UPN
UNIVERSIDAD PRIVADA DEL NORTE

FACULTAD DE INGENIERÍA

Carrera de Ingeniería de Sistemas Computacionales

"SCRIPT DE AUTOMATIZACIÓN PARA EL CONTROL FUNCIONAL DE MÓDULOS EN EL SISTEMA DE FACTURACIÓN DE LA EMPRESA ASYSTEM COMPANY PERU, LIMA 2024."

Trabajo de suficiencia profesional para optar el título profesional de:

INGENIERO DE SISTEMAS COMPUTACIONALES

Autor:

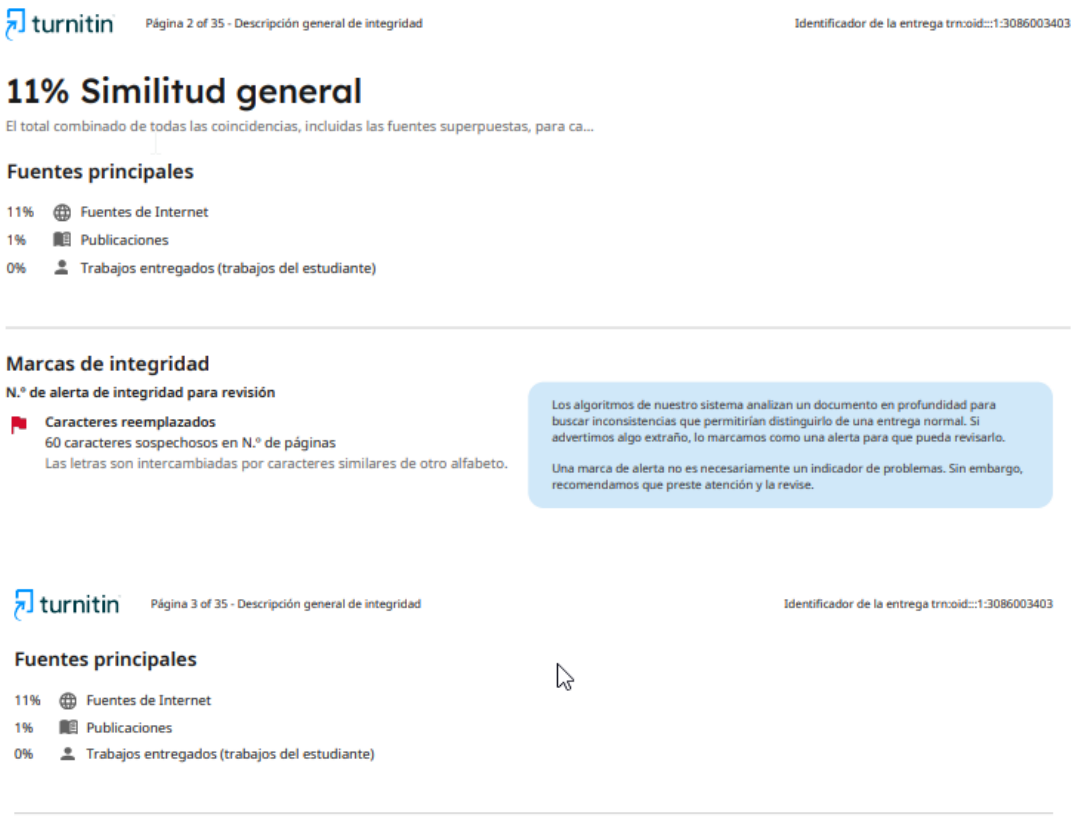
David Brian Urbano Osorio

Asesor:

Ing. Raúl Eduardo Huarote Zegarra
https://orcid.org/0000-0001-7466-7404
Lima - Perú

2024

INFORME DE SIMILITUD



turnitin Página 2 of 35 - Descripción general de integridad Identificador de la entrega trn:oid:::1:3086003403

11% Similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para ca...

Fuentes principales

11% Fuentes de Internet
1% Publicaciones
0% Trabajos entregados (trabajos del estudiante)

Marcas de integridad

N.º de alerta de integridad para revisión

Caracteres reemplazados
60 caracteres sospechosos en N.º de páginas
Las letras son intercambiadas por caracteres similares de otro alfabeto.

Los algoritmos de nuestro sistema analizan un documento en profundidad para buscar inconsistencias que permitirían distinguirlo de una entrega normal. Si advertimos algo extraño, lo marcamos como una alerta para que pueda revisarlo.

Una marca de alerta no es necesariamente un indicador de problemas. Sin embargo, recomendamos que preste atención y la revise.

turnitin Página 3 of 35 - Descripción general de integridad Identificador de la entrega trn:oid:::1:3086003403

Fuentes principales

11% Fuentes de Internet
1% Publicaciones
0% Trabajos entregados (trabajos del estudiante)

DEDICATORIA

Dedico este trabajo a mis padres por ser mi mayor fuente de inspiración y sacrificio para poder lograr las cosas que me propongo cada día.

A mi asesor, porque sin su orientación y consejo no se hubiera culminado este trabajo.

A todas las personas que, me animaron a que de alguna manera pueda lograr este trabajo.

AGRADECIMIENTO

Agradezco a mis padres por siempre brindarme su apoyo incondicional siempre para poder lograr mis sueños.

Asimismo, agradezco a mis hermanos que siempre me alientan a seguir adelante y no rendirme.

A mi novia que es un motor adicional para seguir luchando por mis sueños.

Finalmente agradecer a mi asesor quien siempre me ayudo en resolver las dudas que tuve para realizar este trabajo.

TABLA DE CONTENIDO

INDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El procedimiento de creación de Scripts para pruebas automatizadas de software para una aplicación web para una organización que brinda servicios de facturación se describe en este trabajo profesional a fin de garantizar la eficiencia y funcionalidad de las pruebas funcionales sen los módulos de sistema. Asimismo, la empresa gestiona múltiples proyectos utilizando un marco de trabajo tradicional, lo que resulta tedioso porque no hay reducción de tiempo durante la fase de ejecución de Scripts en las pruebas funcionales de los módulos o porque el producto fue entregado más tarde de lo esperado. Como resultado, se produjeron fallos de producción,

En ese sentido, el equipo de calidad (QA) incluirá el marco de trabajo Scrum en sus procedimientos y operaciones a petición del cliente. Con respecto a los resultados y beneficios de este trabajo, se puede ver fehacientemente en cuanto a tiempo y ahorro de dinero, para efectos de la investigación pues si ejecutamos toda la Test Suite, la cual consta de 5 casos de prueba automatizados se ejecutarán en 10 minutos aproximadamente, mientras que, si lo ejecutamos de la forma manual, tomaría alrededor de 1 hora. Esto representará un importante ahorro monetario y de tiempo para la empresa.

Palabras Clave: Script de Automatización de pruebas funcionales, pruebas de software, Marco de trabajo ágil, scrum, aplicación web, control de módulos.

INTRODUCCIÓN

Dentro del sector Facturación, el cual en la actualidad es muy competitivo, la eficacia y la calidad del software desarrollado en Perú son esenciales para brindar servicios seguros y confiables a los clientes. La garantía de calidad del software desempeña un papel crucial en este proceso, y las pruebas de software son una parte primordial de esta garantía. En un entorno en constante evolución, la necesidad de optimizar los tiempos de ejecución de pruebas se convierte en un desafío crítico. Este trabajo se centra en la implementación de Scripts de pruebas automatizadas como un planteamiento clave para mejorar el desempeño de las pruebas en la empresa. Para lograrlo, se utiliza Selenium, una herramienta ampliamente reconocida para la automatización de pruebas web, que permite simular la interacción del usuario con aplicaciones web y evaluar su funcionalidad de manera sistemática y repetible. Además, se incorpora Scrum, un marco de trabajo ágil, para gestionar y organizar el proceso de desarrollo y pruebas del software de la empresa. Scrum promueve la colaboración interdisciplinaria y la entrega continua de funcionalidades, lo que es fundamental para mantenerse al día con las cambiantes demandas del mercado.

El propósito de este trabajo para la obtención de la suficiencia profesional es demostrar cómo la implementación de scripts de pruebas automatizadas con Selenium, la adopción de Scrum como marco de trabajo, pueden optimizar significativamente la duración del tiempo de ejecución de las pruebas de los requerimientos funcionales de los módulos del sistema. Esto denota también en la reducción de costos de los proyectos de desarrollo de Software de la empresa, contribuye también a la mejora de la calidad del sistema y a que este sea adaptable a las exigencias cambiantes del mercado.

En el transcurso de este trabajo, exploraremos el procedimiento de implementación de Scripts de pruebas automatizadas, a partir de la planificación y elaboración de los casos de

prueba hasta la realización y mantenimiento. Al final de esta tesis, se espera proporcionar una guía sólida para otras organizaciones que buscan optimizar sus procesos de pruebas a través de la automatización y enfoques ágiles. La aplicación de esta estrategia no solo mejora la eficiencia en la validación de software, sino que también fortalece la capacidad de adaptación de las organizaciones en un entorno tecnológico en constante evolución.

Para su comodidad, este trabajo ha sido separado en los siguientes seis capítulos:

- En el CAPÍTULO I – MARCO TEORICO, definimos conceptos a utilizarse dentro de este trabajo de tesis.
- En el CAPÍTULO II - DESCRIPCIÓN DE LA EXPERIENCIA, se mencionará a la empresa en la cual se hará la solución.
- En el CAPÍTULO III - ACTIVIDADES DESARROLLADAS, se describirá el problema, las etapas, los objetivos, los alcances, del marco de trabajo implementado, la solución y el diagnóstico final.
- En el CAPÍTULO IV - CONCLUSIONES Y RECOMENDACIONES, se explican los resultados y sugerencias del proyecto realizado con base en los resultados obtenidos.

CAPÍTULO I. MARCO TEÓRICO

Las pruebas automáticas son pruebas de software que no requieren de intervención humana alguna para crearlas, ejecutarlas, evaluarlas e informarlas; son autónomos en todas sus magnitudes. A la vez las pruebas automatizadas al utilizar un equipo hay participación del recurso en las distintas magnitudes. En consecuencia, la automatización facilita verificar el software en ambientes de prueba reales en cualquier circunstancia, y es un proceso autónomo que permite la retroalimentación inmediata de los resultados si un software se rige con los criterios de calidad" (Duquino, A., 2020).

A si mismo Cabrera Serna, J., & Pareja Verastegui en su investigación “Automatización de Pruebas Funcionales Web para mejorar el Área de Calidad de Software en una empresa del rubro de Retails en el año 2021”. enfatizan en que la automatización de pruebas funcionales es una práctica fundamental para reducir el esfuerzo manual en las pruebas, permitiendo una mayor cobertura de escenarios de prueba de manera más eficiente.

Debemos agregar que en el artículo de investigación “Propuesta de Automatización de Pruebas Funcionales Durante el Ciclo de Vida del Software en Desoft” se indica que automatizar las pruebas tiene el objetivo de disminuir su tiempo de ejecución, aumentar la calidad del proceso en cuanto a cantidad de errores encontrados y aumentar la profesionalidad de los especialistas probadores con la implementación de un plan de capacitación sobre la automatización de las pruebas funcionales.

En las siguientes paginas definimos conceptos que se utilizaran en la realización de este documento.

1. SCRIPT DE AUTOMATIZACIÓN:

"Un script de automatización para pruebas funcionales es un conjunto de instrucciones que se ejecutan en un orden específico para realizar una tarea específica, como la automatización de pruebas de interfaz de usuario o la automatización de pruebas de rendimiento." (López et al., 2020) además Hernández et al., 2019, p. 15 en su investigación Automatización de pruebas de software utilizando Selenium Web Driver nos dice que "La automatización de pruebas funcionales es una práctica común en el desarrollo de software, ya que permite reducir el tiempo y el esfuerzo necesarios para la realización de pruebas, y mejorar la calidad del software"

CONTROL FUNCIONAL:

"El control funcional de módulos es esencial para garantizar que cada componente del sistema opere de acuerdo con las especificaciones establecidas, lo que contribuye a la estabilidad y confiabilidad del software" (García et al., 2021, p. 45). algo semejante se menciona en la investigación de Martínez & López, 2020, p. 78 donde se dice que "Implementar un control funcional efectivo en los módulos permite detectar errores en etapas tempranas del desarrollo, lo que reduce costos y mejora la calidad del producto final"

SISTEMA DE FACTURACIÓN:

Un sistema de facturación se define como una importante herramienta de gestión financiera para una empresa que puede emitir facturas, realizar un seguimiento de los pagos y generar informes financieros.

Según López (2022), "un sistema de facturación eficiente no solo facilita la emisión de documentos fiscales, sino que también optimiza la gestión de cobros y mejora la relación con los clientes" (p. 15).

Además, Martínez y Pérez (2021) destacan que "la implementación de un sistema de facturación automatizado reduce significativamente los errores humanos y acelera el proceso de facturación, lo que se traduce en una mayor satisfacción del cliente" (p. 30).

METODOLOGÍA AGILES:

Según Atlassian (s.f.), "La metodología ágil es un enfoque de gestión de proyectos que implica dividir el proyecto en fases y enfatiza la colaboración y mejora continua" (párr. 1). Esta definición destaca la importancia del trabajo en equipo y la adaptabilidad en la gestión de proyectos. El Manifiesto Ágil articula valores clave que guían las prácticas ágiles. Establece que "valoramos a los individuos y las interacciones sobre los procesos y las herramientas" y "Software funcionando sobre documentación exhaustiva" (Beck et al., 2001). Estos principios destacan la importancia de la colaboración humana y la entrega de productos funcionales en lugar de enfocarse en la documentación.

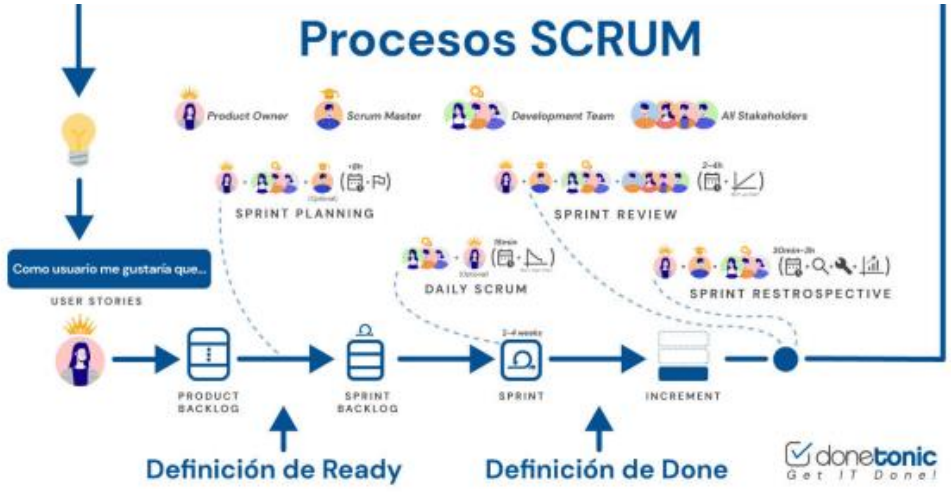
MARCO DE TRABAJO SCRUM:

El Marco de trabajo Scrum es un marco ágil centrado en ofrecer productos funcionales en un entorno de equipo colaborativo y flexible. Esto es útil para proyectos en lo que se requieren entregas rápidas y que puedan adaptarse a los cambios.

Según Scrum.org (s.f.), "Scrum es un marco de trabajo que ayuda a los equipos a trabajar juntos" (párr. 1). Esta definición destaca la importancia del trabajo en equipo y la colaboración en la gestión de proyectos. El marco de trabajo Scrum define tres roles clave: el Product Owner, el Scrum Master y el equipo de desarrollo. Según Schwaber y Sutherland (2017), "El Product Owner es responsable de maximizar el valor del producto" (p. 4), mientras que el Scrum Master "es responsable de garantizar que el equipo de Scrum entienda y siga las prácticas de Scrum" (p. 5).

Scrum se caracteriza por sus iteraciones, conocidas como Sprints. Según Scrum.org (s.f.), "Un Sprint es un período de tiempo fijo durante el cual se desarrolla un conjunto de trabajo" (párr. 1). Estas iteraciones permiten a los equipos entregar productos funcionales de manera regular y recibir retroalimentación continua.

Figura 1: Procesos Scrum



MAVEN:

Maven es una herramienta de automatización utilizada en la construcción de software y también para gestionar y construir proyectos de código. Este resulta muy útil en un entorno de desarrollo de software donde la gestión de dependencias y la construcción de sistemas relacionados con el desarrollo de software son métodos críticos utilizados en la industria.

Según Apache Maven (s.f.), "Maven es una herramienta de automatización de construcción de software que se utiliza para gestionar y construir proyectos de código" (párr. 1). Esta definición destaca la importancia de la automatización y la gestión de proyectos en el desarrollo de software. Maven se utiliza para gestionar dependencias de proyectos de código. Según Apache Maven (s.f.), "Maven maneja las dependencias de un proyecto mediante la declaración de dependencias en el archivo pom.xml" (párr. 2). Esta función permite a los desarrolladores gestionar fácilmente las dependencias de sus proyectos además Maven se utiliza para construir proyectos de código de manera automatizada. Según Apache Maven (s.f.), "Maven proporciona una forma sencilla de construir y empaquetar proyectos de código" (párr. 3). Esta función permite a los desarrolladores construir y empaquetar proyectos de manera rápida y eficiente.

JAVA:

Java es un lenguaje de programación de alto valor utilizado para desarrollar una amplia gama de sistemas, como aplicaciones web, móviles y de escritorio. Java principal característica es que es una plataforma independiente, esto implica que los programas desarrollados en Java pueden funcionar en cualquier equipo con una máquina virtual JAVA. Según Oracle (s.f.), "Java es un lenguaje de programación de alto nivel que se utiliza para desarrollar aplicaciones de software" (párr. 1). Esta definición destaca la

importancia de Java como lenguaje de programación para el desarrollo de software. Java se usa en varias aplicaciones, incluyendo web, móviles y de escritorio. Según Tutorials Point (s.f.), "Java se utiliza en aplicaciones como Android, web, desktop, juegos, etc." (párr. 1). Esta variedad de usos hace que Java sea un lenguaje de programación muy popular entre los desarrolladores.

SELENIUM LIBRARY:

Selenium es una herramienta que nos ayuda a probar sitios web y asegurarnos de que funcionan correctamente. Es muy útil para comprobar si los sitios web y las aplicaciones se ven y funcionan de la misma manera en diferentes navegadores y dispositivos. Según Selenium.dev (s.f.), "Selenium es una herramienta de automatización de pruebas de software que se utiliza para probar aplicaciones web" (párr. 1). Esta definición destaca la importancia de Selenium como herramienta de automatización de pruebas para aplicaciones web.

Selenium es conocido por su capacidad de probar aplicaciones web en diferentes navegadores y plataformas. Según Selenium.dev (s.f.), "Selenium soporta una variedad de navegadores, incluyendo Chrome, Firefox, Safari, Edge, Internet Explorer y Opera" (párr. 2). Esta característica permite a los desarrolladores probar sus aplicaciones web en diferentes entornos sin necesidad de modificar el código.

Selenium se utiliza en una variedad de industrias, incluyendo la industria de la tecnología, la industria financiera y la industria de la salud. Según Guru99 (s.f.), "Selenium es ampliamente utilizado en la industria de la tecnología para probar aplicaciones web" (párr. 1). Esta variedad de usos hace que Selenium sea una herramienta muy popular entre los desarrolladores y testers.

JUNIT:

Junit es una herramienta que te ayuda a escribir y ejecutar pruebas para tu código Java, asegurándote de que todo funcione como debería. Es muy importante asegurarse de que el código sea confiable y no tenga ningún error.

Según JUnit.org (s.f.), "JUnit es una biblioteca de pruebas de unidad para Java que se utiliza para escribir y ejecutar pruebas de software" (párr. 1). Esta definición destaca la importancia de JUnit como herramienta de pruebas de unidad para Java. JUnit es conocido por su capacidad de escribir pruebas de unidad para clases y métodos individuales. Según JUnit.org (s.f.), "JUnit proporciona una forma sencilla de escribir pruebas de unidad para clases y métodos individuales" (párr. 2). Esta característica permite a los desarrolladores probar individualmente las partes del código para garantizar que funcionen correctamente. JUnit se utiliza en una variedad de industrias, incluyendo la industria de la tecnología, la industria financiera y la industria de la salud. Según Tutorials Point (s.f.), "JUnit es ampliamente utilizado en la industria de la tecnología para probar aplicaciones Java" (párr. 1). Esta variedad de usos hace que JUnit sea una herramienta muy popular entre los desarrolladores y testers.

CAPÍTULO II. DESCRIPCIÓN DE LA EXPERIENCIA

En el año 2023, me uní a ASYSTEM COMPANY PERU como Analista de Calidad en el área de Proyectos e Implementaciones. En este cargo, he tenido la oportunidad de trabajar gestionando y desarrollando proyectos de desarrollo de automatizaciones utilizando lenguajes de programación como Java, Python, framework Junit, POM en entornos de prueba como Selenium.

En mi rol, he sido responsable de garantizar la funcionalidad correcta de los módulos del sistema de la empresa, lo que ha implicado realizar pruebas y análisis exhaustivos para identificar y solucionar problemas. También he trabajado en estrecha colaboración con el equipo de desarrollo para implementar mejoras y soluciones innovadoras.

En un proyecto reciente, se me asignó la tarea de implementar un script de automatización para la validación de distintos requerimientos funcionales de módulos en el sistema de facturación de la empresa. Para ello, realicé el análisis de los requisitos y necesidades de la empresa, y luego desarrollar el script de automatización utilizando Java y Selenium.

Durante el proceso de desarrollo, surgieron nuevas ideas y propuestas que requirieron realizar pruebas y análisis adicionales para implementar las mejoras respectivas. Sin embargo, gracias a mi experiencia y habilidades en el área, pude garantizar que el Script de automatización se desarrollara de manera eficiente y efectiva, y que cumpliera con todo lo especificado en los casos de prueba.

Mi experiencia como Analista de Calidad en ASYSTEM COMPANY PERU ha sido enriquecedora y ha permitido desarrollar mis habilidades en el desarrollo de automatizaciones.

CAPÍTULO III. ACTIVIDADES DESARROLLADAS:

Identificación del problema

La empresa Asystem company Perú es una empresa joven en crecimiento en nuestro País, la .

do a la validación de requerimientos funcionales en los diferentes módulos del sistema.

En la actualidad la empresa viene implementando distintas medidas para mitigar el costo de horas hombre respecto del tiempo que toma realizar las pruebas funcionales que son repetitivas y rutinarias, en este proceso se ha identificado que, dentro del ciclo de desarrollo del software, la empresa no tiene implementado como recurso, la automatización de pruebas funcionales, el cual permitiría la reducción de costo en horas hombre y el rápido despliegue de distintos requerimientos y proyectos.

En la actualidad se sigue manteniendo el formato de pruebas manuales y exhaustivas para la validación de los requerimientos funcionales, esto en algunas ocasiones ha provocado el retraso en la entrega de los proyectos o incluso el rechazo del acta de aceptación porque en el checklist del proyecto se identificaban algunos errores de validación.

A consecuencia de ello se está optando por la implantación de scripts de automatización para la validación de las pruebas funcionales, esto permitirá tener un control eficiente de los recursos en horas hombre y le permitirá ejecutar el script de automatización en distintos entornos de prueba en simultaneo esto supone una mejora en cuanto a la administración de horas hombre y aportara en mejorar la gestión de los equipos para su asignación en distintos proyectos.

Objetivos

- Reducir el tiempo y el esfuerzo humano necesario para realizar tareas de validación de requerimientos en los módulos del sistema.
- Desarrollar script de automatización para mejorar la precisión y la exactitud de la validación de requerimientos funcionales en los módulos del sistema.

Estrategias

Se opto por el marco de trabajo Scrum que nos permite llevar una correcta gestión del proyecto. Por lo que se aplica las Etapas de Scrum para nuestro trabajo de creación de script automatización de pruebas funcionales:

Inicio:

Definición de Objetivos: En esta etapa, se establecerían los objetivos específicos de la tesis. Se debe definir claramente qué aspectos de las pruebas funcionales deseas investigar y mejorar.

Planificación y Creación del Backlog: Identificación de Historias de Usuario: Identificar historias de usuario relacionadas con la automatización de pruebas. Por ejemplo, podrían ser historias como "Automatizar las pruebas de ingreso de nuevo cliente" o "Automatizar las pruebas de registro de facturas".

Sprint 1: Investigación y Diseño Inicial:

Este sprint podría centrarse en la revisión de la información existente acerca de automatización de pruebas funcionales. Debemos identificar las tendencias, herramientas y mejores prácticas.

Diseño de la Estructura de Pruebas: Debemos diseñar una estructura base para las pruebas automatizadas. Se debe incluir la definición de los casos de prueba, escenarios de los test y la elección de herramientas de automatización.

Sprint 2: Desarrollo de Scripts de Automatización, pruebas Iniciales:

Escribe scripts iniciales para automatizar pruebas funcionales de acuerdo con la estructura de pruebas definida en el sprint anterior.

Sprint 3: Implementación y ejecución de Pruebas Automatizadas:

En este Sprint se ejecuta las pruebas automatizadas dentro de un entorno de prueba. Se documenta los resultados y se realiza los ajustes en los scripts según sea necesario.

Sprint 4: Análisis de Resultados:

Comparamos la eficiencia y la cobertura con las pruebas manuales y documentamos los hallazgos y examinamos los resultados de pruebas automatizadas.

En la tabla que presentaremos a continuación analizamos el mapeo de tiempos de ejecución de pruebas manuales esto constituye el resumen de la ejecución del plan de pruebas del módulo se describe la relación de los tiempos tomados en cada prueba manual.

Tabla 1:
Resumen del plan de pruebas Manuales

Nombre del Test	Descripción	Duración (min)	Comentarios
CP-001	Registrar factura con serie F001	0.15	Demora por validación e ingreso de data de prueba
CP-002	Registrar boleta con serie B001	0.15	Demora por validación e ingreso de data de prueba
CP-003	Registrar factura con operación de exportación	0.3	Demora por validación e ingreso de data de prueba y demora en el ambiente de prueba

CP-004	Registrar boleta con operación no domiciliada	0.3	Demora por validación e ingreso de data de prueba y demora en el ambiente de prueba
CP-005	Registrar compra interna	0.1	Sin novedad
Total		1 hora	

Podemos decir que un set de 5 CP se ejecuto en 1 hora, este tiempo es no solo por el tipo de prueba si no porque para las pruebas manuales se tiene dependencia como son accesos a los entornos de prueba desplegados, espera en la validación de la data de prueba disponible y entre otros factores operativos como el llenado de data manual.

En la tablas 2 describimos el resumen de tiempos de las pruebas automatizadas para el mismo set de prueba de 5 CP y verificamos que la reducción de tiempo es considerables

Tabla 2:
Resumen del plan de pruebas Automatizado

Nombre del Test	**Descripción**	**Duración (min)**	**Comentarios**
CP-001	Registrar factura con serie F001	0.046	Sin novedad (data precargada)
CP-002	Registrar boleta con serie B001	0.07	Demora por validación de series disponibles, y servicio de resumen diario
CP-003	Registrar factura con operación de exportación	0.033	Demora por validación de números de DAM para la exportación
CP-004	Registrar boleta con operación no domiciliada	0.02	Sin novedad
CP-005	Registrar compra interna	0.01	Sin novedad
Total		10m 44 segundos	

Realizamos los ajustes y mejoras en las pruebas automatizadas en función de los resultados y la retroalimentación.

Podemos determinar que el tiempo empleado en las pruebas automatizadas reduce considerablemente respecto de las pruebas manuales, en función de lo expuesto en las tablas 1 y 2 los resultados concluyen que reducimos el tiempo en un 80% este porcentaje puede ser relativo dependiendo de la suite de pruebas y de la complejidad de cada caso de prueba.

Sprint 5: Validación de Resultados:

Validamos si la automatización de las pruebas funcionales aportó al cumplimento de los objetivos definidos en la etapa de inicio.

En base a lo anterior citado, las etapas del presente trabajo serán:

Planificación:

En esta etapa seleccionaremos las historias de usuario para automatizar:

El team de desarrollo Scrum identifica las historias de usuario, estas serán candidatas para la automatización de pruebas. Las historias por lo general son aquellas que se consideran críticas o que se espera que estén presentes durante múltiples Sprints.

Definición de criterios de aceptación: Se definen los criterios de aceptación para cada historia de usuario, estos deben incluir los casos de prueba que se encuentran mapeadas en la tabla 1y estas deben pasar para que la historia se considere completada con éxito.

Tabla 3:
Diseño de casos de prueba

ID del Caso de Prueba	Descripción	Entrada	Resultado Esperado	Estado
CP-001	Registrar factura con serie F001	Tipo de comprobante: Factura Serie: F001 Tipo de operación: Venta interna Moneda: Soles	Comprobante registrado exitosamente.	Pendiente
CP-002	Registrar boleta con serie B001	Tipo de comprobante: Boleta Serie: B001 Tipo de operación: Venta interna Moneda: dólares americanos	Comprobante registrado exitosamente.	Pendiente
CP-003	Registrar factura con operación de exportación	Tipo de comprobante: Factura Serie: F001 Tipo de operación: Exportación de Bienes Moneda: Soles	Comprobante registrado exitosamente.	Pendiente
CP-004	Registrar boleta con operación no domiciliada	Tipo de comprobante: Boleta Serie: B001 Tipo de operación: Ventas no domiciliados que no califican como exportación Moneda: dólares americanos	Comprobante registrado exitosamente.	Pendiente
CP-005	Registrar compra interna	Tipo de comprobante: Factura Serie: F001 Tipo de operación: Compra interna Moneda: Soles	Comprobante registrado exitosamente.	Pendiente

Nota: Elaboración propia

Desarrollo:

Diseño de casos de prueba automatizados:

Los Analistas de Calidad diseñan los casos de prueba automatizados en función de los criterios de aceptación de las historias de usuario que fueron seleccionados por el equipo Scrum. Se espera que estos casos de prueba sean claros y específicos, se debe describir las interacciones esperadas con los módulos del Sistema.

Tabla 4:

Diseño de pruebas automatizadas (proceso de registro de Factura)

Caso de Prueba 1: Registrar Factura con Serie F001 y Operación Venta Interna		
Método:	Datos de Prueba	Resultado esperado:
registrarFacturaF001VentaInterna()	Tipo de comprobante: Factura Serie: F001 Tipo de operación: Venta Interna Moneda: Soles	Comprobante registrado exitosamente.

Nota: Elaboración propia

Tabla 5:

Diseño de pruebas automatizadas (proceso de registro de Boleta)

Caso de Prueba 2: Registrar Boleta con Serie B001 y Operación Venta Interna		
Método:	Datos de Prueba	Resultado esperado:
registrarBoletaB001VentaInterna()	Tipo de comprobante: Boleta Serie: B001 Tipo de operación: Venta Interna Moneda: Dólares Americanos	Comprobante registrado exitosamente.

Nota: Elaboración propia

Tabla 6:

Diseño de pruebas automatizadas (proceso de registro de Factura con Operación Exportación de Bienes)

Caso de Prueba 3: Registrar Factura con Operación Exportación de Bienes		
Método:	Datos de Prueba	Resultado esperado:
registrarFacturaExportacionBienes()	Tipo de comprobante: Factura Serie: F001 Tipo de operación: Exportación de Bienes Moneda: Soles	Comprobante registrado exitosamente.

Nota: Elaboración propia

Tabla 7:

Diseño de pruebas automatizadas (proceso de registro de Boleta con Operación Ventas no Domiciliados)

Caso de Prueba 4: Registrar Boleta con Operación Ventas no Domiciliados		
Método:	Datos de Prueba	Resultado esperado:
registrarBoletaVentasNoDomiciliados()	Tipo de comprobante: Boleta Serie: B001 Tipo de operación: Ventas no Domiciliados que no califican como exportación Moneda: Dólares Americanos	Comprobante registrado exitosamente.

Nota: Elaboración propia

Tabla 8:

Diseño de pruebas automatizadas (proceso de registro de Compra Interna)

Caso de prueba 5: Registrar Compra Interna		
Método:	Datos de Prueba	Resultado esperado:
registrarCompraInterna()	Tipo de comprobante: Factura Serie: F001 Tipo de operación: Compra Interna Moneda: Soles	Comprobante registrado exitosamente.

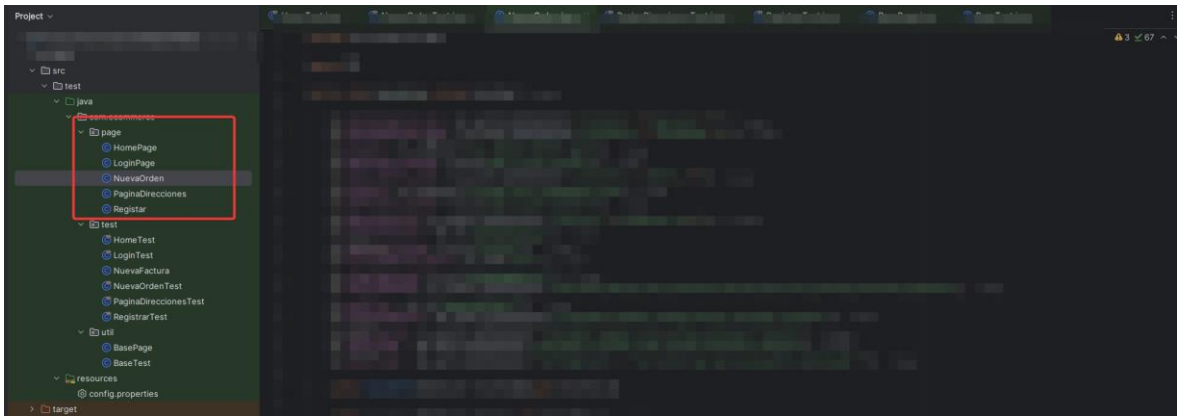
Nota: Elaboración propia

Desarrollo de scripts de automatización: Los analistas de Calidad o los ingenieros de automatización desarrollan scripts de automatización de pruebas utilizando las herramientas tecnológicas adecuadas. Estos scripts están diseñados para ejecuta pruebas automáticamente, interactuando con la aplicación y validando su comportamiento.

Describimos el marco utilizado en la investigación, la estructura del proyecto (POM) para la automatización de pruebas se organiza de la siguiente manera:

PAGE: Este paquete es clave dentro del patrón de diseño Page Object Model (POM) el cual ayuda a mantener el código de pruebas limpio y ordenado. Cada clase dentro de este paquete se ocupa de una página de la aplicación y contiene la lógica para acceder a los elementos de esa página, esto lo podemos denotar en la Figura 2 en la sección enmarcada de rojo.

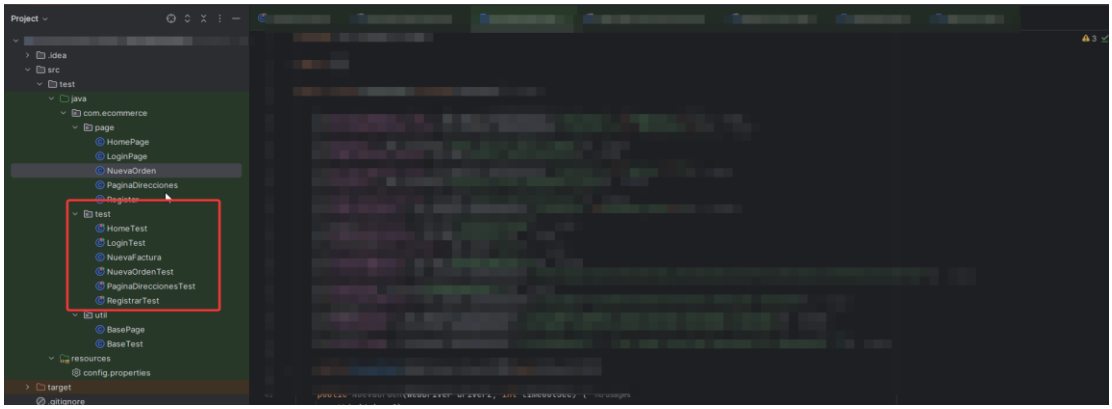
Figura 2:
Imagen estructura PAGE



Nota: Elaboración propia (la imagen enmarca en rojo la sección del PAGE que está compuesto por clases que individualmente se ocuparan de una página de la aplicación).

Test: En la Figura 3 abordamos la explicación del paquete TEST aquí es donde se colocan las clases que contienen los métodos de prueba. Usando TestNG, se pueden definir diferentes grupos de pruebas, configuraciones y afirmaciones para comprobar el comportamiento de la aplicación. Este es el paquete en el que se realiza las pruebas automatizadas, lo podemos visualizar en la sección enmarcada en color rojo.

Figura 3:
Imagen estructura TEST



Nota: Elaboración propia

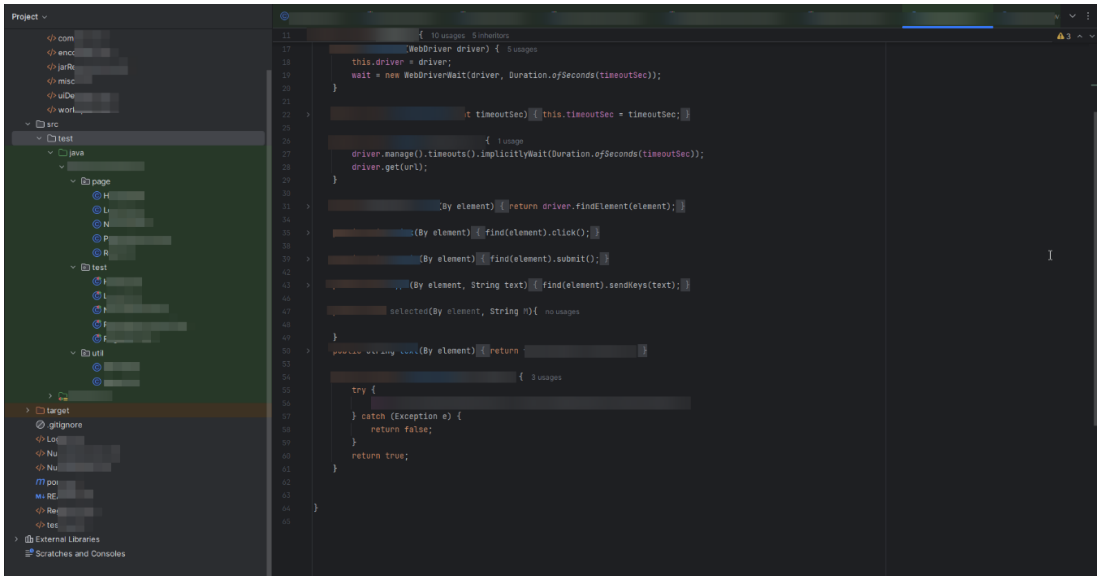
UTIL: Este paquete es bastante útil para encapsular clases que realizan ciertas funciones que pueden implementarse en otros lugares a lo largo del proyecto. Como lo podemos ver en la figura 4 que para efectos de la investigación se ha mapeado 2 clases, puedes tener una clase dedicada a leer el archivo de configuración config.properties, o a realizar la apertura y cierre del navegador.

Figura 4:
Imagen estructura UTIL

En la imagen 5 denotamos el mapeo de las clases del paquete UTIL, la clase base es la que proporcionara los métodos útiles para interactuar con los elementos del sistema web y que forma parte de un paquete de utilidades que proporciona métodos comunes que facilitaran la interacción con la interfaz de usuario.

En esta clase importaremos las clases necesarias del Selenium y java para manejar la automatización y declararemos los métodos recurrentes.

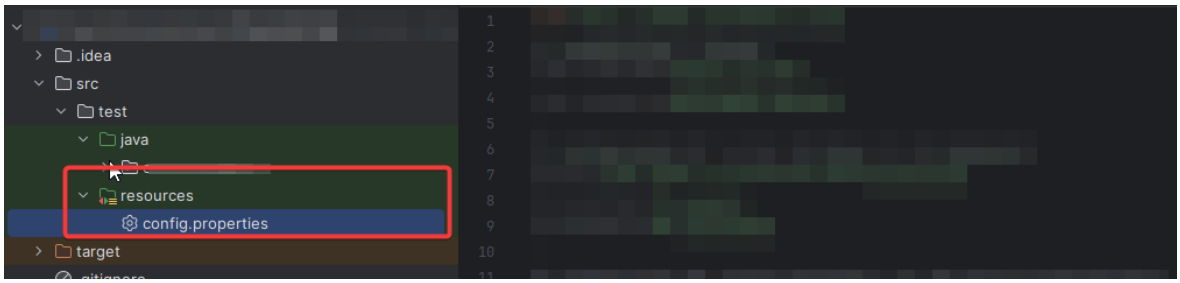
Figura 5:
Clase Base.



Nota: Elaboración propia

RESOURCES: En la figura 5 mostramos el mapeo del archivo config.properties, este archivo es crucial para ejecutar la aplicación en un entorno de pruebas. Contiene datos tales como la URL de la aplicación, acceso a la cuenta de usuario, y otros parámetros que son variables dependiendo del ambiente (desarrollo, pruebas, producción). Esto hace que el código se mantenga con facilidad.

Figura 6:
Imagen estructura RESOURCE



Nota: Elaboración propia

En esta sección del proyecto también mapearemos los locator que nos permitirán identificar e interactuar con componentes específicos del sistema, existen muchos tipos de locators, para este proyecto estaremos utilizando locators como ID, nombre, clase, Xpath y CSS Selector.

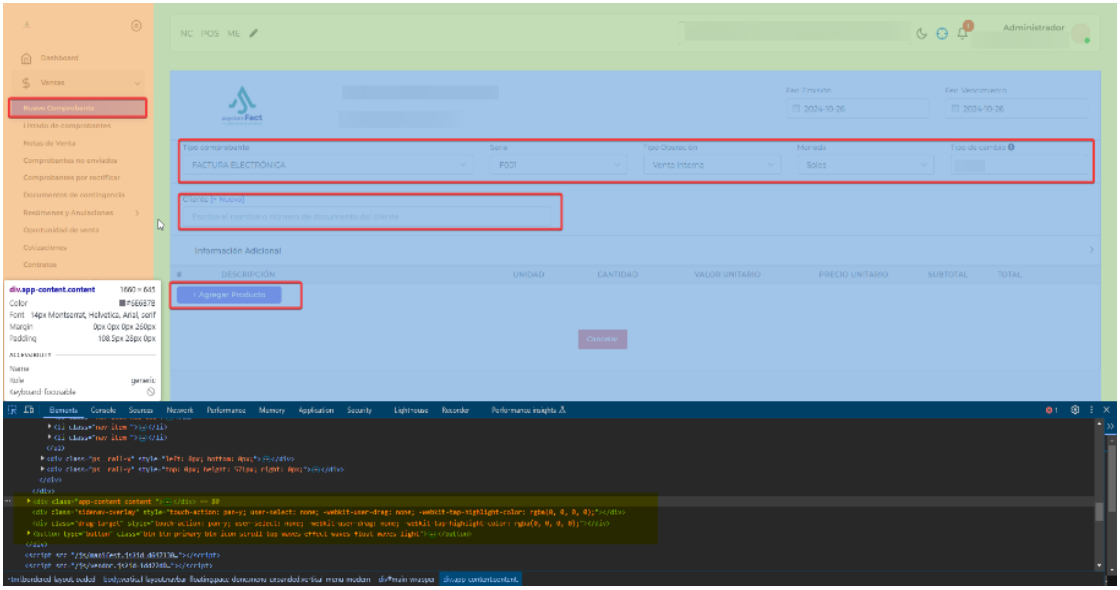
Para nosotros el uso del locator ID es más rápido y confiable no obstante los Xpath también ofrecen una mayor flexibilidad para navegar en elementos o estructuras más complejos. En la siguiente tabla mapearemos los locator de los objetos marcados en rojo en la figura 7 y la figura 8.

Tabla 9:
Mapeo de Locators para registrar un nuevo comprobante

Elemento	Descripción	Locator	Tipo Locator
<li class="active">	Elemento de lista activo	//li[@class='active']	XPath
Nuevo Comprobante	Atributo del span	//span[@data-i18n='Nuevo Comprobante']	XPath
Tipo comprobante	Etiqueta para el tipo de comprobante	//label[text()='Tipo comprobante']	XPath
Factura Electrónica	Opción seleccionable en el dropdown	//div[@dusk='document_type_id']//span[text()='FACTURA ELECTRÓNICA']	XPath
Boleta De Venta Electrónica	Opción seleccionable en el dropdown	//div[@dusk='document_type_id']//span[text()='BOLETA DE VENTA ELECTRÓNICA']	XPath
Establecimiento	Etiqueta del campo	//label[text()='Establecimiento']	XPath

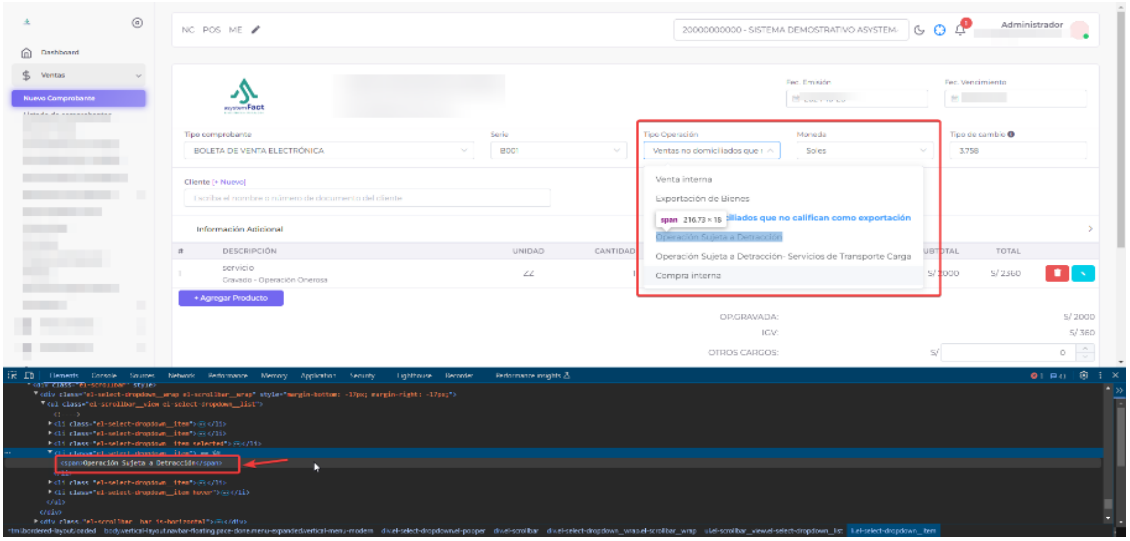
	"Establecimiento"		
Serie	Etiqueta del campo "Serie"	//label[text()='Serie']	XPath
Tipo Operación	Etiqueta del campo "Tipo Operación"	//label[text()='Tipo Operación']	XPath
Tipo de cambio	Tipo de cambio	//label[text()='Tipo de cambio']	XPath

Figura 7:
Mapeo de objetos de la ventana registrar un nuevo comprobante.



Nota: Elaboración propia

Figura 8:
Mapeo de objetos de la ventana registrar nueva factura tipo comprobante



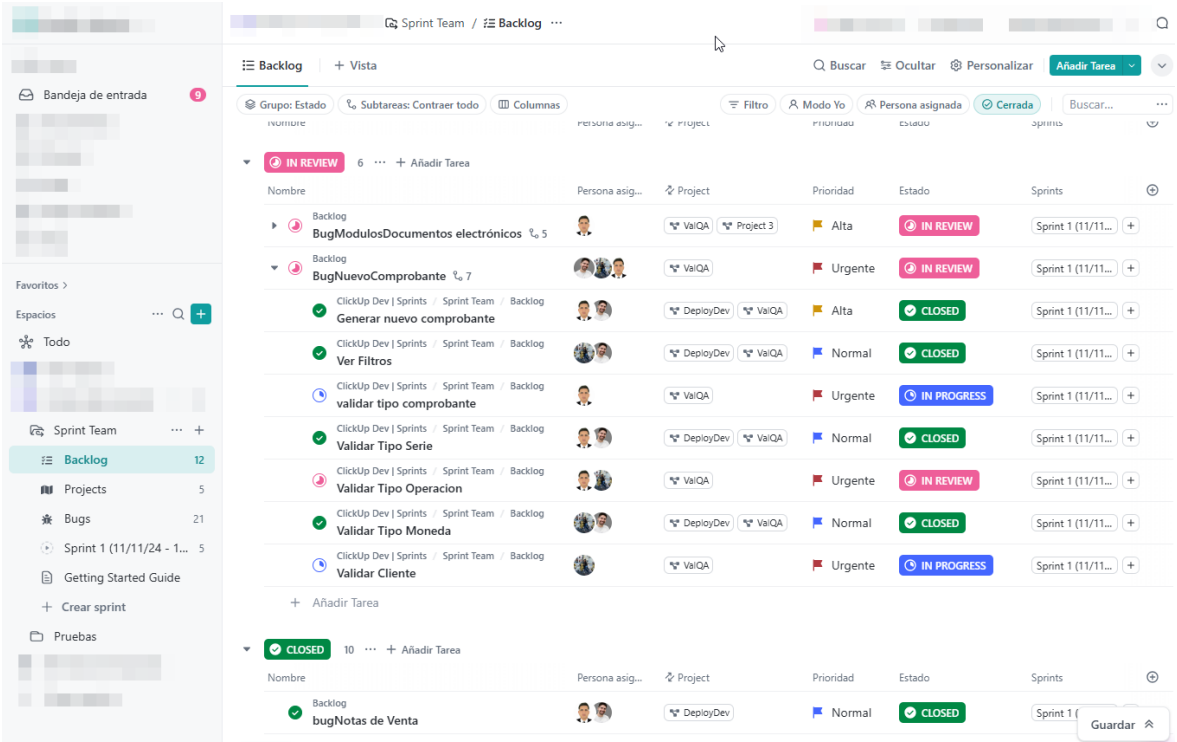
Nota: Elaboración propia

Implementación:

Ejecución de Scripts automatizados: Durante el Sprint, se ejecutan las pruebas automatizadas diseñadas en las historias de usuario. Esto se puede hacer de manera continua a medida que se completan los scripts de automatización.

Como podemos visualizar en la Figura 9 mapeamos en el backlog los defectos o problemas que se detectaron durante la ejecución de las pruebas automatizadas, estos se pueden registrar en distintos softwares de Gestión ya sea en DevOps y/o ClickUp para comunicar al área de desarrollo para su inmediata corrección.

Figura 9:
Backlog de Actividades



Nota: Elaboración propia

Revisión y Retroalimentación:

Al final del Sprint, el equipo Scrum hace una evaluación en conjunto, y se analiza el estado de la automatización de pruebas y se evalúa su efectividad.

Retroalimentación y ajustes: Se recopila feedback de todos los integrantes del equipo, incluyendo ambos equipos: de desarrollo y de pruebas, para realizar ajustes y mejoras en la automatización de pruebas.

Iteración y Mejora Continua:

La automatización de pruebas continúa en los sprints subsiguientes, con la adición de nuevos casos de prueba y la mejora de los existentes. La retroalimentación constante del equipo de desarrollo y pruebas ayuda a ajustar y mejorar la suite de pruebas automatizadas a lo largo del tiempo. El ciclo se repite en cada Sprint, lo que permite una automatización de pruebas continua y una mayor confiabilidad en el procedimiento de desarrollo de software ágil. La automatización de pruebas es una componente integral de Scrum y contribuye al envío de software de elevada calidad de manera iterativa.

Alcance del proyecto

Desarrollar un script de automatización que permita realizar pruebas funcionales automatizadas de los requerimientos de manera eficiente para reducir la inversión de horas hombre y destinar estos recursos a otros proyectos dentro de la empresa, El navegador web por usar será Google Chrome versión 118.0.5993.89 o superior...

Los puntos que se tomaran en cuenta para el desarrollo del Script de Automatización son:

- Seleccionar las herramientas y tecnologías adecuadas para el desarrollo del script de automatización.
- Definir casos de prueba para validar la funcionalidad de los módulos del sistema.
- Probar que el script de automatización pueda registrar una factura correctamente
- Probar que el script de automatización pueda buscar una factura por número, cliente o descripción.

Limitaciones:

Consideramos que son limitaciones en los proyectos de Scripts de automatización a los costos de los servidores en la nube, la infraestructura de almacenamiento ,la seguridad y la dependencia de otros equipos. Por ejemplo, el costo de los servicios en la nube limita nuestra capacidad de aumentar la capacidad de almacenamiento ya que puede no ser suficiente para satisfacer las necesidades del proyecto. En paralelo, las normas de seguridad existentes pueden limitar el uso de determinadas herramientas de automatización, también el hecho de depender de otros equipos para la habilitación de ciertos ambientes. Estamos explorando opciones para mitigar estos problemas, como optimizar el almacenamiento, implementar medidas de seguridad necesarias y así encontrar alternativas un poco más rentables respecto de los servicios en la nube.

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

Conclusiones

- Se comprueba que la automatización de pruebas funcionales reduce hasta en un 80% el tiempo de ejecución de pruebas manuales y aporta mayor fluides al momento de realizar alguna Regresión de los módulos del sistema.
- La automatización de pruebas funcionales no solo reduce el esfuerzo del personal involucrado, si no que mejora la precisión y escalabilidad de las pruebas. Estos hechos se traducen en que se tendrá un desarrollo de software más eficiente.
- Se Comprueba que la automatización de pruebas funcionales en las empresas mejora la calidad del software optimizando los tiempos de ejecución de las pruebas manuales, amplía la cobertura de los escenarios de prueba, reduce el esfuerzo del equipo, y nos permite detectar errores con mayor rapidez.
- Se comprueba que al igual que las investigaciones citadas en el marco teórico sobre la automatización de pruebas funcionales, concluimos que se reduce el esfuerzo humano y permite agilizar los procedimientos de pruebas, detección de errores y/o procedimientos de rollback.

Recomendaciones

Se recomienda el uso de buenas prácticas en la codificación de la automatización, esto permitirá que cualquier otra persona pueda entender y dar mantenimiento al código.

Además, se sugiere siempre tener comunicación contante con el equipo DEV para tener siempre mapeado en el diccionario de datos los locator a utilizar en la automatización.

También se recomienda que los casos que se vayan a automatizar sean tareas muy recurrentes y que consuman mucho tiempo del personal.

Recomendamos además que los casos a automatizar tengan expectativas realistas y que por más que se automatice esto no significa que se deba dejar de realizar pruebas manuales para algunos escenarios.

Se recomienda implementar nuevos scripts de automatización para los módulos más recurrentes del sistema.

Se recomienda tomar como base esta investigación para la realización de futuros scripts de automatización de pruebas funcionales.

Se recomienda tomar como referencia este marco de trabajo como inicio para la automatización de pruebas funcionales.

REFERENCIAS

Alianza Ágil. (s.f.). ¿Qué es la gestión de proyectos ágil? Recuperado de https://www.agilealliance.org/agile101/

Atlassian. (s.f.). ¿Qué es la metodología ágil? Recuperado de https://www.atlassian.com/agile.

Apache Maven. (s.f.). What is Maven? Recuperado de https://maven.apache.org/what-is-maven.html

Cabrera Serna, J., & Pareja Verastegui, C. E. (2021). Automatización de Pruebas Funcionales Web para mejorar el Área de Calidad de Software en una empresa del rubro de Retails en el año 2021. Tesis, Facultad de Ingeniería, Universidad Tecnológica del Perú, Lima, Perú.

Duquino, A. (2020). Automatización de un Sistema de Pruebas de Software para la Optimización del Proceso de Calidad de DetectID.

García, M., Pérez, R., & Torres, L. (2021). Estrategias para el control funcional en el desarrollo de software. Journal of Software Engineering, (2021).

Guru99. (s.f.). What is Selenium? Recuperado de https://www.guru99.com/selenium-tutorial.html

Hernández, R., Martínez, J., & Sánchez, J. (2019). Automatización de pruebas de software utilizando Selenium WebDriver. Revista de Ingeniería de Sistemas.

JUnit.org. (s.f.). What is JUnit? Recuperado de: https://junit.org/junit5/docs/current/user-guide/

López, J., García, J., & Hernández, R. (2020). Automatización de pruebas funcionales en el desarrollo de software: un caso de estudio. Revista de Ingeniería Industrial, 27(2), 123-135.

López, A. (2022). La importancia de un sistema de facturación en la gestión empresarial. Editorial Empresarial.

Martínez, J., & López, S. (2020). Importancia del control funcional en módulos de software. Revista de Tecnología y Desarrollo.

Martínez, J., & Pérez, R. (2021). Sistemas de facturación: optimización y eficiencia en la gestión financiera. Revista de Administración y Finanzas.

Ramos Marén, Y., Morales Tabares, Z. E., & Trujillo Casañola, Y. (2019). Propuesta de automatización de pruebas funcionales durante el ciclo de vida del software en Desoft. Serie Científica de la Universidad de las Ciencias Informáticas, 12(9), 112-127. http://publicaciones.uci.cu

Scrum.org. (s.f.). ¿Qué es Scrum? Recuperado de: https://www.scrum.org/resources/what-is-scrum

Schwaber, K., & Sutherland, J. (2017). The Scrum Guide. Recuperado de: https://scrumguides.org/scrum-guide.html

Selenium.dev.¿ Que es Selenium? Recuperado de: https://www.selenium.dev/documentation/en/introduction/getting_started/

TutorialsPoint. ¿Que es JUnit? Recuperado de: https://www.tutorialspoint.com/junit/junit_overview.htm