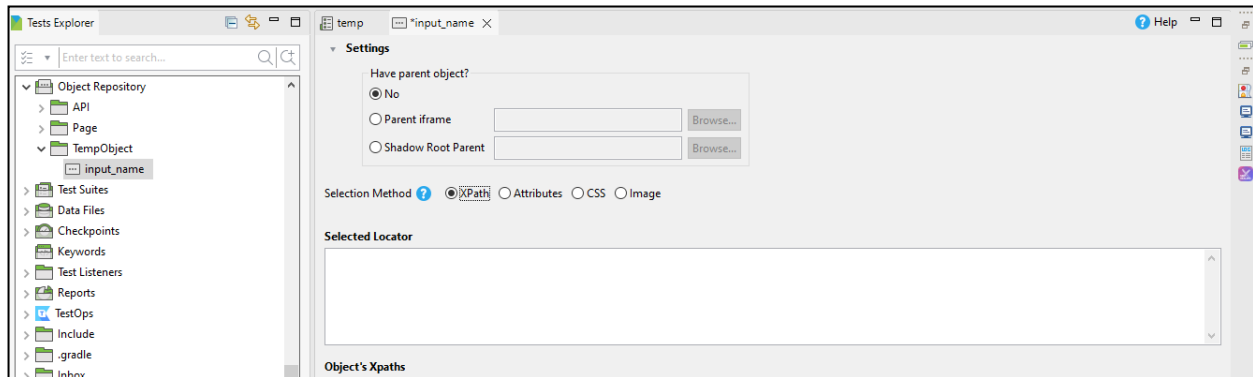
**Full Xpath** : Kita akan mendapatkan Xpath berupa id element hingga root elementnya.

**Xpath :**

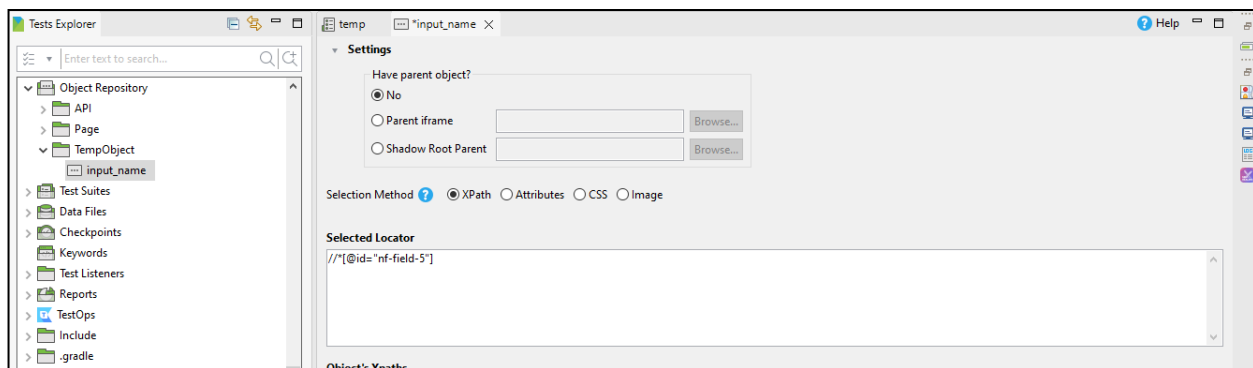
```
//*[@id="nf-field-5"]
```

### Full Xpath :

```
/html/body/div[1]/div/footer/div[1]/div/div[2]/div[3]/div/div/div/div[4]/form/div/div[2]/nf-fields-wrap/nf-field[1]/div/div[2]/div/div[2]/input
```



Gambar 57

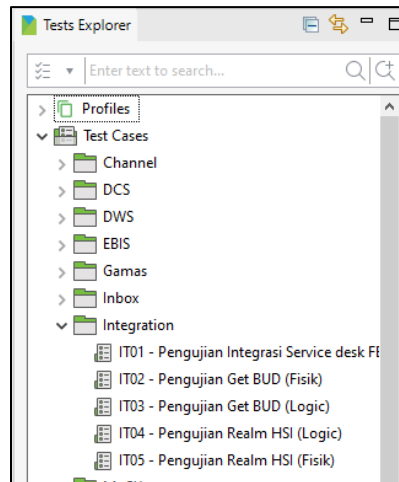


Gambar 58

Ketika sudah berhasil menyalin Xpathnya, bisa kita tempel seperti gambar diatas, pastikan Selection Methodnya adalah XPath pada properties Test Object.

## Test Case

---



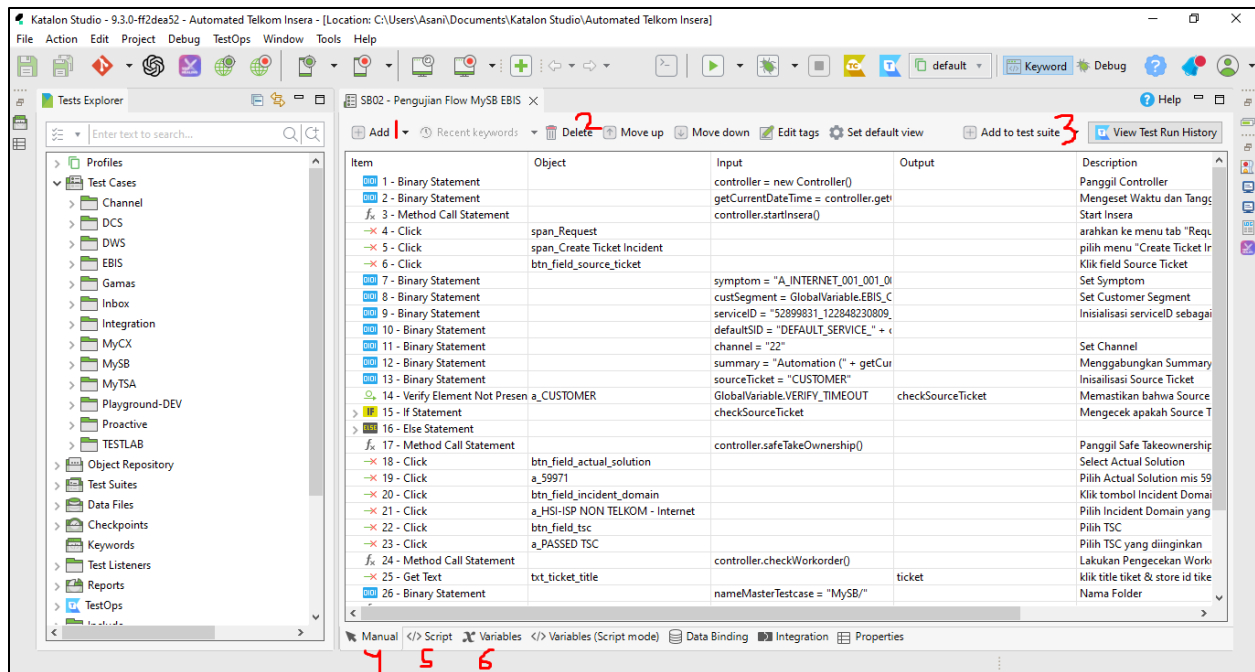
Gambar 59

Test Case ini digunakan untuk menampung seluruh flow atau scenario yang ada untuk menjalankan proses Automation, Misalnya kita ingin membuat pengujian untuk kalkulator kalo 5x5 adalah 25, nanti dibuatlah sebuah testcase nya, dengan scenario positif, Dimana scenario positif itu sebagai hasil pengujian yang seharusnya dan Skenario Negatif Dimana melakukan flow yang tidak semestinya / kemungkinan error.

Balik lagi ke kalkulator, nah di test case kita bikin alur untuk melakukan perkalian  $5 \times 5 = 25$  dengan flow.

1. **Masukan Bilangan Pertama**
2. **Klik Tombol "X"**
3. **Masukan Bilangan Kedua**
4. **Klik Tombol "="**
5. **Tampilkan Hasil**

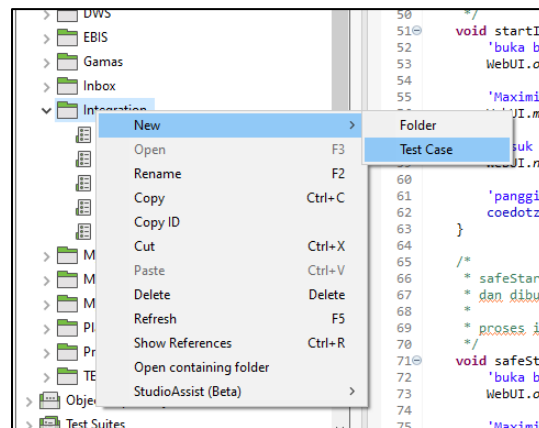
Nah dari situ baru dibuat testcase berdasarkan flow yang sudah ada.



Gambar 60

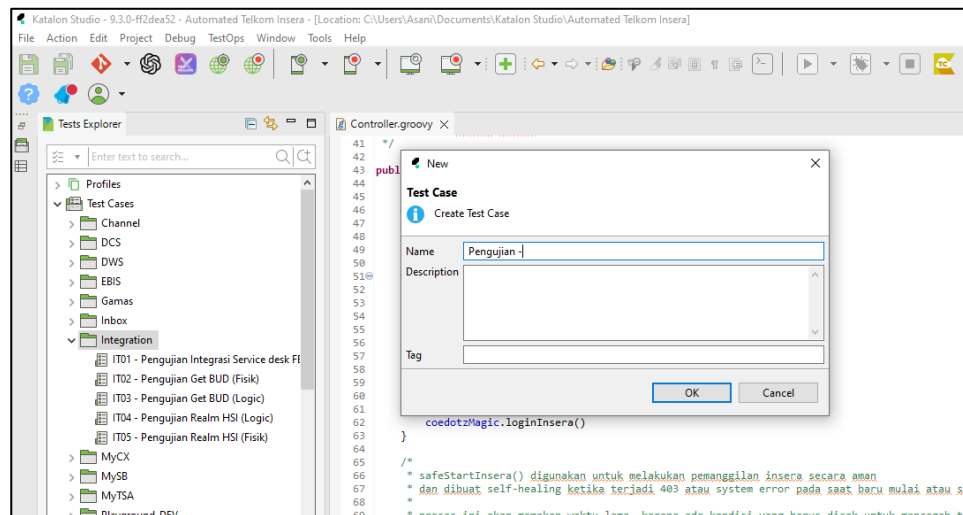
1. **Add** : Digunakan untuk membuat keyword atau perintah automation.
2. **Delete** : Digunakan untuk menghapus keyword atau perintah automation.
3. **Add to Test Suite** : Digunakan untuk menambahkan Testcase yang dibuat ke Test Suite, dan melakukan pengujian automation secara Borongan dengan testcase lain.
4. **Manual** : Digunakan untuk melakukan pembuatan perintah automation secara manual (via UI).
5. **Script** : Digunakan untuk melakukan pembuatan perintah automation menggunakan script (via Ngoding).
6. **Variables** : Tempat untuk menampung variable local, bisa juga menggunakan Global Variable agar dapat menggunakan variable dikeseluruhan projek.

## Membuat Test Case



Gambar 61 - Membuat Testcase

Untuk membuat testcase sebelumnya perlu dikelompokkan berdasarkan folder testcase dengan alasan, agar lebih rapih dan terorganisir, jika ingin membuat testcase, pastikan sudah ada folder testcasenya, jika belum bisa diklik kanan → Folder, jika sudah ikuti gambar diatas untuk melakukan pembuatan testcase.

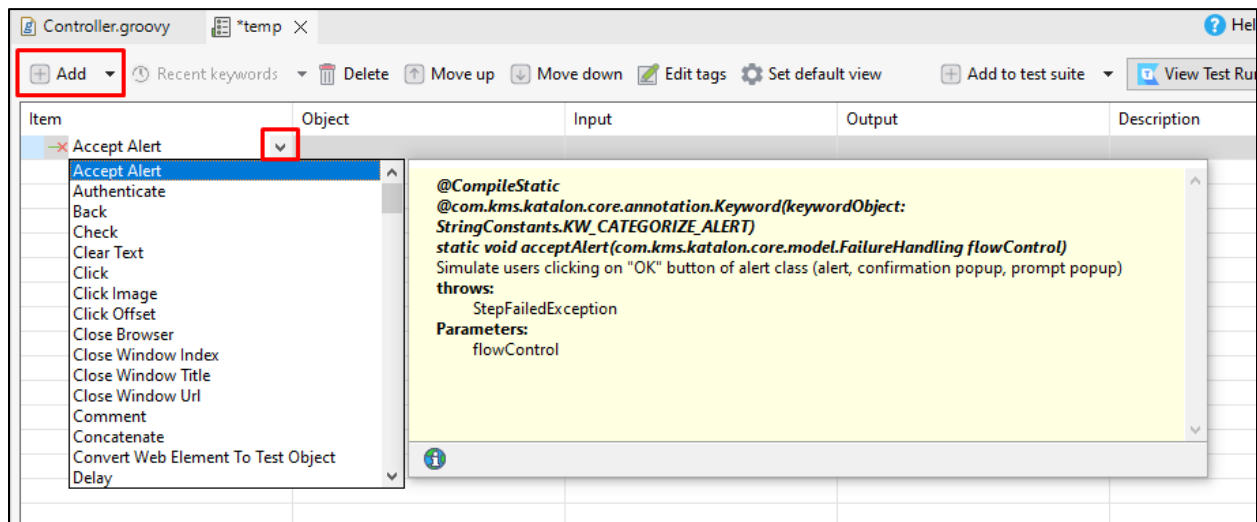


Gambar 62 - Form Pembuatan Testcase

Nah jika muncul form seperti ini, tinggal diisi aja Nama Testcasenya apa, untuk Deskripsi dan Tag bisa dikosongin, tetapi jika ada terjadi kesalahan nama, tidak disarankan untuk Klik Kanan → Rename, **Katalon sangat sensitif dan bisa saja kode atau perintah automation yang sudah dibuat akan hilang semua karena direname.** Mungkin saran bisa dihapus dan dibuat ulang (pastikan sudah di backup perintah automationnya, ambil seluruh kode perintahnya di bagian "Script").

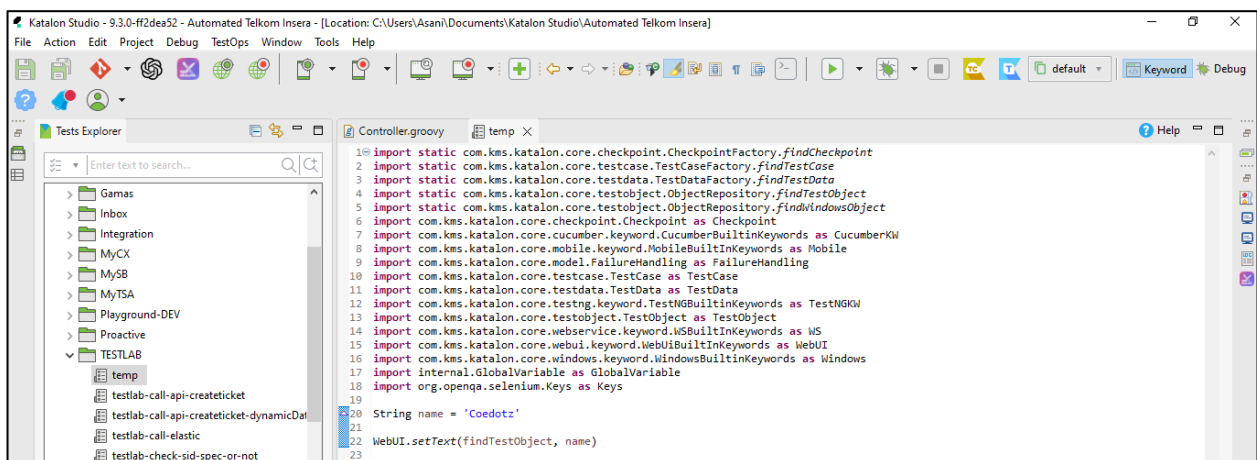


## Membuat Perintah Automation pada Test Case



Gambar 63 - Add via UI (Manual)

Untuk membuat perintah Automation pada Testcase bisa menggunakan Manual atau Script, pada gambar diatas dilakukan Pembuatan perintah menggunakan Manual (UI), dengan cara klik tombol “Add” jika sudah nanti muncul item perintahnya, nah untuk merubah dan menampilkan dropdown seperti gambar diatas bisa dengan cara double klik pada item dan nanti muncul ikon pilih dropdown seperti gambar diatas, gunakan perintah automation sesuai yang diinginkan.



Gambar 64 - Membuat Perintah dengan Script

Pada gambar diatas adalah contoh pembuatan Perintah automation menggunakan Script (ngoding), sebenarnya lebih mudah jika menggunakan UI (Manual), tetapi kemungkinan ada beberapa kasus yang mungkin harus di custom perintah Automationnya.

## Failure Handling pada Test Case (Test Script)

---

Gunakan FailureHandling jika tidak pasti apakah bisa berjalan atau tidak karena, FailureHandling mencegah jika terjadi kesalahan karena lokasi objek tidak ditemukan, contoh penggunaan :

```
WebUI.setText(findTestObject, 'Coedotz', FailureHandling.CONTINUE_ON_FAILURE)
```

Pada penggunaan FailureHandling akan mencegah terjadinya permasalahan jika objeknya tidak muncul atau tidak terlihat, Berikut penjelasan pemanggilan FailureHandling.

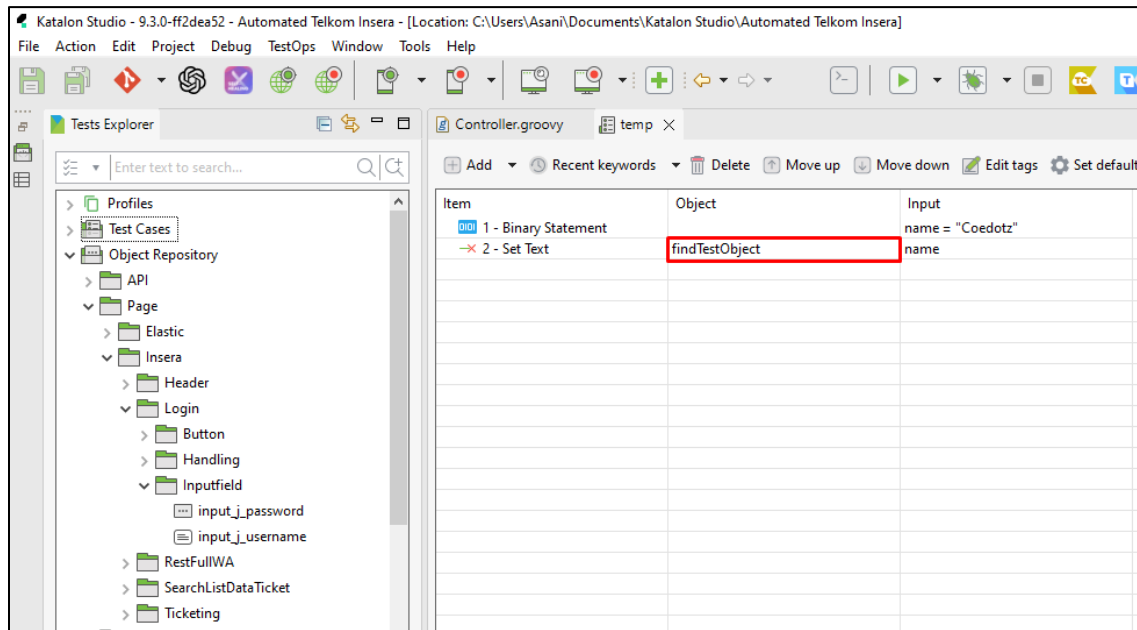
<b>FailureHandling.CONTINUE_ON_FAILURE</b>	Dilakukan Ketika objeknya tidak muncul / terlihat dia akan terus menjalankan automationnya.
<b>FailureHandling.OPTIONAL</b>	Kondisi ini sama seperti CONTINUE_ON_FAILURE
<b>FailureHandling.STOP_ON_FAILURE</b>	Dilakukan Ketika objeknya tidak muncul / terlihat dia akan stop / eror automationnya.

Pastikan tidak ada teks underline di keyword perintah automation, seperti contoh dibawah ini.

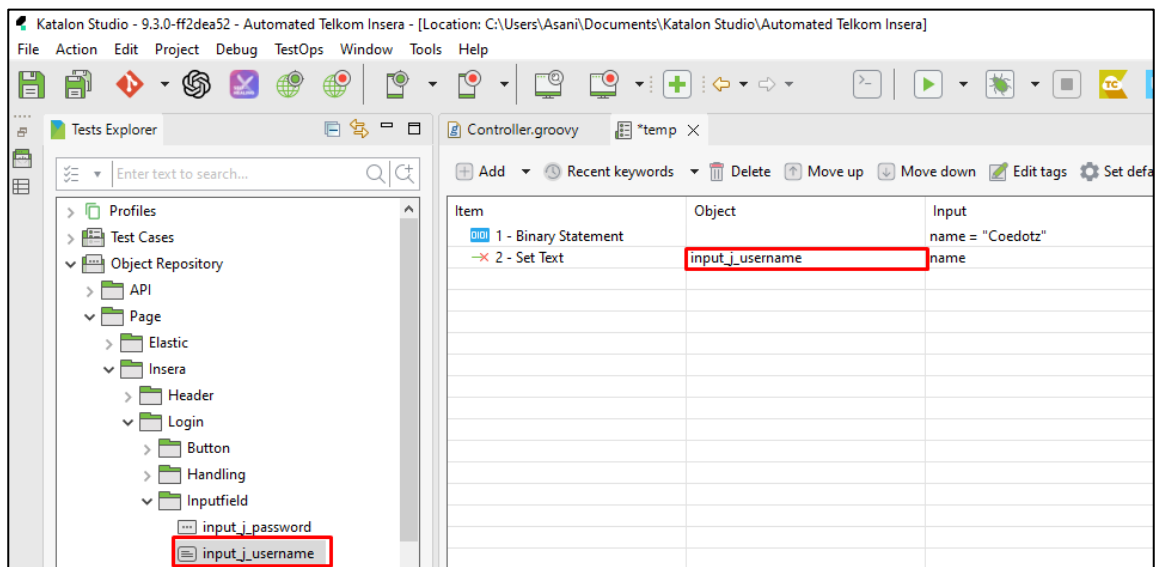
```
WebUI.setText(findTestObject, 'Coedotz', FailureHandling.CONTINUE_ON_FAILURE)
```

Jika di keyword setText atau pas pemanggilan fungsi atau method terdapat teks underline, mungkin bisa dicek lagi, pastikan parameter isianya sudah benar atau cara pemanggilannya benar atau keywordnya terdaftar / ada.

## Drag & Drop di Test Case



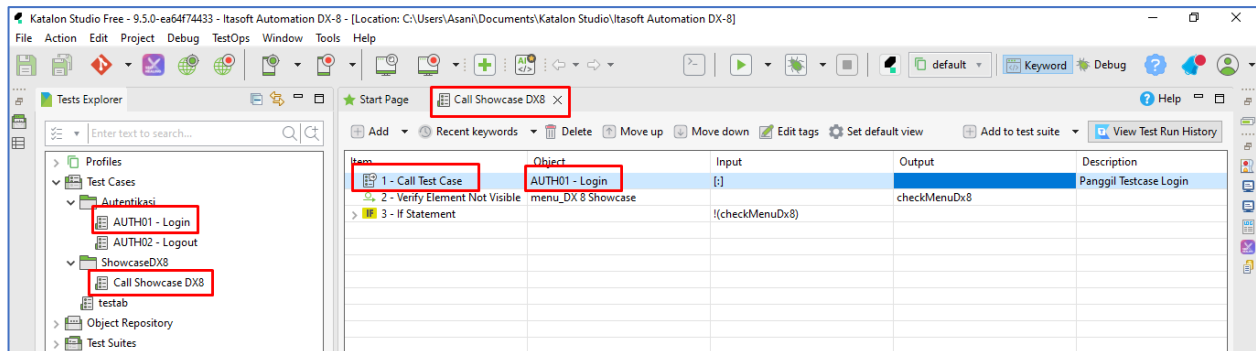
Gambar 65 - Drag&Drop 1



Gambar 66 - Drag&Drop 2

Pada gambar diatas dilakukan drag & drop Test Objek dari folder Object Repository ke Test Case seperti gambar diatas.

## Call Test Case



Gambar 67

Call Test Case ini berguna untuk menggunakan kembali test case yang sudah ada, jadi kita tidak perlu membuat test script yang sudah dibuat pada test case sebelumnya atau test case lainnya, jadi kita perlu panggil test case yang ada

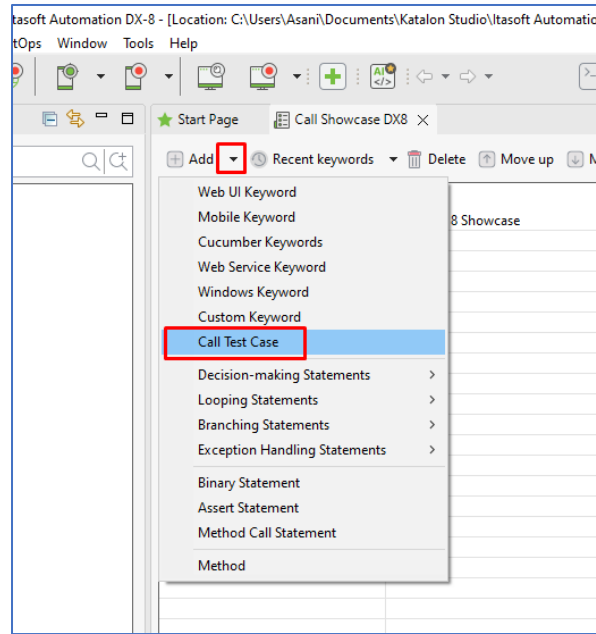
### Contohnya :

Kita mempunyai 2 Test Case yaitu :

- Test Case Login
- Test Case Tambah Data

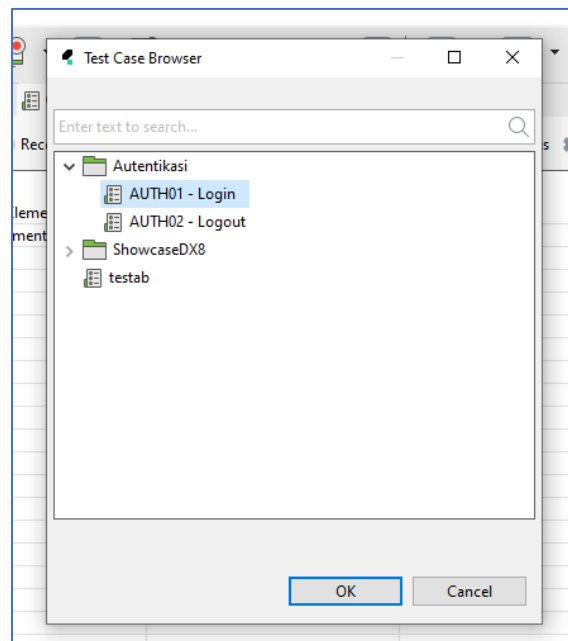
Kita sudah membuat test case login dengan test script yang berupa memasukan username dan password lalu klik tombol submit, dan pada saat di test case tambah data, kita perlu melakukan login juga, karena jika ingin masuk ke halaman tambah data harus login, maka kita perlu membuat test script yang sama seperti test case login, jadi karena test script tersebut sudah ada, maka dari pada kita membuat kembali kode yang sama, lebih baik kita melakukan penggunaan kembali test case login, dengan cara menggunakan “Call Test Case” Login kedalam Test Case Tambah Data.

Pada gambar dibawah, kita akan mencoba untuk melakukan “Call Test Case” dengan contoh kasus Panggil Test case Login.



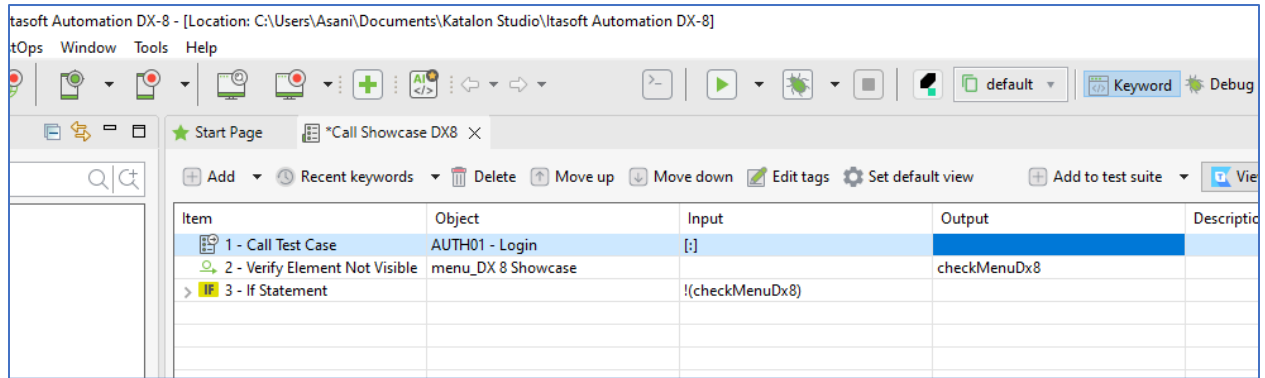
Gambar 68

Pada gambar diatas, kita perlu membuka test case terlebih dahulu, untuk test case bebas sesuai yang diinginkan, jika sudah klik tombol sebelah “Add”, lalu pilih “Call Test Case”.



Gambar 69

Jika sudah, akan ada dialog untuk memilih test case yang akan di panggil (pada gambar diatas), pada contoh kasus ini kita akan panggil test case login, jika sudah klik tombol OK.



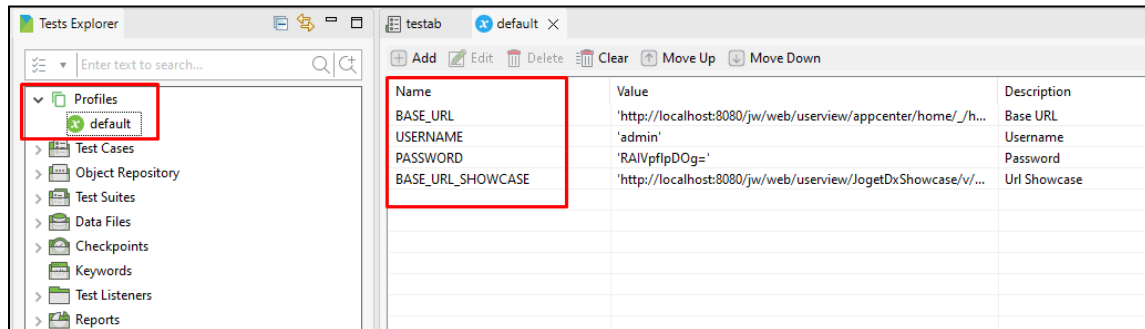
Gambar 70

Pada gambar diatas, test case login sudah berhasil di panggil ke test case yang diinginkan, lalu ketika test case ini dijalankan, maka yang pertama kali di eksekusi adalah test case login, lalu jika test case login sudah selesai, maka dilanjutkan ke proses test case yang kita inginkan (tergantung dari susunan item pada test case).

## Variable

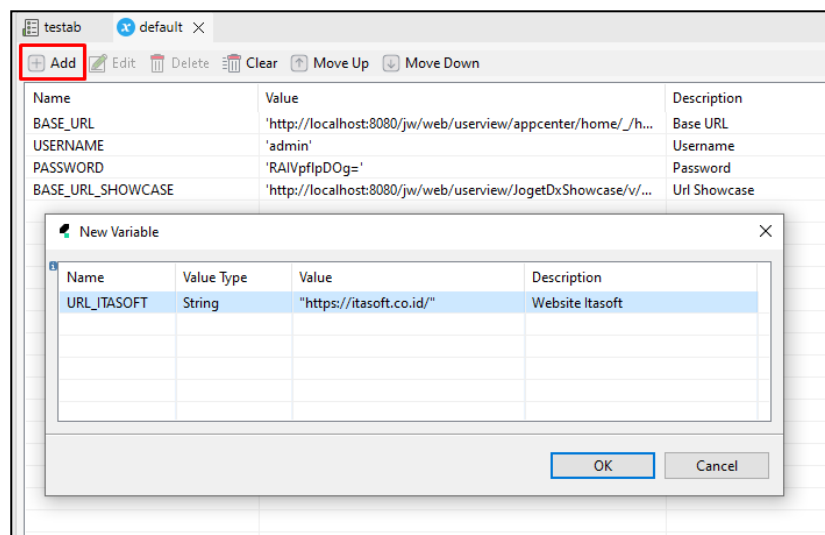
Pada Katalon Studio terdapat beberapa Variable, yaitu **Global Variabel**, **Local Variabel** & **Local Variable Data Binding**, mereka mempunyai masing-masing karakter yang sedikit berbeda tapi masih sama, yaitu untuk menampung sebuah data, berikut penjelasan masing-masing dari variable di katalon studio.

- **Global Variable**



Gambar 71

Global variable ini adalah variable yang bentuknya global dan bisa diakses pada test case atau di kelas lainnya, mengetahui Lokasi global variable ini, kita bisa lihat pada bagian “Profiles” dan disitu ada default, kita juga bisa membuat profiles lainnya untuk membedakan data di global variable, dan pada gambar diatas adalah merupakan halaman Global Variable, untuk penamaan Global Variable ini diharuskan menggunakan kapital semua (berdasarkan best practice), untuk melakukan pembuatan Global Variabel bisa dilihat pada gambar dibawah.



Gambar 72

testab		
*default		
Add Edit Delete Clear Move Up Move Down		
Name	Value	Description
BASE_URL	'http://localhost:8080/jw/web/userview/appcenter/home/_/h...	Base URL
USERNAME	'admin'	Username
PASSWORD	'RAIVpflpDOg='	Password
BASE_URL_SHOWCASE	'http://localhost:8080/jw/web/userview/logetDeShowcase/v/...	Url Showcase
URL_ITASOFT	'https://itasoft.co.id/'	Website Itasoft

Gambar 73

Pada gambar diatas, kita akan melakukan sebuah value global variable baru, awalnya kita akan klik tombol Add, lalu masukan value global variable yang diinginkan, masukan name, value type, value dan deskripsi, jika sudah klik OK, dan variable yang kita buat sudah ada di global variable, lalu jika sudah, kita akan menerapkan global variable ini pada test case seperti gambar dibawah.

```

1 import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint
19
20 'Open Browser'
21 WebUI.openBrowser('')
22
23 'Maximize Window'
24 WebUI.maximizeWindow()
25
26 'Navigate URL'
27 WebUI.navigateToUrl('itasoft.co.id') // Hardcoded
28 WebUI.navigateToUrl(GlobalVariable.URL_ITASOFT) // Global Variable
29
30 'Close Browser'
31 WebUI.closeBrowser()
  
```

Gambar 74

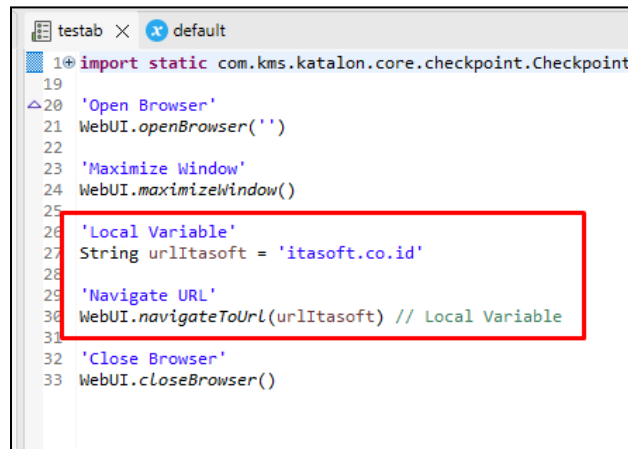
Pada gambar diatas adalah penerapan global variable pada testcase, untuk penerapan pada file kelas juga bisa, kita hanya perlu melakukan pemanggilan seperti dibawah ini.

```
GlobalVariable.(NAMA VARIABEL)
```



- **Local Variable**

Di Katalon, Local Variabel adalah variable yang hanya di akses pada lingkup tersebut, dan tidak bisa digunakan oleh kelas lain atau test case lain, dan hanya bisa digunakan oleh kelas atau test case yang membuatnya.



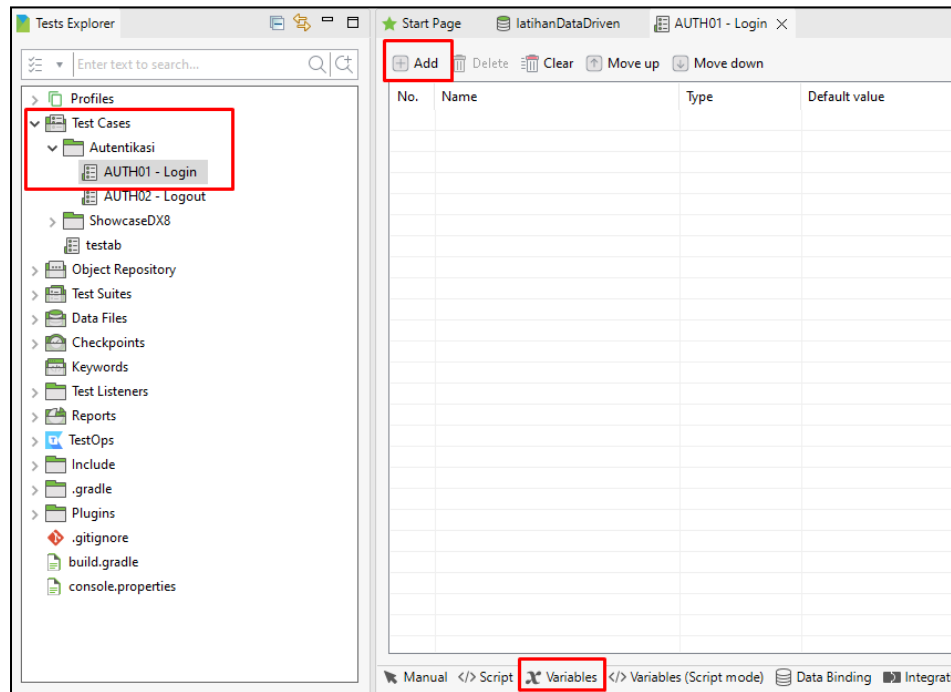
```
testab x default
19 import static com.kms.katalon.core.checkpoint.Checkpoint
20 'Open Browser'
21 WebUI.openBrowser('')
22
23 'Maximize Window'
24 WebUI.maximizeWindow()
25
26 'Local Variable'
27 String urlItasoft = 'itasoft.co.id'
28
29 'Navigate URL'
30 WebUI.navigateToUrl(urlItasoft) // Local Variable
31
32 'Close Browser'
33 WebUI.closeBrowser()
```

Gambar 75

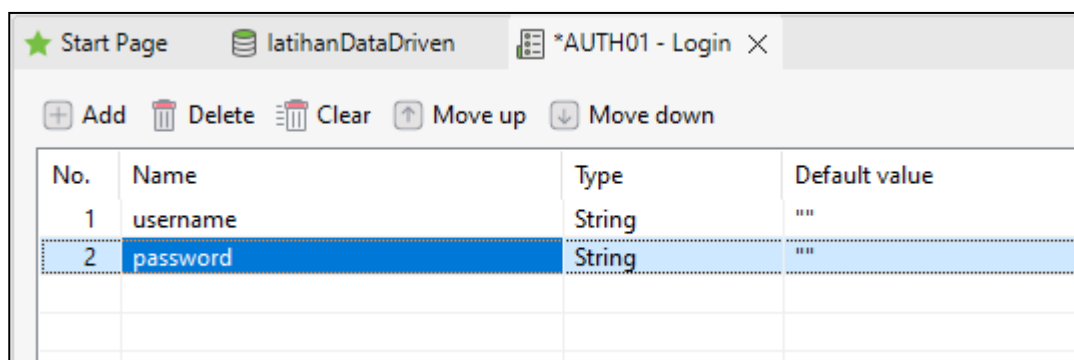
Pada gambar diatas adalah penerapan dari Local Variabel pada test case, disini kita membuat sebuah local variable berbentuk String dan diisi dengan 'itasoft.co.id' lalu selanjutnya, kita akan memanggil local variable tersebut kedalam navigateUrl, bisa dilihat pada gambar diatas contoh penggunaan local variable.

- **Local Variable Binding**

Local Variable Binding, ini merupakan variable yang digunakan untuk melakukan pengujian Data Test (ada di materi Test Data Driven untuk lengkapnya), berikut gambar dibawah cara pembuatan local variable binding.



Gambar 76



Gambar 77

Pada gambar diatas, kita akan melakukan pembuatan local variable binding didalam sebuah test case, disini, kita akan menggunakan test case login dalam pembuatan local variable binding. Ketika sudah masuk kedalam halaman test case, pilih tab variables (dibawah), lalu klik tombol Add, buat 2 variabel yaitu Username & Password.

```

18 import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint
19
20 'Buka Browser'
21 WebUI.openBrowser('')
22
23 'Tampilkan penuh browser'
24 WebUI.maximizeWindow()
25
26 'Arahkan ke URL yang diinginkan'
27 WebUI.navigateToUrl(GlobalVariable.BASE_URL)
28
29 // cek tombol login apakah terlihat atau tidak
30 boolean checkBtnLogin = WebUI.verifyElementNotVisible(findTestObject('Page/Autentikasi/Button/btn_login'))
31
32 // Pengecekan tombol login terlihat atau tidak
33 if (!checkBtnLogin) {
34   'Klik login'
35   WebUI.click(findTestObject('Page/Autentikasi/Button/btn_login'))
36
37   'Masukan Username'
38   WebUI.setText(findTestObject('Page/Autentikasi/Inputfield/input_username'), username)
39
40   'Masukan Password'
41   WebUI.setEncryptedText(findTestObject('Page/Autentikasi/Inputfield/input_password'), password)
42
43   'Klik tombol submit'
44   WebUI.click(findTestObject('Page/Autentikasi/Button/btn_submit'))
45 }
  
```

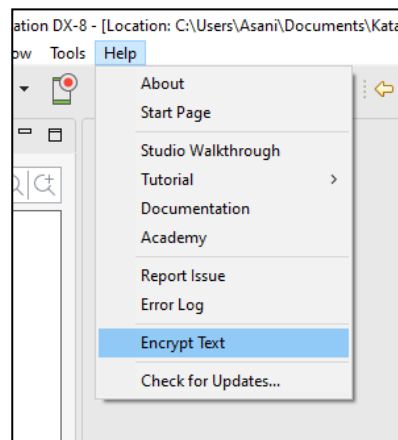
Manual **</> Script** Variables </> Variables (Script mode) Data Binding Integration Properties

Gambar 78

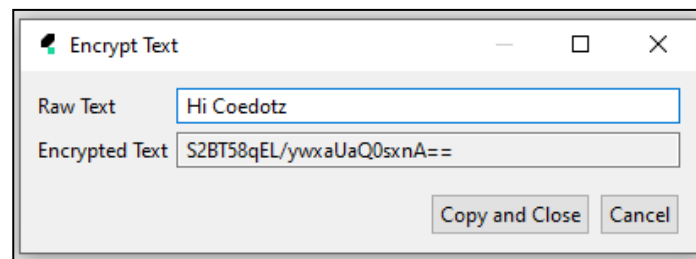
Pada gambar diatas, lakukan pemanggilan local variable binding kedalam test script seperti gambar diatas, jangan khawatir ketika variable nya terdapat tulisan bergaris bawah (underline), variable ini bisa berfungsi walaupun terdapat tulisan bergaris bawah (underline), ini hanya berlaku jika menggunakan variable local binding.

## Encrypt Text

Encrypt Text merupakan merupakan proses mengamankan teks agar tidak dibaca oleh orang lain atau pihak lain, biasanya Encrypt Text ini digunakan pada saat input password di test case atau object yang ada di file kelas, untuk menggunakan di katalon, kita perlu menggunakan encrypt text bawaan dari katalon, karena jika kita menggunakan encrypt text dari website luar selain didalam aplikasi katalon, kemungkinan akan mengalami error pada saat dijalankan oleh automation dikatalon, berikut cara merubah teks menjadi encrypt text pada gambar dibawah.



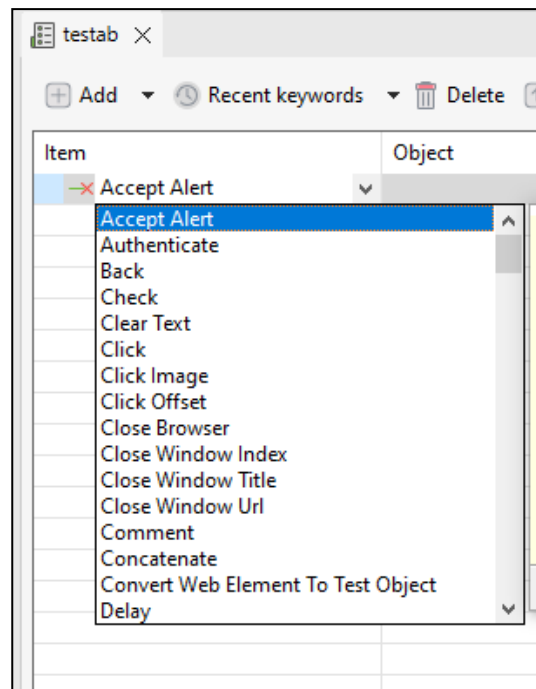
Gambar 79



Gambar 80

Pada gambar diatas, kita perlu ke menu help lalu pilih Encrypt Text, setelah itu akan muncul dialog Encrypt Text, masukan text yang diinginkan kedalam Raw Text, lalu jika sudah klik tombol Copy and Close, dan pastekan encrypt text tersebut ke object atau variable yang kita inginkan. Pastikan jika ingin menggunakan Encrypt Text, Keyword atau Object pada test case harus berupa tipe "Set Encrypt Text", dan untuk penggunaan pada global variable hanya perlu menggunakan tipe String, lalu ketika di implementasi ke test case, kita perlu menggunakan keyword atau object berupa "Set Encrypt Text".

## Keywords



Gambar 81

Keyword ini merupakan perintah yang dilakukan pada Test Case di automation katalon studio, keyword ini berfungsi untuk mengetahui tujuan yang akan dilakukan, semisal jika ingin melakukan klik tombol maka diperlukan keyword click, ataupun memasukan teks menggunakan keyword set text, disini akan memberikan contoh penggunaan keyword yang sering digunakan dan menggunakan keyword WebUI (untuk pengujian sebuah web), berikut contoh penggunaan keyword berserta failure handling.

- **Accept Alert**

Accept Alert biasa digunakan untuk melakukan Tindakan klik tombol OK pada dialog prompt, atau dialog lainnya.

**Penggunaan :**

WebUI.acceptAlert()

**Dengan Failure Handling :**

WebUI.acceptAlert(FailureHandling.OPTIONAL)

- **Click**

Click, digunakan untuk melakukan Tindakan click, entah klik button atau lainnya.

**Penggunaan :**

```
WebUI.click(findTestObject(testobject))
```

**Dengan Failure Handling :**

```
WebUI.click(findTestObject(testobject), FailureHandling.OPTIONAL)
```

- **Delay**

Delay, digunakan untuk memberikan jeda, semisal kita ingin memberikan jeda 2 detik sebelum klik tombol tambah atau lainnya, penggunaan delay ini tidak boleh terus menerus dan berulang-ulang, karena dapat membuat automation kita tidak efisien dan tidak sesuai dengan best practice, gunakan “Wait” agar lebih baik daripada menggunakan delay.

**Penggunaan :**

```
WebUI.delay(3) // delay 3 detik
```

**Dengan Failure Handling :**

```
WebUI.delay(3, FailureHandling.OPTIONAL)
```

- **Dismiss Alert**

Dismiss Alert, digunakan untuk melakukan Tindakan cancel pada prompt dialog, atau dialog lainnya.

**Penggunaan :**

```
WebUI.dismissAlert()
```

**Dengan Failure Handling :**

```
WebUI.dismissAlert(FailureHandling.OPTIONAL)
```

- **Double Click**

Double Click, digunakan untuk melakukan klik sebanyak 2x.

**Penggunaan :**

```
WebUI.doubleClick(findTestObject(testobject))
```

**Dengan Failure Handling :**

```
WebUI.doubleClick(findTestObject(testobject), FailureHandling.OPTIONAL)
```

- **Focus**

Focus, digunakan untuk mengarahkan kursor atau menargetkan ke bagian yang telah difokuskan, semisal ada sebuah halaman tambah data, dan tombol tambah datanya ada dipaling bawah, ketika kita menggunakan focus, maka nanti akan langsung mengarahkan kursor ke bagian tambah data yang ada dipaling bawah.

**Penggunaan :**

```
WebUI.focus(findTestObject(testobject))
```

**Dengan Failure Handling :**

```
WebUI.focus(findTestObject(testobject), FailureHandling.OPTIONAL)
```

- **Get Text**

Get Text, digunakan untuk mengambil teks yang ada pada komponen label.

**Penggunaan :**

```
WebUI.getText(findTestObject(testobject))
```

**Dengan Failure Handling :**

```
WebUI.getText(findTestObject(testobject), FailureHandling.OPTIONAL)
```

- **Get Attribute**

Get Attribute, digunakan untuk mengambil sebuah nilai yang biasanya terdapat pada komponen input field, text area atau lainnya, selain dikomponen label.

**Penggunaan :**

```
WebUI.getAttribute(findTestObject(testobject))
```

**Dengan Failure Handling :**

```
WebUI.getAttribute(findTestObject(testobject), FailureHandling.OPTIONAL)
```

- **Navigate to Url**

Navigate to Url, digunakan untuk memanggil halaman situs yang kita inginkan, semisal kita awalnya ada di situs google, dan kita lakukan navigate to url dengan parameter situs itasoft, maka nanti akan berubah ke halaman situs itasoft.

**Penggunaan :**

```
WebUI.navigateToUrl(findTestObject(testobject))
```

**Dengan Failure Handling :**

```
WebUI.navigateToUrl(findTestObject(testobject), FailureHandling.OPTIONAL)
```

- **Open Browser**

Open Browser, digunakan untuk melakukan pemanggilan browser / web driver.

**Penggunaan :**

```
WebUI.openBrowser("") // kosongkan saja isi dari petiknya
```

**Dengan Failure Handling :**

```
WebUI.openBrowser("", FailureHandling.OPTIONAL)
```



- **Close Browser**

Close Browser, digunakan untuk menutup browser / web driver.

**Penggunaan :**

```
WebUI.closeBrowser()
```

**Dengan Failure Handling :**

```
WebUI.closeBrowser(FailureHandling.OPTIONAL)
```

- **Maximize Window**

Maximize Window, digunakan untuk membuat aplikasi browser / web driver menjadi maximize.

**Penggunaan :**

```
WebUI.maximizeWindow()
```

**Dengan Failure Handling :**

```
WebUI.maximizeWindow(FailureHandling.OPTIONAL)
```

- **Refresh**

Refresh, digunakan untuk melakukan refresh halaman pada browser / web driver.

**Penggunaan :**

```
WebUI.refresh()
```

**Dengan Failure Handling :**

```
WebUI.refresh(FailureHandling.OPTIONAL)
```

- **Select Option by Index**

Select Option by Index, digunakan untuk membantu melakukan pemilihan pada komponen dropdown dan pemilihan datanya berupa Index dari data dropdown.

**Penggunaan :**

```
WebUI.selectOptionByIndex(findTestObject(testobject), index)
```

**Dengan Failure Handling :**

```
WebUI.selectOptionByIndex(findTestObject(testobject), index, FailureHandling.OPTIONAL)
```

- **Select Option by Value**

Select Option by Value, digunakan untuk membantu melakukan pemilihan pada komponen dropdown dan pemilihan datanya berupa value dari data dropdown.

**Penggunaan :**

```
WebUI.selectOptionByValue(findTestObject(testobject), value)
```

**Dengan Failure Handling :**

```
WebUI.selectOptionByValue(findTestObject(testobject), value, FailureHandling.OPTIONAL)
```

- **Select Option by Label**

Select Option by Label, digunakan untuk membantu melakukan pemilihan pada komponen dropdown dan pemilihan datanya berupa Label dari data dropdown.

**Penggunaan :**

```
WebUI.selectOptionByLabel(findTestObject(testobject), stringLabel)
```

**Dengan Failure Handling :**

```
WebUI.selectOptionByLabel(findTestObject(testobject), stringLabel, FailureHandling.OPTIONAL)
```

- **Set Text**

Set Text, digunakan untuk melakukan pengisian teks, biasanya digunakan ketika ingin mengisi teks pada inputfield, text area atau lainnya.

**Penggunaan :**

```
WebUI.setText(findTestObject(testobject))
```

**Dengan Failure Handling :**

```
WebUI.setText(findTestObject(testobject), FailureHandling.OPTIONAL)
```

- **Set Encrypt Text**

Set Encrypt Text, sama seperti Set Text, tetapi Encrypt text ini digunakan untuk data yang menggunakan encrypt text.

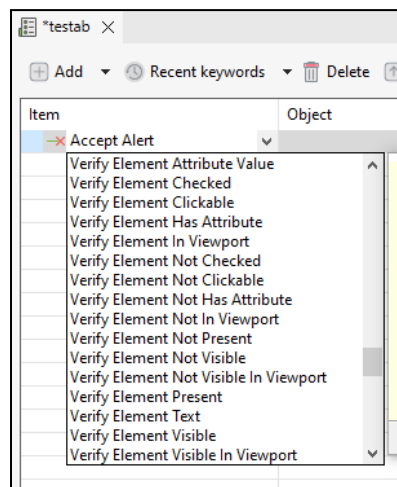
**Penggunaan :**

```
WebUI.setEncryptText(findTestObject(testobject))
```

**Dengan Failure Handling :**

```
WebUI.setEncryptText(findTestObject(testobject), FailureHandling.OPTIONAL)
```

- **Verify**



Gambar 82

Verify, digunakan untuk melakukan validasi dan verifikasi terhadap komponen, apakah dia terlihat atau tidak, muncul atau tidak, memiliki teks atau tidak, bisa diklik atau tidak dan lainnya.

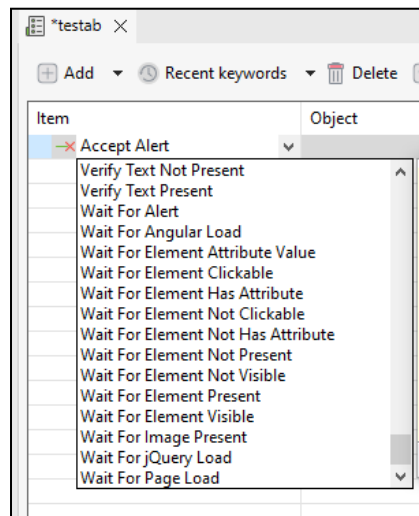
**Penggunaan :**

WebUI.verify.....(findTestObject(testobject)) // sesuaikan dengan masing-masing parameter

**Dengan Failure Handling :**

WebUI.verify...(findTestObject(testobject), FailureHandling.OPTIONAL) // sesuaikan dengan masing-masing parameter

- **Wait**



Gambar 83

Wait, digunakan untuk melakukan proses menunggu memuat komponen secara sempurna, tanpa harus melakukan delay, karena jika melakukan delay tetapi komponennya belum terlihat akan mengakibatkan error atau gagal pada saat menjalankan automation.

**Penggunaan :**

WebUI.wait.....(findTestObject(testobject)) // sesuaikan dengan masing-masing parameter

**Dengan Failure Handling :**

WebUI.wait...(findTestObject(testobject), FailureHandling.OPTIONAL) // sesuaikan dengan masing-masing parameter