

Índex

1. Proves Funcionals	3
2. Requisits	3
2.1. Instal·lar Docker	3
2.2. Instal·lar JDK	4
2.3. Instal·lar Maven	4
2.4. Instal·lar IDE Visual Studio Code	4
2.5. Accés a la intranet de git i/o github enterprise	4
2.6. Crear variables d'entorn en l'equip local	4
3. Descarregar el projecte.....	10
3.1 – Descarregar la plantilla des de Gitlab	11
3.2 – Descarregar la plantilla des de Github	14
4. Aixecar la arquitectura de serveis definida en el YAML anterior.	14
5. Explicació de la plantilla/llibreria	17
5.1. Arxius pertanyents de la llibreria	18
5.1.1. ConfigParameters	18
5.1.2. BaseTest.java.....	21
5.1.3. Utils.java.....	21
5.1.4. BrowserOptions.java	22
5.1.5. ExtentManager.java	22
5.1.6. ResultSender.java i ExecutionListener.java.....	22
5.2. Compilar i executar la plantilla.....	22
6. Crear casos de proves	25
6.1. Definir Page Objet Models	26
6.2. Definir Test.....	34
6.3. Definir test.xml	38
7. Executar les proves des de Visual Studio Code	39

8.	Anàlisi de resultats en el report HTML.	41
8.1.	Anàlisi de l'error	44
8.1.1.	Solucionar l'error	46
9.	Integració amb Xray	54
9.1	Abans de començar	54
9.2	Crear una Prova	55
10.	Integració amb GitHub Enterprise	57
10.1	Anar a la secció d'Actions a GitHub Enterprise	57

1. Proves Funcionals

Les proves funcionals d'aplicacions a través de Selenium són un enfocament crucial en el desenvolupament de programari, on Selenium, una eina d'automatització de proves, s'utilitza per verificar el correcte funcionament d'aplicacions web. Aquestes proves es centren en validar la funcionalitat de l'aplicació des de la perspectiva de l'usuari, simulant interaccions reals de l'usuari, com fer clic en botons, omplir formularis i navegar per diferents pàgines.

Selenium permet crear scripts que emulen accions humanes, la qual cosa garanteix que l'aplicació compleixi amb els requisits funcionals i que es mantingui estable davant possibles canvis o actualitzacions.

En el següent manual, podeu executar proves funcionals de Selenium, a través de Selenium Grid, una eina que permet orquestrar i escalar els navegadors.

2. Requisites

Nota

En el projecte base que se us proporciona, ja existeix un **DevContainer** configurat. Per tant, si teniu la capacitat de fer-lo servir, els punts **2.2** i **2.3** són **opcionals**.

Podeu trobar la documentació oficial sobre com executar un **DevContainer** sobre **VS Code** aquí:

 <https://code.visualstudio.com/docs/devcontainers/containers>

2.1. Instal·lar Docker

Assegureu-vos que teniu Docker y Docker Compose instal·lat al vostre sistema. Podeu descarregar-lo des del lloc web oficial de Docker o utilitzar un gestor de paquets si està disponible per al vostre sistema operatiu.

Link: [Get Docker](#)

Link: [Get Docker Compose](#)

2.2. Instal·lar JDK

1. Descàrrega i instal·lació de [Java Development Kit \(JDK\)](#).
2. Establir la variable d'entorn *JAVA_HOME*: *C:\Program Files\Java\jdk1.8.0_301*
3. Actualitzar la variable d'entorn *PATH*: *%JAVA_HOME%\bin*

2.3. Instal·lar Maven

1. Descàrrega i instal·lació de [Maven](#).
2. Establir la variable d'entorn *MAVEN_HOME*: *C:\Program Files\Maven\apache-maven-3.8.8\bin*
3. Actualitzar la variable d'entorn *PATH*: *%MAVEN_HOME%\bin*

2.4. Instal·lar IDE Visual Studio Code

Per a la creació d'aquestes proves és necessari tenir instal·lat una eina d'entorn de desenvolupament integrat.

En aquest exemple, s'utilitzarà el IDE [Visual Studio Code](#).

2.5. Accés a la intranet de git i/o github enterprise

Comprovar que es té accés a git.intranet.gencat.cat/devsecopsctti/3632-mat-functional-tests o <https://github.com/ctti-arq/functional-test-template.git> per a poder descarregar la plantilla del projecte.

2.6. Crear variables d'entorn en l'equip local

S'hauran de crear les següents variables d'entorn en l'equip local:

- GITHUB_USER (El teu usuari de Github)
- GITHUB_PASSWORD (token)

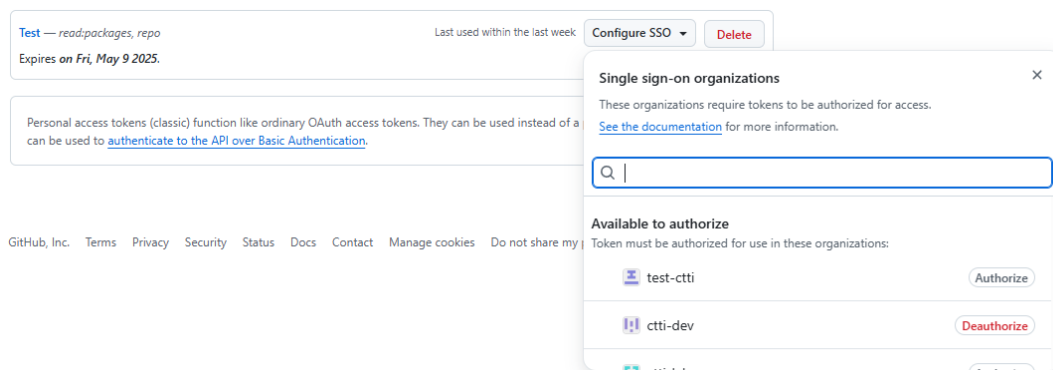
Per generar el token, amb la sessió iniciada a Github Enterprise, heu de navegar a <https://github.com/settings/tokens/new> amb el permisos repo i read:packages

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

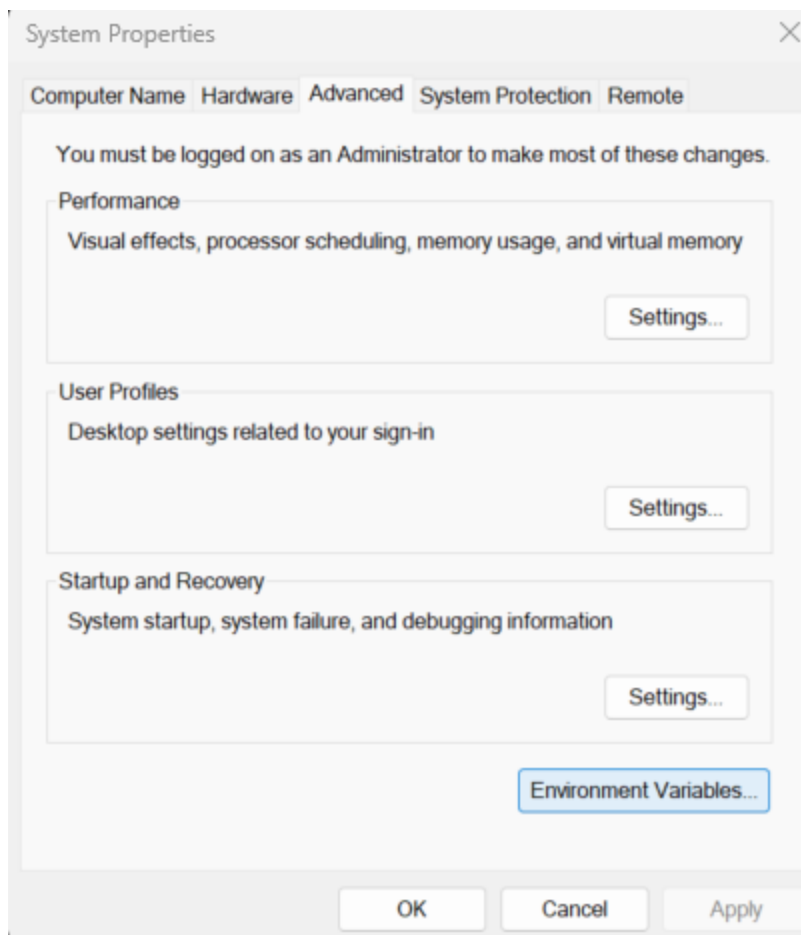
☒ repo	Full control of private repositories
☒ repo:status	Access commit status
☒ repo_deployment	Access deployment status
☒ public_repo	Access public repositories
☒ repo:invite	Access repository invitations
☒ security_events	Read and write security events
☐ workflow	Update GitHub Action workflows
☐ write:packages	Upload packages to GitHub Package Registry
☒ read:packages	Download packages from GitHub Package Registry
☐ delete:packages	Delete packages from GitHub Package Registry
☐ admin:org	Full control of orgs and teams, read and write org projects

Quan el creeu, haureu d'autoritzar a l'organització CTTI-DEV

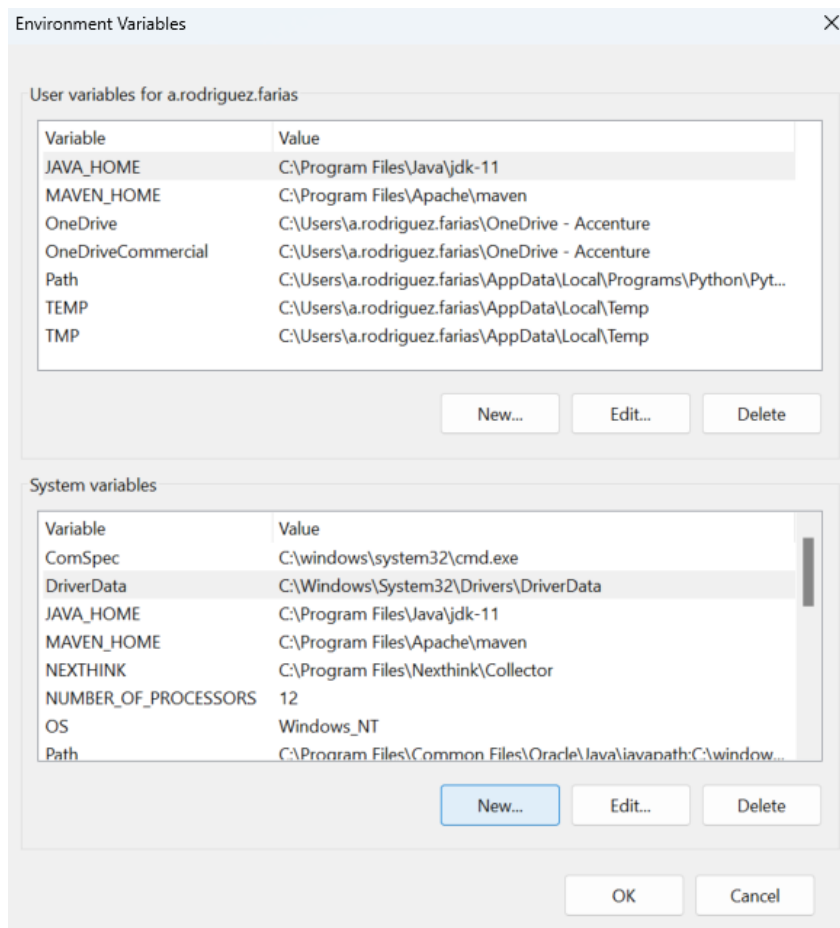


En el cas d'equips amb sistema operatiu Windows, de la següent manera:

1. Obrir el menú de variables d'entorn del sistema i fer clic a Variables d'entorn



2. En variables del sistema, fer clic a Nova



3. Afegim ambdues variables esmentades anteriorment (amb l'usuari i el personal access token de GitHub)



Environment Variables

User variables for a.rodriuez.farias

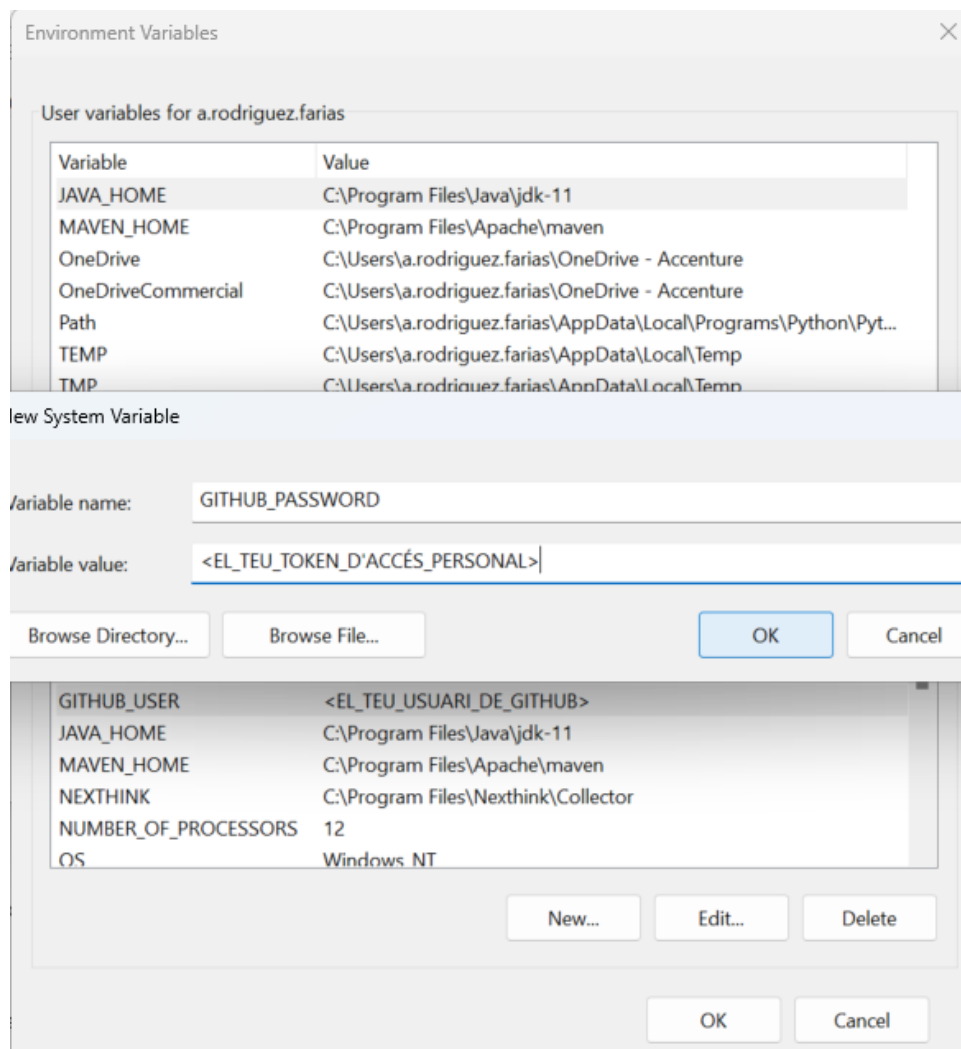
Variable	Value
JAVA_HOME	C:\Program Files\Java\jdk-11
MAVEN_HOME	C:\Program Files\Apache\maven
OneDrive	C:\Users\a.rodriuez.farias\OneDrive - Accenture
OneDriveCommercial	C:\Users\a.rodriuez.farias\OneDrive - Accenture
Path	C:\Users\a.rodriuez.farias\AppData\Local\Programs\Python\Pyt...
TEMP	C:\Users\a.rodriuez.farias\AppData\Local\Temp
TMP	C:\Users\a.rodriuez.farias\AppData\Local\Temp

New System Variable

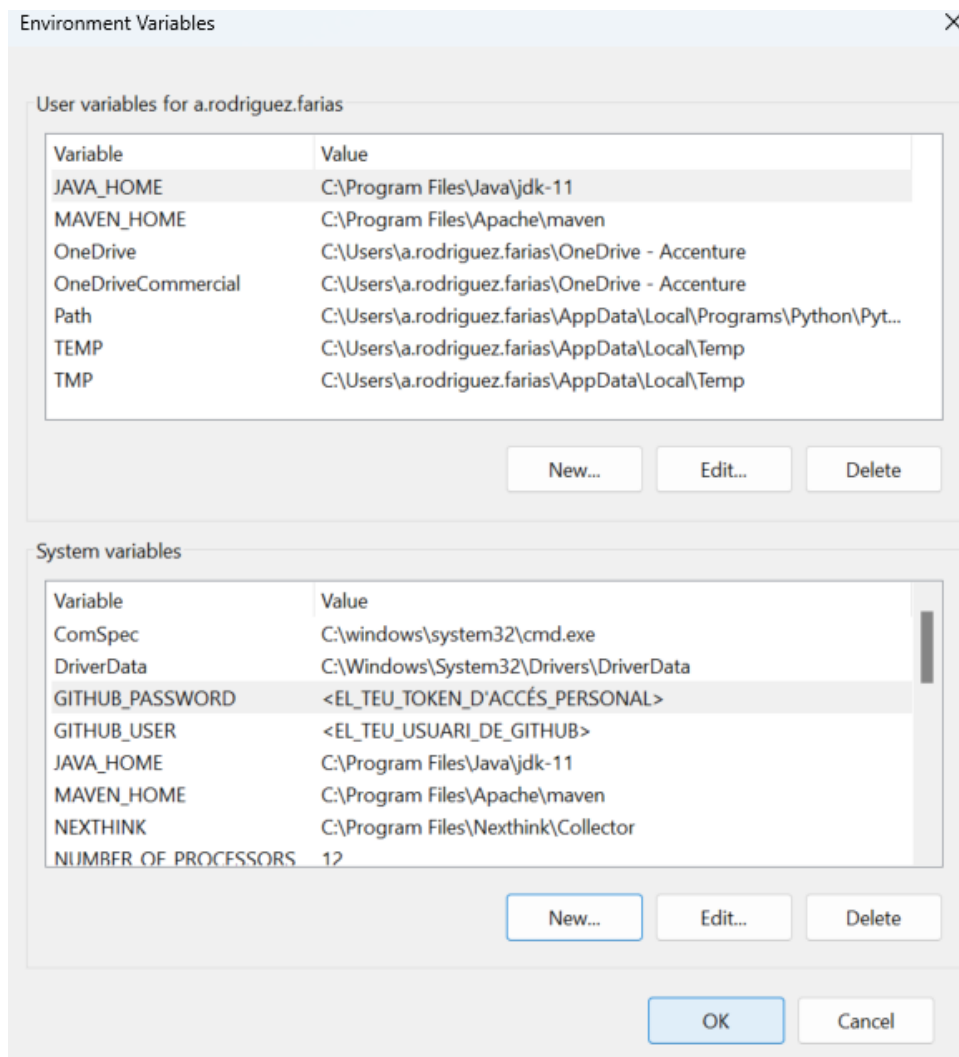
Variable name:

Variable value:

JAVA_HOME	C:\Program Files\Java\jdk-11
MAVEN_HOME	C:\Program Files\Apache\maven
NEXTHINK	C:\Program Files\Nexthink\Collector
NUMBER_OF_PROCESSORS	12
OS	Windows_NT
Path	C:\Program Files\Common Files\Oracle\Java\javapath;C:\window...



4. Premem OK per acceptar els canvis realitzats i finalitzar la incorporació d'aquestes variables.



3. Descarregar el projecte

Accedir al repositori per a descarregar la plantilla del projecte.

Aquest projecte conté els següents arxius:

ELEMENT	DESCRIPCIÓ
config.properties	Aquest arxiu conté els camps de configuració que es necessitin per a l'execució del projecte.
src/test	La carpeta emmagatzemarà tots els scripts, artefactes i recursos que es necessiten per a executar les proves, comprèn una jerarquia de carpeta entre les quals es troben main i test.
docker-compose.yaml	Fitxer d'execució del docker per a aixecar Selenium Grid.
pom.xml	És un arxiu XML del model d'objectes del projecte que conté informació sobre els detalls de configuració que Maven necessita.
README	Aquest arxiu conté informació sobre el projecte i com utilitzar-lo. És una forma de documentació de programari, usualment en un arxiu de text pla en format TXT, o MD.
launchTest.sh	Aquest script permet executar el projecte de forma local transversalment, creant tots els components, executant les proves i destruint els components.

3.1 – Descarregar la plantilla des de Gitlab (preferible Github).

Enable in settings

3

3632-mat-functional-tests

 master ▾

3632-mat-functional-tests /

+ ▾



Update QualityGenCatTest.java

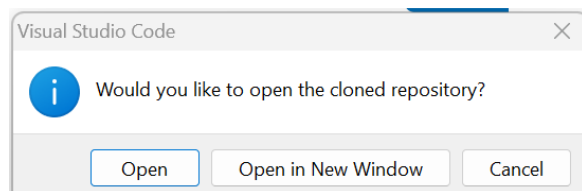
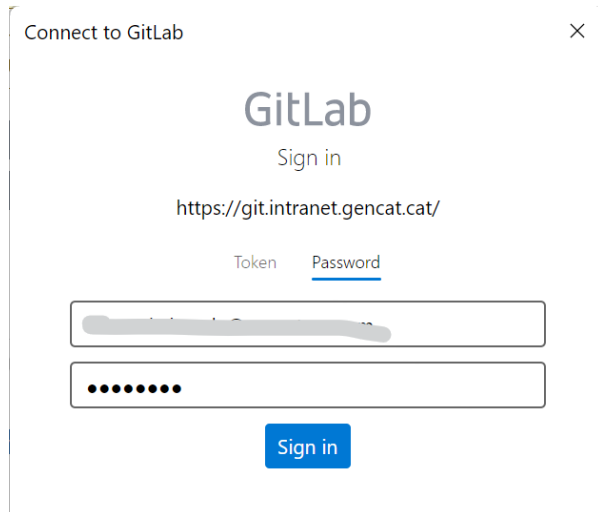
Mora Alonso, Juan Antonio authored 1 month ago

Name	Last commit
 .devcontainer	Agregar configuración inicial del proye...
 .github	Agregar configuración inicial del proye...
 .mvn	Agregar configuración inicial del proye...
 maven	Add new directory
 report	Agregar configuración inicial del proye...
 src/test	Update QualityGenCatTest.java
 .gitignore	Agregar configuración inicial del proye...
 LICENSE	Add LICENSE
 README.md	Initial commit
 config.properties	Agregar configuración inicial del proye...
 docker-compose.yaml	Agregar configuración inicial del proye...
 launchTest.sh	Agregar configuración inicial del proye...
 pom.xml	Agregar configuración inicial del proye...

 README.md

Obrir el projecte en el IDE De Visual Studio Code (HTTPS).

Connectar-se a GitLab amb les mateixes credencials i obre el projecte.



Important!

Per seguretat, Gitlab no permet allotjar fitxers de configuració, on s'es genera la credencial per descarregar el projecte, afegiu a la carpeta maven del projecte un fitxer settings.xml amb el contingut:

```
<settings>
  <servers>
    <server>
      <id>github</id>
      <username>${env.GITHUB_USER}</username>
      <password>${env.GITHUB_PASSWORD}</password>
    </server>
  </servers>
</settings>
```

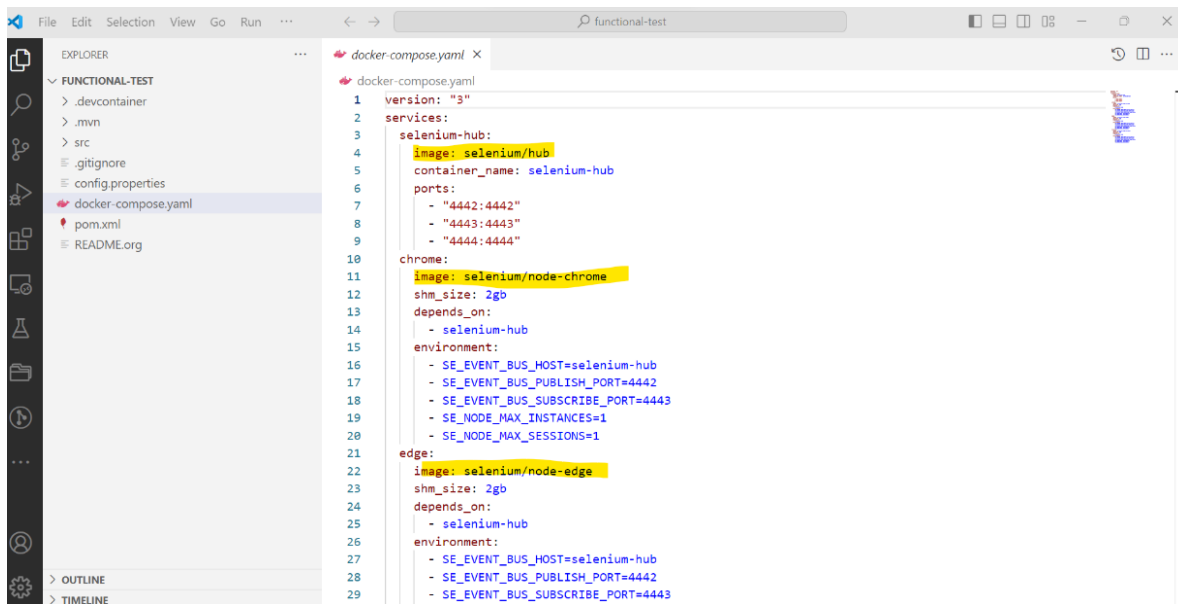
Aquesta casuística només passa amb Maven, a Github Enterprise disposeu del codi complet.

3.2 – Descarregar la plantilla des de Github

Podeu descarregar la plantilla simplement amb `git clone <Nom del repositori>`, us demanar automàticament les credencials.

4. Aixecar la arquitectura de serveis definida en el YAML anterior.

L'arxiu de *docker-compose.yaml*, es compon de la informació necessària per a executar Selenium Grid amb Docker.



```
1 version: "3"
2 services:
3   selenium-hub:
4     image: selenium/hub
5     container_name: selenium-hub
6     ports:
7       - "4442:4442"
8       - "4443:4443"
9       - "4444:4444"
10  chrome:
11    image: selenium/node-chrome
12    shm_size: 2gb
13    depends_on:
14      - selenium-hub
15    environment:
16      - SE_EVENT_BUS_HOST=selenium-hub
17      - SE_EVENT_BUS_PUBLISH_PORT=4442
18      - SE_EVENT_BUS_SUBSCRIBE_PORT=4443
19      - SE_NODE_MAX_INSTANCES=1
20      - SE_NODE_MAX_SESSIONS=1
21  edge:
22    image: selenium/node-edge
23    shm_size: 2gb
24    depends_on:
25      - selenium-hub
26    environment:
27      - SE_EVENT_BUS_HOST=selenium-hub
28      - SE_EVENT_BUS_PUBLISH_PORT=4442
29      - SE_EVENT_BUS_SUBSCRIBE_PORT=4443
```

Hi ha dos paràmetres que a tenir en compte que poden ser modificats pel tester, en funció de la concurrència que necessiti:

- SE_NODE_MAX_INSTANCES=X
- SE_NODE_MAX_SESSIONS=X

A la plantilla, es proporciona un fitxer `launchTest.sh` que executa automàticament tot el contingut d'aquest apartat, creant i destruint els components automàticament. Per això, cal obrir el contingut en un devcontainer o des de Linux, ja que és un fitxer Shell.

Obrir la consola (per exemple, Powershell) per a aixecar la arquitectura definida en el `docker-compose.yaml` i accedir a Selenium Grid.

Accedir al directori on es trobi el projecte i executar la següent instrucció (els valors de cada navegador indiquen el nombre de nodes (contenidors) per cada navegador):

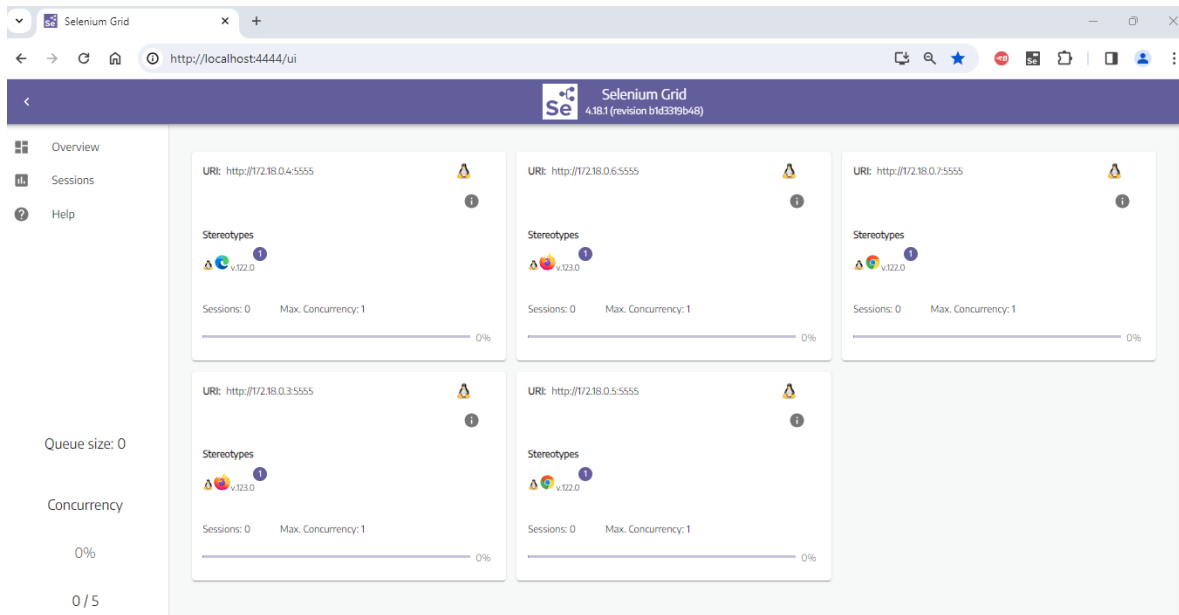
```
docker-compose -f docker-compose.yaml up -d --scale chrome=2 --scale firefox=2 --scale edge=1.
```

```
PS C:\Users\... \Selenium\ctti\functional-test> docker-compose -f docker-compose.yaml up -d --scale chrome=2
--scale firefox=2 --scale edge=1
[+] Running 0/7
- Network functional-test_default      Created                               11.1s
- Container selenium-hub               Starting                            10.4s
- Container functional-test-chrome-2   Created                               9.2s
- Container functional-test-firefox-2  Created                               9.2s
- Container functional-test-chrome-1   Created                               9.2s
- Container functional-test-edge-1     Created                               9.2s
- Container functional-test-firefox-1  Created                               9.2s
```

Esperar que tots els contenidors estiguin aixecats.

```
PS C:\Users\... \Selenium\ctti\functional-test> docker-compose -f docker-compose.yaml up -d --scale chrome=2
--scale firefox=2 --scale edge=1
[+] Running 6/7
- Network functional-test_default      Created                               20.8s
✓ Container selenium-hub               Started                               13.1s
✓ Container functional-test-chrome-2   Started                               15.1s
✓ Container functional-test-firefox-2  Started                               16.9s
✓ Container functional-test-chrome-1   Started                               18.9s
✓ Container functional-test-edge-1     Started                               14.7s
✓ Container functional-test-firefox-1  Started                               14.0s
PS C:\Users\... \Selenium\ctti\functional-test>
```

Accedir a la WebUI de Selenium Grid: <http://localhost:4444/ui>



També es pot veure per consola els contenidors aixecats:

docker container ls

```

PS C:\Users\... Selenium\ctti\functional-test> docker container ls
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
79d199128dbf   selenium/node-firefox               "/opt/bin/entry_poin... 24 minutes ago Up 23 minutes 5900/tcp
acc8f8e74225   selenium/node-edge                 "/opt/bin/entry_poin... 24 minutes ago Up 23 minutes 5900/tcp
3208e496295f   selenium/node-chrome               "/opt/bin/entry_poin... 24 minutes ago Up 23 minutes 5900/tcp
2337d4857a16   selenium/node-chrome               "/opt/bin/entry_poin... 24 minutes ago Up 23 minutes 5900/tcp
2acaecbe307f   selenium/node-firefox               "/opt/bin/entry_poin... 24 minutes ago Up 23 minutes 5900/tcp
7ffc03d5294d   selenium/hub                       "/opt/bin/entry_poin... 24 minutes ago Up 23 minutes 0.0.0.0:4442->4442
PS C:\Users\... Selenium\ctti\functional-test>
  
```

Una vegada que tots els contenidors estiguin aixecats i que tots els nodes estiguin disponibles en la url de Selenium Grid, ja es pot començar a executar les proves.

Recordatori:

Una vegada que hagueu acabat d'executar les proves, és recomanable **eliminar els components prèviament aixecats** per a no consumir recursos.

```
docker-compose -f docker-compose.yaml down
```

```

PS C:\Users\... \Selenium\ctti\functional-test> docker-compose -f docker-compose.yaml down
[+] Running 0/5
- Container functional-test-firefox-2 Stopping 15.5s
- Container functional-test-chrome-1 Stopping 15.4s
- Container functional-test-firefox-1 Stopping 15.4s
- Container functional-test-edge-1 Stopping 15.4s
- Container functional-test-chrome-2 Stopping 15.4s

PS C:\Users\... \Selenium\ctti\functional-test> docker-compose -f docker-compose.yaml down
[+] Running 7/7
✓Container functional-test-firefox-2 Removed 18.9s
✓Container functional-test-chrome-1 Removed 19.4s
✓Container functional-test-firefox-1 Removed 18.7s
✓Container functional-test-edge-1 Removed 17.9s
✓Container functional-test-chrome-2 Removed 19.3s
✓Container selenium-hub Removed 8.2s
✓Network functional-test_default Removed 1.6s
PS C:\Users\... \Selenium\ctti\functional-test>
    
```

5. Explicació de la plantilla/libreria

La plantilla descarregada en l'apartat 3 inclou la llibreria del MAT que inclou classes comunes que permeten la correcta execució de les proves. Aquestes classes inclouen mètodes com aixecar el Drive de Selenium, obrir el navegador, realitzar captures, monitorar els resultats en InfluxDB i Grafana, etc.

Arxius visibles en la plantilla	Arxius pertanyents de la llibreria
• config.properties. • src/test. ◦ Pages > QualityGenCatPage.java ◦ QualityGenCatTest.java • docker-compose.yaml. • pom.xml • README: aquí es pot trobar l'explicació de la plantilla.	• src/main/java: ◦ BaseTest.java ◦ BrowserOptions.java ◦ ConfigParameters.java ◦ ExecutionListener.java ◦ ExtentManager.java ◦ ResultSender.java ◦ Utils.java

Els arxius pertanyents a la llibreria no estan visibles en la plantilla descarregada, ja que aquestes són classes comunes i no necessiten ser modificades pel tester.

Dins de la carpeta **src/test/java** es troba la carpeta Java que al seu torn conté dos elements:

- Pages > QualityGenCatPage.java: aquesta classe java és en la qual es defineixen els Pages Object Model (explicat en apartat 6.1)
- QualityGenCatTest.java: aquí es defineixen els passos del test que s'executaran. Aquest test estén de la classe **BaseTest.java** i utilitza mètodes definits en la classe **Utils.java**. Totes dues pertanyents a la llibreria.

5.1. Arxius pertanyents de la llibreria

5.1.1. ConfigParameters

Aquesta classe analitza i guarda els paràmetres de configuració (la url de l'aplicació, el mantenidor, les dades de InfluxDB...).

Aquestes són les variables d'entorn disponibles:

```
Protected static String app=System.getenv("MAT_TF_APP");

Protected static String app_url=System.getenv("MAT_TF_APP_URL");

Protected static String maintainer=System.getenv("MAT_TF_MAINTAINER");

protected static String ambit = System.getenv("MAT_TF_AMBIT");

Protected static String selenium_url=System.getenv("MAT_TF_SELENIUM_URL");

Protected static String influxdb_url=System.getenv("MAT_TF_INFLUXDB_URL");

Protected static String influxdb_token=System.getenv("MAT_TF_INFLUXDB_TOKEN");

Protected static String influxdb_bucket=System.getenv("MAT_TF_INFLUXDB_BUCKET");

Protected static String influxdb_company=System.getenv("MAT_TF_INFLUXDB_COMPANY");

Protected static String selenium_firefox_driver=System.getenv("MAT_TF_SELENIUM_FIREFOX_DRIVER");

Protected static String environment=System.getProperty("environment");

Protected static String build_id=System.getProperty("build_id");

Protected static String job_name=System.getProperty("job_name");

Protected static String jira_pk=System.getProperty("jira_pk");

Protected static String jira_issue=System.getProperty("jira_issue");
```

Les variables de environment, build_id, job_name i les relacionades amb influx són necessàries per a la connexió del projecte amb Grafana.

En l'apartat 5.2 es mostra com es realitza l'execució de la plantilla. En aquest exemple, no es realitzarà cap anomenada a les variables de Influx.

Diferents formes de aplicar les variables en els paràmetres d'entrada.

- Directament des d'un comandament en el terminal. (-D)

Example: *mvn clean test -Dselenium_url="http://localhost:4444/wd/hub"*

- A través del fitxer “config.properties” de la plantilla.

```
≡ config.properties
1   maintainer = LOT-AM01
2   app = template
3   app_url = https://qualitat.solucions.gencat.cat
4   ambit = Cultura
5
```

La lògica de la classe està formada de manera que si el tester inclou els paràmetres en l'arxiu de *config.properties*, no és necessari que els cridi en el moment de l'execució.

No obstant això, si en l'arxiu de *config.properties* està declarada una variable i de nou es diu en la terminal amb el comando -D, aquesta tindrà preferència.

És a dir, el flux del projecte serà mirar si algun paràmetre va ser manat amb el comando -D, en cas que no haver-hi res en -D, buscarà en el *config.properties* i en cas de no veure res en *config.properties*, reconecrà la variable com nul·la.

Per exemple, en el fitxer de *config.properties* es defineix la variable d'*app_url*.

En la terminal es pot manar simplement la instrucció:

```
mvn clean test --settings maven/settings.xml -Dselenium_url="http://localhost:4444/wd/hub"
```

En cas de manar per consola la variable *app_url*, el projecte llegirà primer la variable definida amb el comando -D, ignorant la que està en *config.properties*.

```
mvn clean test --settings maven/settings.xml -Dselenium_url="http://localhost:4444/wd/hub"
-Dapp_url="https://qualitat.solucions.gencat.cat"
```

Què és InfluxDB?

InfluxDB és una base de dades de sèries temporals que generen mètriques de l'aplicació.

InfluxDB s'integra amb **Grafana**, que és una plataforma que permet el monitoratge i visualització de panells de control.

5.1.2. BaseTest.java

La classe **BaseTest**.java pertany a la llibreria, i conté mètodes comuns (iniciar el driver, tancar el driver...) vàlids per a qualsevol projecte que s'executaran abans de cada classe, abans de cada mètode, en iniciar la suite, etc.

Anotació	Mètodes	Explicació
@BeforeSuite	suiteInit()	Inicialitza el conjunt de proves.
@BeforeMethod	testInit(String browser, Method method)	Procés de configuració per a un cas de prova específic.
@AfterMethod	testQuit()	Atura el controlador web.
@AfterClass	testShutdown()	Neteja tots els recursos relacionats amb el cas de prova.
@AfterSuite	suiteShutdown()	Atura el conjunt de proves.

5.1.3. Utils.java

La classe Utils conté mètodes que poden ser necessaris en qualsevol projecte com accedir a l'aplicació, maximitzar la finestra, scroll, verificar un element...

Mètodes	Explicació
gotoApp()	Accedir a la url de l'app a testar. Aquest mètode diu el paràmetre app_url.
maximize()	Maximitzar la finestra del navegador
getElement(By selector)	Obtenir l'element web d'un selector.
getElement(By selector, int timeout)	Obtenir l'element web d'un selector amb un temps d'espera.
scroll(int percent_x, int percent_y, int timeout)	Desplaça la finestra (en percentatge) amb un temps d'espera.
scrollToBottom(int timeout)	Desplaça la finestra cap a la part inferior (eix Y) amb un temps d'espera.
scrollToTop(int timeout)	Desplaceu la finestra cap a la part superior (eix Y) amb un temps d'espera.
step(String name)	Definiu un nou pas dins d'un cas de prova.
anotate(LogLevel level, String msg)	Afegeix una anotació dins del context del cas de prova actual. Sent "level" l'estat del pas i "msg" el missatge explicatiu
screenshot(String caption)	Feu una captura de pantalla nova de l'àrea de visualització del navegador actual i adjunteu-la al context del cas de prova actual.
endTestAsOK(String browser, Method method)	Gestiona el final del cas de prova amb èxit.



```
endTestAsKO(String browser,  
Method method, Throwable  
e)
```

Gestiona el final del cas de prova amb error.

5.1.4. BrowserOptions.java

La classe BrowserOptions conté totes característiques pròpies per a cada navegador.

5.1.5. ExtentManager.java

Aquesta classe defineix els mètodes els mètodes necessaris per a bolcar els resultats al document .html.

5.1.6. ResultSender.java i ExecutionListener.java

Aquestes classes són les que gestionen la connexió amb InfluxDB.

5.2. Compilar i executar la plantilla

Com ja està Selenium Grid aixecat (veure pas 4), ja es pot executar la plantilla de prova, per a comprovar que tot funciona correctament.

Comprovar que el projecte compila correctament:

```
mvn clean compiler:compile compiler:testCompile
```



```
docker-compose.yaml
2 services:
10 chrome:
15 environment:
17   - SE_EVENT_BUS_PUBLISH_PORT=4442
18   - SE_EVENT_BUS_SUBSCRIBE_PORT=4443
19   - SE_NODE_MAX_INSTANCES=1
20   - SE_NODE_MAX_SESSIONS=1
21 edge:
22   image: selenium/node-edge

PS C:\Users\... \Selenium\CTTI\functional-test> mvn clean compiler:compile compiler:testCompile
[INFO] Scanning for projects...
[INFO]
[INFO] -----< ctti:template >-----
[INFO] Building template undefined
[INFO] from pom.xml
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ template ---
[INFO] Deleting C:\Users\m.martin.losada\Selenium\CTTI\functional-test\target
[INFO]
[INFO] --- maven-compiler-plugin:3.6.1:compile (default-compile) @ template ---
[INFO] No sources to compile
[INFO]
[INFO] --- maven-compiler-plugin:3.6.1:testCompile (default-compile) @ template ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 2 source files to C:\Users\m.martin.losada\Selenium\CTTI\functional-test\target\test-classes
[INFO]
[INFO] BUILD SUCCESS
```

Executar el projecte:

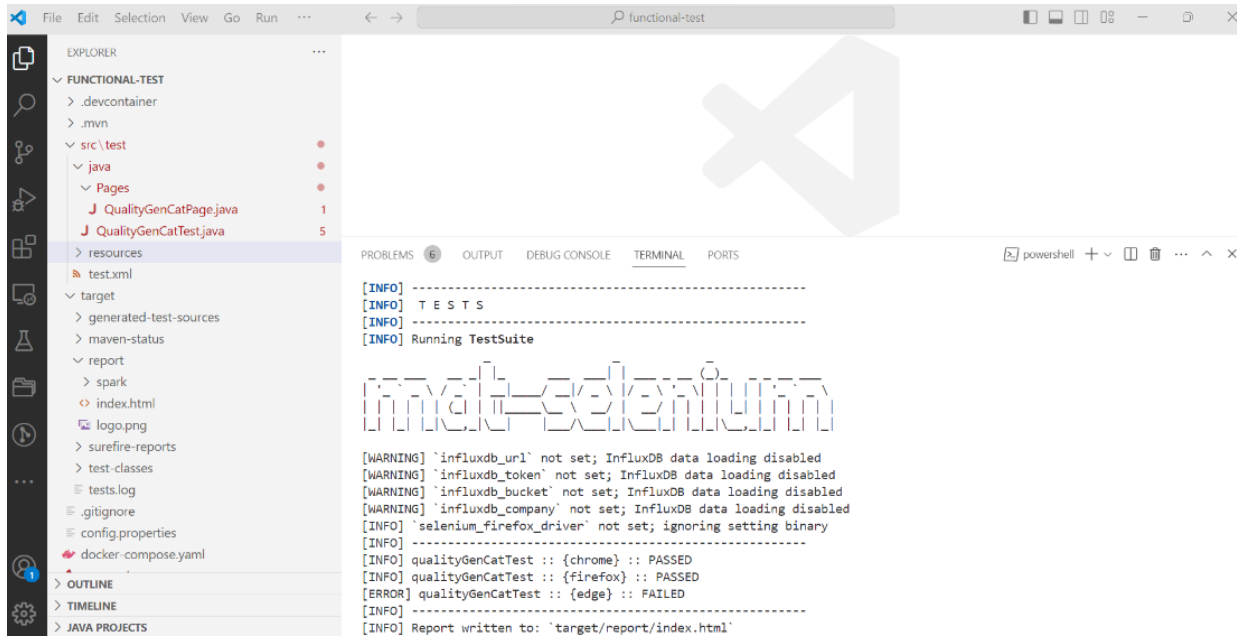
```
mvn clean test --settings maven/settings.xml -
Dselenium_url="http://localhost:4444/wd/hub" -
Dapp_url="https://qualitat.solucions.gencat.cat"
```

```
mvn clean test --settings maven/settings.xml -Dselenium_url="[url Selenium Grid]" -
```

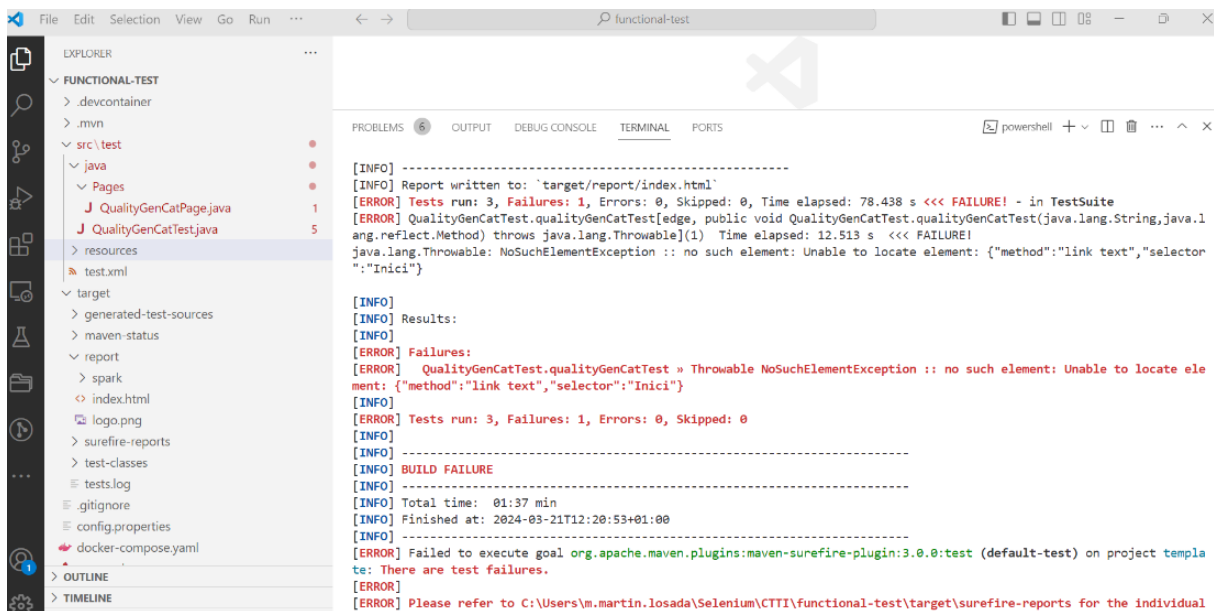


```
PS C:\Users\m.martin.losada\Selenium\CTTI\functional-test> mvn clean test -Dwebdriver.chrome.driver=C:\Users\m.martin.losada\Selenium\CTTI\functional-test\src\main\resources\chromedriver.exe
Dapp_url="https://qualitat.soluciones.gencat.cat"
[INFO] Scanning for projects...
[INFO]
[INFO] -----< ctti:template >-----
[INFO] Building template undefined
[INFO] from pom.xml
[INFO]
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ template ---
[INFO] Deleting C:\Users\m.martin.losada\Selenium\CTTI\functional-test\target
[INFO]
[INFO] --- maven-resources-plugin:3.0.2:resources (default-resources) @ template ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\Users\m.martin.losada\Selenium\CTTI\functional-test\src\main\resources
[INFO]
[INFO] --- maven-resources-plugin:3.0.2:copy-resources (copy-report-logo) @ template ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.6.1:compile (default-compile) @ template ---
```

```
PS C:\Users\m.martin.losada\Selenium\CTTI\functional-test> mvn clean test -Dwebdriver.chrome.driver=C:\Users\m.martin.losada\Selenium\CTTI\functional-test\src\main\resources\chromedriver.exe
Dapp_url="https://qualitat.soluciones.gencat.cat"
[INFO] Scanning for projects...
[INFO]
[INFO] -----< ctti:template >-----
[INFO] Building template undefined
[INFO] from pom.xml
[INFO]
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ template ---
[INFO] Deleting C:\Users\m.martin.losada\Selenium\CTTI\functional-test\target
[INFO]
[INFO] --- maven-resources-plugin:3.0.2:resources (default-resources) @ template ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\Users\m.martin.losada\Selenium\CTTI\functional-test\src\main\resources
[INFO]
[INFO] --- maven-resources-plugin:3.0.2:copy-resources (copy-report-logo) @ template ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.6.1:compile (default-compile) @ template ---
```



```
[INFO] -----  
[INFO] T E S T S  
[INFO] -----  
[INFO] Running TestSuite  
  
[WARNING] 'influxdb_url' not set; InfluxDB data loading disabled  
[WARNING] 'influxdb_token' not set; InfluxDB data loading disabled  
[WARNING] 'influxdb_bucket' not set; InfluxDB data loading disabled  
[WARNING] 'influxdb_company' not set; InfluxDB data loading disabled  
[INFO] 'selenium_firefox_driver' not set; ignoring setting binary  
[INFO] -----  
[INFO] qualityGenCatTest :: {chrome} :: PASSED  
[INFO] qualityGenCatTest :: {firefox} :: PASSED  
[ERROR] qualityGenCatTest :: {edge} :: FAILED  
[INFO] -----  
[INFO] Report written to: 'target/report/index.html'
```



```
[INFO] -----  
[INFO] Report written to: 'target/report/index.html'  
[ERROR] Tests run: 3, Failures: 1, Errors: 0, Skipped: 0, Time elapsed: 78.438 s <<< FAILURE! - in TestSuite  
[ERROR] QualityGenCatTest.qualityGenCatTest(edge, public void QualityGenCatTest.qualityGenCatTest(java.lang.String,java.1  
ang.reflect.Method) throws java.lang.Throwable)(1) Time elapsed: 12.513 s <<< FAILURE!  
java.lang.Throwable: NoSuchElementException :: no such element: Unable to locate element: {"method":"link text","selector  
":"Inici"}  
  
[INFO] Results:  
[INFO] Failures:  
[ERROR] QualityGenCatTest.qualityGenCatTest » Throwable NoSuchElementException :: no such element: Unable to locate ele  
ment: {"method":"link text","selector":"Inici"}  
[INFO] Tests run: 3, Failures: 1, Errors: 0, Skipped: 0  
[INFO] BUILD FAILURE  
[INFO] Total time: 01:37 min  
[INFO] Finished at: 2024-03-21T12:20:53+01:00  
[INFO] -----  
[ERROR] Failed to execute goal org.apache.maven.plugins:maven-surefire-plugin:3.0.0:test (default-test) on project templa  
te: There are test failures.  
[ERROR] Please refer to C:\Users\m.martin.losada\Selenium\CTTI\functional-test\target\surefire-reports for the individual
```

6. Crear casos de proves

El test d'exemple que es crearà és el següent:

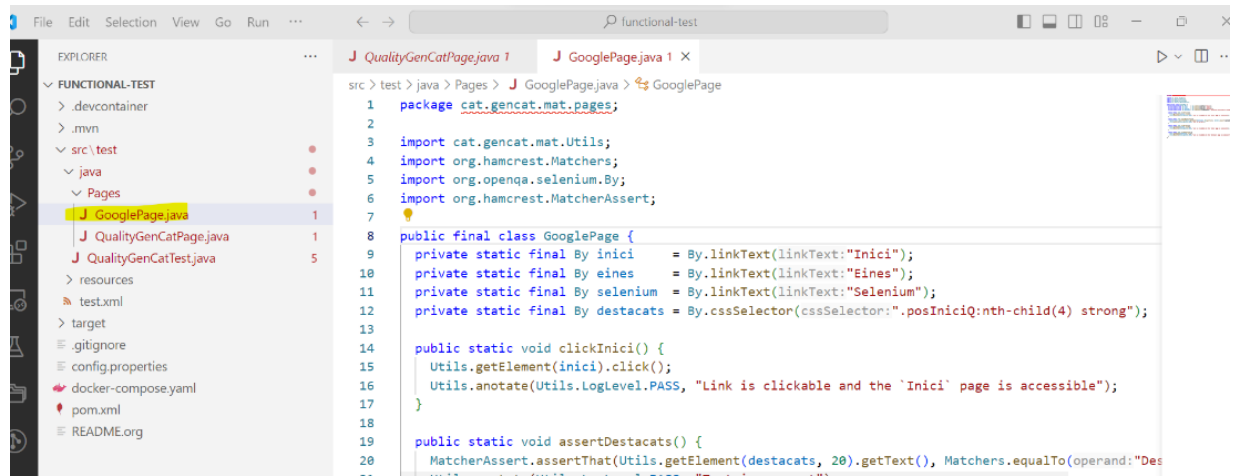
1. Navega per www.google.com
2. Verificar imatge de "Google".

3. En el cercador, cerca "Selenium".
4. Clic en l'opció de "Selenium" del desplegable.
5. Buscar la url de <https://www.selenium.dev/>
6. Clic en Selenium.

6.1. Definir Page Objet Models

Per a crear els casos de prova, primer s'han de definir la "Page". Aquesta classe contindrà tots els elements específics necessaris per a les proves ([page object models](#)), per exemple, validar un text, clic en un element de la web, etc.

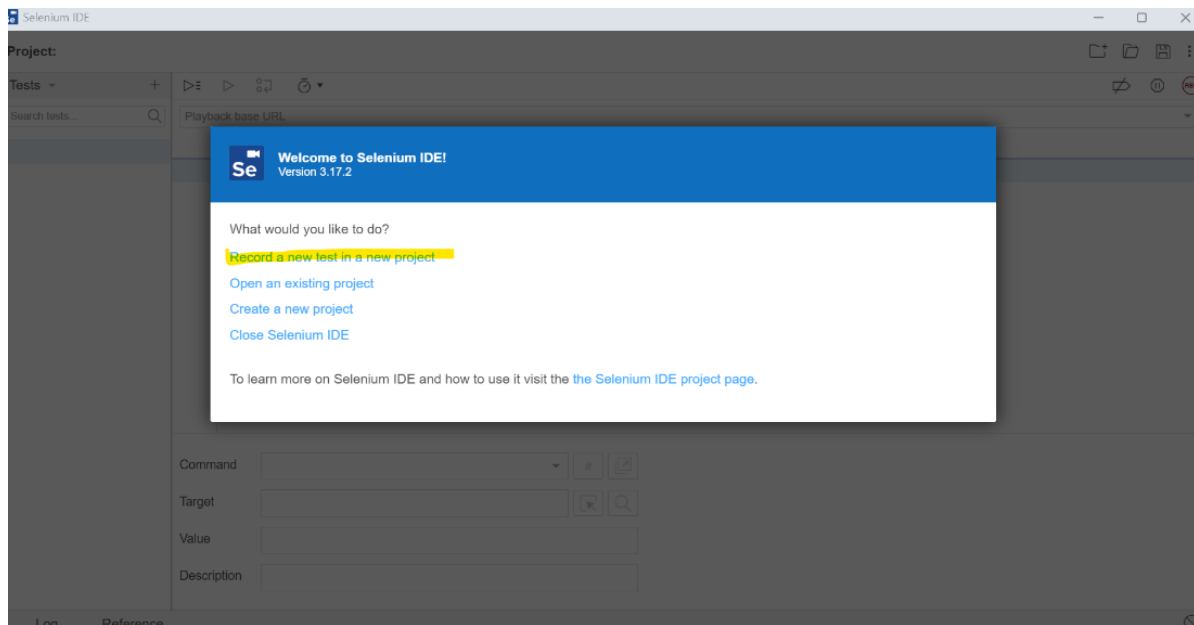
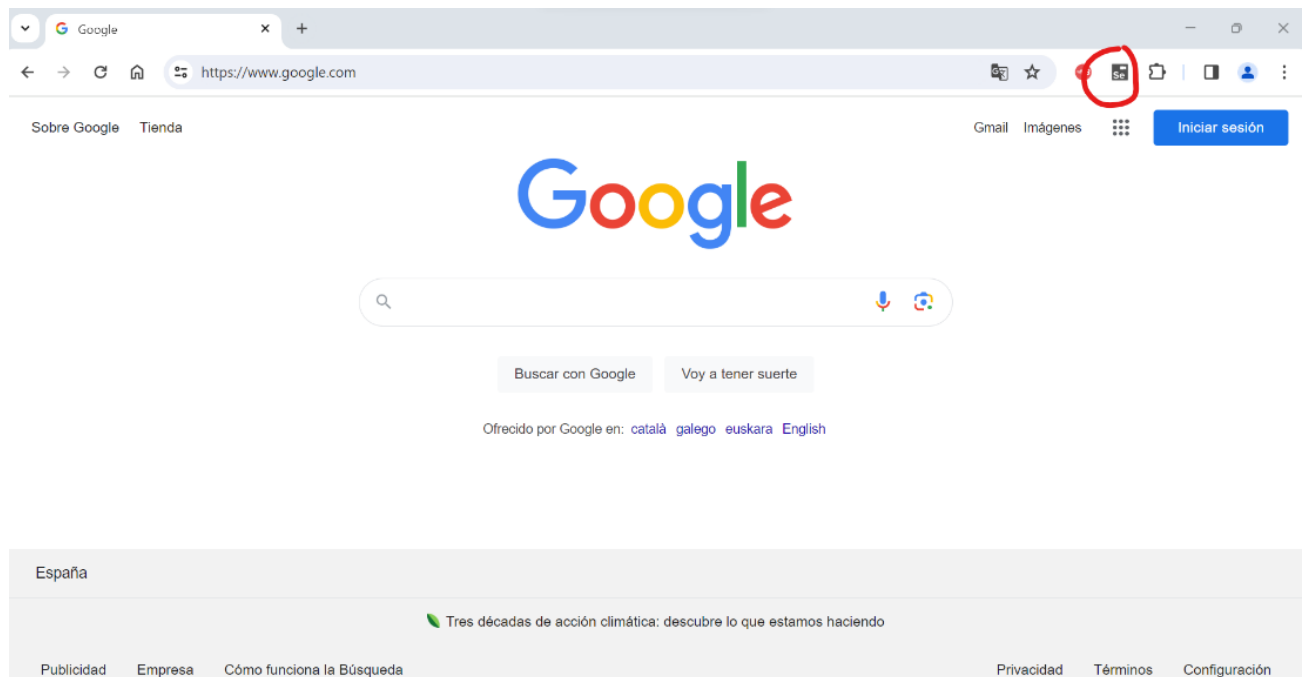
Realitzar una còpia de QualityGenCatPage.java i canviar el nom acord a les teves proves- (**GooglePage.java**).



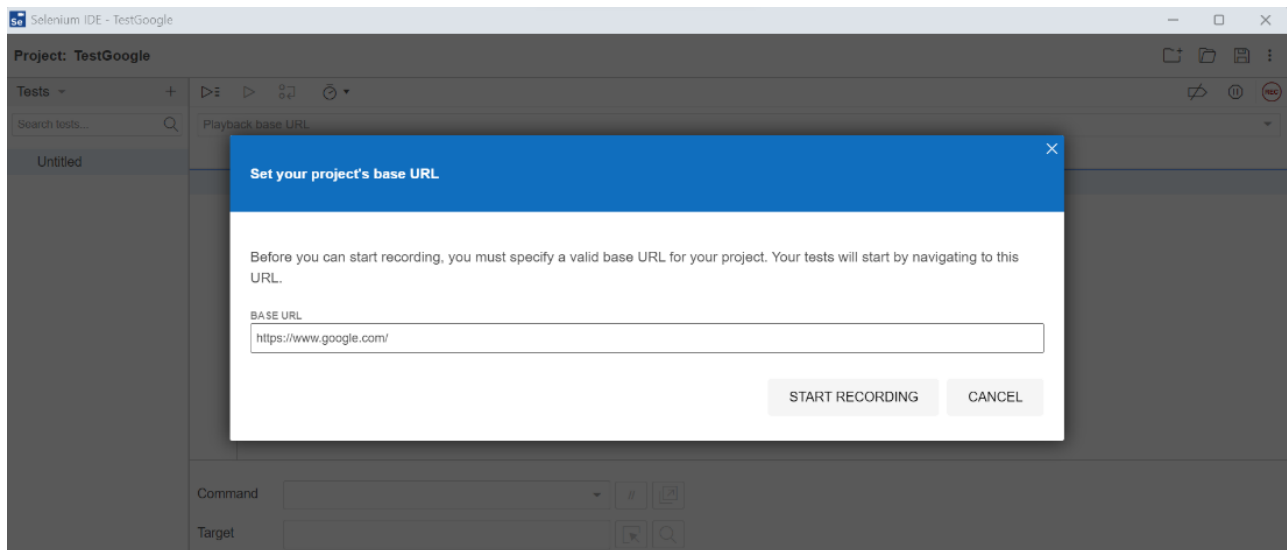
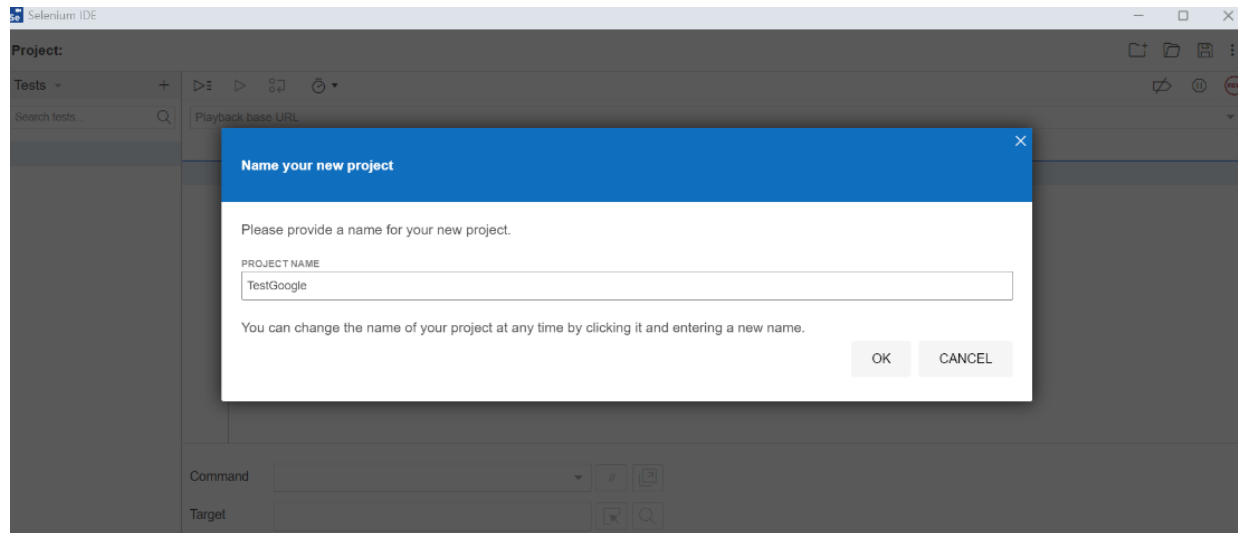
```
1 package cat.gencat.mat.pages;
2
3 import cat.gencat.mat.Utils;
4 import org.hamcrest.Matchers;
5 import org.openqa.selenium.By;
6 import org.hamcrest.MatcherAssert;
7
8 public final class GooglePage {
9     private static final By inici = By.linkText(linkText:"Inici");
10    private static final By eines = By.linkText(linkText:"Eines");
11    private static final By selenium = By.linkText(linkText:"Selenium");
12    private static final By destacats = By.cssSelector(cssSelector:".posIniciQ:nth-child(4) strong");
13
14    public static void clickInici() {
15        Utils.getElement(inici).click();
16        Utils.anoate(Utils.LogLevel.PASS, "Link is clickable and the 'Inici' page is accessible");
17    }
18
19    public static void assertDestacats() {
20        MatcherAssert.assertThat(Utils.getElement(destacats, 20).getText(), Matchers.equalTo("Des
```

Selenium IDE ha sigut descartada del llistat d'extensions de Chrome, segueix sent funcional a Firefox.

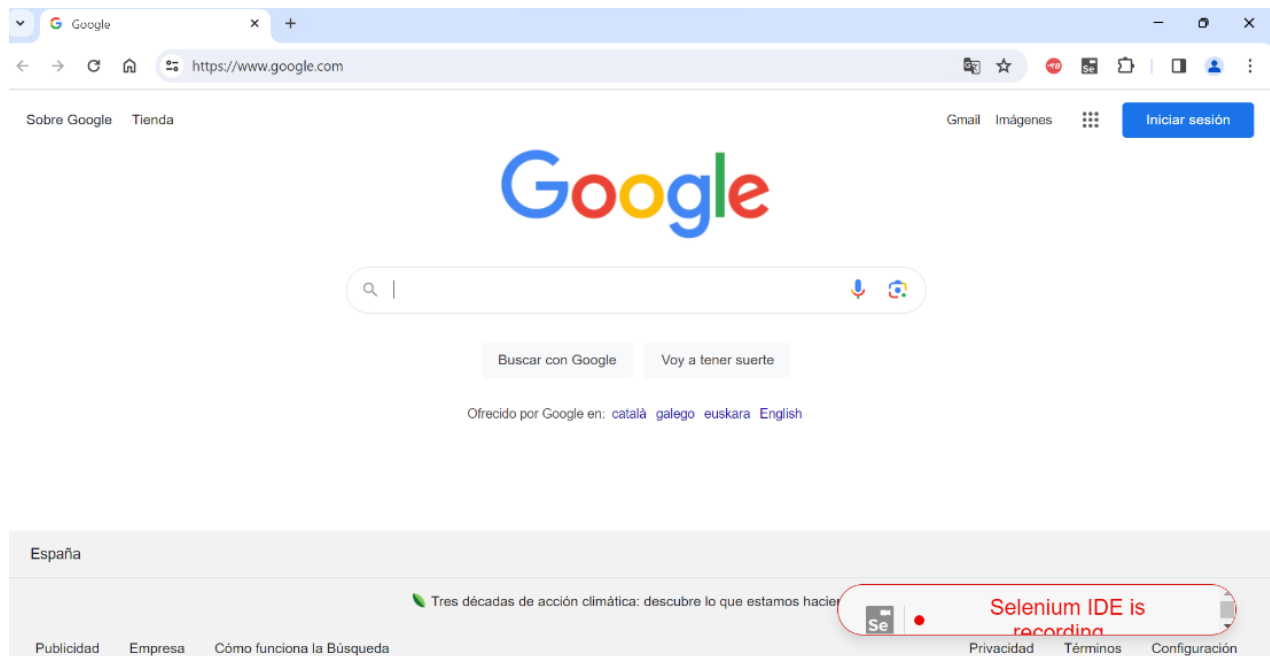
Per a definir els objectes localitzadors, es pot utilitzar l'extensió de Firefox de [Selenium IDE](#). Aquesta extensió permet conèixer tots els possibles localitzadors de cada element de la pàgina.



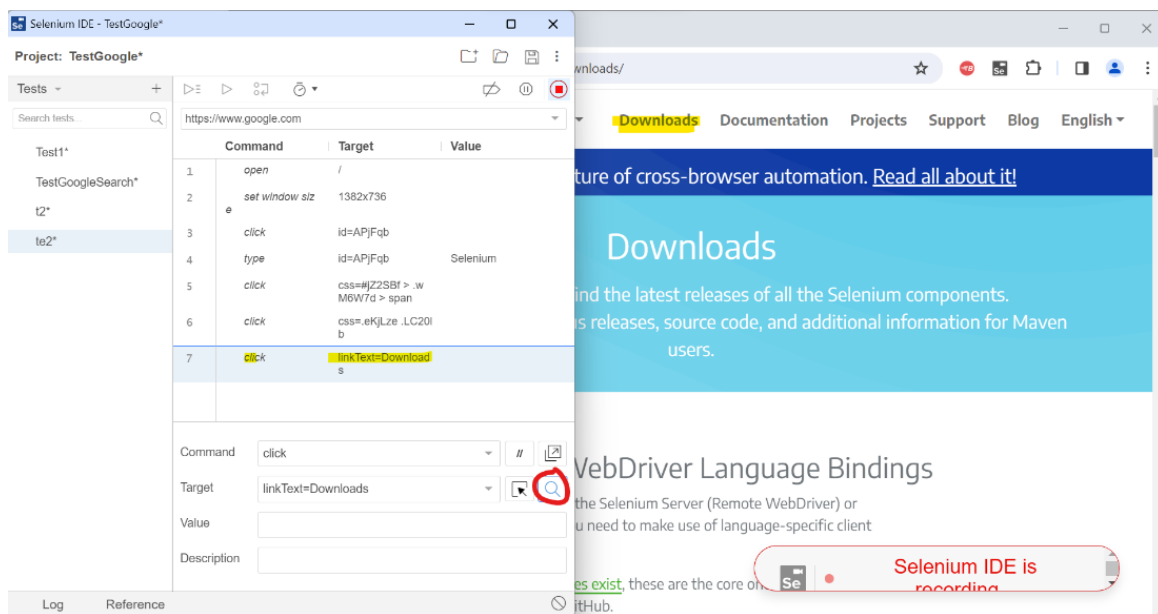
No és necessari guardar aquest projecte de Selenium IDE. Aquesta extensió és útil per a conèixer quins tipus d'elements hem d'incloure a la classe GooglePage.java.



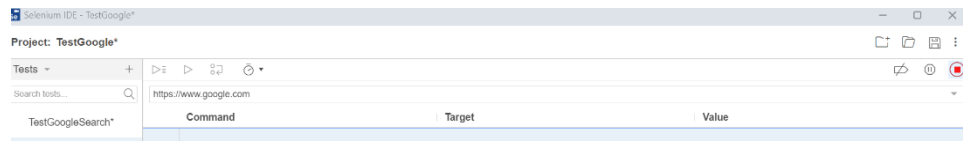
S'aixeca un navegador que registrarà les accions que es realitzin.



Mentre la gravació es trobi activa, es pot comprovar amb la lupa com és element identificat per Selenium IDE en el propi navegador.



Una vegada executades totes les accions necessàries per al test, podem parar la gravació.



Selenium IDE mostra totes les accions realitzades:

Project: TestGoogle*

Tests ▾ +

Search tests...

TestGoogleSearch*

https://www.google.com

	Command	Target	Value
1	open	/	
2	set window size	1382x736	
3	click	linkText=Iniciar sesión	
4	click	css=#APjFqb	
5	type	id=APjFqb	Selenium
6	click	xpath=//span[contains(., 'selenium')]	
7	click	css=.eKjLze .LC20lb	
8	double click	xpath=//h2[contains(., 'Getting Started')]	
9	click	linkText=Downloads	
10	close		

Command:

Target:

Value:

Description:

Log Reference

Cadascuna de les accions realitzades es grava amb un tipus d'objecte. En el desplegable, és possible veure altres maneres d'identificació de l'element.

Selenium IDE - TestGoogle*

Project: TestGoogle*

Tests +

Search tests...

https://www.google.com

TestGoogleSearch*	Command	Target	Value
1	open	/	
2	set window size	1382x736	
3	click	id=APjFqb	
4	type	id=APjFqb	Selenium
5	click	css=#ZZSBf > .wM6W7d > span	
6	click	css=.eKJLze .LC20lb	
7	click	linkText=Downloads	
8	close		

Command: click

Target: linkText=Downloads

Value: linkText=Downloads

Description: css=.nav-item:nth-child(2) > .nav-link

xpath=//div[@id='main_navbar']/u1/1[2]/a

Loc Reference

Tots aquests elements també es poden identificar amb el Inspect del propi navegador.

Google

https://www.google.com/webhp?hl=es&sa=X&ved=0ahUKEwiCh6uzlWFAx39AIHHA8K8LoQPAGJ

Google

textarea#APjFqb.gLFyf 443 x 27

Buscar con Google

Voy a tener suerte

Ofrecido por Google en: català galego euskara

Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder Performance insights

```

<style data-impl="1711033390262"></style>
<div jsname="RNIXgb" class="RNIXgb">
  <div class="SDKEP">
    <div jsname="uFMOf" class="ib1pc">
      <div jscontroller="vZr2rb" jsname="gLFyf" class="a4bIc" data-hpmed="false" data-mnr="10" jsaction="h5M12e;input:d3sQLd;blur:JI3wzf">
        <style data-impl="1711033390262"></style>
        <div jsname="vdlsw" class="VacQv">
          <textarea class="gLFyf" aria-controls="Alh6id" aria-owns="Alh6id" autofocus title="Buscar" value jsaction="paste:puy29d;" aria-label="Buscar" aria-autocomplete="both" aria-expanded="false" aria-haspopup="false" autocapitalize="off" autocomplete="off" autocorrect="off" id="APjFqb" maxlength="2048" name="q" role="combobox" rows="1" spellcheck="false" data-ved="0ahUKEwjYsOFVz4WFAxXVTaQEHtpCDPUQ4tUDCAo"></textarea>
        </div>
        <div class="dRYXxd"></div>
      </div>
    </div>
  </div>
  <div jscontroller="Dvn7fe" jsname="UUBT9" class="UUBT9 EyBRub" style="display:none" jsaction="mouseout:ItzDCd;mouseleave:MMfikb;hBEIVb:nUZ91e;YMF3:VKssTbjvk1u5c;k02QY;ldyIe:CmVOgc" data-ved="0ahUKEwjYsOFVz4WFAxXVTaQEHtpCDPUQ4tUDCAo">
    </div>
</div>
  
```

html body div.L3eUgb div.o3j99.ikrT4e.om7nvf form div div.A8SBwf div.RNIXgb div.SDKEP div.a4bIc textarea#APjFqb.gLFyf

Styles Computed Layout

Filter: :hov .cls + -

element.style {

text webhp?hl=es..BLoQPAGJ:60

rea.gLFyf {

font-family: arial, sans-serif;

line-height: 22px;

margin-bottom: 8px;

overflow-x: hidden;

gLFyf webhp?hl=es..BLoQPAGJ:60

f {

Coneixent els elements, ja es pot definir en GooglePage.java les variables estàtiques:

```

FUNCTION...  src > test > java > Pages > J GooglePage.java > GooglePage
> .devcontainer
> .mvn
✓ src\test
  ✓ java
    ✓ Pages
      J GooglePage.java 1
      J QualityGenCatP... 1
      J GoogleTest.java 1
      J QualityGenCatTe... 5
  > resources

```

```

8  import org.openqa.selenium.WebElement;
9
10 import java.util.List;
11
12 import org.hamcrest.MatcherAssert;
13
14 public final class GooglePage {
15     //Incloure aquí tots els elements localitzadors de la pàgina
16     //google.com
17     private static final By imageGoogleHome = By.className(className:"lnXdpd");
18     private static final By searchBar      = By.cssSelector(cssSelector:"#APjFqb");

```

```

public final class GooglePage extends BaseTest {

//Incloure aquí tots els elements localitzadors de la pàgina
//google.com
private static final By imageGoogleHome      = By.className("lnXdpd");
private static final By searchBar            = By.cssSelector("#APjFqb");

```

Ara hem de crear els mètodes necessaris per a l'execució del test, seguint la mateixa estructura que la plantilla.

```

EXPLORER  ...  J GooglePage.java 1  J GoogleTest.java 7
FUNCTIONAL-TEST
> .devcontainer
> .mvn
✓ src\test
  ✓ java
    ✓ Pages
      J GooglePage.java 1
      J QualityGenCatP... 1
      J GoogleTest.java 7
      J QualityGenCatTe... 5
  > resources
    test.xml
  > target
    .gitignore
    config.properties
    docker-compose.yaml
    pom.xml
    README.org

```

```

14 public final class GooglePage {
44 //Mètodes relacionats amb la pàgina de Google
45 public static void checkImageGoogleHome() {
46     BaseTest.getDriver().findElements(imageGoogleHome);
47     Utils.anoatate(Utils.LogLevel.PASS, "Google Image Home");
48 }
49
50 public static String clickAndSearchBar (String somethingToSearch) {
51     Utils.getElement(searchBar).sendKeys(somethingToSearch);
52     Utils.anoatate(Utils.LogLevel.PASS, "Search Bar is clickable and available to send keys");
53     return somethingToSearch;
54 }
55
56 public static void clickDropdownOption(String label) {
57
58     //Elements del resultat de la cerca (dropdown)
59     List<WebElement> elements = BaseTest.getDriver().findElements(By.cssSelector("div.pcTkSc[aria-label='" + label
60
61     //Iteració - clic
62     for (WebElement element : elements) {
63         element.click();
64         break;
65     }
66 }
67

```



```
//Mètodes relacionats amb la pàgina de Google
public static void checkImageGoogleHome( ) {
    BaseTest.getDriver().findElements(imageGoogleHome);
    Utils.annotate(Utils.LogLevel.PASS, "Google Image Home");
}

public static String clickAndSearchBar (String somethingToSearch) {
    Utils.getElement(searchBar).sendKeys(somethingToSearch);
    Utils.annotate(Utils.LogLevel.PASS, "Search Bar is clickable and available to
send keys");
    return somethingToSearch;
}

public static void clickDropdownOption(String label) {
    //Elements del resultat de la cerca (dropdown)
    List<WebElement> elements =
BaseTest.getDriver().findElements(By.cssSelector("div.pcTkSc[aria-label='" +
label + "']"));

    //Iteració - clic
    for (WebElement element : elements) {
        element.click();
        break;
    }
}

public static void clickElementByHref(String href) {
    //Elements de la cerca trobats en Google
    List<WebElement> elements =
BaseTest.getDriver().findElements(By.cssSelector("span[jscontroller='msmzHf']"));

    //Iteració
    for (WebElement element : elements) {
        WebElement anchor = element.findElement(By.tagName("a")); //Buscar <a>
        String elementHref = anchor.getAttribute("href"); //Obtenir valor de href
en <a>

        //Clic quan es encuetre href
        if (elementHref.equals(href)) {
            anchor.click();
            break;
        }
    }
}
```

Tipus de mètodes	Mètodes	Explicació
Comprovar element	<code>checkImageGoogleHome()</code>	Cerca d'un element de la web.
Clic i escriure un text	<code>clickAndSearchBar()</code>	En la barra de cerca, escriure un text.
Clic	<code>clickDropDownOption()</code>	Després de buscar "Selenium", Clickar en l'opció de Selenium del resultat mostrat
Buscar un element concret i clicar	<code>ClickElementByHref()</code>	Primer es fa una cerca de tots els elements HTML trobats en la pàgina. Després es busca el element que contingui el href que es necessita per al test i finalment es fa clic en ell.

En els mètodes `checkImageGoogleHome()` i `clickDropDownOption()` és necessari cridar al mètode `getDriver()` de la classe `BaseTests`.
Per a això, és necessari que la classe `GooglePage` estengui de `BaseTest`.

6.2. Definir Test

De la mateixa manera que amb la creació dels Page Objet Models, es crea una còpia del test `QualityGenCatTest.java` concorde a les nostres proves (`GoogleTest.java`).

Es recomana que tots els test tinguin la terminació "Test" per a la seva millor identificació.

El test estén de la classe `BaseTest`. Aquesta classe forma part de la llibreria, i és on es defineixen els mètodes comuns per a totes les proves: iniciar el driver, llegir els paràmetres, generar els resultats HTML...



```

src > test > java > J GoogleTest.java > GoogleTest > googleTest(String, Method)
1  import cat.gencat.mat.Utils;
2  import cat.gencat.mat.BaseTest;
3  import java.lang.reflect.Method;
4  import org.testng.annotations.Test;
5  import org.testng.annotations.Parameters;
6
7  import cat.gencat.mat.pages.GooglePage;
8  import cat.gencat.mat.pages.QualityGenCatPage;
9
10 public final class GoogleTest extends BaseTest {
11     @Test @Parameters(value={"browser"})
12     public void googleTest(String browser, Method method) throws Throwable {
13         try {
14             Utils.step("Enter website");
15             Utils.gotoApp();
16             Utils.maximize();
17             Utils.annotate(Utils.LogLevel.PASS, "Website's homepage is accessible");
18             Utils.screenshot("Homepage");
19         }
    
```

Els primers passos del test seran comuns a totes les proves, i no és necessari modificar-los. Aquests es troben definits a la classe Utils (veure apartat 5.1.3). Aquesta classe Utils també forma part de la plantilla i recull mètodes comuns que poden ser d'utilitat en qualsevol projecte (obrir navegador, maximitzar, scroll...).

Aquests mètodes són:

Mètode	Explicació
<code>Utils.step("Enter website");</code>	Generar un pas en el report HTML.
<code>Utils.gotoApp();</code>	Obrir el navegador i accedeix a la url donada.
<code>Utils.maximize();</code>	Maximitzar el navegador.
<code>Utils.annotate(Utils.LogLevel.PASS, "Website's homepage is accessible");</code>	Mostrar el resultat en el report HTML.
<code>Utils.screenshot("Homepage");</code>	Realitzar una captura de pantalla.

```
✓ FUNCTION...  src > test > java > J GoogleTest.java > GoogleTest > googleTest(String, Method)
> .devcontainer
> .mvn
✓ src\test
  ✓ java
    ✓ Pages
      J GooglePage.java 1
      J QualityGenCatP... 1
      J GoogleTest.java 7
      J QualityGenCatTe... 5
  > resources
    test.xml
  > target
  .gitignore
  config.properties
  docker-compose.yaml
  pom.xml
  README.org

10 public final class GoogleTest extends BaseTest {
12 public void googleTest(String browser, Method method) throws Throwable {
23
24     Utils.step("Check Google Image");
25     GooglePage.checkImageGoogleHome();
26     Utils.screenshot("Check Google Image");
27
28     Utils.step("Search Selenium");
29     String somethingToSearch = "Selenium";
30     GooglePage.clickAndSearchBar(somethingToSearch);
31     Utils.screenshot("Search Selenium");
32
33     Utils.step("Click Selenium displayed");
34     String label = "selenium";
35     GooglePage.clickDropdownOption(label);
36     Utils.screenshot("Click Selenium displayed");
37
38     Utils.step("Click https://www.selenium.dev/ ");
39     String href = "https://www.selenium.dev/";
40     GooglePage.clickElementByHref(href);
41     Utils.screenshot("Click https://www.selenium.dev/");
42
43     Utils.endTestAsOK(browser, method);
44 }
```

Després s'afegeixen la resta dels passos necessaris per a la prova. En aquest cas, definits sobre `GooglePage.java`, que compleixin les següents accions:

1. Navega per www.google.com
2. Verificar imatge de "Google".
3. En el cercador, cerca "Selenium".
4. Clic en l'opció de "Selenium" del desplegable.
5. Buscar la url de <https://www.selenium.dev/>
6. Clic en Selenium.

Aquest passos estan dissenyats de manera que el test generarà una fallada a l'inici. En el apartat 8.1 s'analitzarà el motiu d'aquest.

D'aquesta manera, el test quedaria així:



```
public final class GoogleTest extends BaseTest {
    @Test @Parameters(value={"browser"})
    public void googleTest(String browser, Method method) throws Throwable {
        try {
            Utils.step("Enter website");
            Utils.gotoApp();
            Utils.maximize();
            Utils.anoate(Utils.LogLevel.PASS, "Website's homepage is accessible");
            Utils.screenshot("Homepage");

            Utils.step("Check Google Image");
            GooglePage.checkImageGoogleHome();
            Utils.screenshot("Check Google Image");

            Utils.step("Search Selenium");
            String somethingToSearch = "Selenium";
            GooglePage.clickAndSearchBar(somethingToSearch);
            Utils.screenshot("Search Selenium");

            Utils.step("Click Selenium displayed");
            String label = "selenium";
            GooglePage.clickDropdownOption(label);
            Utils.screenshot("Click Selenium displayed");

            Utils.step("Click https://www.selenium.dev/ ");
            String href = "https://www.selenium.dev/";
            GooglePage.clickElementByHref(href);
            Utils.screenshot("Click https://www.selenium.dev/");

            Utils.endTestAsOK(browser, method);
        }
        catch (Exception | AssertionError e) { Utils.endTestAsKO(browser, method, e); }
    }
}
```

Cal tenir en compte que els mètodes de "clickAndSearchBar", "clickDropdownOption" i "clickElementByHref" estan definits per String.

6.3. Definir test.xml

Aquest arxiu .xml és el que s'encarrega de recopilar tots els test del projecte amb els seus característiques. És a dir, aquí es definirà en quin navegador s'executarà cada test i l'anomenada a la classe que defineix el report HTML.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite thread-count="3" name="template" parallel="tests">
  <listeners>
    <listener class-name="cat.gencat.mat.ExecutionListener" />
  </listeners>

  <test name="GoogleTest.Chrome" parallel="methods">
    <parameter name="browser" value="chrome" />
    <classes>
      <class name="GoogleTest"></class>
    </classes>
  </test>

  <test name="GoogleTest.Edge" parallel="methods">
    <parameter name="browser" value="edge" />
    <classes>
      <class name="GoogleTest"></class>
    </classes>
  </test>

  <test name="GoogleTest.Firefox" parallel="methods">
    <parameter name="browser" value="firefox" />
    <classes>
      <class name="GoogleTest"></class>
    </classes>
  </test>
</suite>
```

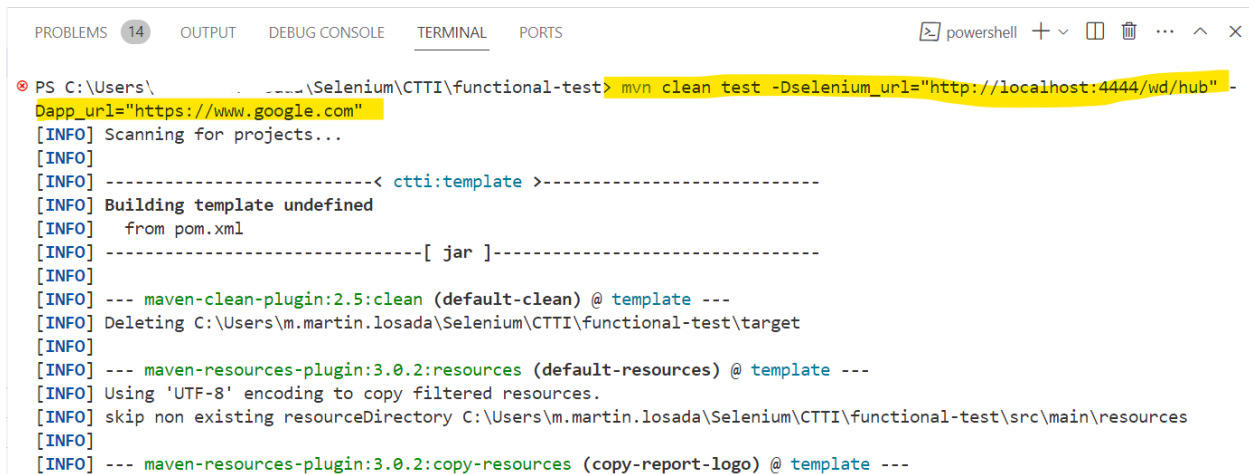
7. Executar les proves des de Visual Studio Code

Una vegada definits els test i implementats en el test.xml, ja es pot executar les proves des de la pròpia consola del Visual Studio Code, a través de la instrucció definida en l'apartat 5 o executant l'script launchTest.sh:

```
mvn clean test --settings maven/settings.xml -Dselenium_url="[url Selenium Grid]" -  
Dapp_url="[url de les proves]"
```

```
mvn clean test --settings maven/settings.xml -  
Dselenium_url="http://localhost:4444/wd/hub" -Dapp_url="https://www.google.com"
```

Comença l'execució del test:



```
PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + - [ ] [ ] ... ^ x
PS C:\Users\... Selenium\CTTI\functional-test> mvn clean test -Dselenium_url="http://localhost:4444/wd/hub" -  
Dapp_url="https://www.google.com"
[INFO] Scanning for projects...
[INFO] -----< ctti:template >-----
[INFO] Building template undefined
[INFO] from pom.xml
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ template ---
[INFO] Deleting C:\Users\m.martin.losada\Selenium\CTTI\functional-test\target
[INFO]
[INFO] --- maven-resources-plugin:3.0.2:resources (default-resources) @ template ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\Users\m.martin.losada\Selenium\CTTI\functional-test\src\main\resources
[INFO]
[INFO] --- maven-resources-plugin:3.0.2:copy-resources (copy-report-logo) @ template ---
```

```
PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS
[INFO] Running TestSuite

[WARNING] `influxdb_url` not set; InfluxDB data loading disabled
[WARNING] `influxdb_token` not set; InfluxDB data loading disabled
[WARNING] `influxdb_bucket` not set; InfluxDB data loading disabled
[WARNING] `influxdb_company` not set; InfluxDB data loading disabled
[INFO] `selenium_firefox_driver` not set; ignoring setting binary
[INFO] -----
[
```

Per consola, s'anirà mostrant tota la informació definida en la llibreria i en els tests.

En aquest cas, els WARNING relatius a InfluxDB indiquen que cap paràmetre de InfluxDB va ser definit, ja que aquests paràmetres solament s'activaran quan es llancin les proves sobre el MAT.

Quan finalitza cada test, mostra l'estat de cadascun d'ells i el navegador on va ser executat.

També s'indica que el report està disponible en la localització de `target/report/index.html`

```
PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS
[INFO] -----
[ERROR] googleTest :: {edge} :: FAILED
[ERROR] googleTest :: {chrome} :: FAILED
[ERROR] googleTest :: {firefox} :: FAILED
[INFO] -----
[INFO] Report written to: `target/report/index.html`
[ERROR] Tests run: 3, Failures: 3, Errors: 0, Skipped: 0, Time elapsed: 69.916 s <<< FAILURE! - in TestSuite
[ERROR] GoogleTest.googleTest[edge, public void GoogleTest.googleTest(java.lang.String,java.lang.reflect.Method) throws java.lang.Throwable](1) Time elapsed: 4.955 s <<< FAILURE!
java.lang.Throwable: ElementNotInteractableException :: element not interactable

[ERROR] GoogleTest.googleTest[chrome, public void GoogleTest.googleTest(java.lang.String,java.lang.reflect.Method) throws java.lang.Throwable](1) Time elapsed: 5.618 s <<< FAILURE!
java.lang.Throwable: ElementNotInteractableException :: element not interactable

[ERROR] GoogleTest.googleTest[firefox, public void GoogleTest.googleTest(java.lang.String,java.lang.reflect.Method) throws java.lang.Throwable](1) Time elapsed: 28.767 s <<< FAILURE!
java.lang.Throwable: TimeoutException :: Expected condition failed: waiting for presence of element located by: By.xpath: //span[contains(., 'selenium')] (tried for 20 second(s) with 500 milliseconds interval)
```

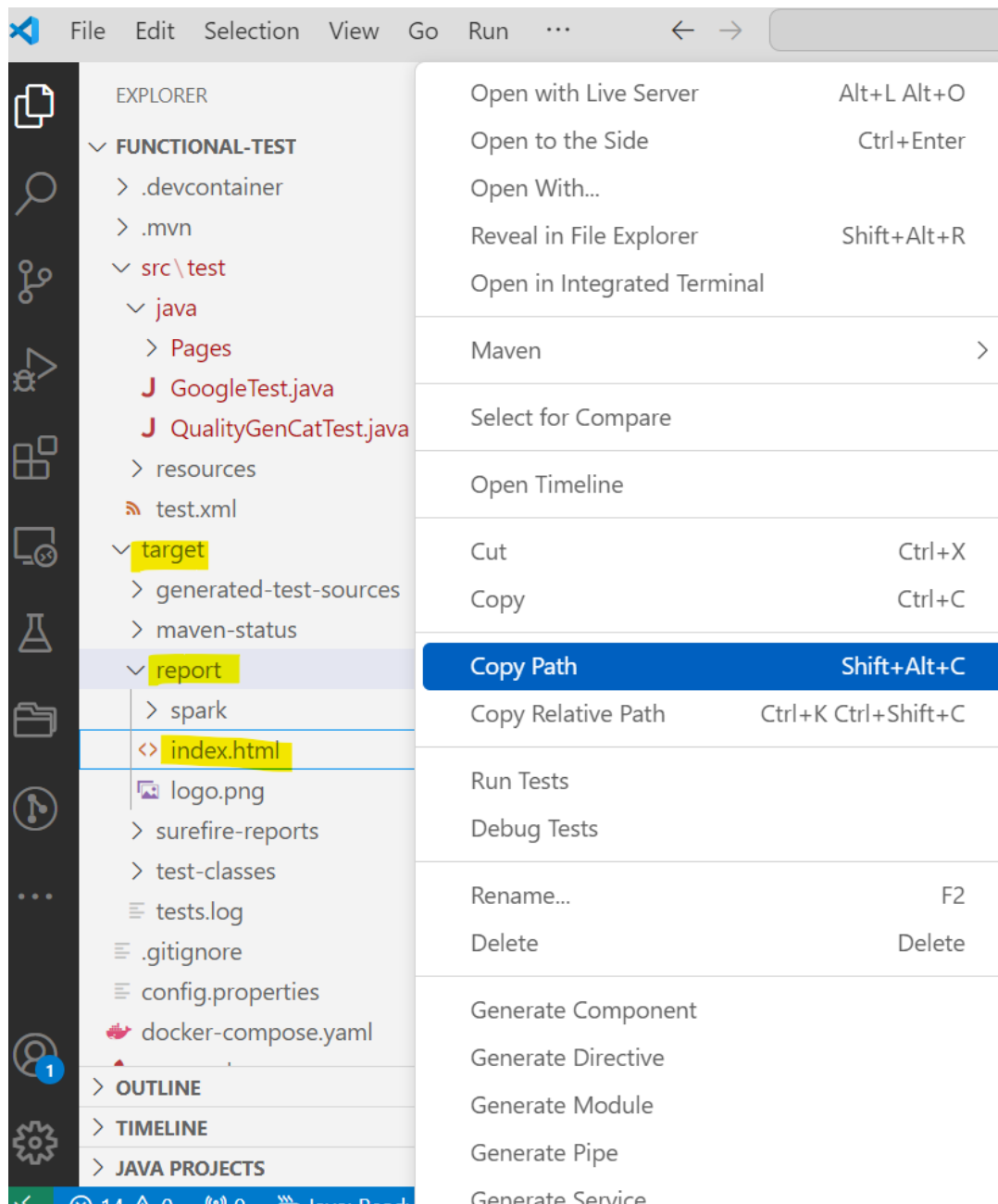
Finalment, també mostra una breu descripció del motiu de la fallada.



```
[ERROR] Failures:
[ERROR]   GoogleTest.googleTest » Throwable ElementNotInteractableException :: element not interactable
[ERROR]   GoogleTest.googleTest » Throwable ElementNotInteractableException :: element not interactable
[ERROR]   GoogleTest.googleTest » Throwable TimeoutException :: Expected condition failed: waiting for presence of element located by: By.xpath: //span[contains(.,'selenium')] (tried for 20 second(s) with 500 milliseconds interval)
[INFO]
[ERROR] Tests run: 3, Failures: 3, Errors: 0, Skipped: 0
[INFO]
[INFO] -----
[INFO] BUILD FAILURE
[INFO] -----
[INFO] Total time: 01:24 min
[INFO] Finished at: 2024-03-25T09:21:35+01:00
[INFO] -----
[ERROR] Failed to execute goal org.apache.maven.plugins:maven-surefire-plugin:3.0.0:test (default-test) on project template: There are test failures.
[ERROR]
[ERROR] Please refer to C:\Users\m.martin.losada\Selenium\CTTI\functional-test\target\surefire-reports for the individual test results.
[ERROR] Please refer to dump files (if any exist) [date].dump, [date]-jvmRun[N].dump and [date].dumpstream.
[ERROR] -> [Help 1]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the following articles:
[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/MojoFailureException
PS C:\Users\m.martin.losada\Selenium\CTTI\functional-test>
```

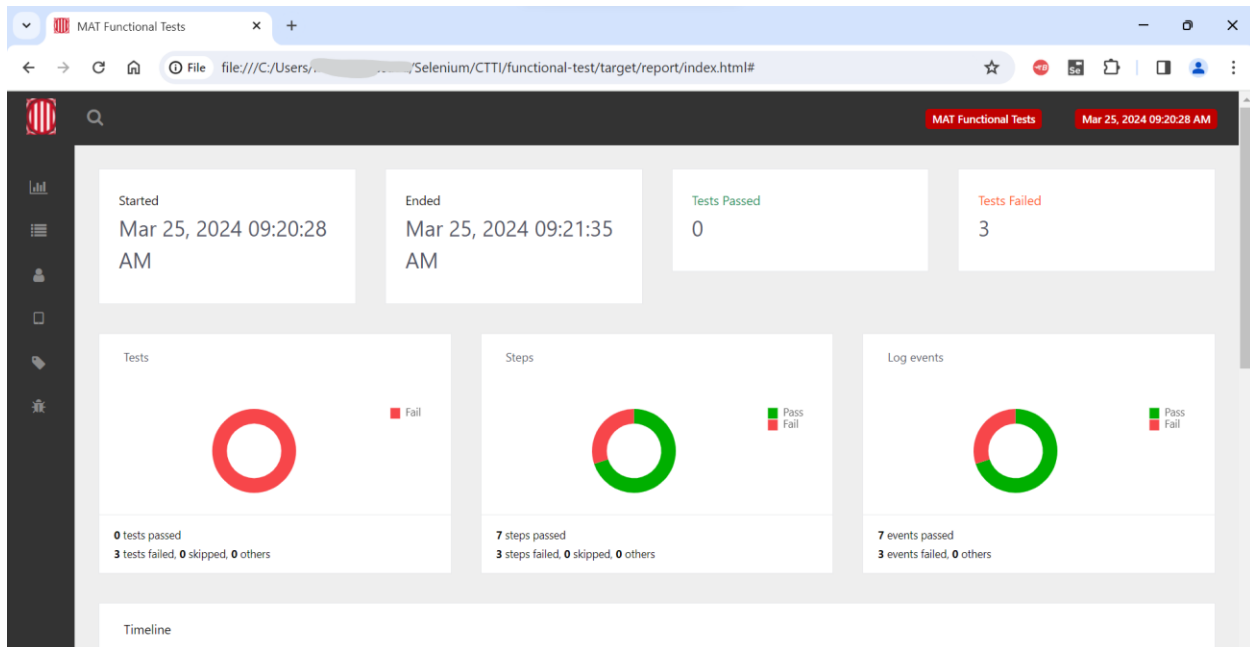
8. Anàlisi de resultats en el report HTML.

Una vegada acabada l'execució, es genera automàticament un reporti amb els resultats en format *.html*. Aquest es troba guardat en ***target/report/index.html***.



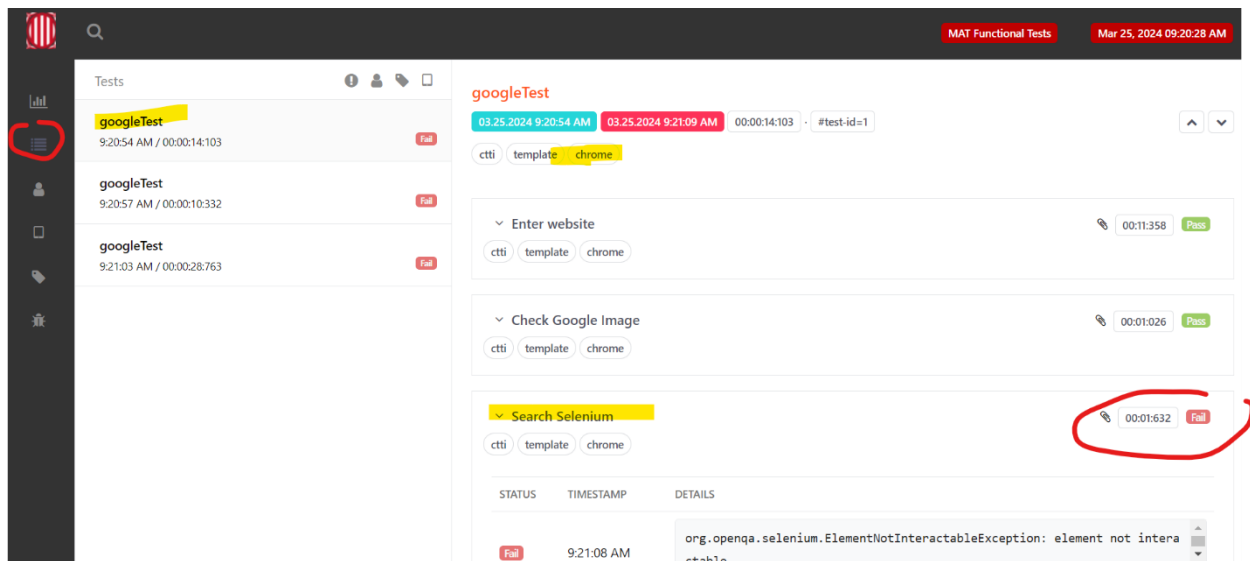
Es copia la direcció en un navegador, per a poder visualitzar els resultats de forma més còmoda.

En una primera visió, es pot veure el resum de l'execució i els casos passats o fallats.



Accedint a la llista dels test, es pot veure cada test executat i dins de cadascun d'ells, els passos realitzats (indicant si van ser passats o fallats i les seves captures de pantalla).

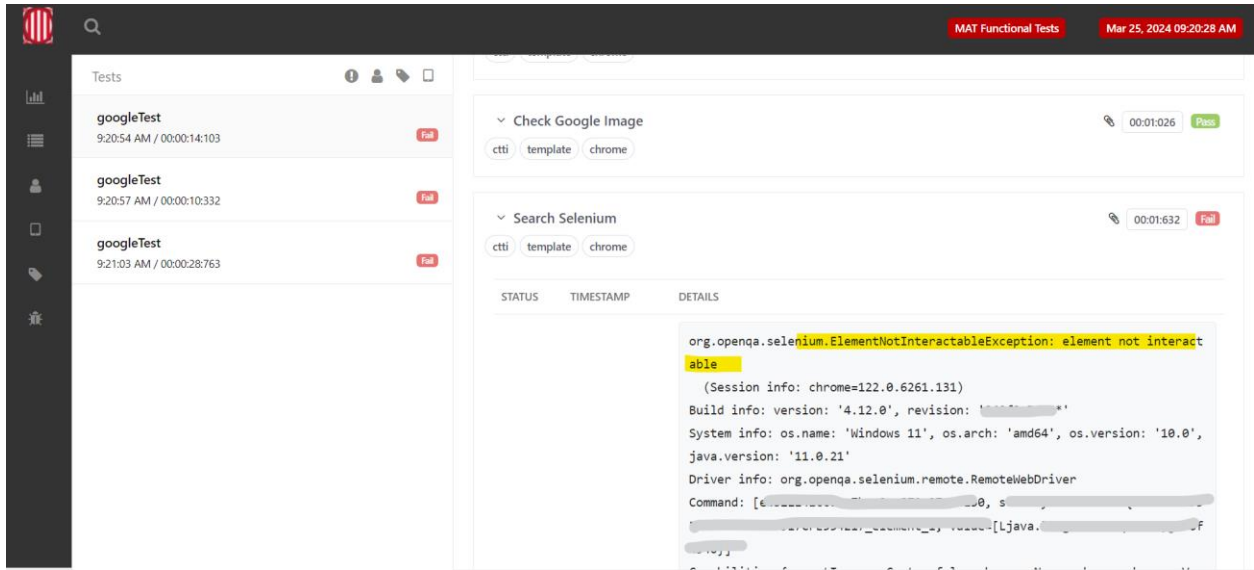
Dins del detall del pas, es pot veure el motiu de l'error.



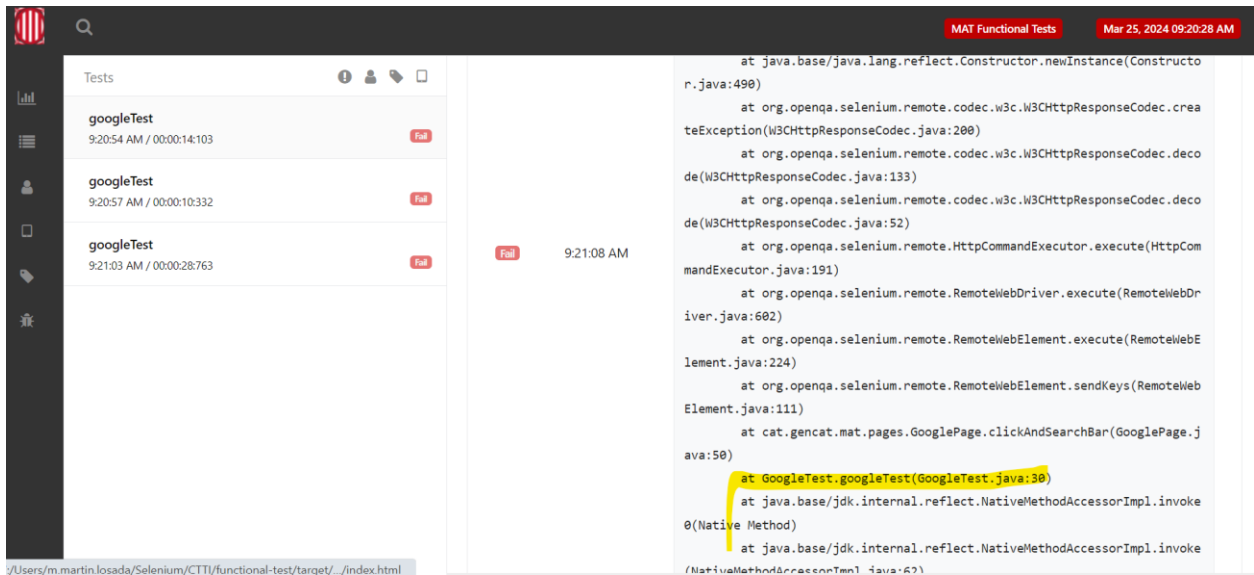
8.1. Anàlisi de l'error

L'error ocorre en el pas de "Search Selenium".

Desplegant el detall de l'error, es pot veure que no es pot interactuar amb l'element, i que es troba en la línia 30 del test `GoogleTest.java`.



The screenshot shows the MAT Functional Tests interface. On the left, a list of tests is displayed, all marked as 'Fail'. The 'Search Selenium' test is selected, and its details are shown on the right. The error message is: `org.openqa.selenium.ElementNotInteractableException: element not interactable`. The details section shows the session info, build info, system info, driver info, and the command executed. The error occurs at the `clickAndSearchBar` method in `GooglePage.java`.



The screenshot shows the MAT Functional Tests interface with the 'Search Selenium' test selected. The stack trace is displayed on the right, showing the sequence of method calls leading to the failure. The error occurs at the `clickAndSearchBar` method in `GooglePage.java`, which is called by the `googleTest` method in `GoogleTest.java`.

És a dir, l'error ocorre en l'anomenada al mètode `clickAndSearchBar()`.

```

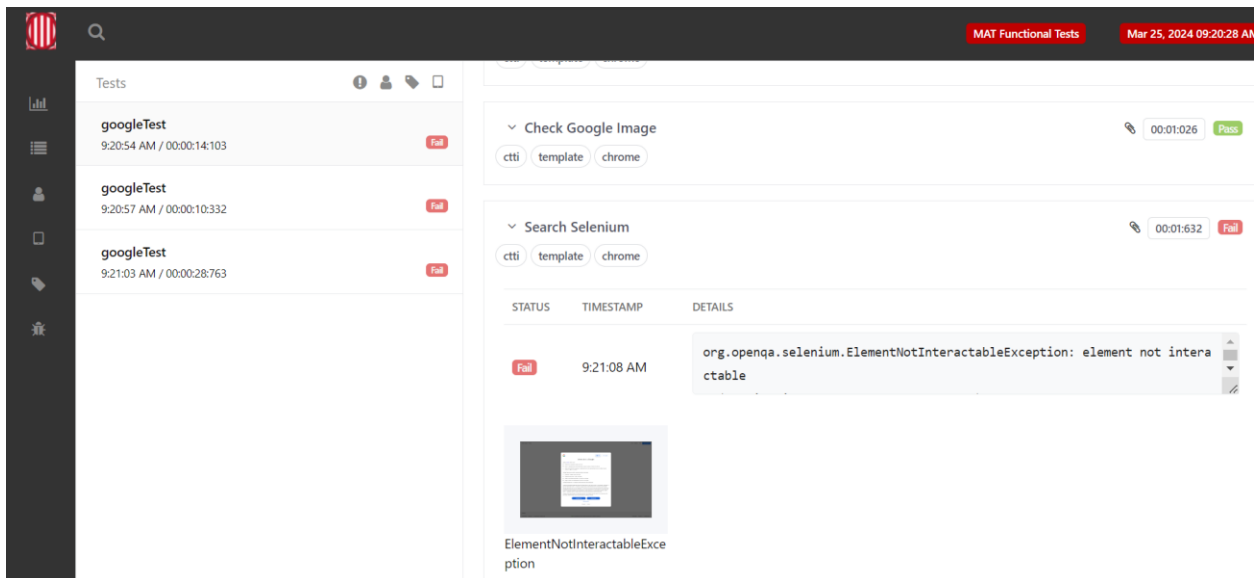
src > test > java > J GoogleTest.java > GoogleTest > googleTest(String, Method)
10  public final class GoogleTest extends BaseTest {
12      public void googleTest(String browser, Method method) throws Throwable {
24          Utils.step("Check Google Image");
25          GooglePage.checkImageGoogleHome();
26          Utils.screenshot("Check Google Image");
27
28          Utils.step("Search Selenium");
29          String somethingToSearch = "Selenium";
30          GooglePage.clickAndSearchBar(somethingToSearch);
31          Utils.screenshot("Search Selenium");
32
    
```

Tal com s'explica en l'apartat 6.1, aquest mètode busca un element i el compara amb un text.

Tipus de mètodes	Mètodes	Explicació
Clic i escriure un text	<code>clickAndSearchBar()</code>	En la barra de cerca, escriure un text.

Si no es troba l'element o no es pot interactuar amb ell per algun motiu, el test falla.

En el index.html també es pot veure la captura de pantalla del pas, per a entendre millor el motiu de la fallada.



The screenshot shows the MAT Functional Tests interface. On the left, a sidebar lists three test runs, all marked as 'Fail'. The main panel displays details for the 'Search Selenium' test, which failed at 9:21:08 AM. The error message is 'org.openqa.selenium.ElementNotInteractableException: element not interactable'. Below the error message, there is a small screenshot of the web page where the error occurred, labeled 'ElementNotInteractableException'.

És per això, que en executar el mètode `clickAndSearchBar()` no pot veure l'element correctament.

Per a solucionar aquest error, s'hauria de crear un mètode que Accepti o Rebutgi les cookies en cas que es desplegui el pop up.

Seguint els passos realitzats en l'apartat 6 de creació de casos de proves, primer s'ha de definir el POM relacionat amb el pop up.

46

The screenshot shows a Google cookie consent banner. The banner text is in Spanish, explaining that Google uses cookies to enhance site navigation, analyze site usage, and assist in our marketing efforts. It offers two main options: 'Rechazar todo' (Reject all) and 'Aceptar todo' (Accept all). Below these are links for 'Más opciones' (More options), 'Privacidad' (Privacy), and 'Términos' (Terms). A web inspector is overlaid on the banner, highlighting the 'Rechazar todo' button with a red circle. The HTML structure shows a flex container with the buttons and links.

```
public final class GooglePage {
    //Incloure aquí tots els elements localitzadors de la pàgina
    //google.com
    private static final By rechazarCookiesBtn = By.id(id:"W0wltc");
    private static final By imageGoogleHome = By.className(className:"lnXdpd");
    private static final By searchBar = By.cssSelector(cssSelector:"#APjFqb");
    private static final By dropdownSeleniumOption = By.xpath(xpathExpression:"//span[contains(., 'selenium')]");
}
```

s'afegeix un mètode que primer comprovi si existeix el pop up de les cookies, i en cas que existeixi, faci clic a Rebutjar.

The screenshot shows an IDE with the GooglePage.java file open. The file is part of a project named 'FUNCTIONAL-TEST'. The code defines a class GooglePage with several static methods. A new method, checkAndClickRechazarCookiesBtn, has been added. This method checks if the 'Rechazar todo' button exists and clicks it. It includes a try-catch block to handle potential exceptions and a sleep statement to ensure the button is clickable before clicking.

```
src > test > java > Pages > J GooglePage.java > GooglePage
14 public final class GooglePage {
26
27 //Mètodes relacionats amb la pàgina de Google
28 public static void checkAndClickRechazarCookiesBtn( ) {
29 //Guardar tots els elements que contingui l'id W0wltc (Reject btn)
30 List<WebElement> element = BaseTest.getDriver().findElements(By.id(id:"W0wltc"));
31
32 //Si algun element existeix, fer clic en el primer (get(0)). En aquest cas només apareixeria un element.
33 if (!element.isEmpty()) {
34 | element.get(index:0).click();
35 }
36
37 try {
38 | Thread.sleep(millis:5000); // Pausa de 2 segundos (ajusta el tiempo según sea necesario)
39 | catch (InterruptedException e) {
40 | e.printStackTrace();
41 }
42
43 Utils.annotate(Utils.LogLevel.PASS, "Cookies Rechazadas");
44 }
```

Per tant, ara hi ha aquests mètodes:

Tipus de mètodes	Mètodes	Explicació
Comprovar element i si existeix, realitzar una acció.	<code>checkAndClickRechazarCookiesBtn()</code>	Comprovar pop up i si existeix fer clic.
Comprovar element	<code>checkImageGoogleHome()</code>	Cerca d'un element de la web.
Comprovar text	<code>getSeleniumOption()</code>	Cerca d'un element de la web i comprovar el seu text.
Clic i escriure un text	<code>clickAndSearchBar()</code>	En la barra de cerca, escriure un text.
Clic	<code>ClickDropdownSeleniumOption()</code>	Després de buscar "Selenium", Clickar en l'opció de Selenium del resultat mostrat
Buscar un element concret i clicar	<code>ClickElementByHref()</code>	Primer es fa una cerca de tots els elements HTML trobats en la pàgina. Després es busca el element que contingui el href que es necessita per al test i finalment es fa clic en ell.

D'aquesta manera, la classe GooglePage.java queda d'aquesta manera:



```
public final class GooglePage {
    //Incloure aquí tots els elements localitzadors de la pàgina
    //google.com
    private static final By rechazarCookiesBtn        = By.id("W0wltc");
    private static final By imageGoogleHome           = By.className("lnXdpd");
    private static final By searchBar                 = By.cssSelector("#APjFqb");
    private static final By dropdownSeleniumOption    =
By.xpath("//span[contains(., 'selenium')]");

    //Mètodes relacionats amb la pàgina de Google

    public static void checkAndClickRechazarCookiesBtn( ) {
        //Guardar tots els elements que contingui l'id W0wltc (Reject btn)
        List<WebElement> element = BaseTest.getDriver().findElements(By.id("W0wltc"));

        //Si algun element existeix, fer clic en el primer (get(0)). En aquest cas
        només apareixeria un element.
        if (!element.isEmpty()) {
            element.get(0).click();
        }

        try {
            Thread.sleep(5000); // Pausa de 5 segons
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        Utils.anoate(Utils.LogLevel.PASS, "Cookies Rechazadas");
    }

    public static void checkImageGoogleHome( ) {
        BaseTest.getDriver().findElements(imageGoogleHome);
        Utils.anoate(Utils.LogLevel.PASS, "Google Image Home");
    }
}
```



```
public static String clickAndSearchBar (String somethingToSearch) {  
    Utils.getElement(searchBar).sendKeys(somethingToSearch);  
    Utils.anoate(Utils.LogLevel.PASS, "Search Bar is clickable and available to send  
keys");  
    return somethingToSearch;  
}  
  
public static void getSeleniumOption() {  
    MatcherAssert.assertThat(Utils.getElement(dropdownSeleniumOption, 20).getText(),  
Matchers.equalTo("Selenium"));  
    Utils.anoate(Utils.LogLevel.PASS, "Text is correct");  
}  
  
public static void clickDropdownSeleniumOption() {  
    Utils.getElement(dropdownSeleniumOption).click();  
    Utils.anoate(Utils.LogLevel.PASS, "Selenium option is clickable");  
}  
  
public static void clickElementByHref(String href) {  
    //Elements de la cerca trobats en Google  
    List<WebElement> elements =  
BaseTest.getDriver().findElements(By.cssSelector("span[jscontroller='msmzHf']"));  
  
    //Iteració  
    for (WebElement element : elements) {  
        WebElement anchor = element.findElement(By.tagName("a")); //Buscar <a>  
        String elementHref = anchor.getAttribute("href"); //Obtenir valor de href en <a>  
  
        //Clic quan es encuetre href  
        if (elementHref.equals(href)) {  
            anchor.click();  
            break;  
        }  
    }  
}
```

8.1.1.2. *Modificar el Test*

Després d'iniciar el navegador, i abans de cridar al mètode de `checkImageGoogleHome()`, cal cridar al mètode de `checkAndClickRechazarCookiesBtn()`.



```
public final class GoogleTest extends BaseTest {
    @Test @Parameters(value={"browser"})
    public void googleTest(String browser, Method method) throws Throwable {
        try {
            Utils.step("Enter website");
            Utils.gotoApp();
            Utils.maximize();
            Utils.anoate(Utils.LogLevel.PASS, "Website's homepage is accessible");
            Utils.screenshot("Homepage");

            Utils.step("Rechazar Cookies");
            GooglePage.checkAndClickRechazarCookiesBtn();
            Utils.screenshot("Rechazar Cookies");

            Utils.step("Check Google Image");
            GooglePage.checkImageGoogleHome();
            Utils.screenshot("Check Google Image");

            Utils.step("Search Selenium");
            String somethingToSearch = "Selenium";
            GooglePage.clickAndSearchBar(somethingToSearch);
            Utils.screenshot("Search Selenium");

            Utils.step("Click Selenium displayed");
            String label = "selenium";
            GooglePage.clickDropdownOption(label);
            Utils.screenshot("Click Selenium displayed");

            Utils.step("Click https://www.selenium.dev/ ");
            String href = "https://www.selenium.dev/";
            GooglePage.clickElementByHref(href);
            Utils.screenshot("Click https://www.selenium.dev/");

            Utils.endTestAsOK(browser, method);
        }
        catch (Exception | AssertionError e) { Utils.endTestAsKO(browser, method,
e); }
    }
}
```

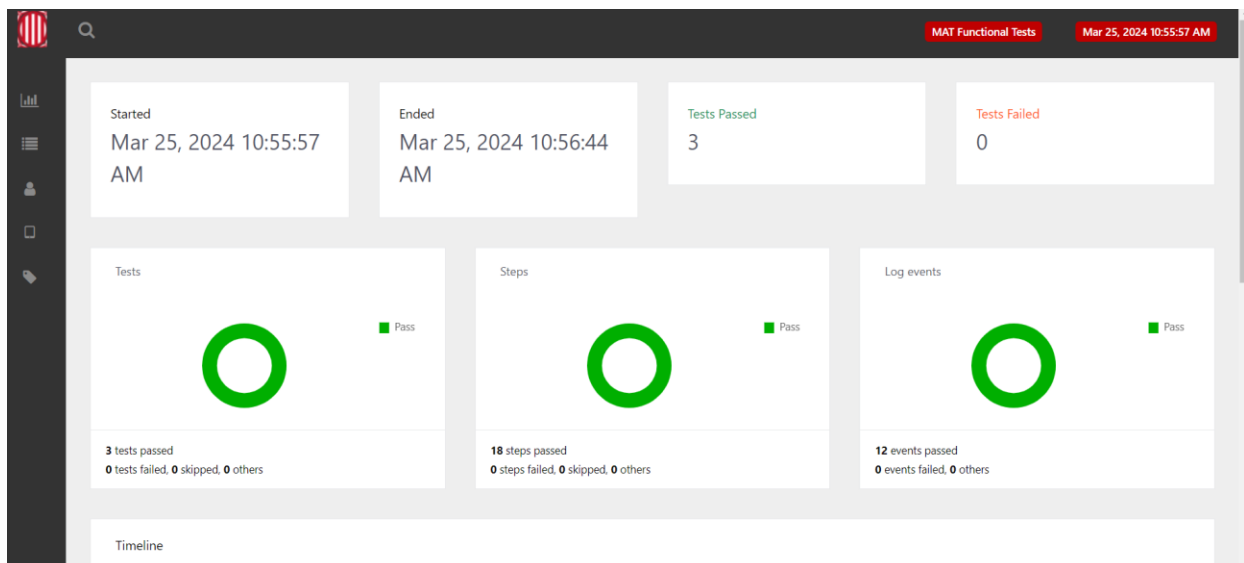
8.1.1.3. Executar i comprovar el report HTML.

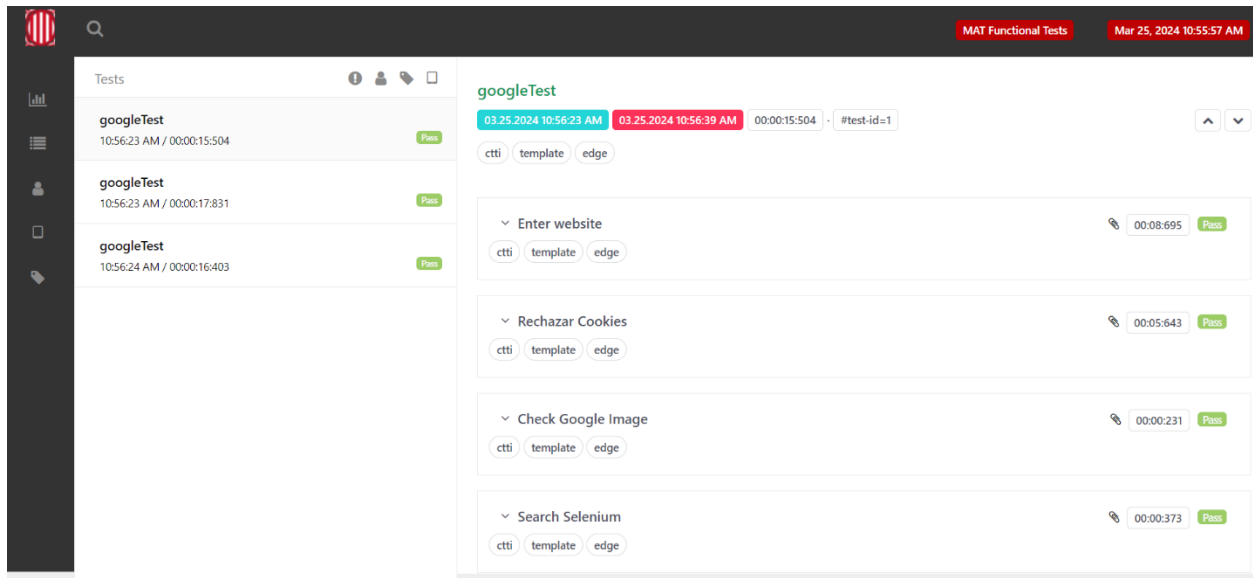
Executar de nou les proves per a comprovar que s'ha solucionat l'error.

```
mvn clean test --settings maven/settings.xml -  
Dselenium_url="http://localhost:4444/wd/hub" -Dapp_url="https://www.google.com"
```

```
PROBLEMS 14 OUTPUT DEBUG CONSOLE TERMINAL PORTS [E]  
  
[INFO] -----  
[INFO] googleTest :: {edge} :: PASSED  
[INFO] googleTest :: {firefox} :: PASSED  
[INFO] googleTest :: {chrome} :: PASSED  
[INFO] -----  
[INFO] Report written to: `target/report/index.html`  
[INFO] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 50.305 s - in TestSuite  
[INFO] Results:  
[INFO] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0  
[INFO] -----  
[INFO] BUILD SUCCESS  
[INFO] -----  
[INFO] Total time: 01:05 min  
[INFO] Finished at: 2024-03-25T10:56:45+01:00  
[INFO] -----  
PS C:\Users\... \Selenium\CTTI\functional-test> [ ]
```

En el index.html, ja es veuen totes les proves passades.





9. Integració amb Xray

9.1 Abans de començar

Abans de començar, assegura't que en els vostres projectes tingueu aplicada la configuració bàsica.

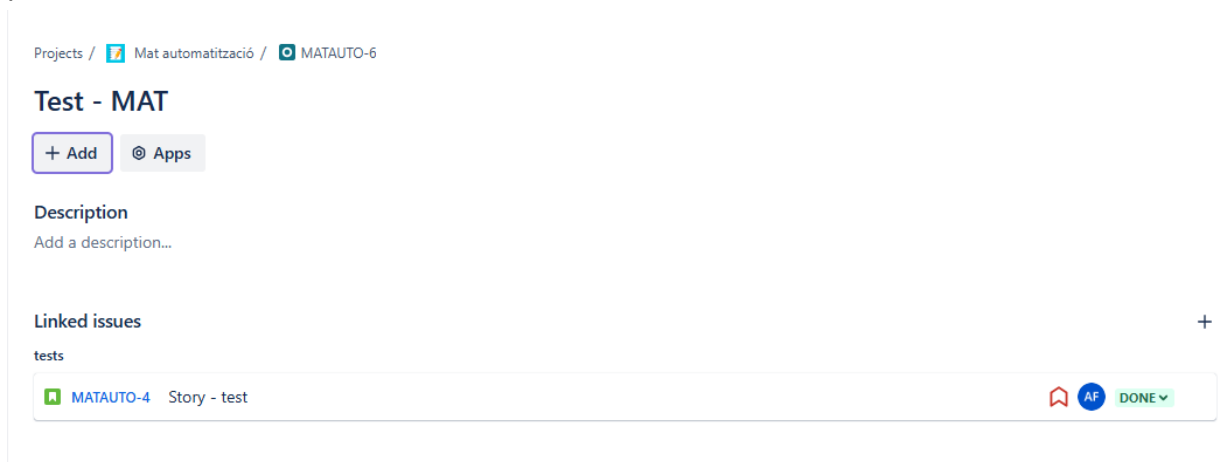
Per comprovar si la configuració bàsica es troba aplicada hauries de veure que:

- Pots crear issues específiques d'Xray com: Prova, Conjunt de Proves, Pla de Proves i Execució de Proves.
- En els teus Requisits (tipus de issue Història o qualsevol altre que hagi configurat com a requisit) has de veure el panell "Cobertura de Proves".

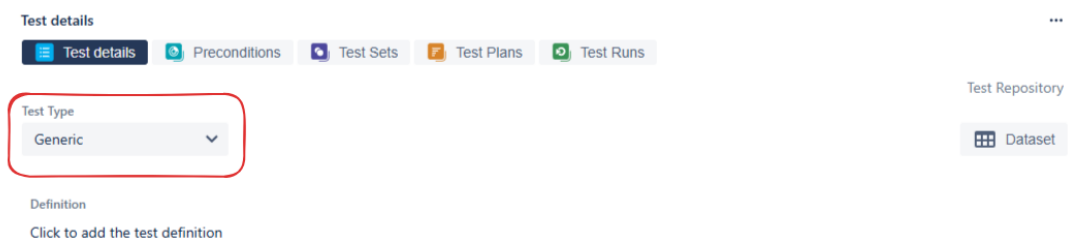
9.2 Crear una Prova

Per tal de crear proves **automatitzades**, hem de crear unes Issues sobre el projecte.

1. Fes clic al botó **Crear** o **Crea Prova** des del "requisit/història d'usuari que estem provant".



2. Després que s'hagi creat el issue de tipus Test , **edita'l** i a la pestanya **Detalls de la Prova** selecciona el tipus de prova **Generic**.



3. Confirma que la teva Prova està enllaçada amb un requisit. Per analitzar la cobertura, has d'enllaçar les teves proves amb els requisits. Vés a la pestanya Enllaçar issues o comprova els **issues enllaçats**.



Projects / Mat automatització / MATAUTO-6

Test - MAT

[+ Add](#) [@ Apps](#)

Description
Add a description...

Linked issues +

tests

MATAUTO-4 Story - test AF DONE

Test details ...

[Test details](#) [Preconditions](#) [Test Sets](#) [Test Plans](#) [Test Runs](#)

Test Type
Generic ▼

Test Repository
[Dataset](#)

4. Des del teu codi de proves, hauràs d'afegir el Test i el requisit/història d'usuari que està provant en els espais mostrats a continuació, i en cada bloc de proves que es creïn.

```
QualityGenCatTest.java X
3632.00-mat-functional-tests-master > src > test > java > QualityGenCatTest.java
1  import cat.gencat.mat.Utills;
2  import cat.gencat.mat.BaseTest;
3  import java.lang.reflect.Method;
4  import org.testng.annotations.Test;
5  import org.testng.annotations.Parameters;
6  import cat.gencat.mat.pages.QualityGenCatPage;
7  import app.getxray.xray.testng.annotations.XrayTest;
8  import app.getxray.xray.testng.annotations.Requirement;
9
10 public final class QualityGenCatTest extends BaseTest {
11
12     @Test @Parameters(value = {"browser"})
13     @XrayTest(key = "DEVSECOPS2-769")
14     @Requirement(key = "DEVSECOPS2-768")
```

5. És important afegir al pom.xml l'identificador del projecte Jira al qual es pertany, ja que basant-se en aquest projecte de Jira i afegint la Història d'Usuari i el Test Case vistos en el pas anterior, es podran vincular allí els resultats de les proves realitzades.

```
<plugin>
  <groupId>app.getxray</groupId>
  <artifactId>xray-maven-plugin</artifactId>
  <version>0.8.0</version>
  <configuration>
    <cloud>true</cloud>
    <projectKey>JIRA-PROJECT-ID</projectKey>
    <reportFormat>testng</reportFormat>
    <reportFile>target/surefire-reports/testng-results.xml</reportFile>
  </configuration>
</plugin>
```

10. Integració amb GitHub Enterprise

10.1 Anar a la secció d'Actions a GitHub Enterprise

En la qual es troba l'arxiu .yaml '**Test Reusable CD**'



El qual es podrà executar manualment des del botó que fa referència a 'Run Workflow'



On s'hauran d'indicar les dades requerides -marcades amb un asterisc-, com per exemple, la branca que es desitja executar, l'entorn, la URL de l'aplicació... I fins i tot dades opcionals com el projecte de Jira i Test Plan, i fins i tot el número de Pull Request.

Use workflow from

Branch: master ▾

Environments to test in *

Produccio ▾

Application URL *

https://qualitat.solucions.gencat.cat

Umbral de pruebas *

20

Jira Project

Jira Issue Test Plan

GitHub Pull Request Number

Run workflow

En cas d'haver indicat el projecte de Jira i Test plan, podrà veure el resultat de l'execució en les respectives tasques indicades anteriorment, així com el resultat de l'execució en l'històric d'execucions de GitHub Actions.

