Informe Pruebas

Añasco Silvia, Enriquez Sheylee, Proaño Jose & Oña Yorman

Departamento de Ciencias de la Computación

Universidad de las Fuerzas Armadas ESPE

14568: Ingeniería de Requisitos

Ing. Jenny Alexandra Ruíz Robalino

Versión 1

25 de Julio de 2024

Historial de versiones

Fecha	Versión	Descripción	Autor
25/07/2024	V 1.0	Creación, redacción y análisis de los casos de prueba	Silvia Añasco, Sheylee Enriquez, Yorman Oña

Tabla de contenido

1. Objetivo

Validar y asegurar la funcionalidad del proceso de registro de usuarios en el sistema "Banana's Cocktails". Esto incluye verificar que los nuevos usuarios puedan crear cuentas de manera exitosa con información válida, y que el sistema maneje adecuadamente los casos donde la información ingresada es incompleta o inválida. Además, se busca garantizar la integridad y seguridad de los datos de usuario, así como la facilidad de uso del proceso de registro. La validación de este requisito es fundamental para asegurar una experiencia de usuario positiva y la eficiencia operativa del sistema.

2. Metodología

La metodología empleada para validar el proceso de registro de usuarios en el sistema "Banana's Cocktails" se basa en la automatización de pruebas utilizando las herramientas Selenium y Cucumber, con implementación en Python.

3. Resultado de las pruebas

Requisito probado: Registro de usuarios

3.1 Escenario 1: Registro exitoso

Objetivo: Verificar que un usuario pueda registrarse exitosamente con datos válidos.

Flujo de pasos:

1. Navegar a la página de registro.
2. Ingresar datos válidos en cada campo.
3. Hacer clic en el botón de registro.

Resultado esperado:

- El sistema valida la información.
- Se crea una nueva cuenta de usuario.
- Se muestra un mensaje de confirmación.

Resultado Real: Passed

Análisis: El equipo al haber sido el que diseñó el formulario de registro, sabe cuál es el formato idóneo para cada uno de los campos; por lo que, los datos que se usaron en la prueba se pensaron bajo las mismas concepciones con las que los campos del formulario fueron creados.

3.2 Escenario 2: Campos en blanco

Objetivo: Verificar que el sistema muestre un mensaje de alerta cuando los campos están vacíos.

Flujo de pasos:

1. Navegar a la página de registro.
2. Dejar todos los campos en blanco.
3. Hacer clic en el botón de registro.

Resultado esperado:

- El sistema detecta los campos vacíos.
- Se muestra un mensaje de alerta.

Resultado Real: Passed

Análisis: Desde el punto de vista del usuario, es común que entre tantos campos dentro de un formulario que se debe llenar, exista uno que se pasa por alto. Razón por la cuál lenguajes como HTML ya incluyen esta condición (conocida como required) que permiten manejar los campos en blanco y

mostrar un mensaje al usuario, lo que facilitó al equipo de trabajo manejar dicho problema sin mayor déficit que, por ende, la prueba en este escenario pueda pasar.

3.3 Escenario 3: Nombre con números

Objetivo: Verificar que el sistema muestre un mensaje de alerta cuando el campo nombre contiene números.

Flujo de pasos:

1. Navegar a la página de registro.
2. Ingresar un nombre con números.
3. Hacer clic en el botón de registro.

Resultado esperado:

- El sistema detecta el nombre inválido.
- Se muestra un mensaje de alerta.

Resultado Real: Failed

Análisis: El fallo se debe a un problema principal, en el frontend. En el frontend, no se implementó una validación adecuada para el campo de nombre que evite la entrada de números. Esto permite que los usuarios puedan ingresar cualquier cadena de texto, incluyendo números, en el campo de nombre sin recibir ningún tipo de alerta o mensaje de error inmediato. Adicionalmente, en el backend dentro del modelo para la tabla de usuarios, el campo de nombre está configurado simplemente como un campo de texto (string). Django no realiza ninguna validación específica para evitar que se ingresen números en este campo, ya que considera cualquier entrada de texto como válida. Por lo

tanto, los números se aceptan y se almacenan en la base de datos como cadenas de texto sin generar ningún mensaje de error.

3.4 Escenario 4: Apellido con números

Objetivo: Verificar que el sistema muestre un mensaje de alerta cuando el campo apellido contiene números.

Flujo de pasos:

1. Navegar a la página de registro.
2. Ingresar un apellido con números.
3. Hacer clic en el botón de registro.

Resultado esperado:

- El sistema detecta el apellido inválido.
- Se muestra un mensaje de alerta.

Resultado Real: Failed

Análisis: El fallo, al igual que en escenario anterior, se debe al problema en el frontend. En el frontend, no se implementó una validación adecuada para el campo de apellido que evite la entrada de números. Esto permite que los usuarios puedan ingresar cualquier cadena de texto, incluyendo números, en el campo de apellido sin recibir ningún tipo de alerta o mensaje de error inmediato. Así, los datos se envían sin problema y se guardan en la BD normalmente.

3.5 Escenario 5: Fechas de nacimiento muy antiguas

Objetivo: Verificar que el sistema muestre un mensaje de alerta cuando se ingresa una fecha de nacimiento excesivamente antigua.

Flujo de pasos:

1. Navegar a la página de registro.
2. Ingresar una fecha de nacimiento excesivamente antigua.
3. Hacer clic en el botón de registro.

Resultado esperado:

- El sistema detecta la fecha de nacimiento inválida.
- Se muestra un mensaje de alerta.

Resultado Real: Failed

Análisis: El fallo en este escenario se debe a la falta de validación específica tanto en el frontend como en el backend para fechas de nacimiento excesivamente antiguas. Aunque se utiliza un campo de entrada de tipo fecha en HTML, este no realiza una validación para determinar si la fecha ingresada es razonable en el contexto actual. Así mismo, dentro del modelo de la tabla en la BD, el campo para fecha de nacimiento solo almacena la fecha como una cadena, permitiendo cualquier valor de fecha sin considerar su antigüedad.

3.6 Escenario 6: Edad menor a 18 años

Objetivo: Verificar que el sistema muestre un mensaje de alerta cuando se ingresa una fecha de nacimiento que indica que el usuario es menor de 18 años.

Flujo de pasos

1. Navegar a la página de registro.
2. Ingresar la fecha de nacimiento de una persona menor de edad.
3. Hacer clic en el botón de registro.

Resultado esperado:

- El sistema detecta la edad menor a 18 años.

- Se muestra un mensaje de alerta.

Resultado Real: Failed

Análisis: Este escenario falló porque no se implementaron validaciones específicas para verificar la edad del usuario en el momento de la inscripción. Aunque el campo de entrada de tipo fecha en HTML permite seleccionar una fecha, no hay lógica que compare la fecha ingresada con la fecha actual para determinar si el usuario es menor de 18 años, un aspecto importante a tener en consideración puesto que dicho registro pertenece a una página destinada a la reserva de servicios de bartender.

3.7 Escenario 7: Cédula con valor negativo

Objetivo: Verificar que el sistema muestre un mensaje de alerta cuando se ingresa un número negativo en el campo de cédula.

Flujo de pasos:

1. Navegar a la página de registro.
2. Ingresar un número negativo en el campo de cédula.
3. Hacer clic en el botón de registro.

Resultado esperado:

- El sistema detecta el número de cédula inválido.
- Se muestra un mensaje de alerta.

Resultado Real: Failed

Análisis: El fallo en este escenario se debe a la ausencia de validaciones para asegurar que el número de cédula sea positivo. El campo de entrada de tipo número permite el ingreso de cualquier número, incluidos los negativos. Es crucial que la cédula contenga únicamente valores numéricos positivos, ya que

una cédula con números negativos no tiene sentido en el contexto de identificación personal.

4. **Script**

Feature: Registro de usuarios

Scenario: Registro exitoso

Given I navigate to the register page

When I enter valid data for each field

Then I should see a confirmation message

Scenario: Campos en blanco

Given I navigate to the register page

When I leave all fields empty

Then I should see a message

Scenario: Nombre con números

Given I navigate to the register page

When I enter a name with numbers

Then I should see a alert message

Scenario: Apellido con números

Given I navigate to the register page

When I enter a lastname with numbers

Then I should see a alert message

Scenario: Fechas de nacimiento muy antiguas

Given I navigate to the register page

When I enter an excessively old birth date

Then I should see a alert message

Scenario: Edad menor a 18 años

Given I navigate to the register page

When I enter the birth date of a underage person

Then I should see a alert message

Scenario: Cedula con valor negativo

Given I navigate to the register page

When I enter a negative number in id field

Then I should see a alert message

5. Conclusiones

Los resultados de las pruebas muestran que el escenario de registro exitoso pasó sin problemas, validando que el sistema permite a los usuarios registrarse con datos válidos de manera correcta. Sin embargo, los escenarios que requerían la visualización de mensajes de alerta fallaron debido a errores de localización de elementos y la falta de validaciones específicas en el frontend y backend. Esto indica que existen deficiencias en la implementación de validaciones para campos específicos, lo que puede afectar la integridad de los datos y la experiencia del usuario.

6. Recomendaciones

Es fundamental implementar validaciones específicas tanto en el frontend como en el backend para asegurar la integridad de los datos ingresados por los usuarios. Se recomienda restringir la entrada a caracteres alfabéticos para los campos de nombre y apellido, utilizar expresiones regulares (regex) y asegurar que el campo de cédula solo acepte valores numéricos positivos. Además, es importante verificar que las fechas de

nacimiento no sean excesivamente antiguas y que los usuarios tengan al menos 18 años. Una vez implementadas estas validaciones, se deben volver a realizar las pruebas correspondientes para verificar que las validaciones están funcionando correctamente y que el sistema maneja adecuadamente todos los casos de entrada de datos.