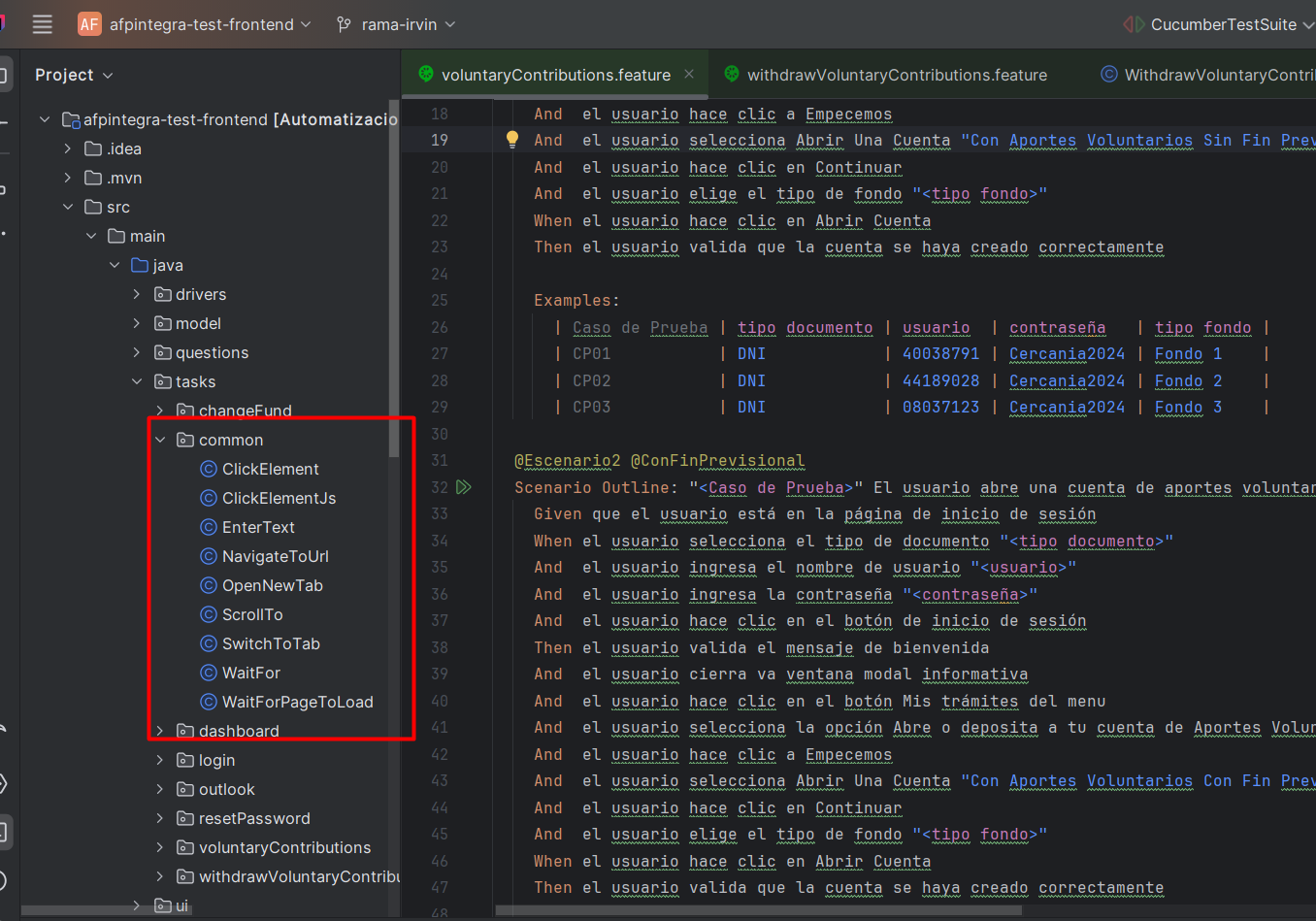
Comun

1-click element

package tasks.common;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Click;  
import net.serenitybdd.screenplay.actions.Scroll;  
import net.serenitybdd.screenplay.targets.Target;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
  
import static net.serenitybdd.screenplay.matchers.WebElementStateMatchers.*isClickable*;  
import static net.serenitybdd.screenplay.matchers.WebElementStateMatchers.*isVisible*;  
  
public class ClickElement implements Task {  
  
 private final Target target;  
 private final boolean shouldScroll;  
 private final String[] values;  
  
 // Constructor que acepta un Target, un indicador de desplazamiento y un número variable de valores  
 public ClickElement(Target target, boolean shouldScroll, String... values) {  
 this.target = target;  
 this.shouldScroll = shouldScroll;  
 this.values = values;  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 // Determinar el Target final basado en si se proporcionaron valores o no  
 Target finalTarget = (values == null || values.length == 0) ? target : target.of(values);  
  
 // Desplazarse al elemento si shouldScroll es verdadero  
 if (shouldScroll) {  
 actor.attemptsTo(Scroll.*to*(finalTarget));  
 }  
  
 // Esperar hasta que el elemento sea visible y clickeable  
 actor.attemptsTo(  
 WaitUntil.*the*(finalTarget, *isVisible*()).forNoMoreThan(15).seconds(),  
 WaitUntil.*the*(finalTarget, *isClickable*()).forNoMoreThan(15).seconds(),  
 // Hacer clic en el elemento  
 Click.*on*(finalTarget)  
 );  
 }  
  
 // Mé*todo estático para facilitar la creación de la tarea*  
public static ClickElement on(Target target, boolean shouldScroll, String... values) {  
 return new ClickElement(target, shouldScroll, values);  
 }  
}

Como se usa

@When("el usuario hace clic en Confirmar Retiro")  
public void elUsuarioHaceClicEnConfirmarRetiro() {  
 *theActorInTheSpotlight*().attemptsTo(ClickElement.*on*(WithdrawVoluntaryContributionsPage.*CONFIRM\_WITHDRAW\_BUTTON*, true));  
}

2-Click Element JS

package tasks.common;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.targets.Target;  
import org.openqa.selenium.JavascriptExecutor;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.WebElement;  
  
import static net.serenitybdd.screenplay.Tasks.*instrumented*;  
  
public class ClickElementJs implements Task {  
 private final Target target;  
  
 public ClickElementJs(Target target) {  
 this.target = target;  
 }  
  
 public static ClickElementJs on(Target target) {  
 return *instrumented*(ClickElementJs.class, target);  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 WebDriver driver = net.serenitybdd.screenplay.abilities.BrowseTheWeb.*as*(actor).getDriver();  
 WebElement element = target.resolveFor(actor);  
  
 // Usar JavaScript para hacer clic en el elemento  
 ((JavascriptExecutor) driver).executeScript("arguments[0].click();", element);  
 }  
}

Como se usa

@And("el usuario hace clic en la opción Abrir una cuenta de Aportes Aporte Voluntarios")  
public void elUsuarioHaceClicEnLaOpciónAbrirUnaCuentaDeAportesAporteVoluntarios() {  
 *theActorInTheSpotlight*().attemptsTo(WaitFor.*seconds*(3));  
 *theActorInTheSpotlight*().attemptsTo(ClickElementJs.*on*(HomePage.*BUTTON\_ABRIR\_CUENTA*));  
  
}

3-Enter Text

package tasks.common;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Enter;  
import net.serenitybdd.screenplay.actions.Scroll;  
import net.serenitybdd.screenplay.targets.Target;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
  
import static net.serenitybdd.screenplay.matchers.WebElementStateMatchers.*isVisible*;  
  
public class EnterText implements Task {  
  
 private final Target target;  
 private final String value;  
  
 public EnterText(Target target, String value) {  
 this.target = target;  
 this.value = value;  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 actor.attemptsTo(  
 // Desplazarse al elemento  
 Scroll.*to*(target),  
 // Esperar hasta que el elemento sea visible  
 WaitUntil.*the*(target, *isVisible*()).forNoMoreThan(15).seconds(),  
 // Ingresar el valor en el elemento  
 Enter.*theValue*(value).into(target)  
 );  
 }  
  
 // Mé*todo estático para facilitar la creación de la tarea*  
public static EnterText into(Target target, String value) {  
 return new EnterText(target, value);  
 }  
}

Como se usa

@And("el usuario ingresa a outlook con su correo: {string} y clave: {string}")  
public void elUsuarioIngresaAOutlookConSuCorreoYClave(String correo, String clave) {  
 // theActorInTheSpotlight().attemptsTo(LoginOutlook.login(correo,clave));  
  
 String outlookUrl = EnvironmentSpecificConfiguration.*from*(SystemEnvironmentVariables.*createEnvironmentVariables*())  
 .getProperty("environments.default.outlook.url");  
  
 *theActorInTheSpotlight*().attemptsTo(  
 OpenNewTab.*now*(),  
 NavigateToUrl.*to*(outlookUrl),  
 ClickElement.*on*(LoginOutlookPage.*INP\_EMAIL\_USER*,false),  
 EnterText.*into*(LoginOutlookPage.*INP\_EMAIL\_USER*,correo),  
 ClickElement.*on*(LoginOutlookPage.*NEXT\_BUTTON*,false),  
 WaitFor.*seconds*(5),  
 ClickElement.*on*(LoginOutlookPage.*INP\_PASSWORD\_USER*,false),  
 EnterText.*into*(LoginOutlookPage.*INP\_PASSWORD\_USER*,clave),  
 ClickElement.*on*(LoginOutlookPage.*NEXT\_BUTTON*,false),  
 ClickElement.*on*(LoginOutlookPage.*BTN\_YES\_NOT\_OPTION*,false, "Sí"),  
 WaitForPageToLoad.*withTimeout*(10),  
 ClickElement.*on*(OutlookPage.*ACCOUNT\_1*, false),  
 WaitFor.*seconds*(5)  
 // WaitForPageToLoad.withTimeout(30)  
 );  
}

@And("el usuario ingresa {string} en número de cuenta")  
public void elUsuarioIngresaEnNúmeroDeCuenta(String numero\_cuenta) {  
 *theActorInTheSpotlight*().attemptsTo(EnterText.*into*(WithdrawVoluntaryContributionsPage.*INPUT\_ACCOUNTNUMBER*,  
 numero\_cuenta.trim()));  
}

4-NavigateToURL

package tasks.common;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Open;  
import net.serenitybdd.screenplay.annotations.Subject;  
import net.serenitybdd.screenplay.abilities.BrowseTheWeb;  
import org.openqa.selenium.JavascriptExecutor;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.support.ui.WebDriverWait;  
import org.openqa.selenium.support.ui.ExpectedCondition;  
  
import java.time.Duration;  
  
import static net.serenitybdd.screenplay.Tasks.*instrumented*;  
  
@Subject("Navigate to a specific URL and wait for the page to load")  
public class NavigateToUrl implements Task {  
 private final String url;  
  
 public NavigateToUrl(String url) {  
 this.url = url;  
 }  
  
 public static NavigateToUrl to(String url) {  
 return *instrumented*(NavigateToUrl.class, url);  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 // Navegar a la URL  
 actor.attemptsTo(Open.*url*(url));  
  
 // Obtener el WebDriver del actor  
 WebDriver driver = BrowseTheWeb.*as*(actor).getDriver();  
  
 // Esperar a que el estado del documento sea 'complete'  
 WebDriverWait wait = new WebDriverWait(driver, Duration.*ofSeconds*(10)); // Tiempo máximo de espera: 10 segundos  
 wait.until((ExpectedCondition<Boolean>) wd ->  
 ((JavascriptExecutor) wd).executeScript("return document.readyState").equals("complete"));  
 }  
}

Como se usa

@When("el usuario abre la página de inicio de sesión para probar el cambio de contraseña")  
 public void elUsuarioAbreLaPáginaDeInicioDeSesiónParaProbarElCambioDeContraseña() {  
  
 String loginurl = EnvironmentSpecificConfiguration.*from*(SystemEnvironmentVariables.*createEnvironmentVariables*())  
 .getProperty("environments.default.login.url");  
  
 *theActorInTheSpotlight*().attemptsTo(  
 OpenNewTab.*now*(),  
 NavigateToUrl.*to*(loginurl)  
 );  
 }  
}

5-OpenNewTab

package tasks.common;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.abilities.BrowseTheWeb;  
import net.serenitybdd.screenplay.annotations.Subject;  
import org.openqa.selenium.JavascriptExecutor;  
import org.openqa.selenium.WebDriver;  
  
import java.util.ArrayList;  
import java.util.List;  
  
import static net.serenitybdd.screenplay.Tasks.*instrumented*;  
  
@Subject("Open a new browser tab")  
public class OpenNewTab implements Task {  
  
 public static OpenNewTab now() {  
 return *instrumented*(OpenNewTab.class);  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 WebDriver driver = BrowseTheWeb.*as*(actor).getDriver();  
  
 ((JavascriptExecutor) driver).executeScript("window.open()");  
 List<String> tabs = new ArrayList<>(driver.getWindowHandles());  
  
 driver.switchTo().window(tabs.get(tabs.size() - 1));  
 }  
  
}

Como se usa

@When("el usuario abre la página de inicio de sesión para probar el cambio de contraseña")  
public void elUsuarioAbreLaPáginaDeInicioDeSesiónParaProbarElCambioDeContraseña() {  
  
 String loginurl = EnvironmentSpecificConfiguration.*from*(SystemEnvironmentVariables.*createEnvironmentVariables*())  
 .getProperty("environments.default.login.url");  
  
 *theActorInTheSpotlight*().attemptsTo(  
 OpenNewTab.*now*(),  
 NavigateToUrl.*to*(loginurl)  
 );  
}

6-ScrollTo

package tasks.common;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.targets.Target;  
import org.openqa.selenium.JavascriptExecutor;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.WebElement;  
  
import static net.serenitybdd.screenplay.Tasks.*instrumented*;  
  
public class ScrollTo implements Task {  
 private final Target target;  
 public ScrollTo(Target target) {  
 this.target = target;  
 }  
  
 public static ScrollTo element(Target target) {  
 return *instrumented*(ScrollTo.class, target);  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 WebDriver driver = net.serenitybdd.screenplay.abilities.BrowseTheWeb.*as*(actor).getDriver();  
 WebElement element = target.resolveFor(actor);  
 // Usar JavaScript para hacer scroll al elemento  
 ((JavascriptExecutor) driver).executeScript("arguments[0].scrollIntoView(true);", element);  
 }  
}

Como se usa

Aun no veo el uso

7-SwitchTotab

package tasks.common;  
  
import net.serenitybdd.annotations.Step;  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.Tasks;  
import net.serenitybdd.screenplay.abilities.BrowseTheWeb;  
import net.serenitybdd.screenplay.actions.Switch;  
import org.openqa.selenium.WebDriver;  
  
import java.util.ArrayList;  
  
public class SwitchToTab implements Task {  
 private final int tabIndex;  
  
 public SwitchToTab(int tabIndex) {  
 this.tabIndex = tabIndex;  
 }  
  
 public static SwitchToTab to(int tabIndex) {  
 return Tasks.*instrumented*(SwitchToTab.class, tabIndex);  
 }  
  
 @Override  
 @Step("Cambiar a la pestaña número {0}")  
 public <T extends Actor> void performAs(T actor) {  
 WebDriver driver = BrowseTheWeb.*as*(actor).getDriver();  
 ArrayList<String> tabs = new ArrayList<>(driver.getWindowHandles());  
 if (tabs.size() > tabIndex) {  
 actor.attemptsTo(Switch.*toWindow*(tabs.get(tabIndex)));  
 } else {  
 throw new IllegalStateException("No hay suficientes pestañas abiertas para cambiar a la pestaña número " + tabIndex);  
 }  
 }  
}

Como se usa

@And("el usuario busca el correo {string}")  
public void elUsuarioBuscaElCorreo(String titleEmail) {  
 *theActorInTheSpotlight*().attemptsTo(InputSearchEmail.*withData*(titleEmail));  
 // theActorInTheSpotlight().attemptsTo(new ButtonSearch());  
 *theActorInTheSpotlight*().attemptsTo(new SelectFirstEmail());  
 *theActorInTheSpotlight*().attemptsTo(ButtonShowContent.*now*());  
 *theActorInTheSpotlight*().attemptsTo(ClickLinkChoose.*now*());  
 *theActorInTheSpotlight*().attemptsTo(SwitchToTab.*to*(2));  
 *theActorInTheSpotlight*().attemptsTo(new ButtonOk());  
}

8-WaitFor

package tasks.common;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
  
import static net.serenitybdd.screenplay.Tasks.*instrumented*;  
  
public class WaitFor implements Task {  
 private final int seconds;  
  
 public WaitFor(int seconds) {  
 this.seconds = seconds;  
 }  
  
 public static WaitFor seconds(int seconds) {  
 return *instrumented*(WaitFor.class, seconds);  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 try {  
 Thread.*sleep*(seconds \* 1000L); // Convertir segundos a milisegundos  
 } catch (InterruptedException e) {  
 Thread.*currentThread*().interrupt(); // Restaurar el estado de interrupción  
 throw new RuntimeException("La espera fue interrumpida", e);  
 }  
 }  
}

Como se usa

@And("el usuario selecciona el tipo de retiro {string}")  
public void elUsuarioSeleccionaElTipoDeRetiro(String tipo\_retiro) {  
 *theActorInTheSpotlight*().attemptsTo(WaitFor.*seconds*(20));  
 *theActorInTheSpotlight*().attemptsTo(SelectWithdrawType.*click*(tipo\_retiro));  
}

9-WaitForPageToLoad

package tasks.common;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.abilities.BrowseTheWeb;  
import org.openqa.selenium.JavascriptExecutor;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.support.ui.ExpectedCondition;  
import org.openqa.selenium.support.ui.WebDriverWait;  
  
import java.time.Duration;  
  
import static net.serenitybdd.screenplay.Tasks.*instrumented*;  
  
public class WaitForPageToLoad implements Task {  
  
 private final int timeoutInSeconds;  
  
 public WaitForPageToLoad(int timeoutInSeconds) {  
 this.timeoutInSeconds = timeoutInSeconds;  
 }  
  
 public static WaitForPageToLoad withTimeout(int timeoutInSeconds) {  
 return *instrumented*(WaitForPageToLoad.class, timeoutInSeconds);  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 WebDriver driver = BrowseTheWeb.*as*(actor).getDriver();  
  
 // Esperar a que el estado del documento sea 'complete'  
 WebDriverWait wait = new WebDriverWait(driver, Duration.*ofSeconds*(timeoutInSeconds));  
 wait.until((ExpectedCondition<Boolean>) wd ->  
 ((JavascriptExecutor) wd).executeScript("return document.readyState").equals("complete"));  
 }  
  
}

Como se usa

@And("el usuario ingresa a outlook con el correo {string} y clave {string}")  
public void elUsuarioIngresaAOutlookConElCorreoYClave(String correo, String clave) {  
  
 String outlookUrl = EnvironmentSpecificConfiguration.*from*(SystemEnvironmentVariables.*createEnvironmentVariables*())  
 .getProperty("environments.default.outlook.url");  
   
 *theActorInTheSpotlight*().attemptsTo(  
 OpenNewTab.*now*(),  
 NavigateToUrl.*to*(outlookUrl),  
 ClickElement.*on*(LoginOutlookPage.*INP\_EMAIL\_USER*,false),  
 EnterText.*into*(LoginOutlookPage.*INP\_EMAIL\_USER*,correo),  
 ClickElement.*on*(LoginOutlookPage.*NEXT\_BUTTON*,false),  
 WaitFor.*seconds*(5),  
 ClickElement.*on*(LoginOutlookPage.*INP\_PASSWORD\_USER*,false),  
 EnterText.*into*(LoginOutlookPage.*INP\_PASSWORD\_USER*,clave),  
 ClickElement.*on*(LoginOutlookPage.*NEXT\_BUTTON*,false),  
 ClickElement.*on*(LoginOutlookPage.*BTN\_YES\_NOT\_OPTION*,false, "Sí"),  
 WaitForPageToLoad.*withTimeout*(10),  
 ClickElement.*on*(OutlookPage.*ACCOUNT\_1*, false),  
 WaitForPageToLoad.*withTimeout*(10)  
 );  
}

FEATURE

@inicioSesion @web  
Feature: Iniciar sesión en la aplicación web  
 Como usuario de la aplicación web  
 Quiero iniciar sesión con mis credenciales válidas  
 Para acceder a mi Agencia Digital  
  
 @esc01 @login @happy  
 Scenario Outline: "<Caso de Prueba>" El usuario inicia sesión con tipo documento "<Tipo Documento>" con credenciales válidas  
 Given que el usuario está en la página de inicio de sesión  
 When el usuario selecciona el tipo de documento "<Tipo Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 And el usuario ingresa la contraseña "<Contraseña>"  
 And el usuario hace clic en el botón de inicio de sesión  
 And el usuario cierra va ventana modal informativa  
 Then el usuario valida el mensaje de bienvenida  
  
 Examples:  
 | Caso de Prueba | Tipo Documento | Usuario | Contraseña |  
 | CP01 | DNI | 72159283 | Cercania2024 |  
 | CP02 | CE | 001108718 | Cercania2024x |  
 | CP03 | PAS | D588394 | Cercania2024 |  
*# | CP04 | PTP | 85808405 | Cercania2024 |*  
*# | CP05 | CRE | 09783628 | Cercania2024 |*  
*# | CP06 | CIE | 0962710232 | Cercania2024 |*  
*# | CP07 | CSR | 742402050 | Cercania2024 |*  
*# | CP08 | CPP | 1234567890 | Cercania2024 |*  
*# | CP09 | CIA | 72777884 | Cercania2024 |*  
*# | CP10 | CFA | 31974330 | Cercania2024 |*  
  
@unhappy @login @esc02  
 Scenario Outline: "<Caso de Prueba>" El usuario no puede iniciar sesión por el usuario incompleto  
 Given que el usuario está en la página de inicio de sesión  
 When el usuario selecciona el tipo de documento "<Tipo Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 Then el usuario debería ver el mensaje de campo user incompleto "No es un campo valido"  
  
 Examples:  
 | Caso de Prueba | Tipo Documento | Usuario |  
 | CP11 | DNI | 71 |  
  
 @unhappy @login @esc03  
 Scenario Outline: "<Caso de Prueba>" El usuario no puede iniciar sesión porque el usuario es inválido  
 Given que el usuario está en la página de inicio de sesión  
 When el usuario selecciona el tipo de documento "<Tipo Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 And el usuario ingresa la contraseña "<Contraseña>"  
 And el usuario hace clic en el botón de inicio de sesión  
 Then el usuario debería ver el mensaje de campo user inválido "Puede que no hayas ingresado correctamente tus datos o que no seas cliente de AFP Integra."  
  
 Examples:  
 | Caso de Prueba | Tipo Documento | Usuario | Contraseña |  
 | CP12 | CE | 001108710 | Cercania2021 |  
  
 @unhappy @login @esc04  
 Scenario Outline: "<Caso de Prueba>" El usuario no puede iniciar sesión porque la contraseña es incompleta  
 Given que el usuario está en la página de inicio de sesión  
 When el usuario selecciona el tipo de documento "<Tipo Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 And el usuario ingresa la contraseña "<Contraseña>"  
 Then el usuario debería ver el mensaje de campo password incompleta "Debe ingresar mínimo 8 dígitos"  
  
 Examples:  
 | Caso de Prueba | Tipo Documento | Usuario | Contraseña |  
 | CP13 | PAS | D588394 | Cercan |  
  
 @unhappy @login @esc05  
 Scenario Outline: "<Caso de Prueba>" El usuario no puede iniciar sesión porque la contraseña es inválida  
 Given que el usuario está en la página de inicio de sesión  
 When el usuario selecciona el tipo de documento "<Tipo Documento>"  
 And el usuario ingresa el nombre de usuario "<Usuario>"  
 And el usuario ingresa la contraseña "<Contraseña>"  
 And el usuario hace clic en el botón de inicio de sesión  
 Then el usuario debe ver mensaje de contraseña inválida  
  
 Examples:  
 | Caso de Prueba | Tipo Documento | Usuario | Contraseña |  
 *# | CP14 | CIA | 72777884 | Cercania2020 |*  
| CP14 | DNI | 45661984 | Cercania2020 |  
  
 @unhappy @login @esc06  
 Scenario Outline: "<Caso de prueba>" El usuario no se encuentra registrado  
 Given que el usuario está en la página de inicio de sesión  
 When el usuario selecciona el tipo de documento "<tipoDocumento>"  
 And el usuario ingresa el nombre de usuario "<usuario>"  
 And el usuario ingresa la contraseña "abchg5g55"  
 And el usuario hace clic en el botón de inicio de sesión  
 Then el usuario debería ver el mensaje de datos incorrectos o no es cliente de AFP Integra  
  
 Examples:  
 | Caso de prueba | tipoDocumento | usuario |  
 | CP13 | DNI | 99999999 |

@Given("que el usuario está en la página de inicio de sesión")  
public void queElUsuarioEstáEnLaPáginaDeInicioDeSesión() {  
 *theActorCalled*("user").wasAbleTo(OpenUrl.*loginAFP*());  
}

package tasks.login;  
  
import net.serenitybdd.annotations.Step;  
import net.serenitybdd.model.environment.ConfiguredEnvironment;  
import net.serenitybdd.model.environment.EnvironmentSpecificConfiguration;  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Open;  
import net.thucydides.model.util.EnvironmentVariables;  
import static net.serenitybdd.screenplay.Tasks.*instrumented*;  
  
public class OpenUrl implements Task {  
  
 private final String url;  
 private final EnvironmentVariables environmentVariables;  
  
 public OpenUrl(String url) {  
 this.url = url;  
 this.environmentVariables = ConfiguredEnvironment.*getEnvironmentVariables*();  
 }  
  
 public static Task loginAFP() {  
 String url = "login.url";  
 return *instrumented*(OpenUrl.class, url);  
 }  
  
 @Override  
 @Step("{0} Inicia la página #url")  
 public <T extends Actor> void performAs(T actor) {  
  
 String targetUrl = EnvironmentSpecificConfiguration.*from*(environmentVariables).getProperty(url);  
  
 if (targetUrl == null) {  
 throw new IllegalArgumentException("URL not found in environment variables: " + url);  
 } else {  
 actor.attemptsTo(Open.*url*(targetUrl));  
 }  
 }  
}

@And("el usuario selecciona el tipo de documento {string}")  
public void elUsuarioSeleccionaElTipoDeDocumento(String typeDocument) {  
 *theActorInTheSpotlight*().attemptsTo(SelectDocumentType.*withData*(typeDocument));  
}

package tasks.login;  
  
import net.serenitybdd.annotations.Step;  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Click;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import ui.CookiesPage;  
  
public class SelectDocumentType implements Task {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(SelectDocumentType.class);  
 private final String documentType;  
  
 public SelectDocumentType(String documentType) {  
 this.documentType = documentType;  
 }  
  
 public static SelectDocumentType withData(String documentType) {  
 return new SelectDocumentType(documentType);  
 }  
  
 @Override  
 @Step("{0} selecciona el tipo de documento #documentType")  
 public <T extends Actor> void performAs(T actor) {  
 long endTime = System.*currentTimeMillis*() + 10000; // 10 segundos  
 boolean clicked = false;  
  
 while (System.*currentTimeMillis*() < endTime && !clicked) {  
 try {  
 Thread.*sleep*(500); // Espera inicial de 500 milisegundos  
 if (CookiesPage.*DOCUMENT\_TYPE\_DROPDOWN*.resolveFor(actor).isVisible()) {  
 actor.attemptsTo(  
 WaitUntil.*the*(CookiesPage.*DOCUMENT\_TYPE\_DROPDOWN*, WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(),  
 Click.*on*(CookiesPage.*DOCUMENT\_TYPE\_DROPDOWN*)  
 );  
 clicked = true;  
 } else {  
 *LOGGER*.info("El menú desplegable de tipo de documento no está visible inicialmente.");  
 }  
 } catch (Exception e) {  
 *LOGGER*.warn("Intento fallido de hacer clic en el menú desplegable de tipo de documento. Reintentando...", e);  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de reintentar  
 } catch (InterruptedException ie) {  
 Thread.*currentThread*().interrupt();  
 *LOGGER*.error("El hilo fue interrumpido", ie);  
 }  
 }  
 }  
  
 if (!clicked) {  
 *LOGGER*.error("No se pudo hacer clic en el menú desplegable de tipo de documento después de 10 segundos.");  
 return;  
 }  
  
 // Paso 2: Selecciona la opción del DropDown  
 actor.attemptsTo(  
 WaitUntil.*the*(CookiesPage.*CLICK\_OPTION\_TYPE\_DROPDOWN*.of(documentType), WebElementStateMatchers.*isVisible*()).forNoMoreThan(15).seconds(),  
 WaitUntil.*the*(CookiesPage.*CLICK\_OPTION\_TYPE\_DROPDOWN*.of(documentType), WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(),  
 Click.*on*(CookiesPage.*CLICK\_OPTION\_TYPE\_DROPDOWN*.of(documentType))  
 );  
 }  
}

@And("el usuario ingresa el nombre de usuario {string}")  
public void elUsuarioIngresaElNombreDeUsuario(String user) {  
 *theActorInTheSpotlight*().attemptsTo(InputUser.*withData*(user));  
}

package tasks.login;  
  
import net.serenitybdd.annotations.Step;  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Performable;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.Tasks;  
import net.serenitybdd.screenplay.actions.Enter;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
import ui.CookiesPage;  
  
public class InputUser implements Task {  
  
 private static final Logger *LOGGER* = LoggerFactory.*getLogger*(InputUser.class);  
 private final String enterUser;  
  
 public InputUser(String enterUser) {  
 this.enterUser = enterUser;  
 }  
  
 public static Performable withData(String enterUser) {  
 return Tasks.*instrumented*(InputUser.class, enterUser);  
 }  
  
 @Override  
 @Step("{0} ingresa usuario")  
 public <T extends Actor> void performAs(T actor) {  
 long endTime = System.*currentTimeMillis*() + 10000; // 10 segundos  
 boolean entered = false;  
  
 while (System.*currentTimeMillis*() < endTime && !entered) {  
 try {  
  
 if (CookiesPage.*USERNAME\_FIELD*.resolveFor(actor).isVisible()) {  
 actor.attemptsTo(  
 WaitUntil.*the*(CookiesPage.*USERNAME\_FIELD*, WebElementStateMatchers.*isClickable*()).forNoMoreThan(15).seconds(),  
 Enter.*theValue*(enterUser).into(CookiesPage.*USERNAME\_FIELD*)  
 );  
 entered = true;  
 } else {  
 *LOGGER*.info("El campo de entrada de usuario no está visible inicialmente.");  
 }  
 } catch (Exception e) {  
 *LOGGER*.warn("Intento fallido de ingresar el usuario. Reintentando...", e);  
 try {  
 Thread.*sleep*(500); // Espera medio segundo antes de reintentar  
 } catch (InterruptedException ie) {  
 Thread.*currentThread*().interrupt();  
 *LOGGER*.error("El hilo fue interrumpido", ie);  
 }  
 }  
 }  
  
 if (!entered) {  
 *LOGGER*.error("No se pudo ingresar el usuario después de 10 segundos.");  
 }  
 }  
}

@And("el usuario ingresa la contraseña {string}")  
public void elUsuarioIngresaLaContraseña(String password) {  
 *theActorInTheSpotlight*().attemptsTo(InputPassword.*withData*(password));  
}

package tasks.login;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Enter;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import ui.CookiesPage;  
  
public class InputPassword implements Task {  
  
 private final String enterPassword;  
  
 public InputPassword(String enterPassword) {  
 this.enterPassword = enterPassword;  
 }  
  
 public static InputPassword withData(String enterPassword) {  
 return new InputPassword(enterPassword);  
 }  
  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 actor.attemptsTo(  
 WaitUntil.*the*(CookiesPage.*PASSWORD\_FIELD*, WebElementStateMatchers.*isVisible*()).forNoMoreThan(15).seconds(),  
 Enter.*theValue*(enterPassword).into(CookiesPage.*PASSWORD\_FIELD*));  
 }  
}

@And("el usuario hace clic en el botón de inicio de sesión")  
public void elUsuarioHaceClicEnElBotónDeInicioDeSesión() {  
 *theActorInTheSpotlight*().attemptsTo(new ButtonEnter());  
}

package tasks.login;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Click;  
import net.serenitybdd.screenplay.matchers.WebElementStateMatchers;  
import net.serenitybdd.screenplay.waits.WaitUntil;  
import ui.CookiesPage;  
  
public class ButtonEnter implements Task {  
  
 public <T extends Actor> void performAs(T actor) {  
 actor.attemptsTo(  
 WaitUntil.*the*(CookiesPage.*LOGIN\_BUTTON*, WebElementStateMatchers.*isVisible*()).forNoMoreThan(15).seconds(),  
 Click.*on*(CookiesPage.*LOGIN\_BUTTON*));  
 }  
}

public static final Target *LOGIN\_BUTTON* = Target.*the*("Click on the login button")  
 .located(By.*xpath*("//button[@type='submit']"));

package com.sura.web.stepdefinitions;  
  
import io.cucumber.java.en.And;  
import tasks.dashboard.CloseModalPresent;  
  
import static net.serenitybdd.screenplay.actors.OnStage.*theActorInTheSpotlight*;  
  
public class OutlookStepDefinitions {  
  
 @And("el usuario cierra va ventana modal informativa")  
 public void el\_usuario\_cierra\_va\_ventana\_modal\_informativa() {  
 *theActorInTheSpotlight*().attemptsTo(CloseModalPresent.*close*());  
 }  
  
}

package tasks.dashboard;  
  
import net.serenitybdd.screenplay.Actor;  
import net.serenitybdd.screenplay.Task;  
import net.serenitybdd.screenplay.actions.Click;  
import net.serenitybdd.screenplay.actions.Scroll;  
import net.serenitybdd.screenplay.conditions.Check;  
import ui.DashboardPage;  
  
public class CloseModalPresent implements Task {  
 @Override  
 public <T extends Actor> void performAs(T actor) {  
 actor.attemptsTo(  
 Check.*whether*(DashboardPage.*MODAL\_BUTTON*.resolveFor(actor).isVisible())  
 .andIfSo(  
 Scroll.*to*(DashboardPage.*MODAL\_BUTTON*),  
 Click.*on*(DashboardPage.*MODAL\_BUTTON*)  
 )  
 );  
 }  
  
 // Mé*todo estático para facilitar la creación de la tarea*  
public static CloseModalPresent close() {  
 return new CloseModalPresent();  
 }  
}

public static final Target *MODAL\_BUTTON* = Target.*the*("Botón del modal")  
 .locatedBy("//div[contains(@class,'IsOpen')]//button");

@Then("el usuario valida el mensaje de bienvenida")  
public void elUsuarioValidaElMensajeDeBienvenida() {  
 *theActorInTheSpotlight*().should(  
 *seeThat*(  
 "El mensaje de bienvenida es correcto",  
 LoginQuestions.*ValidateMessage*(),  
 *anyOf*(  
 *containsString*("Bienvenido a tu Agencia Digital"),  
 *containsString*("Bienvenida a tu Agencia Digital")  
 )  
 )  
 );  
}

public static final Target *DOCUMENT\_TYPE\_DROPDOWN* = Target.*the*("Document Type Dropdown")  
 .located(By.*cssSelector*(".Dropdown\_c\_dropdown\_\_relative\_\_zbfiJ"));

public static final Target *USERNAME\_FIELD* = Target.*the*("Enter Username")  
 .located(org.openqa.selenium.By.*id* ("inpDocumentNumber"));

public static final Target *PASSWORD\_FIELD* = Target.*the*("Enter Password")  
 .located(org.openqa.selenium.By.*id*("inpPassword"));

public static final Target *LOGIN\_BUTTON* = Target.*the*("Click on the login button")  
 .located(By.*xpath*("//button[@type='submit']"));

public static final Target *MODAL\_BUTTON* = Target.*the*("Botón del modal")  
 .locatedBy("//div[contains(@class,'IsOpen')]//button");

STEP DE LOGIN

package com.sura.web.stepdefinitions;  
  
import io.cucumber.java.en.And;  
import io.cucumber.java.en.Given;  
import io.cucumber.java.en.Then;  
import io.cucumber.java.en.When;  
import questions.ElementVisible;  
import questions.LoginQuestions;  
import tasks.login.\*;  
import ui.LoginPage;  
  
import static net.serenitybdd.screenplay.GivenWhenThen.*seeThat*;  
import static net.serenitybdd.screenplay.actors.OnStage.*theActorCalled*;  
import static net.serenitybdd.screenplay.actors.OnStage.*theActorInTheSpotlight*;  
import static org.hamcrest.Matchers.*containsString*;  
import static org.hamcrest.Matchers.*equalTo*;  
import static org.hamcrest.Matchers.*anyOf*;  
import static org.hamcrest.Matchers.*is*;  
  
public class LoginStepDefinitions {  
  
 @Given("que el usuario está en la página de inicio de sesión")  
 public void queElUsuarioEstáEnLaPáginaDeInicioDeSesión() {  
 *theActorCalled*("user").wasAbleTo(OpenUrl.*loginAFP*());  
 }  
  
 @And("el usuario selecciona el tipo de documento {string}")  
 public void elUsuarioSeleccionaElTipoDeDocumento(String typeDocument) {  
 *theActorInTheSpotlight*().attemptsTo(SelectDocumentType.*withData*(typeDocument));  
 }  
  
 @And("el usuario ingresa el nombre de usuario {string}")  
 public void elUsuarioIngresaElNombreDeUsuario(String user) {  
 *theActorInTheSpotlight*().attemptsTo(InputUser.*withData*(user));  
 }  
  
 @And("el usuario ingresa la contraseña {string}")  
 public void elUsuarioIngresaLaContraseña(String password) {  
 *theActorInTheSpotlight*().attemptsTo(InputPassword.*withData*(password));  
 }  
  
 @And("el usuario hace clic en el botón de inicio de sesión")  
 public void elUsuarioHaceClicEnElBotónDeInicioDeSesión() {  
 *theActorInTheSpotlight*().attemptsTo(new ButtonEnter());  
 }  
  
 @Then("el usuario debería ver el mensaje de error {string}")  
 public void elUsuarioDeberíaVerElMensajeDeError(String message) {  
 *theActorInTheSpotlight*().should(*seeThat*(LoginQuestions.*ValidateMessage*(), *equalTo*(message)));  
 }  
  
 @Then("el usuario debería ver el mensaje de campo user incompleto {string}")  
 public void elUsuarioDeberíaVerElMensajeDeCampoUserInvalido(String message) {  
 *theActorInTheSpotlight*().should(*seeThat*(LoginQuestions.*ValidateMessageFieldUser*(), *containsString*(message)));  
 }  
  
 @Then("el usuario debería ver el mensaje de campo user inválido {string}")  
 public void elUsuarioDeberíaVerElMensajeDeCampoUserInválido(String message) {  
 *theActorInTheSpotlight*().should(*seeThat*(LoginQuestions.*ValidateMessageModalUser*(), *equalTo*(message)));  
 }  
  
 @Then("el usuario debería ver el mensaje de campo password incompleta {string}")  
 public void elUsuarioDeberíaVerElMensajeDeCampoPasswordIncompleta(String message) {  
 *theActorInTheSpotlight*().should(*seeThat*(LoginQuestions.*ValidateMessageFieldPassword*(), *equalTo*(message)));  
 }  
  
  
 @Then("el usuario valida el mensaje de bienvenida")  
 public void elUsuarioValidaElMensajeDeBienvenida() {  
 *theActorInTheSpotlight*().should(  
 *seeThat*(  
 "El mensaje de bienvenida es correcto",  
 LoginQuestions.*ValidateMessage*(),  
 *anyOf*(  
 *containsString*("Bienvenido a tu Agencia Digital"),  
 *containsString*("Bienvenida a tu Agencia Digital")  
 )  
 )  
 );  
 }  
  
 @Then("el usuario debe ver mensaje de contraseña inválida")  
 public void elUsuarioDebeVerMensajeDeContraseñaInválida() {  
 *theActorInTheSpotlight*().should(  
 *seeThat*(  
 "El mensaje de bienvenida es correcto",  
 LoginQuestions.*ValidateMessagedos*(),  
 *anyOf*(  
 *containsString*("Tienes un (1) intento más para ingresar tu contraseña."),  
 *containsString*("Tienes dos (2) intentos más para ingresar tu contraseña.")  
 )  
 )  
 );  
  
  
 }  
  
 @Then("el usuario debería ver el mensaje de datos incorrectos o no es cliente de AFP Integra")  
 public void elUsuarioDeberíaVerElMensajeDeDatosIncorrectosONoEsClienteDeAFPIntegra() {  
  
 *theActorInTheSpotlight*().should(*seeThat*(ElementVisible.*isVisible*(  
 LoginPage.*VALIDAR\_MENSAJE\_DATOSINCORRECT*), *is*(true)));  
 }  
  
}