**INFORME DE PRUEBAS PARA LOS CASOS DE USO**

Alfonso Arroyo

David Pilaguano

Stalin Uvidia

**Departamento de Ciencias de la Computación, Universidad de las Fuerzas Armadas ESPE**

**14538 – Fundamentos de la Ingenieria de Software**

**Ing. Jenny Alexandra Ruiz Robalino**

**02 de septiembre de 2024**

1. **Introducción.**

Este informe detalla el proceso mediante el cual se llevaron a cabo las pruebas de aceptación en el prototipo de los casos de uso del proyecto. Se utilizaron diversas herramientas y metodologías, con un enfoque particular en la herramienta Cucumber, que permite la automatización de pruebas mediante la creación de escenarios en lenguaje natural. El propósito de estas pruebas es verificar que cada funcionalidad del sistema opere de manera correcta y eficiente, asegurando que los usuarios finales puedan interactuar con el sistema de manera intuitiva y sin errores.

1. **Objetivos**

**2.1. Objetivo General**

Desarrollar pruebas de aceptación en el prototipo de los casos de uso del proyecto, todo esto mediante la herramienta de Cucumber para validar el correcto funcionamiento en reglas aprobadas o fallidas.

**2.2. Objetivos Específicos**

- Establecer un entorno de desarrollo necesario para el desarrollo de pruebas automatizadas mediante la configuración del lenguaje de programación Java en ellos prototipos de casos de uso de todas las pantallas

- Realizar capturas de la ejecución para simular la interacción con el usuario en cada pantalla verificando todo los casos de uso y requisitos funcionales

- Recopilar los resultados de las pruebas automatizadas en un archivo Excel para analizar los resultados y obtener posibles mejoras

1. **Entradas a utilizar**

Se emplearán todos los campos de texto de cada pantalla utilizada como entradas a utilizar. En las tablas 1 2, y 3 se visualizan las entradas, con sus respectivos estados esperados al realizar las pruebas de aceptación del programa.

Cada campo de texto será sometido a diferentes pruebas, incluyendo la introducción de datos válidos, inválidos, datos límite, y pruebas de campo vacío, para asegurar que la aplicación maneje correctamente todas las posibles entradas de los usuarios.

Tabla

Descripción generada automáticamenteTabla 1 Ingreso al sistema | Login

Tabla 2 Añadir productos | AddProducts verify the ID

Tabla

Descripción generada automáticamente

Tabla

Descripción generada automáticamenteTabla 3 Editar Productor | EditProducts verify the ID

En las tablas mencionadas, se detallan cada uno de estos campos, los valores de entrada probados, y el comportamiento esperado del sistema en cada caso. Esto proporciona una base clara y completa para evaluar si el programa cumple con los criterios de aceptación definidos.

1. **Ambiente de Desarrollo de pruebas**

Para el desarrollo de pruebas, se utilizó la herramienta Java con el IDE Intellij en el cual se creo un ambiente, en el mismo se creo diferentes archivos con el propósito de desarrollar las pruebas, esto se lo realiza mediante el uso de algunas dependencias como se recomendaba en la página web Cucumber <https://cucumber.io/docs/installation/java/>

A continuación, se detallará mas el contenido de los diferentes archivos dentro del proyecto de pruebas:

* Archivo iniciarSesion.feature: en este se encuentran los casos a evaluar dentro del caso de uso Iniciar Sesión, en el cual se puede encontrar los escenarios y las preguntas que se realiza en cada caso.

Figura 1

*Archivo iniciarSesion.feature:*

*Texto

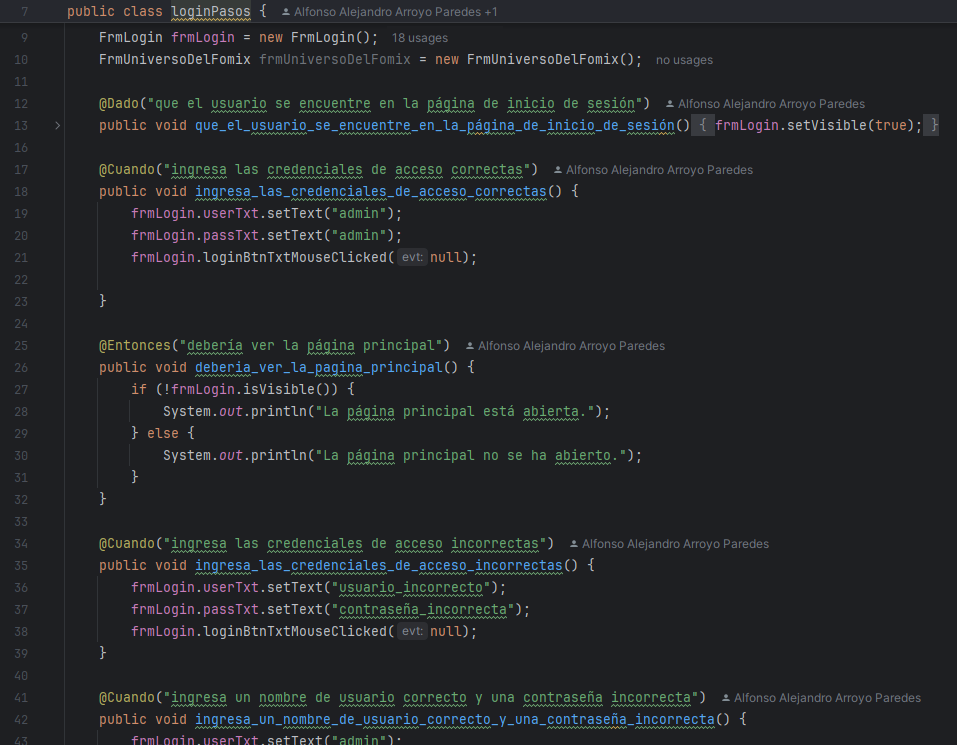
Descripción generada automáticamente*

En la figura 1 se muestra el feature "Inicio de sesión", en el cual existe un primer escenario llamado "Iniciar sesión exitosamente". Este escenario es considerado exitoso ya que se verifica que, al ingresar las credenciales de acceso correctas en la página de inicio de sesión, el usuario debería ser redirigido a la página principal, lo que confirma que el proceso de autenticación ha sido completado correctamente.

* Archivo loginPasos.java: en este archivo se encuentran los pasos evaluados en lenguaje java con entradas de datos para probar el escenario de inicio de sesión del proyecto de pruebas automatizadas

Figura 2

*Archivo loginPasos.java*



Dentro de la figura anterior se muestra como se crearon las funciones en base al .feature anterior determinado en el escenario. Para acceder a todos los atributos y entradas del programa se usa las importaciones del proyecto.

* Archivo añadirProductos.feature: Este archivo define los escenarios de prueba para la funcionalidad de añadir productos. Incluye casos de uso tanto exitosos como fallidos, especificando los pasos y condiciones a evaluar. Los escenarios detallan las acciones del usuario y las respuestas esperadas del sistema, asegurando una prueba exhaustiva de la capacidad de añadir productos y manejar entradas incorrectas o situaciones especiales.

Figura 3

Texto

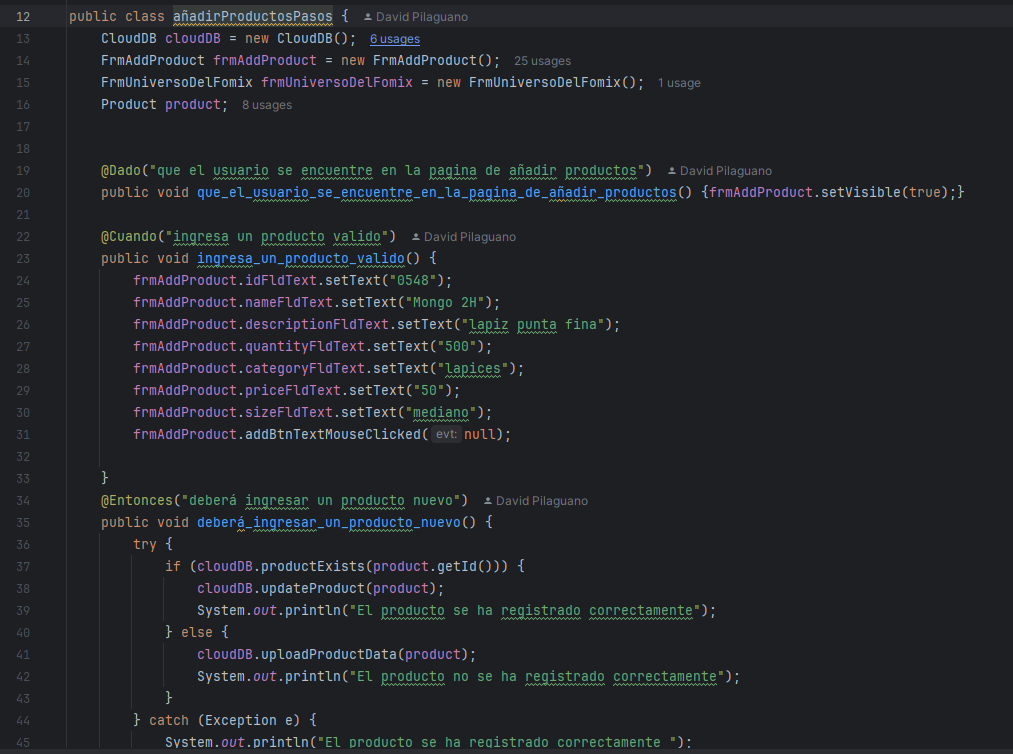
Descripción generada automáticamente*Archivo añadirProductos.feature:*

En la figura 3 se muestra el feature "Añadir Producto", en el cual existe un primer escenario llamado "Añadir un producto nuevo exitosamente". Este escenario es considerado exitoso ya que se verifica que, el usuario ingrese un producto valido para que este sea subido a la base de datos exitosamente y se mantenga en la misma pantalla.

* Archivo añadirProductosPasos.java: en este archivo se encuentran los pasos evaluados en lenguaje java con entradas de datos válidas para probar el escenario de ingreso de un producto a la base de datos.

Figura 4

*Archivo añadirProductosPasos.java*

**

Dentro de la figura 4 se muestra cómo se crearon las funciones en base al .feature anterior determinado en el escenario, esto mediante los atributos y entradas del programa se pueden usar por las importaciones del proyecto.

* Archivo eliminarProductos.feature: Este archivo define los escenarios de prueba para la funcionalidad de eliminar productos. Incluye casos de uso exitosos, donde se verifica que el usuario pueda eliminar productos existentes, así como situaciones fallidas, como cuando se intenta eliminar un producto que no existe en la base de datos.

Figura 5

Texto

Descripción generada automáticamente*Archivo eliminarProductos.feature:*

En la figura 5 se muestra el feature "Eliminar Producto" en el cual existe un primer escenario llamado "Eliminar un producto exitosamente". Este escenario es considerado exitoso ya que se verifica que el usuario selecciona un producto existente y confirma su eliminación. El sistema debe eliminar el producto de la base de datos y notificar al usuario que la operación se completó correctamente.

* Archivo eliminarProductosPasos.java: En este archivo se encuentran los pasos evaluados en lenguaje Java para probar el escenario de eliminación de productos del proyecto de pruebas automatizadas. Aquí se implementan las funciones necesarias para acceder a la base de datos, eliminar el producto seleccionado, y verificar que la eliminación se realizó correctamente, asegurando que el sistema responde adecuadamente a esta operación.

Figura 6

Texto

Descripción generada automáticamente*Archivo eliminarProductosPasos.java*

Dentro de la figura 6 se muestra cómo se crearon las funciones en base al .feature anterior determinado en el escenario. Se utilizan los métodos necesarios para buscar el producto en la base de datos, ejecutar la operación de eliminación, y luego verificar que el producto ya no está disponible. Esto asegura que el proceso de eliminación funcione sin errores y que el sistema responda de acuerdo a lo esperado.

* Archivo editarProductos.feature: Este archivo define los escenarios de prueba para la funcionalidad de edición de productos. Los escenarios incluyen la modificación de detalles como el nombre o el precio de un producto y verifican que el sistema actualice correctamente la base de datos con los nuevos valores introducidos.

Figura 7

*Archivo editarProductos.feature:*

Texto

Descripción generada automáticamente

En la figura 7 se presenta el feature "Editar Producto", que contiene un escenario llamado "Editar un producto existente exitosamente". En este escenario, se valida que un usuario pueda modificar los detalles de un producto existente, como el nombre o el precio. El sistema debe actualizar la información en la base de datos y reflejar los cambios de manera inmediata.

* Archivo editarProductosPasos.java: Dentro de este archivo se encuentran los pasos evaluados en lenguaje Java para probar el escenario de edición de productos. Se implementan las funciones que permiten modificar los atributos de un producto, como el nombre o el precio, y verificar que los cambios se guardan correctamente en la base de datos, manteniendo la integridad de la información.

Figura 8

*Archivo editarProductosPasos.java:*

La figura 8 muestra cómo se implementaron los pasos en Java para el escenario de edición de productos. Aquí se manejan los cambios en los datos del producto, asegurando que los nuevos valores se guarden correctamente y que el sistema maneje adecuadamente cualquier error, como la introducción de valores no válidos.Texto

Descripción generada automáticamente

* Archivo universoDelFomix.feature: Este archivo abarca un conjunto de escenarios para probar la funcionalidad integral del sistema de inventario del "Universo del Fomix". Los escenarios incluyen la interacción entre diferentes módulos del sistema, como añadir, editar y eliminar productos, para garantizar que todas las funcionalidades trabajen en conjunto de manera coherente.

Figura 9

*Archivo universoDelFomix.feature:*

Texto

Descripción generada automáticamente

En la figura 9 se muestra el feature "Universo del Fomix", que incluye un conjunto de escenarios para probar la funcionalidad integral del sistema. Este archivo abarca la interacción entre diferentes módulos, como añadir, editar y eliminar productos, para garantizar que el sistema funcione de manera coherente cuando se utilizan varias funciones en secuencia.

* Archivo universoDelFomixPasos.java: Este archivo contiene los pasos en lenguaje Java que corresponden a los escenarios globales del proyecto, abarcando la funcionalidad integral del sistema "Universo del Fomix". Aquí se coordinan las operaciones de los distintos módulos, asegurando que funcionen en conjunto sin errores y validando que todas las acciones del usuario son manejadas correctamente por el sistema.

Figura 10

Texto

Descripción generada automáticamente*Archivo universoDelFomixPasos.java:*

La figura 10 presenta el archivo "universoDelFomixPasos.java", que implementa los pasos para probar el sistema de manera global. Aquí se coordinan las acciones de los diferentes módulos para asegurarse de que todas las funciones trabajen juntas sin problemas. Los métodos en este archivo validan que la interacción entre las diversas partes del sistema no cause conflictos y que los resultados sean consistentes con las expectativas del usuario.

1. **Ejecución de pruebas**

Los archivos .feature ejecutarán el código de las clases de pasos para cada caso de prueba. Estas clases sirven como un puente entre los escenarios de prueba y la implementación técnica que interactúa con la aplicación. Los métodos en estas clases inicializan el entorno de pruebas, aseguran que comience en un estado controlado, y validan si el sistema se comporta según lo esperado, generando reportes de fallas si es necesario. Cada paso en un archivo .feature es implementado por un método en la clase de pasos, que realiza acciones como navegar, llenar formularios, o interactuar con la interfaz. Al ejecutar los archivos .feature en Cucumber, se muestran los resultados de las pruebas en la terminal, indicando los escenarios y pasos exitosos, junto con el tiempo de ejecución. Además, las clases de pasos manejan casos especiales, garantizando la estabilidad de la aplicación frente a entradas inválidas o situaciones excepcionales.

Figura 11

*Captura de pantalla de un celular

Descripción generada automáticamenteTerminal de ejecución del archivo inicioSesion.feature*

Figura 12

*Terminal de ejecución del archivo añadirProductos.feature*

*Texto

Descripción generada automáticamente*

Figura 13

*Terminal de ejecución del archivo editarProductos.feature*

Texto

Descripción generada automáticamente

Figura 14

*Terminal de ejecución del archivo eliminarProductos.feature*

Texto

Descripción generada automáticamente

Figura 15

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente*Terminal de ejecución del archivo universoDelFomix.feature*

1. **Resultados obtenidos**

Los resultados obtenidos de las pruebas automatizadas se documentaron para cada archivo .feature ejecutado en el proyecto. A través de la ejecución de estos escenarios en Cucumber, se verificó el comportamiento del sistema en diversas situaciones, asegurando que las funcionalidades respondan conforme a las especificaciones. A continuación, se presenta un resumen de los resultados obtenidos, con capturas de pantalla que ilustran el desempeño y las áreas que requieren mejoras en cada caso.

A continuación, se visualizará imágenes donde consta los escenarios tanto el Passed como el Failed de los requisitos hechos con la herramienta “Cucumber”:

Imagen que contiene Escala de tiempo

Descripción generada automáticamenteFigura 15

Figura 16

Interfaz de usuario gráfica

Descripción generada automáticamenteFigura 17



Figura 18

Figura 19

Figura 20

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

Figura 21

Tabla

Descripción generada automáticamente

Figura 22

Imagen que contiene Escala de tiempo

Descripción generada automáticamente

Figura 23

Escala de tiempo

Descripción generada automáticamente

Figura 24

Imagen que contiene Escala de tiempo

Descripción generada automáticamente

1. **Conclusiones**
2. Este proceso de pruebas nos demostró que, cuando seguimos una buena metodología y usamos las herramientas adecuadas, podemos asegurarnos de que el sistema funcione como se espera ya que todo fue como un rompecabezas que armamos poco a poco hasta conseguir un resultado eficiente.
3. Las pruebas automatizadas no solo ayudaron a validar las funciones del sistema, sino que también nos permitieron identificar y corregir errores que de otra manera podrían haber pasado desapercibidos, fue como tener un control de calidad constante durante el desarrollo de este software.
4. La herramienta llamada “Cucumber” fue clave para realizar las pruebas de manera efectiva porque nos permitió simular diferentes escenarios y asegurarnos de que el sistema responde bien en cada uno de ellos, lo que hace que el software sea más robusto y confiable.
5. **Recomendaciones**
6. Aunque el sistema pasó las pruebas, es importante seguir buscando formas de optimizarlo y no conformarnos con que funcione bien sino también buscar cómo mejorarlo más en futuras versiones o actualizaciones.
7. En el desarrollo de este software surgieron muchas ideas de agregar más y más funciones al sistema, pero lo más importante es asegurarse de que cada nueva función sea realmente útil y que el sistema siga siendo fácil de usar para los q van a utilizarlo a diario.
8. Es importante planificar el mantenimiento continuo y las actualizaciones para que el sistema siga siendo confiable a largo plazo y también hay que estar atentos a posibles problemas y estar listos para solucionarlos rápidamente cuando surjan.