*Prueba técnica Automatizador*

# Escenarios de pruebas

La empresa EUNO, requiere que por medio de una pruebas exploratorias sean presentados los escenarios y casos de pruebas de su portal web inicialmente del login y registro.

**URL**: <https://euno.lms.doctustest.com/>

**Criterios de aceptación**:

* Se debe entregar la documentación de los diferentes escenarios con cada uno de los casos de pruebas.

# Automatización

Automatice por lo menos dos de los escenarios planteados en el primer punto, uno del login y otro del registro.

**Criterios de aceptación**:

* Se debe entregar la solución o código fuente donde se encuentre la automatización de estos escenarios.
* Se debe entregar los resultados generados al correr la automatización de los escenarios.
* Se debe entregar la documentación del desarrollo realizado, donde se explique la arquitectura del proyecto y qué patrones de diseño fueron utilizados.
* Se debe entregar un manual que indique cómo ejecutar el proyecto.

*Plan de pruebas*

*Alcance:*

*Se probará la aplicación en el inicio de sesión con diferentes usuarios validando el usuario y la contraseña ingresada, adicional se probará el registro de usuario validando los tipos de datos de cada campo y la validación de los formatos.*

*Estrategia:*

Se probará el inicio de sesión principalmente con usuario y contraseñas validos validando el correcto inicio de sesión, adicional a esto se probará ingresando usuarios y contraseña incorrectos y que se muestren los respectivos mensajes de alerta a los usuarios, después esto se validará ingresando al menos un valor correcto y por último sin valores y mostrando el respectivo mensaje al usuario.

Se probará el registro de usuario principalmente ingresando todos los campos correctos y su respectivo registro exitoso, posterior se probará el registro ingresando correos electrónicos inválidos, por último, se probará realizando el registro ingresando solo algunos campos y ningún campo el cual se debería mostrar el respectivo mensaje al usuario.

Escenarios inicio de sesión:

* Iniciar sesión con usuario y contraseña validos
* Iniciar sesión con usuario y contraseña inválidos
* Iniciar sesión con usuario valido y contraseña inválidos
* Iniciar sesión con usuario inválidos y contraseña valido
* Iniciar sesión ingresando solo usuario
* Iniciar sesión ingresando solo contraseña
* Iniciar sesión sin ingresar ningún campo

Supuestos:

* Suponiendo que el email es el usuario
* Suponiendo que el capo de contraseña no tiene límite de longitud, no se hicieron pruebas de longitud sobre estos campos

Escenarios registro de usuarios:

* Registro con todos los campos validos y posterior inicio de sesión con usuario y contraseña diligenciados
* Registro ingresando letras en el campo de documento
* Registro ingresando numero en los campos de nombre y apellido
* Registro con email invalido
* Registro con ningún campo
* Registro con email en formato invalido

Supuestos:

* Suponiendo que el email debe estar en alguna db para poder hacer el registro exitoso
* Suponiendo que el capo de contraseña no tiene limite de longitud, no se hicieron pruebas de longitud sobre estos campos

Casos de prueba: Inicio de sesión

Precondiciones: se debe tener las credenciales de un usuario ya registrado para poder iniciar sesión exitosamente.

* Iniciar sesión con usuario y contraseña validos

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La pagina debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Ingresar usuario y contraseña validos |  |  |
| 3 | Clic en iniciar sesión | Se debe iniciar sesión sin problema | True |

* Iniciar sesión con usuario y contraseña inválidos

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Ingresar usuario y contraseña inválidos |  |  |
| 3 | Clic en iniciar sesión | NO se debe iniciar sesión, debe mostrar mensaje de usuario y contraseña invalido | True |

* Iniciar sesión con usuario valido y contraseña inválidos

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Ingresar usuario valido |  |  |
| 3 | Ingresar contraseña invalida |  |  |
| 4 | Clic en iniciar sesión | NO se debe iniciar sesión, debe mostrar mensaje de usuario y contraseña invalido | True |

* Iniciar sesión con usuario inválidos y contraseña valido

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Ingresar usuario invalido |  |  |
| 3 | Ingresar contraseña valida |  |  |
| 4 | Clic en iniciar sesión | NO se debe iniciar sesión, debe mostrar mensaje de usuario y contraseña invalido | True |

* Iniciar sesión ingresando solo usuario

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Ingresar usuario valido |  |  |
| 3 | Clic en iniciar sesión | NO se debe iniciar sesión, debe mostrar mensaje indicando falta del campo contraseña | True |

* Iniciar sesión ingresando solo contraseña

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Ingresar contraseña valido |  |  |
| 3 | Clic en iniciar sesión | NO se debe iniciar sesión, debe mostrar mensaje indicando falta del campo usuario | True |

* Iniciar sesión sin ingresar ningún campo

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 3 | Clic en iniciar sesión | NO se debe iniciar sesión, debe mostrar mensaje indicando falta del campo contraseña | True |

Casos de prueba: Registro de usuario

Precondiciones: se debe tener el correo electrónico de un usuario valido en la DB relacionada.

* Registro con todos los campos válidos y posterior inicio de sesión con usuario y contraseña diligenciados

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Clic en registro de usuario | Se debe direccionar a la pantalla de registro | True |
| 3 | Ingresar información personal de usuario valida |  |  |
| 4 | Clic en registrar | Se debe mostrar mensaje de registro exitoso | True |
| 5 | Iniciar sesión con usuario y contraseña diligenciados | Se debe iniciar sesión sin problema | True |

* Registro ingresando letras en el campo de documento

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Clic en registro de usuario | Se debe direccionar a la pantalla de registro | True |
| 3 | Ingresar información personal de usuario valida |  |  |
| 5 | Ingresar letras en el campo documento |  |  |
| 4 | Clic en registrar | Se debe mostrar mensaje de registro fallido por valores en campo documento invalido | True |

* Registro ingresando números en los campos de nombre y apellido

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Clic en registro de usuario | Se debe direccionar a la pantalla de registro | True |
| 3 | Ingresar información personal de usuario valida |  |  |
| 5 | Ingresar números en los campos de nombre y apellidos |  |  |
| 4 | Clic en registrar | Se debe mostrar mensaje de registro fallido por valores en los campos nombre y apellidos invalido | True |

* Registro con email invalido

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Clic en registro de usuario | Se debe direccionar a la pantalla de registro | True |
| 3 | Ingresar información personal de usuario valida |  |  |
| 5 | Ingresar email no existente en DB |  |  |
| 4 | Clic en registrar | Se debe mostrar mensaje de registro fallido por que el email no esta registrado | True |

* Registro con ningún campo

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Clic en registro de usuario | Se debe direccionar a la pantalla de registro | True |
| 4 | Clic en registrar | Se debe mostrar mensaje de registro fallido porque todos los campos son requeridos | True |

* Registro con email en formato invalido

|  |  |  |  |
| --- | --- | --- | --- |
| Pasos | Descripción | Resultado esperado | Resultado obtenido |
| 1 | Ingresar a Home Page de la pagina | La página debe cargar completamente mostrando los campos de inicio de sesión | True |
| 2 | Clic en registro de usuario | Se debe direccionar a la pantalla de registro | True |
| 3 | Ingresar información personal de usuario valida |  |  |
| 5 | Ingresar email sin @ |  |  |
| 4 | Clic en registrar | Se debe mostrar mensaje de registro fallido porque el email no tiene el formato correcto | True |

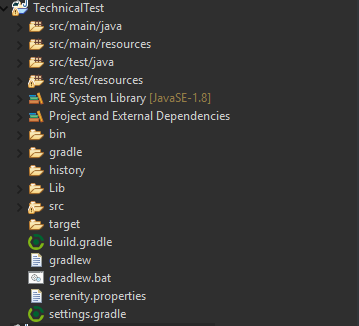
**Patrón de diseño:**

El patrón de diseño utilizado fue Screenplay implementando metodología DBB.

**Arquitectura de la automatización:**

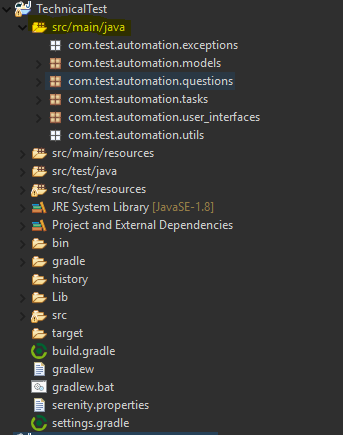
La arquitectura de la automatización esta realizada de la siguiente manera:

El proyecto está construido en el ide eclipse lo cual lo hace fácil de usar y el proyecto se puede abrir desde otros ide como IntelliJ, en este proyecto se utilizo Gradle el cual es un compilador de código abierto pero también es posible construir o crear el mismo proyecto sobre Maven el cual se puede decir que es similar a gradle par al compilación de código. En este proyecto se implemento serenity BDD con screenplay y cucumber



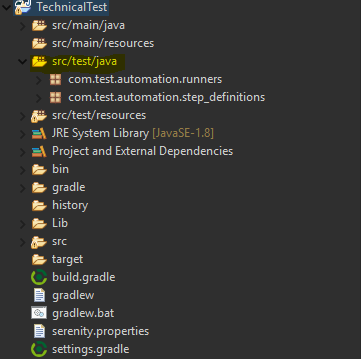
El proyecto esta construido de la siguiente manera. En la ruta de capetas src/main/java tenemos los siguientes contenedores.

* Exceptions: en este contenedor se crean las clases las cuales van a controlas los fallos en el sistema el cual puede ser un problema de ambiente o algún otro fallo cual no sea fallo de la prueba
* Models: en este contenedor van las clases tipo model las cuales contienen la información, obteniendo y enviando la información necesaria para la prueba.
* Questions: en este contenedor van las clases tipo questions las cuales se usan para realizar la validación final de la prueba, sin estas clases la validación no estaría completa y la ejecución diría que termino sin saber el estatus de la prueba.
* Tasks: en este contenedor se tienen las clases tipo tasks las cuales llevan a cabo el paso a paso de la prueba ingresando información, dándole clic a los elementos definidos.
* User\_interfaces: en este contenedor se tienen las clases tipo pages las cuales definen cada página de la aplicación, donde se identifican y definen los elementos de cada página.
* Utils: en este contenedor se tienen las clases de utilidades para la aplicación tales como conexiones a DB, cargar y lectura de archivo en otras utilidades las cuales se usan para hacer la prueba

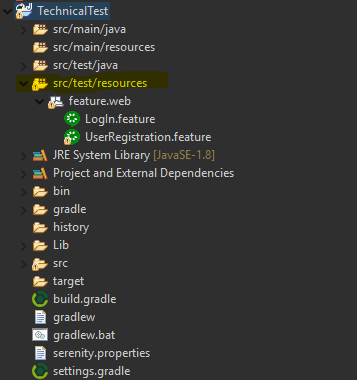


En la ruta de capetas src/test/java tenemos los siguientes contenedores.

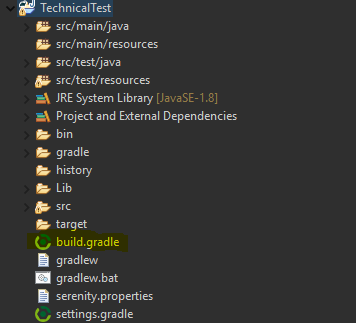
* Runners: en este contenedor se tienen las clases las cuales ejecutan la prueba de cada Feature definido.
* Step\_definitions: en este contenedor están las clases las cuales tienen los escenarios definidos en los archivos features

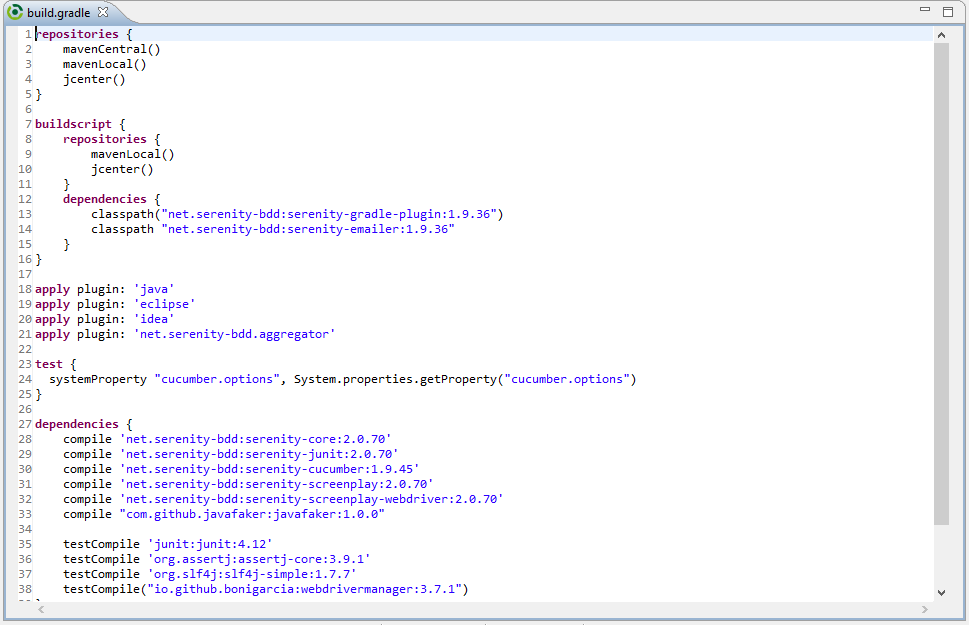


En la ruta src/test/resources se tienen los contenedores con los archivo .features los cuales son de cucumber y es donde se definen los escenarios de pruebas con la notación Gherkin que definen el lenguaje de negocio en escenarios para hacer las pruebas.

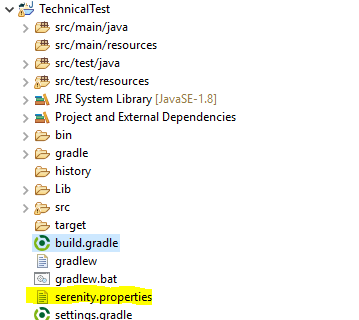


En este proyecto se tiene la configuración de Gradle en el archivo **build.gradle** en el cual se definen los repositorios desde los cuales se vana a descargar los dependencias (librerías), las cuales también están definidas en este archivo, las que se usan para llagar la cabo la codificación de la automatización y en este archivo está definida también la configuración y plugins para la ejecución exitosa.

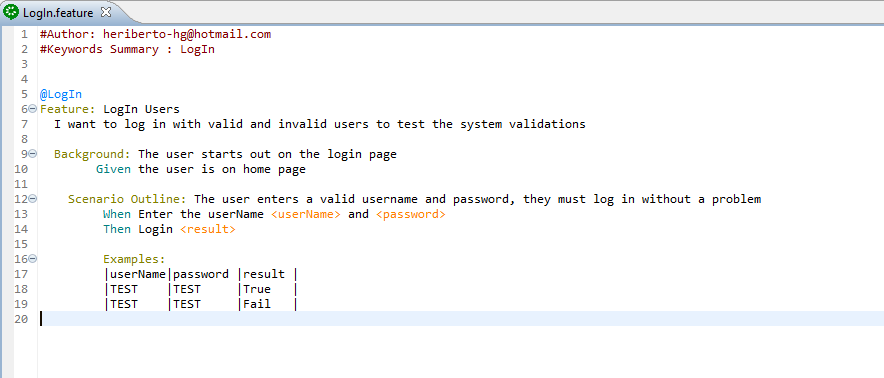




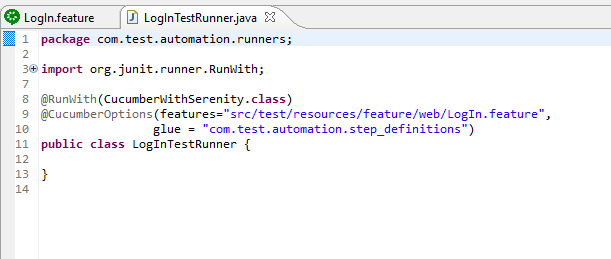
Se tiene también el archivo serenity.properties el cual define alguna configuración para la ejecucion como lo seria el driver con el cual se va a ejecutar la prueba, en este caso se va a ejecutar con Google Chrome se usa chromedriver y por estar en un equipo Windows se define en este archivo la ruta donde está el driver el cual es en la carpeta Lib, en el caso de estar en un Linux no es necesario definir la ruta en este archivo.



Archivo Feature: en este archivo se define los escenarios de pruebas en un lenguaje de negocio definido anteriormente como Gherkin. En este archivo están definidos los comandos Given, When y Then los cuales definen el Given como las precondiciones que se deben llevar a cabo para hacer la prueba, el When como la prueba que se va realizar y el Then como el resultado que se debería obtener. En este archivos hay varias maneras de definir un escenarios en este caso se tiene un Scenario Outline el cual tiene una tabla con datos el cual define cada fila como una iteración de la prueba. Pero también se pueden tener escenarios sin tablas ejecutando una sola ves la prueba con los datos definidos en la descripción.

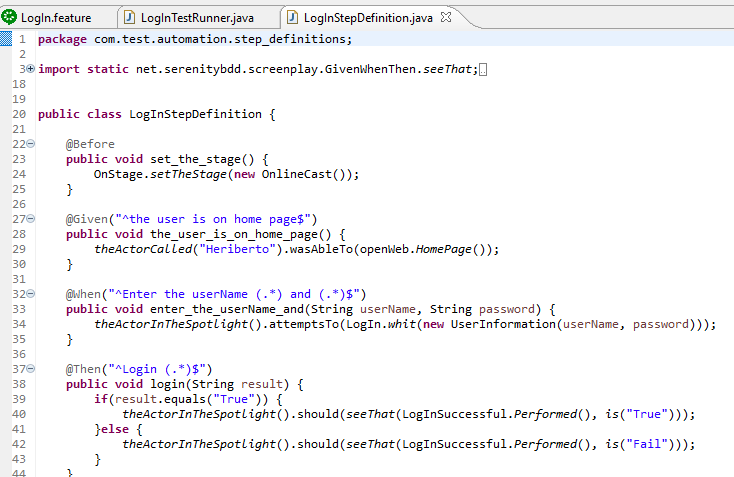


Runner: en esta clase de tipo runner se define el archivo .feature el cual se va a ejecutar y la ruta donde se buscan los pasos a ejecutar.



Step\_definition: en este clase se definen los pasos codificados del archivo .feature, se tienen definidos los métodos con las siguientes etiquetas.

* @Before: define como el método que se va a ejecutar antes de comenzar la prueba y es donde se especifican los actores y se les da los permisos que se los actores pueden hacer durante la prueba. En este momento se está utilizando un metodo avanzado el cual tienen embebido la definición de los permisos a los actores, los buscadores a usar.
* @Given: define como las precondiciones que se deben llevar a cabo para hacer la prueba especifica. En este caso se está usando un método avanzado el cual define el nombre del actor y la tarea que fue capas de realizar.
* @When: define como la tarea que va a intentar realizar, en este caso se usa un modelo para especificar el usuario y contraseña que se enviar a la tarea que se van a usar para hacer la prueba de login.
* @Then: define como la validación que se va a realizar después de llevado a cabo tarea. En este caso se validara el mensaje que se le muestra al usuario al intentar hacer login con usuario invalido ya que durante las pruebas exploratorias no se logró iniciar sesión exitosamente.

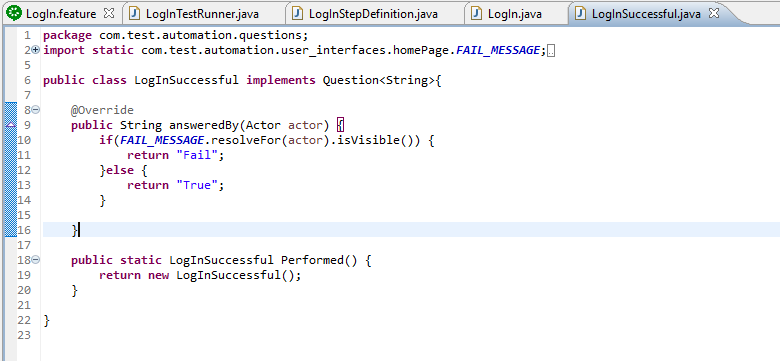


Tasks: en esta clase implementa la clase Task la cual trae con si unos atributos para codificar la prueba. Se define un modelo el cual va a contener la información enviada desde el modelo en el step\_definition.

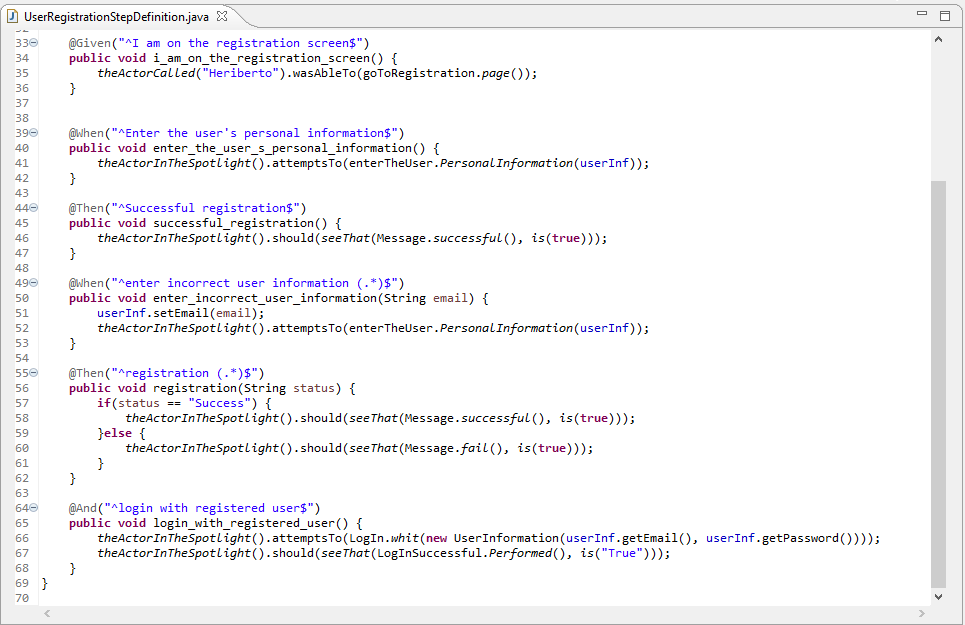
Se define un método performAs el cual es donde se codifica cada paso que se va hacer, en este método el actor intenta ingresar los valores de usuario y contraseña en los campos respectivos y por último dar clic en iniciar sesión. El ultimo método define como la clase y datos usados para la tarea.



Question: en esta clase se implementa la clase question con la cual se pueden hacer validaciones. En este caso se esta validando la visibilidad de un elemento de mensaje ya que si el elemento esta visible significa que el inicio de sesión fallo y si el mensaje no se muestra es porque se inició sesión sin problema.



Se pueden tener dos validaciones como lo es el flujo del registro de usuario donde después del registro se validan dos mensajes diferentes, el primero el registro exitoso y el segundo el registro fallido y adicional después del registro se valida el inicio de sesión con el usuario recientemente registrado.



**Ejecución de la prueba**

Para ejecutar la prueba por consola lo primero estar posicionado en la carpeta del proyecto.



Después para ejecutar se pueden usar los siguientes comandos.

* Para ejecutar la prueba completa con todos sus escenarios se debe lanzar el comando: **gradle clean test**



* Para ejecutar la prueba de un solo feature se lanzar el comando **gradle test -Dtest.single=UserRegistration** definiendo al final el nombre del feature a ejecutar.



* Para ejecutar un solo escenario especifico se debe lanzar el comando **gradle clean -Dcucumber.options="--tags @RegistrySucces" test –info** definiendo después del @ el nombre del tag (sobrenombre) dado al escenario



**Notas importante**: en caso de estar en un equipo windows para la correcta ejecución se debe descargar el chromedriver del buscador Google desde la pagina <https://chromedriver.chromium.org/downloads> con la misma versión del buscador instalado en el equipo y ponerlo en la carpeta /Lib en el proyecto. En caso de estar en un Linux no es necesario descargar el controlador ya que este lo reconoce automáticamente, solo se debe comentar en el archivo serenity.properties la línea numero 2 ya que esta es la que define donde se va a buscar el controlador.

Saludos!