Diseño de un sistema de pruebas para evaluación de funcionalidad y desempeño del software desarrollado por la empresa TCS Solution Center

Elián Andrés Díaz Vargas

Trabajo de grado para optar el título de Ingeniero Mecatrónico

Director

Luis Fernando Perico Remolina

Magíster en Controles Industriales

Codirector

Diego Ricardo Páez Ardila

Magíster en Ingeniería Biomédica

Universidad Santo Tomás, Bucaramanga

División de ingenierías y arquitectura

Facultad de ingeniería mecatrónica

2023

Contenido

Lista de tablas

Lista de figuras

Lista de apéndices

Apéndice A. *MEC PR 03. Plan De Trabajo*

Apéndice B. *Plan_de_Pruebas_linio*

Apéndice C. *Metodologia_de_pruebas*

Apéndice D. *Plan_de_pruebas_despegar*

Apéndice E. *Funcionamiento_prueba_despegar*

Apéndice F. *Reporte_pruebas_serenity_despegar*

Apéndice G. *MEC_PR_01_Identificación Empresa*

Apéndice H. *Carta laboral practicantes-ELIAN ANDRES DIAZ VARGAS*

Apéndice I. *Certificado - Elián Andrés Díaz Vargas*

Nota: Los apéndices anteriores se encuentran externos y se pueden encontrar en la carpeta nombrada Apéndices.

Resumen

Para el presente proyecto se diseñó un banco de pruebas que permitió comprobar mediante la realización de pruebas de software que las funcionalidades de las aplicaciones desarrolladas por la empresa Tata Consultancy Services (TCS) se ejecutan correctamente, son acordes con los requerimientos dados por el cliente y no tienen fallos importantes que puedan ocasionar que funcione diferente a como se diseñó o que la aplicación salga a producción con problemas inesperados que hagan que falle. El desarrollo de la práctica consistió en identificar fallos mediante la aplicación manual y automatizada de pruebas al software desarrollado por la empresa, diseñar planes de pruebas de software para verificar el correcto funcionamiento del mismo, identificar el alcance y las funcionalidades de la aplicación a probar y el entorno de pruebas donde se van a realizar e implementar pruebas automatizadas de software bajo metodología de trabajo BDD, siguiendo el patrón de diseño Screenplay, utilizando el lenguaje de programación JAVA y la librería Serenity. Finalmente se concluye que el trabajo realizado en la práctica permitió aplicar los conocimientos teóricos adquiridos durante la carrera de Ingeniería Mecatrónica en un entorno laboral real, permitiendo aprender nuevas metodologías y herramientas para el desarrollo de software y pruebas de calidad, además de permitir la adquisición de habilidades y competencias importantes para el desempeño profesional en el campo del desarrollo de software.

Palabras clave: casos de prueba, patrón screenplay, aseguramiento de la calidad (QA), desarrollo guiado por comportamiento (BDD), serenity

Abstract

For this project, a test bench was designed that allowed us to verify through software tests that the functionalities of the applications developed by the company Tata Consultancy Services (TCS) are executed correctly, are in accordance with the requirements given by the client and They do not have major bugs that could cause the app to work differently than designed or cause the app to go into production with unexpected problems that cause it to crash. The development of the practice consisted of identifying failures through the manual and automated application of tests to the software developed by the company, designing software test plans to verify its correct operation, identifying the scope and functionalities of the application to be tested and the test environment where automated software tests will be carried out and implemented under the BDD work methodology, following the Screenplay design pattern, using the JAVA programming language and the Serenity library. Finally, it is concluded that the work carried out in practice allowed applying the theoretical knowledge acquired during the Mechatronics Engineering degree in a real work environment, allowing learning new methodologies and tools for software development and quality testing, as well as allowing the acquisition of important skills and competencies for professional performance in the field of software development.

Keywords: test cases, screenplay pattern, quality assurance (QA), behavior-driven development (BDD), serenity

Introducción

El concepto de desarrollo de software abarca un conjunto de actividades informáticas con el fin de diseñar y crear software para diferentes funcionalidades (sistemas operativos, softwares de programación, aplicaciones, entre otras), así como realizar su despliegue a producción y verificar su compatibilidad [1]. El sector del desarrollo de software es una de las áreas de la economía que ha tenido un gran crecimiento en la actualidad, esto debido en gran parte al auge mediado por la pandemia Covid-19, la cual provocó que varios sectores económicos migraran a plataformas mediadas por tecnología para poder seguir prestando los servicios o labores que se realizaban de manera presencial en empresas e instituciones. Por tal motivo, surge la necesidad en el mercado de invertir en tecnologías digitales, principalmente en el desarrollo de software y de esta manera mitigar de manera eficaz la demanda de aplicaciones [2].

La multinacional Tata Consultancy Services ofrece consultoría especializada en soluciones tecnológicas para satisfacer las diferentes necesidades de sus clientes, asignando equipos de trabajo que cumplan con los requisitos para estar en los proyectos que desarrolla la compañía. Cada equipo de trabajo está asignado a un proyecto, estos equipos se conforman siguiendo la metodología SCRUM, por lo tanto, cada persona dentro de él cumple un rol dentro del equipo. En los proyectos realizados por la empresa se realiza tanto el desarrollo del software como las pruebas para verificar la calidad de este, por lo tanto, y debido a que se deben hacer pruebas al software desarrollado en los proyectos, surge la necesidad de atraer personal que se encargue testear las funcionalidades del software desarrollado, dar el reporte de las pruebas realizadas y en consecuencia verificar que la calidad del software cumple con los requisitos del proyecto.

1. Diseño de un sistema de pruebas para evaluación de funcionalidad y desempeño del software desarrollado por la empresa TCS Solution Center

1.1 Reseña de la empresa

Figura 1. *Logo Tata Consultancy Services (TCS)*

Tomado de [3].

1.2 Misión

Nuestra misión es ayudar a los clientes a lograr sus objetivos comerciales al brindar consultoría, soluciones y servicios de TI innovadores y de la mejor calidad, y hacer que trabajar con nosotros sea un placer para todas las partes interesadas. [3]. Conectar a las personas con oportunidades en la economía digital, y construir caminos equitativos e inclusivos para mujeres, jóvenes y grupos marginados.

1.3 Visión

Visión para desvincular el crecimiento comercial y la huella ecológica de su operación para abordar el resultado final ambiental. visión de crecer de forma sostenible y ayudar a nuestros clientes a lograr un crecimiento sostenible a través de nuestras soluciones ecológicas y ofertas de servicios. [3].

1.4 Justificación

De acuerdo con la problemática existente, la empresa TCS SOLUTION CENTER requiere de un practicante de Ingeniería Mecatrónica para diseñar un banco de pruebas que permita comprobar que el software desarrollado cumpla con la calidad esperada por el cliente, así como ejecutar dichas pruebas de forma manual y automatizada en el entorno de pruebas apropiado (navegadores, dispositivos, sistemas operativos, en los que se utilizará el software).

1.5 Identificación de la problemática

Se requiere realizar pruebas de software para comprobar que las funcionalidades de las aplicaciones desarrolladas por la empresa se ejecutan correctamente, son acordes con los requerimientos dados por el cliente y no tienen fallos importantes que puedan ocasionar que funcione diferente a como se diseñó o que luego de que la aplicación salga a producción se encuentren problemas inesperados que hagan que la aplicación falle; por lo tanto, la empresa necesita personal para realizar pruebas de software y comprobar que el desarrollo realizado cumpla con la calidad esperada por el cliente. Las pruebas de software son el mecanismo mediante el cual se comprueba que un sistema se comporta de acuerdo con lo esperado, se realizan de forma manual o automatizada siguiendo una metodología de trabajo llamada BDD (Behavior Driven Development) la cual permite una mejor comunicación entre todas las partes involucradas y se centra en el comportamiento del sistema y el usuario.

2. Marco de referencia

2.1 Marco contextual

2.1.1 Conglomerado Tata

El grupo Tata tiene más de 150 años en el mercado desde su fundación en 1868, es un conglomerado que agrupa más de 100 empresas en 7 sectores comerciales, tiene presencia en 4 continentes (África, América, Asia y Europa), cuenta con más de 695.000 asociados a nivel mundial y es el conglomerado empresarial multinacional más grande en la India [4].

2.1.1.1 Tata Consultancy Services. Es una empresa que se especializa en consultoría y servicios relacionados con las tecnologías de la información TI con un alcance a nivel global. Ofrece servicios y soluciones a nivel empresarial, tecnológico y de ingeniería. Esta multinacional brinda sus servicios de consultoría en más de 55 países [5].

2.1.1.2 Tata Steel. Es la empresa con mayor producción de acero de la India, es uno de los mayores productores de acero de Europa y es una de las principales empresas siderúrgicas a nivel mundial, se distribuye en los 5 continentes con una cantidad de empleados de más de 65.000 [6].

2.1.1.3 Tata Power. Es una empresa que tiene como actividad principal generar, transmitir y distribuir electricidad, es la compañía eléctrica integrada más grande de la India; sus operaciones se encuentran en Bután, India, Indonesia, Singapur y Sudáfrica [7].

2.1.1.4 Tata Motors. Es el mayor fabricante de automóviles en India y se encuentra entre los principales fabricantes de automóviles del mundo, los productos que ofrecen van desde una amplia gama de automóviles hasta vehículos de defensa, pasando por vehículos deportivos, autobuses y camiones [8].

2.1.1.5 Tata Chemicals. Es una empresa india con operaciones a nivel global en India, Europa, América del Norte y África; es una de las empresas químicas más grandes de India. Tata Chemicals tiene intereses en productos químicos, protección de cultivos y productos químicos especializados [9].

2.1.1.6 Tata Communications. Es una empresa de telecomunicaciones de la India. Proporciona servicios de red y plataformas de red definidas por software como Ethernet, SD-WAM, redes de entrega de contenido (CDN), internet, conmutación de etiquetas multiprotocolo (MPLS) y líneas privadas. La compañía tiene más de 15 terabits/s de capacidad de ancho de banda internacional [10].

2.1.1.7 Tata Consumer Products. Es una empresa india de bienes de consumo de rápido movimiento, esta compañía es uno de los principales productores de café y es el segundo mayor fabricante y distribuidor de té del mundo. Actualmente la empresa opera en la industria de alimentos y bebidas [11].

2.2 Marco teórico

2.2.1 Desarrollo de software

En el desarrollo de software se busca que al implementar el código desarrollado este funcione correctamente y cumpla con los requerimientos dados o solicitados por un cliente, esto debido a que los proyectos relacionados con desarrollo de software tienden a tardar demasiado tiempo y recursos mayores a los planeados en el alcance de un proyecto. Por lo tanto, dado que en los proyectos de software en algunas ocasiones puede que no se cumplan los requerimientos en términos de funcionalidad, presupuesto o tiempo, es indispensable que se realice una correcta gestión de proyectos para garantizar que se siga un orden especifico en las tareas de desarrollo de software [12].

2.2.1.1 Planificación. Corresponde al primer paso que se debe dar en la construcción de cualquier proyecto de desarrollo de software, en este se recolectan las especificaciones o funcionalidades que el cliente requiere que cumpla el software, por lo tanto, el equipo de trabajo debe definir los requerimientos del proyecto y estos deben estar en concordancia con los requerimientos dados por el cliente. De igual manera se debe construir una matriz de requisitos que permitan asegurar la calidad del desarrollo y se debe realizar un análisis de riesgos para el proyecto [13].

2.2.1.2 Implementación. Es la parte del proceso de desarrollo de software en la que los desarrolladores escriben el código para implementar las funcionalidades del proyecto, las cuales deben estar directamente relacionadas con los requerimientos dados en la etapa anterior. El

desarrollo suele hacerse partiendo de pruebas unitarias, las cuales consisten en realizar primero las pruebas para probar una funcionalidad en específico del código en el que se está trabajando y posteriormente se escribe el código que realiza dicha funcionalidad. En este punto también se realizan las pruebas de caja blanca, en las cuales se tiene en cuenta la estructura y el código del producto que se va a probar, y las pruebas de caja negra, donde solo se tiene en cuenta las entradas que recibe una funcionalidad del software y las salidas que produce, suelen ser pruebas desde la interfaz de usuario [14].

2.2.1.3 Pruebas de software. Las pruebas se encargan de identificar los errores o fallos en el software lo más pronto posible, intentando que dichos errores o fallos no salgan a etapa de despliegue lo cual conllevaría costos elevados y retrasos en la entrega del producto; las pruebas de software también verifican que el producto desarrollado hace lo que se supone que debería hacer y las funcionalidades implementadas son acordes a los requerimientos del cliente. Por lo tanto, las pruebas son una parte primordial en el desarrollo de software ya que reducen los costos en términos de tiempo y dinero en el desarrollo además que mejora el rendimiento del software desarrollado [15].

- Pruebas Unitarias: Estas pruebas se deben realizar durante la etapa de desarrollo de cada uno de los requisitos o funcionalidades pedidos por el cliente, en estas pruebas se probarán cada uno de los componentes de código desarrollado para cada una de las funcionalidades de la aplicación.
- Pruebas de integración: Con estas pruebas se busca encontrar errores o comprobar el correcto funcionamiento de la aplicación cuando los componentes desarrollados se unan.

- Pruebas de regresión: Con este tipo de pruebas se busca que al hacer cambios a algún componente de la aplicación no se generen errores adicionales en otros componentes. Para disminuir costos y tiempo en las actividades, estas pruebas se realizan en su mayoría automatizadas.
- Pruebas de aceptación: El objetivo de estas pruebas es validar que la implementación de los requerimientos o funcionalidades pedidas por el cliente cumplan con el funcionamiento esperado. Por lo tanto, se busca comprobar que la aplicación cumpla con los requerimientos del cliente.
- Pruebas de UI: con estas pruebas se busca identificar que la interfaz gráfica es la adecuada para el negocio y para el público al que va dirigida, con el fin de que el cliente se sienta conforme con el desarrollo logrado.
- Pruebas de Performance: Se realizan para probar el rendimiento de la aplicación sea bueno al implementar los requerimientos pedidos por el cliente, se realizan a lo largo del desarrollo y contribuyen a que se tenga una buena calidad de software.
- Pruebas de seguridad: Se realizan con el fin de asegurar que la información de un usuario al ingresar a la aplicación no sea infiltrada por algún agente externo, probar que no existan vulnerabilidades y en general garantizar a los usuarios de la aplicación que la información suministrada y las transacciones realizadas en la aplicación estarán protegidas y será seguro usarla.
- Pruebas E2E: Con estas pruebas se comprueba que la aplicación funciona correctamente al probarla por completo, que cumpla con los requisitos establecidos por el cliente y que en general todo su funcionamiento suceda como se espera por el negocio.

- Pruebas Exploratorias: Se realizan con el fin de encontrar posibles errores que no fueron encontrados por las demás pruebas y comprobar que la aplicación funciona correctamente.

2.2.1.4 Documentación. La documentación en los proyectos de desarrollo de software es importante debido a que facilita procesos de mejora y mantenimiento del código, tanto en las etapas de desarrollo del proyecto, como en etapas posteriores al despliegue, con fines de la mantenibilidad de este [16].

2.2.1.5 Despliegue. El despliegue de un software se realiza cuando hayan sido probadas todas sus funcionalidades para garantizar que en producción no se encuentren errores o fallos importantes [17].

2.2.1.6 Mantenimiento. Uno de los puntos a tener en cuenta en desarrollo de software y está directamente relacionado con los costos del proyecto es la mantenibilidad del software, es decir, se busca que el software pueda ser modificado en el tiempo sin mayor dificultad en términos de tiempo y dinero. Para lo cual es necesario utilizar buenas prácticas de programación, por ejemplo, al momento de escribir el código, la utilización de la notación Camel Case para clases y métodos. Aunque cada empresa de desarrollo de software puede tener especificadas las prácticas de programación que se llevan a cabo en ella y las condiciones o requerimientos para que el software pueda ser mantenible en el tiempo [18] [19].

3. Objetivos

3.1 Objetivo general

Diseñar un sistema de pruebas manuales y automatizadas para garantizar la calidad, funcionalidad y desempeño de un software haciendo uso de patrones de diseño Screenplay y POM, así como, de librerías de Java.

3.2 Objetivos específicos

- Identificar fallos mediante la aplicación manual y automatizada de pruebas al software desarrollado por la empresa TCS SOLUTION CENTER, para evitar que salga a producción funcionando incorrectamente o con errores, afectando la calidad esperada por el cliente.
- Diseñar planes de pruebas de software para verificar el correcto funcionamiento del mismo, identificando el alcance, las funcionalidades de la aplicación a probar y el entorno de pruebas donde se van a realizar.
- Implementar pruebas automatizadas de software bajo metodología de trabajo BDD, siguiendo el patrón de diseño Screenplay, utilizando el lenguaje de programación JAVA y la librería Serenity.

4. Desarrollo de la práctica

4.1 Capacitaciones generalidades

Durante el desarrollo de la práctica inicialmente se realizaron capacitaciones en diferentes conceptos relacionados con el desarrollo de software (Ver Tabla 1), las capacitaciones de generalidades corresponden con los fundamentos básicos de programación, metodologías ágiles y conceptos básicos en algunos lenguajes de programación. Se reforzaron conocimientos en programación orientada a objetos aplicándolos en lenguajes de programación como C, Java. Se obtuvo conocimiento en estructuras de datos, formas de organizar la información, patrones de diseño y fundamentos de programación como recursividad, creación de clases y objetos, bases de datos, entre otros.

Tabla 1. *Capacitaciones generalidades*

Capacitaciones Realizadas
Introducción a algoritmos para programación
Algoritmos y Estructuras de Datos
Aprende SCRUM
Fundamentos de la Programación: Estructura de Datos
.NET esencial
Java esencial
HTML esencial
CSS esencial

4.2 Capacitación Java

Durante la capacitación de Java que inicialmente estuvo enfocada en la sintaxis del código, se aprendió a realizar operaciones básicas como la asignación de variables, creación y llamado de funciones, los tipos de variables, programación orientada a objetos, conexión con bases de datos, organización del código en paquetes y patrones de diseño. Algunos patrones de diseño en lo que se estuvo trabajando fueron Singleton para restringir la creación de instancias de un mismo objeto de una clase, utilizado principalmente al crear la conexión con una base de datos, el patrón Screenplay utilizado en pruebas de software. De igual manera se aprendió acerca de bases de datos relacionales MySQL y a realizar la conexión con Java. Por último, durante las capacitaciones en Java se aprendió a escribir código utilizando buenas prácticas de programación, por ejemplo, el uso de Camel Case o Snake Case para nombrar clases, variables, métodos o funciones (Ver Tabla 2).

Tabla 2. *Capacitaciones Java*

Capacitaciones Realizadas
Java - Fundamentos básicos
Java - Programación Orientada a Objetos
Java - Estructuras básicas
Java - Avanzado
Java - Código robusto
Java - Patrones de Diseño
Java - Buenas prácticas

4.3 Capacitación JavaScript

Durante la Capacitación en el lenguaje de programación JavaScript se aprendió acerca de los fundamentos básicos como lo son los tipos de variable, programación orientada a objetos y en general como se programa en JavaScript. Dentro de las funcionalidades de JavaScript está el ser un lenguaje desarrollado para ser utilizado en páginas web. De igual manera dentro de las capacitaciones de este lenguaje se estuvo escribiendo código JavaScript o Typescript, que permita implementar el backend y frontend para que el usuario pueda interactuar con la página web, también se utilizaron bases de datos relacionales y no relacionales como MySQL y MongoDB para acceder y guardar información de una página web. Se realizaron entornos de ejecución multiplataforma como Nodejs, para ejecutar código JavaScript del lado del servidor y desarrollar el backend de una página web, se utilizaron Frameworks de JavaScript como angular, para realizar el frontend de un sitio web (Ver Tabla 3).

Para el desarrollo de la capacitación, se aprendió acerca de las herramientas para la creación de páginas web tales como JavaScript, Typescript, CSS, bases de datos, Nodejs, Angular, donde el frontend como el backend deben implementarse correctamente para que no haya problemas de comunicación entre ellos. Así mismo, en el desarrollo de páginas web se debe tener un diseño y formato de estilos CSS, que permitan una interfaz con el usuario intuitiva. Por último, se aprendió acerca del uso de microservicios utilizando API-REST para consumirlos en una aplicación.

Tabla 3. *Capacitaciones JavaScript*

Capacitaciones Realizadas
Desarrollo Web con JavaScript
jQuery, Angular, NodeJS
JavaScript Avanzado

4.4 Capacitación Python

Durante la capacitación de Python se aprendió inicialmente sobre los fundamentos básicos, su sintaxis, declaración de variables, funciones y programación orientada a objetos. Posteriormente, se obtuvo conocimiento en manejo de ficheros de texto, binarios, CSV y JSON, Utilización de módulos como NumPy y Matplotlib para analizar datos, Django para desarrollar páginas web dinámicas y Turtle para hacer gráficos primitivos, también se aprendió sobre manejo de bases de datos y creación de peticiones asíncronas utilizando la API Fetch de JavaScript (Ver Tabla 4).

Tabla 4. *Capacitaciones Python*

Capacitaciones Realizadas
Python - Fundamentos básicos
Python - desarrollo web
Python - HTML, CSS, MySQL
Python - Desarrollo Web Backend
Python - Django Framework

4.5 Capacitación Quality Assurance (QA)

Para la capacitación de QA, se estuvo aprendiendo acerca de los diferentes tipos de pruebas de software (Ver Tabla 5), las plataformas donde se pueden realizar dichas pruebas, las librerías o frameworks que se deben utilizar para el desarrollo de pruebas, realización de casos de pruebas y en general aprendiendo sobre metodología y ejecución de pruebas. Principalmente se estuvo trabajando en el uso del entorno de pruebas Selenium y Serenity bajo patrones POM (Page Object Model) y Screenplay respectivamente; el patrón de diseño POM es un patrón de diseño que trabaja con elementos web almacenando cada página de una aplicación en una clase, este patrón también

evita el código duplicado y mejora el mantenimiento de los casos de prueba. Se tomaron conceptos de desarrollo guiado por pruebas (TDD – Test Driven Development) utilizando la librería Junit y se complementaron los conceptos sobre buenas prácticas de programación al hacer uso de los principios SOLID, estos principios son guías para escribir el código de forma que sea entendible, mantenible y no se tenga que refactorizar el código debido a malos diseños o malas prácticas de programación. El desarrollo guiado por pruebas (TDD) consiste en hacer las pruebas para comprobar el funcionamiento del componente que se va a realizar antes de escribir el código de dicho componente, es decir, el desarrollo se hace con el objetivo de superar las pruebas.

Tabla 5. *Capacitaciones QA*

Capacitaciones Realizadas
Introducción al Testing de Software
Testing & Quality Assurance - Guía para Pruebas de Software
SCRUM
Diseño de pruebas de software (software testing)
Programación Orientada a Objetos
Test unitarios con Junit - introducción
JavaScript - TDD y pruebas unitarias
Introducción a Git y GitHub
Java - Gestor de proyectos Maven
Java - Gestor de proyectos Gradle
QA Automation con Serenity y Selenium
Principios SOLID y Clean Code

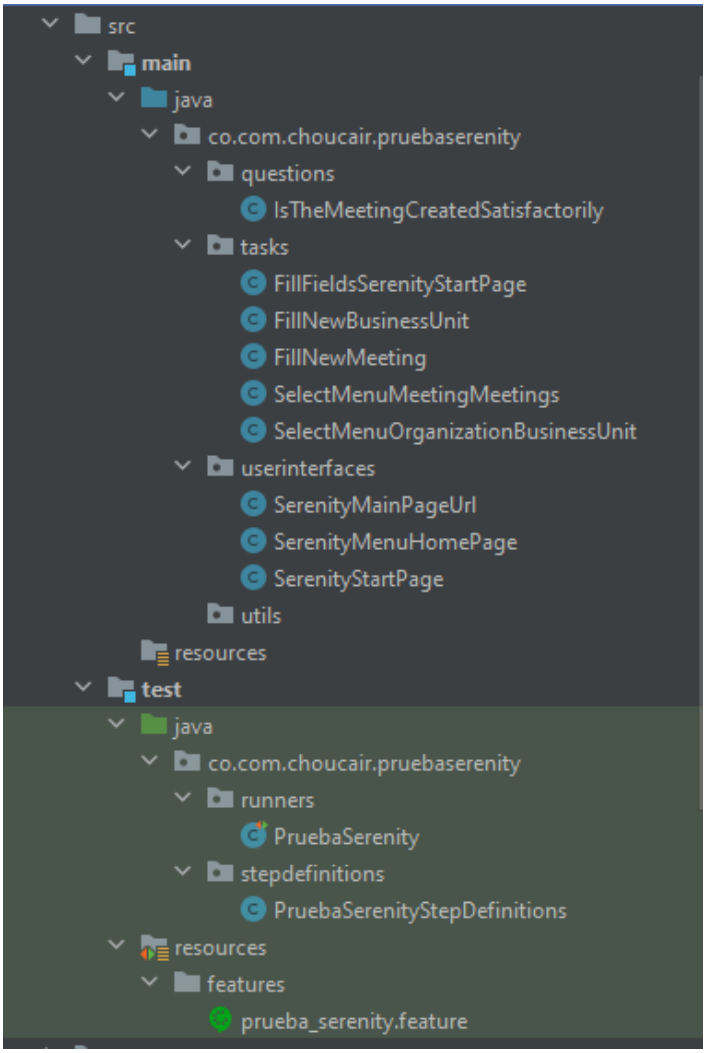
4.6 Etapa practica

Posterior a las capacitaciones fui asignado a un proyecto de la empresa, donde realicé diferentes pruebas automatizadas para las aplicaciones que me asignaban, donde me daban los criterios de aceptación de las pruebas y de acuerdo a estos criterios se diseñaron los casos de prueba que se implementaban en el IDE IntelliJ Idea, en el lenguaje de programación Java, haciendo uso del entorno de pruebas Serenity y el patrón de diseño Screenplay, bajo la metodología de trabajo BDD (Ver apéndices del B al F).

5. Desarrollo pruebas de software

Durante el desarrollo de las practicas se desarrolló código para ejecutar pruebas automatizadas utilizando el patrón de diseño Screenplay. Este patrón de diseño consiste en estandarizar el código desarrollado para las pruebas automatizadas en secciones o paquetes para que este pueda ser entendido y mantenido con mayor facilidad a través del tiempo; tiene como objetivo la organización del código y facilitar la programación de las pruebas, para lo cual, en este patrón para el desarrollo de las pruebas se trabaja con un “Actor” que viene siendo un usuario y las pruebas son desarrolladas de tal forma que describan como ese actor puede interactuar con la aplicación para alcanzar un objetivo específico, el actor puede realizar acciones como presionar un botón, hacer preguntas y puede navegar por la aplicación web. Finalmente, para aplicar el patrón Screenplay se definió el esquema presentado en la figura 2.

Figura 2. *Patrón Screenplay*



Dentro de la carpeta “main” (Ver Figura 2) se crea un paquete con el nombre del proyecto (co.com.choucair.pruebaserenity), y dentro de él se crean sub-paquetes donde se van a implementar el mapeo de los objetos del DOM (Document Object Model) que se van a utilizar en la automatización, las tareas que se quieren ejecutar, las validaciones que se requieran para comprobar que la aplicación funciona correctamente y el paquete “utils”, donde se colocan clases, archivos de texto y en general cualquier archivo que sea requerido para ejecutar las pruebas (como documentos con información para las validaciones).

Dentro de la carpeta “tests” se definen las pruebas o pasos que se van a seguir en la automatización de la prueba, se crea el archivo se ejecución de las pruebas y se establecen las características que se van a probar de la aplicación.

Descripción de la funcionalidad de los paquetes utilizados:

5.1 Paquete “questions”

En este paquete se colocan las clases de java que contienen las validaciones que se requieren hacer para comprobar el correcto funcionamiento de la página, por ejemplo, que al darle clic a un botón en la página se cargue un mensaje en específico. Estas clases se encargan de tomar información que muestra la página web cuando un usuario hace una acción en específico para posteriormente comparar esa información con datos de prueba y verificar que el mensaje que ve el usuario es el correcto.

5.2 Paquete “tasks”

En este paquete se colocan las clases de java que tienen programadas las tareas o acciones que se van a probar en la página, es decir, se programa en estas clases rutinas específicas para cada sección a probar en la aplicación, por ejemplo, darle clic a un botón, posteriormente mover un slider o llenar un formulario y darle clic en el botón que guarda los datos. Todas las acciones que un usuario pueda hacer en la página pueden ser probadas o programadas es las clases que contienen el paquete “tasks”. En las clases de este paquete se utilizan los elementos mapeados del DOM que se encuentran en las clases del paquete “userinterfaces”.

5.3 Paquete “userinterfaces”

En ese paquete se colocan las clases de java que contienen el mapeo de los objetos del DOM (Document Object Model), es decir, donde se hace referencia a los objetos HTML de la sección de la página donde se va a realizar una prueba automatizada, estos objetos se declaran en la clase como constantes. Los objetos del DOM mapeados hacen parte de la interfaz de usuario con la que una persona interactúa.

5.4 Paquete “utils”

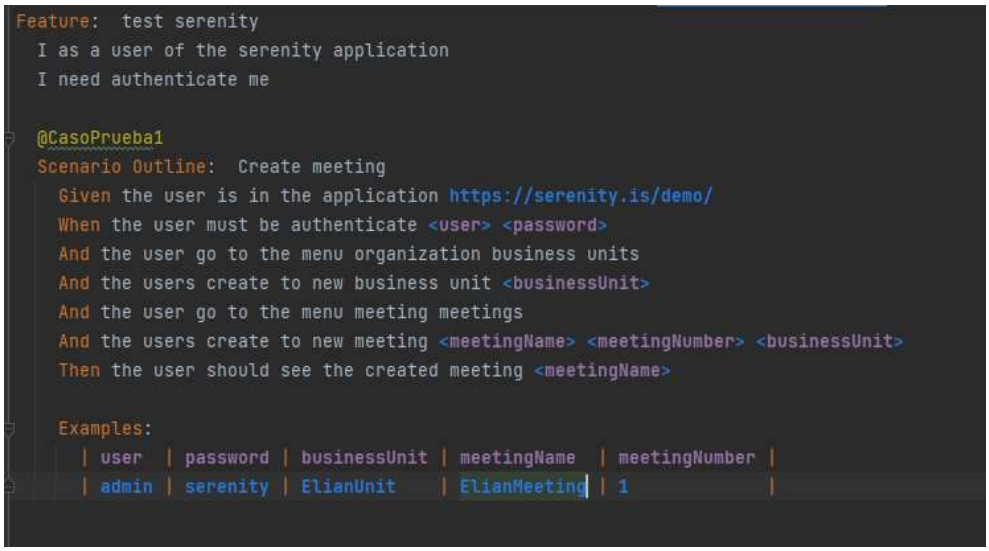
Este paquete es utilizado para crear clases, colocar archivos “. txt”, y en general para colocar cualquier tipo de información que deba ser utilizada para ejecutar correctamente las pruebas, por ejemplo, se pueden crear clases para crear, leer o guardar archivos planos, que contengan información para hacer las validaciones de la prueba o donde se guarde información extraída de la página.

5.5 Paquete “features”

Este paquete contiene los archivos “. feature” (Ver Figura 3), lo cuales son los archivos donde se definen los casos de prueba que se van a ejecutar en la prueba automatizada. Para crear estos archivos se utiliza Cucumber BDD, la cual es una herramienta para el desarrollo de software que admite la metodología BDD (Behavior Driven Development - Desarrollo Guiado por Comportamiento), donde se utiliza un analizador de lenguaje ordinario llamado ”Gherkin”, el cual permite definir los casos de prueba como pasos secuenciales que pueden ser leídos y entendidos tanto por las personas que realizan las pruebas como por el cliente y en general que pueda ser entendido por cualquier persona que participe del proyecto y no tenga conocimiento en lenguajes

de programación, es decir se trata de utilizar el lenguaje natural para crear los casos de prueba, haciendo que estos puedan ser entendidos por cualquier persona.

Figura 3. *Ejemplo Feature*



```
Feature:  test serenity
  I as a user of the serenity application
  I need authenticate me

  @CasoPrueba1
  Scenario Outline:  Create meeting
    Given the user is in the application https://serenity.is/demo/
    When the user must be authenticate <user> <password>
    And the user go to the menu organization business units
    And the users create to new business unit <businessUnit>
    And the user go to the menu meeting meetings
    And the users create to new meeting <meetingName> <meetingNumber> <businessUnit>
    Then the user should see the created meeting <meetingName>

    Examples:
      | user  | password | businessUnit | meetingName  | meetingNumber |
      | admin | serenity | ElianUnit    | ElianMeeting | 1             |
```

5.6 Paquete “runners”

En este paquete se coloca la clase de java con la información de la ubicación del feature que se va a ejecutar, el nombre del paquete del proyecto de la prueba, el tipo de nomenclatura que se está usando en el proyecto (camelCase) y el caso de prueba que se desea ejecutar en caso de que haya más de uno y solo se desee ejecutar uno de ellos.

5.7 Paquete “stepdefinitions”

Este paquete contiene una clase de java donde se encuentran los pasos que se deben ir cumpliendo para cada caso de prueba definido en él archivo “. feature” del paquete feature. En esta clase se llaman las clases que se crearon en los paquetes de “questions” y “tasks”, en un orden

específico para que se ejecuten los pasos de la función de la página que se está probando, por lo tanto, primero deben estar creadas las clases de los paquetes “questions” y “tasks” con la programación de la prueba que se desea automatizar.

6. Resultados

Los resultados de las capacitaciones consistieron en evaluaciones que fueron hechas por los mentores pertenecientes a Kibernum Academy y posteriormente un certificado que acredita el haber cursado y aprobado los cursos o mentorías dadas en los temas mencionados en las capacitaciones (Ver apéndice I).

Posterior a las capacitaciones, en el proyecto de la empresa al que fui asignado, se realizaron pruebas automatizadas de software en Java bajo la metodología de trabajo BDD, utilizando el patrón de diseño Screenplay, así como haciendo uso de Serenity. Los resultados de las pruebas de software corresponden con informes realizados por Serenity durante la ejecución de las pruebas, estos informes se generan automáticamente y son renderizados en una página web (ver apéndices E y F), donde se puede observar si los casos de prueba ejecutados en la automatización fueron comprobados satisfactoriamente, lo que indica que la aplicación está funcionando correctamente, o por el contrario, el caso de prueba falló, lo que indica que hay algún problema en la aplicación; en caso de que un caso de prueba falle, el reporte realizado por Serenity indicará donde está el fallo con capturas del lugar donde ocurrió.

7. Conclusiones

De acuerdo con la información expuesta sobre las capacitaciones se puede evidenciar el fortalecimiento de conocimientos en lógica y lenguajes de programación vistos en el pregrado de Ingeniería Mecatrónica, al igual que se afianzaron competencias en lenguaje de programación Java y Python, orientadas en el desarrollo y pruebas de software.

Se puede concluir que en el desarrollo de software se adquieren y desarrollan competencias en lenguaje de programación JavaScript, el cual es comúnmente utilizado en desarrollo web, comprendiendo los elementos por los que está compuesta una aplicación web, identificando los elementos HTML, JavaScript y CSS, orientadas principalmente en desarrollo de software y realización de pruebas de software.

En el desarrollo profesional de software las pruebas se realizan sistemáticamente utilizando tecnologías como Serenity y Selenium que son librerías que permiten trabajar con elementos Web, donde se deben contar con competencias en Automatización QA en el lenguaje de programación Java o Python, identificando casos de prueba donde la aplicación web funcione con normalidad, así como casos en los que esta pueda fallar, desarrollar la automatización para pruebas funcionales, no funcionales y de regresión para asegurar una buena calidad en el desarrollo de software.

En conclusión, el desarrollo de las pruebas dentro de la empresa se implementaron pruebas automatizadas de software bajo la metodología de trabajo BDD, haciendo uso del patrón de diseño Screenplay y la librería de pruebas Serenity. Todo el desarrollo de las pruebas se realizó en el IDE IntelliJ Idea con el lenguaje de programación Java, se diseñaron planes de pruebas que garantizaron la calidad y las funcionalidades especificadas en los criterios de aceptación de una prueba.

Referencias

[1] "¿Qué es el desarrollo de software? | IBM". IBM - Deutschland | IBM. https://www.ibm.com/co-es/topics/software-development (accedido el 20 de octubre de 2022).

[2] "Claves para entender por qué el desarrollo de software se encuentra en auge". GBSYS. https://www.gbsys.com/claves-para-entender-por-que-el-desarrollo-de-software-se-encuentra-en-auge/#:~:text=En%20síntesis,%20como%20consecuencia%20de,nuevas%20aplicaciones%20que%20les%20permitan (accedido el 20 de octubre de 2022).

[3] "From leadership to events: All about tata consultancy services". IT Consulting Services & Business Solutions | Tata Consultancy Services (TCS). https://www.tcs.com/about-us#0 (accedido el 20 de octubre de 2022).

[4] "About Us | Tata group". The Tata group. Leadership with Trust. https://www.tata.com/about-us (accedido el 20 de octubre de 2022).

[5] "From leadership to events: All about tata consultancy services". IT Consulting Services & Business Solutions | Tata Consultancy Services (TCS). https://www.tcs.com/about-us (accedido el 20 de octubre de 2022).

[6] "Tata steel | company profile & details". Steel Supplier & Manufacturer in India | Tata Steel. https://www.tatasteel.com/corporate/our-organisation/company-profile/ (accedido el 20 de octubre de 2022).

[7] "Tata power - vision, mission and values". Tata Power - India's Largest Integrated Power Company. https://www.tatapower.com/corporate/vision-mission-values.aspx (accedido el 20 de octubre de 2022).

[8] "Tata motors - car manufacturers in India, OEM automotive companies". Tata Motors Limited | Largest Indian Automobile Manufacturer. https://www.tatamotors.com/about-us/company-profile/ (accedido el 20 de octubre de 2022).

[9] "Company profile - about us - tata chemicals limited". Tata Chemicals Limited. https://www.tatachemicals.com/about-us/company-profile (accedido el 20 de octubre de 2022).

[10] "About | tata communications". Tata Communications. https://www.tatacommunications.com/about/ (accedido el 20 de octubre de 2022).

[11] "About | tata consumer products". Tata Consumer Products. https://www.tataconsumer.com/about (accedido el 20 de octubre de 2022).

[12] "Proceso de desarrollo de aplicaciones software". Universidad de Cantabria. https://www.ctr.unican.es/asignaturas/MC_OO/Doc/OO_08_I2_Proceso.pdf (accedido el 20 de octubre de 2022).

[13] P. Concepción. "Artículo: "planificación de proyectos de software" - pedro concepción". Carreras de Informática y Sistemas --- UMSS. https://www.cs.umss.edu.bo/doc/material/mat_gral_130/costo%20del%20software.htm (accedido el 20 de octubre de 2022).

[14] D. J. Colina Gómez, J. C. Pineda y C. Gaitán. "Desarrollo, implementación y puesta en marcha de aplicativo móvil para la gestión y atención de solicitudes e incidentes de usuarios de servicios claro." Universidad Santo Tomás. https://repository.usta.edu.co/bitstream/handle/11634/9915/PinedaJuan2016.pdf (accedido el 20 de octubre de 2022).

[15] "¿Qué son las pruebas de software y cómo funcionan? | IBM". IBM - Deutschland | IBM. https://www.ibm.com/co-es/topics/software-testing (accedido el 20 de octubre de 2022).

[16] "Guía para la documentación de proyectos de software". Universidad Nacional del Sur. https://cs.uns.edu.ar/~ldm/mypage/data/oc/info/guia_para_la_documentacion_de_proyectos_de_software.pdf (accedido el 20 de octubre de 2022).

[17] SUPER INTENDENCIA DE SOCIEDADES. "Entrega y despliegue de sistemas de información". SuperSociedades. https://www.supersociedades.gov.co/sgi/Obsoletos/00_Documentos_Obsoletos_SGI/24_DID/GINT-PR-014_EntregaDespliegueSI_v2.pdf (accedido el 20 de octubre de 2022).

[18] J. Adones Farfán y V. Vega Zepeda, "Mantenibilidad del Software. Consideraciones para su especificación y validación", Ingeniare, vol. 28, n.º 4, diciembre de 2020. Accedido el 20 de octubre de 2022. [En línea]. Disponible: https://www.scielo.cl/scielo.php?pid=S0718-33052020000400654&script=sci_arttext&tlng=en.

[19] F. Ruiz y M. Polo. "Mantenimiento del software". Escuela superior de informática universidad de castilla-la mancha. https://alarcos.esi.uclm.es/per/fruiz/curs/mso/trans/S3.pdf (accedido el 20 de octubre de 2022).