

UNIVERSIDAD CATÓLICA SEDES SAPIENTIAE
FACULTAD DE INGENIERÍA



**Implementación de un Framework de Automatización para Mejorar el
Proceso de Pruebas Funcionales de una OSE, Lima, 2023**

**TRABAJO DE SUFICIENCIA PROFESIONAL PARA OPTAR EL
TÍTULO PROFESIONAL DE INGENIERO DE SISTEMAS**

AUTOR

Sandra Fiorella, Benites Benites

REVISOR

Joel Benigno López del Mar

Lima, Perú

2023

METADATOS COMPLEMENTARIOS

Datos del autor

Nombres	SANDRA FIORELLA
Apellidos	BENITES BENITES
Tipo de documento de identidad	DNI
Número del documento de identidad	75247738
Número de Orcid (opcional)	

Datos del asesor

Nombres	JOEL BENIGNO
Apellidos	LOPEZ DEL MAR
Tipo de documento de identidad	DNI
Número del documento de identidad	08584920
Número de Orcid (obligatorio)	0000-0002-4302-7559

Datos del Jurado

Datos del presidente del jurado

Nombres	
Apellidos	
Tipo de documento de identidad	DNI
Número del documento de identidad	

Datos del segundo miembro

Nombres	
Apellidos	
Tipo de documento de identidad	DNI
Número del documento de identidad	

Datos del tercer miembro

Nombres	
Apellidos	
Tipo de documento de identidad	DNI
Número del documento de identidad	

Datos de la obra

Materia*	Framework, desarrollo de software, testing, automatización
Campo del conocimiento OCDE Consultar el listado: enlace	https://purl.org/pe-repo/ocde/ford#2.00.00
Idioma (Normal ISO 639-3)	SPA - español
Tipo de trabajo de investigación	Trabajo de Suficiencia Profesional
País de publicación	PE - PERÚ
Recurso del cual forma parte (opcional)	
Nombre del grado	Ingeniero de Sistemas
Grado académico o título profesional	Título Profesional
Nombre del programa	Ingeniería de Sistemas
Código del programa Consultar el listado: enlace	612076

*Ingresar las palabras clave o términos del lenguaje natural (no controladas por un vocabulario o tesoro).

FACULTAD DE INGENIERÍA

ACTA N° 002-2023-UCSS-FI/TPISIS

**TRABAJO DE SUFICIENCIA PROFESIONAL PARA OBTENER EL TÍTULO PROFESIONAL DE
INGENIERO DE SISTEMAS**

Los Olivos, 20 de abril de 2023

Siendo el día jueves 20 de abril de 2023, en la Universidad Católica Sedes Sapientiae, se realizó la evaluación y calificación del siguiente informe de Trabajo de Suficiencia Profesional.

**“Implementación de un Framework de Automatización para Mejorar el Proceso de Pruebas
Funcionales de una OSE, Lima, 2023”**

Presentado por la bachiller en Ciencias con mención en Ingeniería de Sistemas de la Sede Lima:

BENITES BENITES, SANDRA FIORELLA

Ante la comisión evaluadora de especialistas conformado por:

MSc. GUERRA GUERRA, JORGE LEONCIO

Mg. RAMIREZ ROMERO, BRANDON VICENTE

Luego de haber realizado las evaluaciones y calificaciones correspondientes la comisión lo declara:

APROBADO

En mérito al resultado obtenido se expide la presente acta con la finalidad que el Consejo de Facultad considere se le otorgue a la Bachiller BENITES BENITES, SANDRA FIORELLA el Título Profesional de:

INGENIERO DE SISTEMAS

En señal de conformidad firmamos,



MSc. GUERRA GUERRA, JORGE LEONCIO
Evaluador especialista 1



Mg. RAMIREZ ROMERO, BRANDON VICENTE
Evaluador especialista 2

Resumen

El presente trabajo propone la implementación de un framework de automatización a fin de mejorar las pruebas funcionales en procesos de mantenimiento y actualización de módulos del sistema de facturación electrónica en respuesta a los nuevos requerimientos tributarios.

Se hace necesario agilizar los tiempos de desarrollo de software para las nuevas versiones del producto, por ello es vital asegurar la calidad en las pruebas funcionales, además de minimizar los tiempos de pruebas o testing.

El framework de automatización de pruebas se implementa con Selenium IDE por ser compatible con los navegadores web más importantes, se diseña un script de automatización con las funcionalidades a probar y los datos de prueba como parte del proceso.

Los resultados de la implementación evidencian una mejora considerable en las pruebas funcionales de un 60%, respecto a la identificación de errores se mejora en 53.84%, finalmente el tiempo de pruebas se redujo en 42.85%, logrando mejorar el proceso y calidad de las pruebas funcionales.

Palabras claves: Framework, desarrollo de software, testing, automatización.

Abstract

This paper proposes the implementation of an automation framework in order to improve functional tests in maintenance processes and updating modules of the electronic invoicing system in response to new tax requirements. It is necessary to speed up software development times for new versions of the product, so it is vital to ensure quality in functional tests, in addition to minimizing testing times. The test automation framework is implemented with Selenium IDE because it is compatible with the most important web browsers, an automation script is designed with the functionalities to be tested and the test data as part of the process. The results of the implementation show a considerable improvement in the functional tests of 60%, with respect to the identification of errors it is improved by 53.84%, finally the testing time was reduced by 42.85%, managing to improve the process and quality of the tests. functional.

Keywords: Framework, software development, testing, automation.

Indice General

Resumen	2
Aabstract	3
Indice general	4
Indice de figuras	6
Indice de tablas	7
1. Introducción	1
2. Trayectoria del autor	4
2.1.Descripción de la empresa/institución (donde labora o laboro).	4
Historia de la empresa	4
Misión	5
Visión	6
Valores	6
Certificaciones	6
Aliados estratégicos	7
2.2. Organigrama de la Empresa	8
Organigrama actual de la empresa	8
Descripción del organigrama	10
2.3. Areas y funciones desempeñadas	10
2.4. Experiencia profesional realizada en la organización	11
3. Problemática	13
3.1. Planteamiento del problema	13
3.2. Determinación del problema	15

3.2.1.	Problema principal	15
3.2.2.	Problemas secundarios	15
3.3.	Objetivo principal	16
3.4.	Objetivos específicos	16
3.5.	Justificación	16
3.6.	Alcances y limitaciones	18
4.	Marco teórico	20
4.1.	Antecedentes bibliográficos	20
4.2.	Bases teóricas	25
4.3.	Definición de términos básicos	32
5.	Propuesta de solución	34
5.1.	Metodología de la solución	34
5.2.	Mesarrollo de la solución	46
5.3.	Factibilidad técnica – operativa	57
5.4.	Cuadro de inversión	60
6.	Análisis de resultados	62
6.1.	Análisis costos – beneficios	62
7.	Aportes destacables a la empresa/ institución	70
8.	Conclusiones	72
9.	Recomendaciones	76
10.	Referencias	77
11.	Anexos	80

Índice de Figuras

Figura 1	Certificación ISO 27001	7
Figura 2	Aliados estratégicos de la Empresa de facturación electrónica	8
Figura 3	Organigrama Empresa Facturación electrónica OSE 2022	9
Figura 4	Estructura Metodología Cascada	37
Figura 5	Proceso Actual de Testing de software	38
Figura 6	Proceso de Testing con la solución	38
Figura 7	conexión de selenium webdriver con chromedriver	47
Figura 8	page of model VS Analista	48
Figura 9	Interfaz login aplicación de Facturación electrónica	49
Figura 10	Script de automatización de interfaz login	49
Figura 11	Estructura del Código en paquetes del Framework de Automatización	50
Figura 12	Script de automatización con selenium webdriver y java	51
Figura 13	Arquitectura de Framework de Automatización para la OSE	52
Figura 14	Actores del sistema	53
Figura 15	Casos de Uso del sistema	53
Figura 16	Diagrama Caso de Uso del Sistema	54

Índice de Tablas

Tabla 1 Roles del Proyecto	39
Tabla 2 Requerimientos funcionales del Framework de automatización	54
Tabla 3 Requerimientos no funcionales.....	55
Tabla 4 Matriz Requisitos VS CUS	56
Tabla 5 Especificaciones del Hardware del servidor de pruebas.....	58
Tabla 6 Especificaciones del hardware para equipos de Analistas QA	58
Tabla 7 Especificaciones del sistema software	58
Tabla 8 características del Analista QA Funcional.....	60
Tabla 9 Características de Automatizador	60
Tabla 10 Costos en Recursos humanos.....	60
Tabla 11 Recursos de software	61
Tabla 12 Recursos de hardware	61
Tabla 13 Costos totales	61
Tabla 14 Van y TIR de la Implementación de un Framework de automatización	64
Tabla 15 Flujo de caja del proyecto	65
Tabla 16 Determinación de un Framework de Automatización para las pruebas	66
Tabla 17 Verificar el proceso de identificación de errores.	67
Tabla 18 verificar el tiempo empleado en las pruebas.....	68
Tabla 19 Costos de la implementación de solución.....	69

1. Introducción

Una Empresa de facturación electrónica es la encargada de brindar al cliente una solución informática ágil, segura y confiable de tributación de comprobante de pagos y Administrativos desde su emisión hasta la comprobación de su validez, todo ello con el respaldo de la Entidad Tributaria Sunat (Superintendencia Nacional de Aduanas y Administración Tributaria), la cual es la encargada de la tributación a nivel nacional.

Para que una Empresa de Facturación electrónica pueda brindar una solución informática optima al cliente final, debe respetar el ciclo de creación de software y realizar unas pruebas adecuadas en la calidad del mismo antes de realizar el pase a producción, la forma de verificar que el sistema cumpla con las especificaciones se hacen a través de pruebas y los responsables son los colaboradores del área de Testing, pero para realizar las pruebas es necesario contar con un plan de pruebas que actualmente es manual, pero con el avance de la tecnología se ha impulsado el uso de framework o marcos de trabajo en automatización para las pruebas de software, estos permiten agilizar el tiempo de pruebas funcionales y de regresión.

Una Empresa de facturación electrónica se encarga de emitir, declarar y validar comprobantes de pago como Boletas y Facturas, comprobantes administrativos como GRR (guía de remisión remitente), guía de remisión transportista (GRT), comprobantes de retención, percepción y resúmenes diarios entre los principales, todo ello a través de sus sistemas implementados.

Para que una empresa de facturación electrónica reciba el respaldo de Sunat y pueda demostrar a sus clientes que es segura y confiable debe contar con una serie de requisitos impuestos por la entidad tributaria antes mencionada, de los cuales resaltan la certificación ISO 27001, también debe contar con un RUC (registro único de contribuyente) habilitado y en estado activo,

ser emisora de comprobantes de pago como boletas y facturas en el rol de persona jurídica y estar en la tercera categoría en lo que corresponde la afección de régimen general de impuesto. A nivel Software la empresa de facturación electrónica debe brindar soporte a sus clientes y comunicación permanente con Sunat para informar el emisor y receptor de los distintos comprobantes que pasan por sus sistemas, es por ello, que debe velar por la calidad de su software, lo cual se realiza a través de pruebas manuales a nivel funcional y de regresión en responsabilidad del Analista de Calidad QA (quality assurance).

La realización de pruebas manuales en calidad de software son la parte inicial pues se apunta a la exploración y verificación de funcionalidades del sistema, sin embargo cuando el sistema se encuentra en una fase robusta con demasiadas características agregadas, el uso de las pruebas automatizadas a través de framework de automatización es recomendable pues permite abarcar la mayor capacidad del sistema, realizar las pruebas en menor tiempo y poder realizar las pruebas de regresión las cuales validan que las antiguas funciones del sistema no hayan sido alteradas, también las pruebas automatizadas son escalables las cuales crecen con el sistema permitiendo crear un control de versiones del código de automatización.

En la actualidad una empresa de Facturación electrónica para poder realizar sus funciones y brindar sus servicios como OSE (operador de servicios electrónicos) necesita tener el respaldo de Sunat, al ser un operador de servicios electrónicos verificado por Sunat, le da una ventaja competitiva frente a otras empresas del mismo rubro que no se actualizan, pues pierden credibilidad ante sus clientes, es por ello que una empresa de facturación electrónica convertida en OSE debe mantenerse actualizada en las normas vigentes que publica Sunat cada cierto tiempo, pues es parte fundamental del negocio.

En el país son pocas las empresas que son actualmente OSE certificados por Sunat, eso debido a burocracias que conlleva, pero es el reto de una empresa de facturación electrónica el apostar por su respaldo, brindad un software de calidad a través de la realización de un desarrollo integro pasando por unas pruebas de Testing de primera mano apuntando a la excelencia y proyectándose a la automatización.

La automatización de pruebas de software es una técnica aplicada para agilizar el proceso de Testing y mejorar los tiempos de subida a producción, sin embargo se debe evaluar de manera correcta el proceso que implica cambiar el proceso de las pruebas de manera manual a la automatización, reconocer el estado actual de la compañía, de los colaboradores, de las herramientas que se cuentan y escoger la mejor solución como se plantea en esta investigación un framework de automatización de pruebas, hecho bajo las necesidades de la organización velando por el logro de los objetivos planteados como mejora.

El objetivo principal de un framework de automatización es proporcionar una estructura sólida y reutilizable para automatizar tareas específicas, como pruebas de software, implementación de aplicaciones, administración de sistemas, entre otros. Esto ayuda a aumentar la eficiencia y la precisión al automatizar tareas repetitivas, reducir el tiempo de implementación y mejorar la confiabilidad del sistema. Además, un framework de automatización también puede proporcionar una interfaz común para diferentes herramientas y tecnologías, lo que facilita la integración y el mantenimiento del sistema.

El objetivo de esta investigación es determinar la mejora que brinda un framework de automatización para las pruebas manuales en un sistema web de una OSE, Lima, 2023.

2. Trayectoria Del Autor

2.1. Descripción de la empresa.

Historia de la Empresa

La Empresa que brinda los servicios de Facturación electrónica a través de una modalidad aprobada por Sunat llamado OSE, posee una historia interesante desde su fundación la cual se detalla a continuación:

Es una empresa de tecnología abarcando su enfoque en brindar servicios de facturación electrónica a diversos clientes a través de diversas soluciones informáticas, empoderándose en diversos modelos de sistemas desde los móviles, pasando por los de escritorio y también incluye los sistemas web con solución en la nube y micro servicios. Todo ello para agilizar el proceso de tributación y cumplimiento de deberes ante la entidad tributaria.

Todo ese proceso se realiza bajo el estricto cumplimiento de las normativas técnicas brindadas por Sunat las cuales cambian con el tiempo y se tiene que estar actualizando constantemente los sistemas para brindar el mejor servicio en facturación electrónica.

Breve Reseña Histórica

Erase el año de 1992 , uno de los cuales muchos peruanos recuerdan y no se pensaba en formar empresa, pero un joven visionario al percibir que se venía un gran avance en tributación a través de medios tecnológicos como ya se hacía en otros países de Latinoamérica, apostó a pesar del caos del país en formar una compañía, la cual hoy cuenta con 30 en el mercado peruano y se encuentra en más de 15 departamentos a nivel nacional, lo cual le ha permitido obtener una cartera de clientes de diversos rubros desde el textil , automotriz, restaurantes y hasta servicios de transporte. Debido a la gran demanda de clientes por el servicio de tributación que es obligatoria

en el Perú, cuenta con más de 120 colaboradores capacitados en su especialidad profesional y además en las normativas que demanda Sunat.

La seguridad que brinda la compañía

Permanecer 30 años en el mercado no es sencillo, pero uno de los factores que la empresa le brinda a sus clientes para fidelizarlos en el uso de sus sistemas y servicios es la seguridad y está la transmiten a través de las normas tecnológicas básicas de Información, las cuales consiste en constar con socios estratégicos en Tecnología los cuales son Google, Microsoft, IBM , HP y diversos más que se pueden mencionar , pero lo principal es usar un software de calidad y licenciado teniendo el respaldo de grandes compañías para poder construir soluciones de calidad a sus clientes.

Su Pilar de Innovación

Para poder mantenerse en el mercado apuestan constantemente por sacar al público productos innovadores de la mano con lo que solicite la entidad tributaria Sunat, lo cual le ha permitido abarcar diversas categorías como son declaración y emisión de comprobantes de Pago, declaración y emisión de comprobantes administrativos, servicios para aduanas y comercio exterior los cuales les brinda a sus clientes propuestas innovadoras que son válidas para cualquier tipo de negocio desde uno pequeño hasta el más complejo permitiéndole estar un paso adelante en soluciones informáticas, tributarias y empresariales.

Misión

La Empresa facturación electrónica Transporte confidencial de Información (TCI,2020) posee misión y visión y valores los cuales detalla en su web:

De acuerdo con TCI (2020) su misión es: “Desarrollar soluciones que cumplan con las normas con funcionalidades únicas, simples y rápidas para aumentar la competitividad de nuestros clientes.”

Visión

“Ser la empresa peruana que impulse el desarrollo de la digitalización de las empresas, reduzca la huella de carbón y colabore con la seguridad virtual y física, a través de nuestra pasión por la tecnología.” es la visión de empresa para TCI (2020).

Valores

Los valores en una empresa son parte de su cultura organizacional, lo que permite que los colaboradores, clientes y todos los que representen a la organización tengan una identidad con la compañía en beneficio de mejorar y seguir avanzado en su crecimiento.

Según TCI (2020) los valores de la compañía son los siguientes

- Innovación con calidad
- Transparencia:
- Trabajo en equipo
- Orientación al cliente
- Compromiso:
- Proactividad:
- Responsabilidad:

Certificaciones

En el Perú para que una compañía se pueda considerar OSE (operador de servicios electrónicos) y ayude a sus clientes con soluciones más ágiles y sencillas a las que se pueden encontrar en la página o aplicaciones de Sunat, es uno de los requisitos que la empresa cuente con

la certificación ISO 27001 sobre los Sistemas de Gestión la seguridad de la información, la cual verifica que la organización cuenta con todos los protocolos de seguridad necesarios para manejar información confidencial, mantener los datos seguros y disponibles desde sus sistemas.

La empresa cuenta con la certificación ISO 27001 el cual le permite demostrar a sus clientes la seguridad y la calidad de sus servicios.

Figura 1

Certificación ISO 27001



Nota: Certificaciones e Informes ISO 27001, por OVHCLOUD 2022,

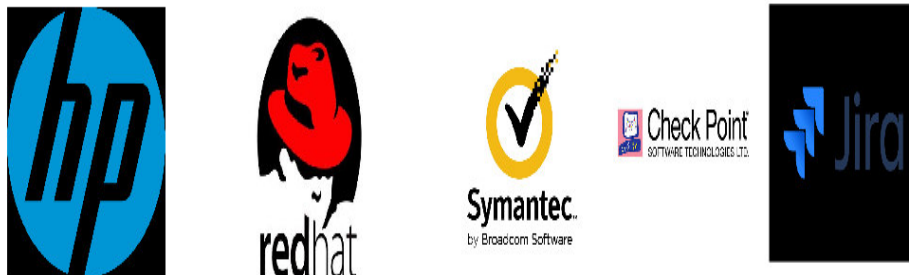
(<https://www.ovhcloud.com/es/enterprise/certification-conformity/iso-27001-27017-27018/>)

Aliados Estratégicos

La Empresa cuenta con aliados estratégicos a nivel nacional e internacional para poder brindar un excelente servicio a sus clientes

Figura 2

Aliados estratégicos de la Empresa de facturación electrónica



Nota: Elaboración propia

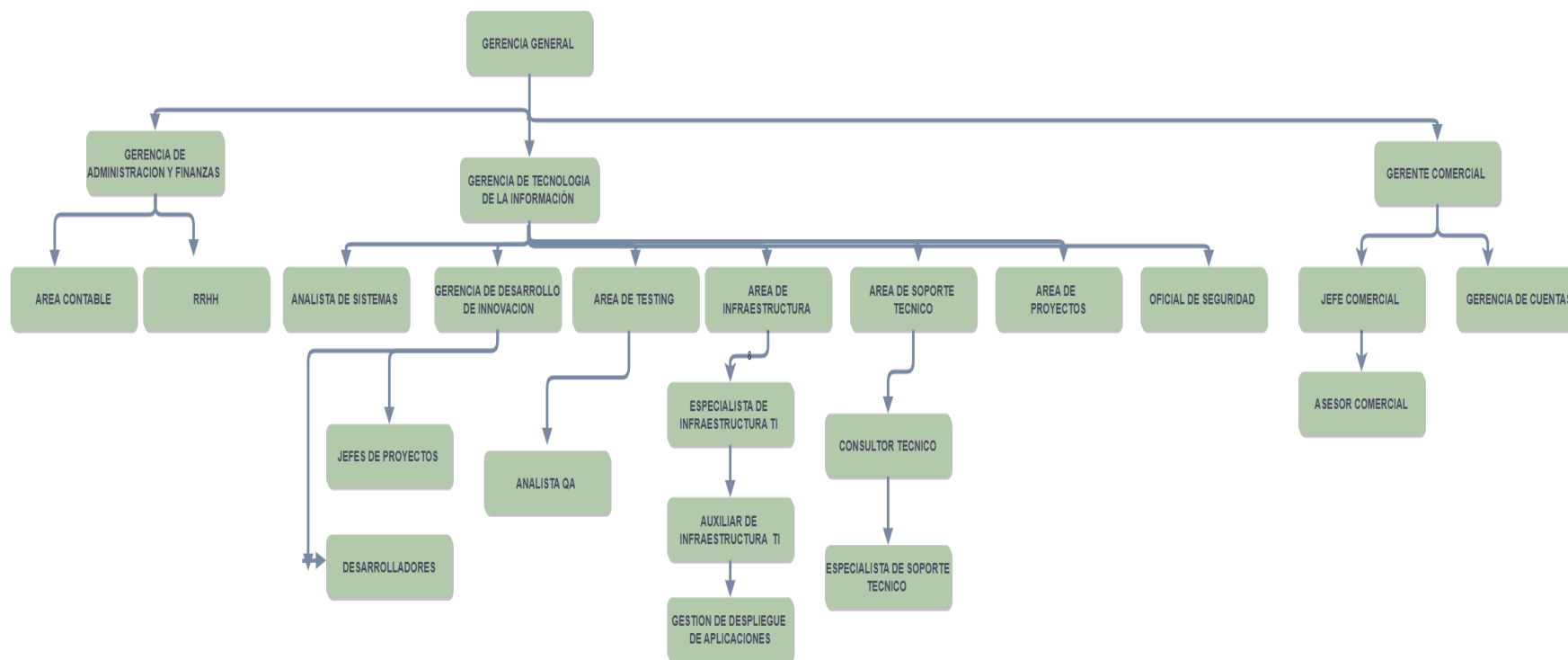
2.2 Organigrama de la empresa

Organigrama actual de la Empresa

El organigrama de la Empresa muestra a los colaboradores y los puestos estratégicos que ocupan en la organización brindando su talento en beneficio de la compañía.

Figura 3

Organigrama Empresa Facturación electrónica OSE 2022



Nota: Organigrama de la Empresa de facturación electrónica, Elaboración propia, 2022.

Descripción del Organigrama

Las áreas más destacadas de la organización son:

- El Área de Gerencia General: Esta área es la encargada de la planificación e innovación de la organización, además de orientar el rumbo de la Empresa.
- El Área de Administración y finanzas: Esta área es la encargada de rendir cuentas, realizar pagos a trabajadores, proveedores. Además, realiza los balances de la compañía.
- El Área de Tecnología de Información: Esta área es la encargada de desarrollar, revisar e innovar los sistemas de la organización mediante normativas, sugerencias de clientes o peticiones de gerencia.
- El Área de Comercial: Esta área es la encargada de ofrecer los sistemas a nuevos posibles clientes, además de fidelizar a los clientes antiguos.

2.3 Áreas y funciones desempeñadas

Área Actual de Trabajo

El área actual en el que labora el Bachiller es el área de Testing o Calidad de Software el cual su función es certificar los cambios o nuevos desarrollos de software de la compañía.

Funciones desempeñadas

Las funciones desempeñadas en el puesto de Analista QA en el área de Testing son las siguientes:

- Diseño de casos de pruebas para los Requerimientos atendidos: Se elabora un diseño de casos de pruebas donde se plasman las variables a usar, los pasos a ejecutar en una prueba y el resultado esperado de la prueba.

- Reunión de Certificación de Requerimientos: Antes de la atención de cada requerimiento se realiza una reunión con desarrollo y los PO (Product Owner) para conocer los alcances de los Requerimientos.
- Ejecución de las pruebas para los Requerimientos: Se ejecutan las pruebas de los requerimientos según el diseño de casos, respetando el tiempo de estimación planeado.
- Reporte de Bugs: En el transcurso de las pruebas, los errores encontrados en el sistema se reportan a DEV con la finalidad de que los soluciones antes que lleguen al cliente.
- Elaborar informes de Pruebas: Al terminar las pruebas, se elabora un informe de certificación explicando las versiones de software revisados, el desarrollador a cargo, el Analista a cargo, y si hubiera alguna observación se añade con la finalidad de mitigar algún inconveniente con la revisión.

2.4 Experiencia profesional realizada en la organización

En la compañía la autora labora desde junio del 2021, en la cual ingreso como QA Junior, en el inicio de sus labores solo revisaba aplicaciones a nivel escritorio y conforme transcurría el tiempo se le asignaron también aplicaciones web y webservices, finalmente ayudo con el testeo de aplicaciones a nivel móvil lo cual le ha permitido revisar la mayoría de los softwares de la empresa en sus distintas versiones.

La empresa cuenta con software de gestión para el manejo de documentación e issues (incidencias o errores de software), estos programas sirven en el desarrollo de las actividades de los colaboradores del Área de Testing y otras áreas de la compañía, los cuales son:

- Jira Software: es un programa de gestión de proyectos donde permite subir documentación, los involucrados en un proyecto, las responsabilidades y funciones y monitorear el avance de cada colaborador.

- Mantis: Es un gestor de reporte de Bugs (errores encontrados en el software) que permite a los desarrolladores y analistas QA mantener comunicación y un historial de los issues encontrados.
- Xray: Es una extensión de Jira Software que permite una gestión del proceso de Testing desde la creación de los casos de prueba, pasando por la ejecución y reporte de errores hasta la terminación de las pruebas con todos los involucrados en tiempo real, permite tener un alcance real del avance del proyecto con referencia a las pruebas de calidad en el software.

3. Problemática

3.1 Planteamiento del problema

En el área de Testing de una empresa de facturación electrónica certificada como OSE atiende una gran cantidad de requerimientos a lo largo del año, esto debido a los constantes cambios en normativas tributarias o nuevos métodos de declaración y emisión de comprobantes de manera electrónica impulsado por el ente tributario máximo del País, pero se ha observado que existen actualizaciones que son repentinas o normativas complejas de las cuales se cuenta con un tiempo corto para desarrollar, probarlas y lanzarlas a producción, pues la empresa de facturación electrónica no puede lanzar un cambio en producción sin realizar las pruebas funcionales correspondientes a los nuevos ajustes requeridos en el sistema para que se mantenga actualizado y vigente.

En el mundo de Testing la mayoría de las pruebas son manuales o funcionales y no se cuenta con un plan o marco de trabajo que apunte a las pruebas automatizadas. Debido a que las pruebas son manuales, demandan más tiempo y causan una demora en la entrega de los requerimientos, adicional a ello se juntan varios requerimientos causando acumulación de trabajo, ocasionando que se usen horas extras para poder entregar un requerimiento a tiempo o sino postergando la fecha de entrega del proyecto a producción.

La regresión de pruebas manuales demanda mucho tiempo, al ser pruebas repetitivas de no impacto ocasionando demora en la estimación de atención para futuros requerimientos.

Es por ello por lo que un framework de automatización adaptado a los requerimientos del sistema y la tecnología existente brinda un aporte de ahorro en tiempo de diseño y ejecución de pruebas, también permite tener mayor alcance en los módulos que se requieren revisar.

Según la International software Testing Qualifications Boards (ISTQB,2018) argumenta lo siguiente La automatización de pruebas ayuda a ejecutar muchos casos de prueba sin problemas adicionalmente que sea coherente y repetible en todas las versiones y/o entornos del SSP (sistema sujeto a prueba). Para la automatización las pruebas no son solo un mecanismo para ejecutar conjuntos de pruebas sin interacción humana. Eso significa que el analista debe comprobar el proceso de diseño del producto y los reportes finales.

Cuando se apunta a la automatización de pruebas es necesario trabajar bajo un marco de trabajo o framework de lo cual Qalified Building Quality (2022) explica lo siguiente Un marco es un conjunto bien definido de pautas y mejores prácticas que deben seguirse para lograr los resultados deseados. Por lo tanto, para lograr los objetivos de automatización, debemos seguir estas pautas y prácticas establecidas por el sistema de automatización.

Comprendiendo la importancia de la automatización de pruebas y tener un marco de trabajo en la resolución de las dificultades de las pruebas manuales en cuanto al tiempo y alcance del sistema Qalified Building Quality (2022) explica un concepto sobre framework de automatización de prueba, Un marco de automatización de pruebas es un conjunto de herramientas y pautas útiles que permiten diseñar, construir y ejecutar casos de prueba. Estas pautas incluyen estándares de codificación, prácticas y procesos para la manipulación de data de prueba e inventario esenciales para las pruebas automatizadas.

En el país las empresas comprenden que la automatización de pruebas incluye un conjunto de herramientas que le permiten codificar fácilmente pruebas funcionales en sus programas. Es decir, si necesita probar una determinada función, simplemente ejecute el programa. Por lo tanto, un marco de automatización de pruebas tiene beneficios muy importantes para los usuarios, incluido un alto grado de reutilización y portabilidad del código, así como un esfuerzo y costo

reducidos asociados con el mantenimiento de scripts, lo cual ayudara a la compañía en sus pruebas de calidad de software.

Para poder determinar la eficacia que brinda un framework de automatización de pruebas en una organización es necesario considerar su rendimiento antes las pruebas, el tipo de diseño que permite realizar en cada caso de prueba y el proceso de ejecución de cada prueba automatizada, todo ello para identificar fallos de manera más optima, reducir el tiempo de pruebas funcionales y mitigar los costos de la solución de los errores del software antes de que lleguen a producción.

Según el ISTQB (2018, p. 51) en su certificador de Programa básico explica que Las pruebas funcionales de un sistema implican pruebas que evalúan las funciones que debe realizar el sistema. Los requisitos funcionales se pueden describir en productos de trabajo, como las especificaciones del producto, requisitos comerciales, épicas, historias de usuarios, casos de uso o especificaciones funcionales, o quizás ningún archivo, aunque es recomendable que siempre se tenga un conocimiento y alcance.

3.2 Determinación del problema

3.2.1 Problema Principal

¿Cómo el framework de automatización mejora el proceso de pruebas funcionales del sistema web de una OSE, ¿Lima, 2023?

3.2.2 Problemas secundarios

¿Cómo el framework de automatización mejora el proceso de pruebas funcionales según la dimensión identificación de errores del sistema web de una OSE, ¿Lima, 2023?

¿Cómo el framework de automatización mejora el proceso de pruebas funcionales según la dimensión tiempo empleado en las pruebas del sistema web de una OSE, ¿Lima, 2023?

¿Cómo el framework de automatización mejora el proceso de pruebas funcionales según la dimensión costo de solución de errores del sistema web de una OSE, Lima, 2023?

3.3 Objetivos

3.3.1 Objetivo Principal

Determinar si la Implementación de un framework de automatización mejora el proceso de pruebas funcionales del sistema web de una OSE, Lima-2023.

3.3.2 Objetivos específicos

Verificar si la implementación de un framework de automatización mejora el proceso de pruebas funcionales según la dimensión identificación de errores del sistema web de una OSE, Lima – 2023

Verificar si la implementación de un framework de automatización mejora el proceso de pruebas funcionales según la dimensión tiempo empleado en las pruebas del sistema web de una OSE, Lima – 2023

Verificar si la implementación de un framework de automatización mejora el proceso de pruebas funcionales según la dimensión costo de solución de errores del sistema web de una OSE, Lima – 2023.

3.4 Justificación

La mejora continua de un área, en este caso el área de Testing es escalar hacia pruebas automatizadas en la evolución de los colaboradores para lograr la realización de los pendientes a tiempo y reducir costos en la compañía pudiéndose invertir en mejoras del área como equipos o

capacitaciones de nuevo conocimiento, estos son solo uno de los beneficios que brinda un framework de automatización de pruebas.

Debido a los avances de la tecnología de información, el hablar de un framework de automatización es un término nuevo que recién está tomando fuerzas en las compañías tecnológicas, por tal motivo los recursos de investigación son escasos, pero se conoce la importancia de esta herramienta en la mejora de los procesos de Testing en una organización.

La investigación permitirá determinar si la implementación de un framework de automatización ayuda en la mejora del proceso de pruebas funcionales en la identificación de errores a mayor escala, en la reducción del tiempo de pruebas y la verificación de los costos en las pruebas del Sistema Web de una OSE para evolucionar a un Testing que va de acuerdo con las mejoras tecnológicas del futuro.

Las implicancias prácticas de la presente investigación es brindar solución a los problemas encontrados después del análisis realizado en los puntos clave hallados, para poder encontrar la mejor manera de realizar automatización de pruebas y escalar de manera robusta el proceso de pruebas funcionales.

Para la investigación la herramienta propuesta como Marco de automatización o Framework es clave fundamental para el apoyo de los analistas de calidad de software funcionales a través del ahorro de tiempo y mayor capacidad de abarcar casos de pruebas lo cual es una metodología aplicada.

Al mejorar el proceso de realización de pruebas funcionales de los analistas QA de una OSE, brinda el apoyo de reducción de tiempo en las pruebas, permitir incluir pruebas de regresión o de no impacto en el proceso de Testing, permitiendo lograr una reducción de issues enviados a producción, lo que provoca un beneficio en el usuario final al momento de realizar el uso del

sistema con menor cantidad de errores , es decir casi mínima , su usabilidad se hace mínima ahora luz, tiempo y también los informes se digitalizan ocasionando ahora de papel eso crea una responsabilidad social en todos los involucrados en el proceso.

Mejorar las pruebas funcionales a una escala de automatización es un beneficio para la compañía que permite ahorrar tiempo, costos y obtener mejores resultados, pero es un reto para los analistas sin embargo adquieren nuevo conocimiento que les permite realizar su trabajo de manera eficiente.

3.5 Alcances y Limitaciones

La presente investigación según su el tipo de resultados que ofrece en la medición de sus variables es de tipo cuantitativo pues una de sus dimensiones es el tiempo, la contabilidad de detención de fallos en el software.

El tipo de estudio considerado es aplicado pues se basa en la implantación de un marco de trabajo para mejorar el proceso pruebas funcionales de regresión de los sistemas o los requerimientos que se soliciten.

Esta investigación aplicada, su objetivo principal es resolver problemas prácticos con un margen de producción limitado. Así, desde un punto de vista de diseño tendrá un enfoque interviniente, aporta al conocimiento, ya que es un estudio basado en el análisis subjetivo e individual de los sistemas de automatización en la mejora de sus pruebas funcionales.

Revisado desde la medición del tiempo es una investigación con un método longitudinal, el estudio se realiza durante un largo período de tiempo, lo que nos permite ver el desarrollo del evento objeto de estudio. Dependiendo de la variable, es un estudio de intervención si se supone que afecta la presencia o el desempeño de la variable dependiente o independiente.

Desde la perspectiva del enfoque es una investigación cuasi experimental, pues se tiene una perspectiva de la situación antes de aplicar la solución planteada esperando mejores resultados.

Los límites a nivel funcional de sistema son las siguientes especificaciones, la solución esta implementada solo para sistemas web, usando en navegador Google Chrome, usando webservices de tipo soap.

4. Marco Teórico

4.1 Antecedentes bibliográficos

Antecedentes Internacionales

Hernández (2021) Desarrollo un framework grafico el cual permitió realizar pruebas automatizadas En el área de Testing al momento de querer implementar soluciones nuevas para la realización de pruebas, surgen muchas dudas sobre cuál es el lenguaje correcto, que tipo de extensión usar , basadas en que navegador, y sobre todo cual es el framework adecuada que permita realiza la transición de las pruebas funcionales manuales a automatizadas de la manera correcta con una curva de aprendizaje no tan grande y extensa, adicionalmente la mayoría de tester no son hábiles en programación su capacidad resalta en el análisis, pero a pesar de ello se necesita innovar en las pruebas a través de la automatización. Por esa razón para el desarrollo de esta investigación se basó en los pilares básicos de la construcción de sistemas de software adaptado a la construcción de un framework básico de automatización, las etapas que el ciclo de vida del desarrollo de software indica son las siguientes: la documentación del desarrollo a implementar , el análisis funcional y no funcional, el diseño los prototipos a nivel interfaz, el desarrollo de lo planeado cumpliendo lo estipulado en las fases anteriores , la implantación de lo desarrollado para su uso y finalmente la evaluación de pruebas para su certificación y continua usabilidad, todo ello bajo la metodología cascada pues cuenta con pasos y métodos tradicionales que permiten desarrollar cualquier modelo de software y en este caso el desarrollo de un framework grafico para pruebas automatizadas. Debido al desarrollo aplicado en el estudio, los resultados obtenidos después de tener listo el framework grafico de automatización para las pruebas web fueron muy prometedoras, se logró reducir en un noventa por ciento el tiempo de la realización de pruebas con referencias a las pruebas funcionales que se hacían de manera manual, adicionalmente el tiempo

empleado de en la implementación, es decir en el diseño de casos y la elaboración del plan de pruebas se logró una reducción al sesenta por ciento en el tiempo. En conclusión, el framework grafico de automatización de pruebas es una gran opción para considerar en compañías donde su área de Testing no cuenta con analistas expertos en programación, lo que permite realizar el proceso de Testing en corto tiempo y con una curva de aprendizaje baja.

Rivera (2018) Formalizar un instrumento en el proceso de Testing para automatizar las pruebas de regresión. La problemática que presenta la investigación referente al área de Testing de la compañía de Tv con sede en Chile son los siguientes: en primer lugar no cuenta con el tiempo suficiente para la realización de las pruebas de regresión además que se realizan de manera manual. En segundo lugar los errores que se encuentran se reportan de manera tardía ya que la revisión se realiza después del pase de desarrollo con pocas semanas de entrega para producción, ocasionando a veces entregas con fallos en el actualizador de producción que serán revisando en futuras versiones, en tercer lugar se solicita apoyo de colaboradores ajenos al área de Testing o calidad de TI de la compañía afectando las actividades y logros internos de las áreas retrasando otros pendientes en la empresa, finalmente los casos ejecutados en las pruebas de regresión no se cumplen en totalidad y no se abarca todo el panorama planeado ya que se realizan de manera manual y toma mucho tiempo para los analistas de calidad de software concluirlos en su totalidad. La metodología usada en la investigación bajo el enfoque de sistemas de Tecnologías de la información es la interactiva incremental, la cual permite entregar avances a través de interacciones las cuales incluyen desde el diseño hasta la ejecución, permitiendo incrementar la solución en cada entregable, se diseñaron scripts automatizados para cada prueba y cada script se detalló paso a paso para realizar la ejecución de manera determinada y sin problemas de

entendimiento por el analista a cargo. Los resultados obtenidos en la investigación fueron medidos por los líderes del área de TI de la compañía abarcando la disponibilidad de la plataforma la cual aumento en un noventa y nueve por ciento y la resolución de issues de categoría 1 en un ochenta y cinco por ciento, reducción de las llamadas a soporte técnico por parte de los clientes y principalmente la reducción del tiempo de pruebas antes de realizar el despliegue a producción. En conclusión con esta investigación se logró eliminar casos de pruebas repetitivos, abarcar mayor cantidad de funcionalidades del sistema, mejorar las prácticas de realización de pruebas de calidad de software, obtener mejores resultados de las mediciones de KPI realizados por los líderes del área de TI y que los colaboradores con el rol de QA tengan un nuevo conocimiento en favor de realizar mejor sus actividades

Cortes (2020) La implementación de automatización de pruebas de regresión a través de un marco de trabajo en selenium. La problemática que presenta la empresa surecomp, compañía dedicada a desarrollar software para el rubro financiero, en donde se realizó la investigación es la no existencia de un proceso formal para el área de Testing, es decir la poca valoración de documentación para la comunicación del área con otras áreas aportando valor al producto revisado, la demora en el diseño de los casos de pruebas para la revisión de las versiones del sistema encargados y finalmente la extensión de entregas de los informes de certificación del software debido a la menor en la ejecución de pruebas. Todo esto ocasionaba pérdida de tiempo tanto para el área de Testing como para el área que mantiene comunicación y procesos compartidos con ella, adicionalmente de generar desorden y descontento de clientes internos y externos de la compañía. Para mitigar toda la problemática se definió un proceso de calidad mediante una bajo la metodología cualitativa-cuantitativa para ordenar los procesos y subprocesos del área, además de

definir una documentación estándar que permita una correcta comunicación interna y externa, adicional a ello se planteó la automatización de los casos de pruebas con el framework selenium web controlador en medida a las necesidades de la compañía. En base a las soluciones planteadas y las metodologías usadas los resultados fueron la disminución del tiempo realizado en los casos de las pruebas de regresión de un tiempo del sesenta por ciento a uno nuevo de treinta y siete con cuarenta y dos por ciento, siento un logro de las mediciones cuantitativas, de las mediciones cualitativas logro una satisfacción de los clientes internos y externos al proyecto, provocando la creación de una nueva área de automatización en la compañía. En conclusión la implementación de la automatización con selenium web controlador en las pruebas de regresión fue tan favorables para la compañía que se contó con una proyección para replicar la solución en sus sedes de Alemania e Israel.

Antecedentes Nacionales

Gutiérrez (2022) Optimización del proceso de pruebas a través de una plataforma de automatización usando selenium web controlador. La problemática que se abarca en esta investigación son las incidencias repostada constantemente reportadas por los clientes al área de soporte después de que la aplicación fuera lanzada a producción, eso causaba demasiado impacto para la organización a través del tiempo y costos en volver a revisar y realizar pruebas de funcionalidad para luego corregir los fallos encontrados, todo esto causaba malestar a los clientes finales por los fallos que tenían al usar al sistema que ya estaba en producción pero que no cumplía con la especificaciones y necesidades que ellos requerían. Finalmente afectaba al área de desarrollo, ellos al trabajar bajo la metodología scrum planeaban sus actividades, pero al presentarse errores en producción que debían ser corregidos afectaba a sus tareas ya planificadas

y acumulando trabajo. La metodología planteada para esta investigación a nivel científico académico se desarrolla bajo un enfoque cuantitativo pues obtiene resultados en métricas y su diseño empleado es el pre-experimental basando en las casuísticas de pruebas sin automatización y luego las pruebas con automatización, por otro lado la metodología a nivel técnico de sistemas de información y tecnologías de información usada es la metodología ágil Scrum, la cual permite la generación de entregables a través de sprint obteniendo un mejor alcance y revisión de posibles fallos a tiempo, a nivel de arquitectura se usó selenium web controlador para la ejecución de automatización. Los resultados obtenidos en la presente investigación fueron gratificantes para la organización pues se mejoró la sustentación de los errores en un cinco por ciento respecto a la situación inicial, se mejoraron los tiempos de pruebas logrando su reducción significativamente, los costos también disminuyeron pues ya no se reportaban fallos en producción que luego serían ser corregidos. Se concluyó que el uso de una plataforma de automatización para las pruebas de sistema usando selenium web controlador optimizaron el tiempo de las pruebas, minimizaron el reporte de errores en producción pues se hallaban a tiempo antes del reléase y finalmente se redujeron los costos de soporte, corrección y repetición de pruebas.

Capcha (2018) Implementación de un Framework de automatización para el área de Testing en un proyecto de Desarrollo. En la investigación la problemática que se expone es la demora en la realización de la ejecución de los casos de pruebas de las pruebas de regresión que se realizan en la última fase del Testing en la empresa consultora de software Belatrix, a pesar de contar con 5 colaboradores analistas de calidad enfocados en dichas pruebas, el tiempo invertido en ese proceso de verificación demandaba hasta tres días considerando un día como 24 horas y no de horario laboral, ocasionado una demora en la continuación del proceso de software volviendo

así su cuello de botella o su eslabón débil, el tiempo y los colaboradores a cargo para la ejecución de casos de pruebas de las pruebas de regresión dependían de la cantidad de sprint que se tendrían que atender, así que si la cantidad de sprint aumentaba también aumentaba el tiempo y la cantidad de colaboradores que se necesitaba para realizar el proceso de verificación, esto debido a que las pruebas se realizaban de manera manual y necesitaban la intervención humana en todo el proceso. La metodología usada en la investigación en el enfoque se desarrolló de software y sistemas de información es la metodología Scrum o conocido como metodología que permite realizar proyectos a través de sprint con sus historias de usuarios, tiempo y responsables permitiendo la creación de un framework de automatización creado a medida para las necesidades de la consultora Belatrix. Los resultados obtenidos fueron los siguientes, en primer lugar se logró reducir de manera exitosa el tiempo de ejecución de pruebas de tres días a dos horas, en segundo lugar se redujo la intervención humana logrando el objetivo de la automatización, finalmente se logró capacitar al área de calidad en el nuevo proceso de automatización en implementarlo en sus actividades de área. En conclusión en la framework de automatización fue de gran ayuda para el área de calidad de software de la consultora Belatrix, adicionalmente que se logró su implementación de manera exitosa, incluyendo el framework en las actividades del proceso de calidad para la verificación de pruebas de regresión.

4.2 Bases teóricas

En el presente capítulo se realizó una búsqueda de los conceptos según diferentes autores que fueron considerados en la investigación y permiten fundamentar el proyecto de investigación alguno de estos conceptos son Framework de pruebas, automatización de pruebas, rendimiento, diseño, proceso para las pruebas automatizadas, pruebas funcionales y de regresión,

identificación de fallos de software, tiempo empleado en las pruebas y costos en la solución de fallos de software.

De acuerdo con Spillner (2014), las pruebas funcionales de regresión constituyen La prueba de regresión es la nueva prueba de un programa previamente probado. Después de la prueba, se cambiará como tal; siempre que no se introduzcan o descubran errores como resultado de los cambios implementados.

Pero debemos entender el concepto inicial del proceso de calidad de software, los casos de pruebas funcionales que en palabras de Jorgensen (2013) afirma que Los casos de prueba funcional son (o deberían ser) productos de trabajo reconocidos. Un caso de prueba completo tendrá un ID de caso de prueba, una breve declaración de propósito (como una regla comercial), una descripción de los requisitos previos, entrada del caso de prueba, resultados esperados, una descripción del estado posterior esperado y un historial del caso de prueba a conducir

Graham (2012) recomienda para las pruebas funcionales de regresión Desarrollar casos de prueba ejecutables que abarquen toda la funcionalidad de la aplicación, teniendo en cuenta ambos aspectos; la aplicación hace lo que debería (una prueba positiva) y la aplicación no hace lo que debería. se espera que no haga nada (pruebas negativas). Debería ser posible agregar incluso casos de prueba generados por eventos, estos últimos deberían ser fácilmente identificables por el nombre dado al caso de prueba.

Las pruebas funcionales definidas según la SSTQB (2008) son Pruebas basadas en el análisis de las características funcionales de un componente o sistema.

Uno de los conceptos básicos de la automatización es el Script el cual McKay (2014) manifiesta que el primer concepto relacionado con las pruebas automatizadas es un script de prueba que especifica los pasos que se realizarán manualmente para probar ese comportamiento. estado del

sistema. Cuando estos scripts de prueba se convierten en scripts, el concepto de automatización se automatiza mediante una herramienta especializada llamada Pruebas automatizadas, que se usa comúnmente para las devoluciones de pruebas automatizadas.

Madi (2013) mencionó que cuándo esto se puede destacar como la mayor ventaja de la automatización. Prueba funcional de regresión reduce el tiempo y el esfuerzo, automatiza la repetibilidad y la amplitud de la cobertura, ofrece software de mayor calidad en comparación con las pruebas manuales del usuario.

El framework de automatización más usado en los proyectos de automatización es Selenium que en palabras de Satish (2015) Selenium consta de varias herramientas de automatización de software como Selenium IDE, Selenium RC (Selenium 1.0) y Selenium webdriver (Selenium 2.0). Selenium IDE es un entorno de desarrollo integrado para crear scripts de prueba. Es un complemento de Firefox que le permite grabar, editar y depurar casos de prueba de Selenium. Registre todas las acciones realizadas por los usuarios finales y genere scripts de prueba. El control remoto (RC) de Selenium ha sido durante mucho tiempo un proyecto importante de Selenium. Selenium RC es más lento que Selenium Web Driver porque utiliza un motor de secuencias de comandos Java llamado Selenium Core. Antes de ejecutar los scripts de prueba en Selenium RC, debe iniciar el servidor.

Un framework de automatización que es fácil de usar sin tanta programación es Selenium Ide en palabras de Satish (2015) Selenium IDE se compone de múltiples herramientas de automatización de software, como Selenium IDE, Selenium RC (selenium 1.0), y Selenium webdriver (selenium 2.0). Selenium IDE es un entorno de desarrollo integrado para construir el guion de prueba. Es un complemento de Firefox que le permite grabar, editar y depurar los casos de prueba de selenio. Graba todas las acciones realizadas por el usuario final y generar los scripts

de prueba. El control remoto de selenio (RC) fue selenio principal proyecto desde hace mucho tiempo. Selenium RC es más lento que el webdriver de selenium porque usa el programa de script java llamado núcleo de selenio. Selenium RC requiere iniciar el servidor antes de ejecutar los scripts de prueba.

Según Pantaleo (2011) Asumimos que Se ha revisado y probado la arquitectura del sistema en desarrollo, y nos enfrentamos al diseño detallado de cada caso de uso. Además de educar a nuestros desarrolladores sobre problemas de diseño, como errores comunes de diseño, buenos estándares de diseño y patrones de diseño, ¿qué más podemos hacer para mejorar nuestros diseños? Una buena práctica que ha dado muy buenos resultados es revisar los planes en reuniones con colegas y otros miembros de la organización con experiencia y conocimientos comprobados. He asistido a estas conferencias y, para ser honesto, en muchas de ellas, el producto que se revisa es tan importante como muchos de los productos que no se revisan. Entonces, ¿qué revisamos y cómo elegimos qué productos revisar? Necesitamos un criterio de selección que, por ejemplo, nos permita revisar secciones críticas, pero también secciones potencialmente incompletas.

Para Pantaleo (2011) es sumamente valioso el código está diseñado y escrito para garantizar que las causas de los errores anteriores se eliminen para que nunca ocurran errores. Para lograr esto, las pruebas de las fases anteriores deben combinarse e implementarse durante el desarrollo del sistema para influir en el diseño desde una perspectiva de prueba. Lo que ofrecemos y promovemos es el cambio que tuvo lugar hace unos años de los métodos de trabajo tradicionales a métodos de trabajo sistemáticos y automatizados.

Según la SSTQB (2008) la técnica de diseño de pruebas no funcionales se define como Un proceso para obtener y/o seleccionar casos de prueba para pruebas no funcionales basado en el análisis de

especificación de un componente o sistema sin referencia a su estructura interna. Véase también el método de desarrollo de pruebas de caja negra.

El diseño de pruebas debe tener un enfoque claro como indica SSTQB (2008) Implementar una estrategia de prueba definida para un proyecto en particular. Por lo general, esto incluye decisiones basadas en los objetivos del proyecto (para el proceso de prueba) y la evaluación de riesgos realizada, puntos de entrada al proceso de prueba, diseño de prueba de práctica utilizado, criterios de salida y tipos de pruebas realizadas.

Las herramientas de diseño de prueba son esenciales para la realización de los casos de pruebas según SSTQB (2008) Una herramienta que respalda las actividades de desarrollo de pruebas mediante la generación de entradas de prueba a partir de una especificación que se puede almacenar en un repositorio de herramientas CASE, como una herramienta de gestión de requisitos, a partir de condiciones de prueba específicas almacenadas en el mismo motor o desde el código.

Las pruebas de ciclo son una técnica de diseño para las pruebas funcionales según la SSTQB (2008) Una técnica de desarrollo de prueba de "caja negra" en la que los casos de prueba están diseñados para ejecutar procedimientos y procesos enfocados en el Core del negocio.

Según la SSTQB (2008) el proceso de pruebas se define como El proceso central de pruebas incluye la planificación y el control de las pruebas, el diseño y el análisis de las pruebas, la implementación y la ejecución de las pruebas, la evaluación y el informe de los criterios de salida y las actividades de cierre de las pruebas.

Una definición de la norma internacional de certificación ISO 12207 (2018) define el proceso como un conjunto de acciones relacionadas para convertir la entrada en salida.

Es importante realizar pruebas basadas en el proceso del negocio como indica la SSTQB (2008) como Un enfoque de prueba en el que los casos de prueba se desarrollan en función de las descripciones y/o el conocimiento del proceso comercial.

En la realización de pruebas es importante tener madurez en sus procesos como indica la SSTQB (2008) La capacidad de la organización para la eficacia y eficiencia de sus procesos y prácticas.

Las pruebas deben de tener en consideración de la fase de requisitos basados en el tiempo de pruebas para SSTQB (2008) indica el período en el ciclo de vida del software durante el cual se definen y documentan los requisitos del software.

El tiempo es la base de ejecución de pruebas es importante medirlo según SSTQB (2008) El período de tiempo en el ciclo de vida del software en el que se definen y documentan los requisitos para un producto de software.

Uno de los factores de las pruebas son las concurrencias según SSTB (2008) indico Periodo de tiempo Definir y documentar el ciclo de vida del software para los requisitos del producto de software.

El tiempo es un indicador que mide la ejecución del estado de software por el cual se necesita herramientas como indica la SSTQB (2008) Una herramienta que proporciona información sobre el estado del código del programa en tiempo de ejecución. Estas herramientas se usan comúnmente para detectar punteros no asignados, verificar la aritmética del puntero, monitorear la asignación de memoria, el uso, la liberación y detectar fugas de memoria.

La revisión de las pruebas demanda una inspección en tiempo real como indica la SSTQB (2008) Un enfoque para el desarrollo de software en el que dos programadores que trabajan en la misma computadora escriben líneas de código (producción y/o prueba) de un componente. Esto significa indirectamente que siempre se realizan pruebas de código en tiempo real.

Aguaded & Fuentes (2006) determina que Es importante que el sistema de TI optimice los procesos manuales que se realizan en la empresa, en este caso de negocio conducirá a una mejor gestión de los procesos de pruebas de la empresa, lo que reduce costos y tiempo. entidad. El propósito del control de proyectos es garantizar que el proyecto se gestione bien sin comprometer la calidad de la gestión de la continuidad del negocio y, para lograrlo, el control de proyectos debe estar respaldado por los medios apropiados.

Martínez Orozco et al., (2014) describe que El caso de negocios para el proyecto se basa en una comparación de los beneficios y costos incurridos a lo largo del tiempo ya una tasa de descuento adecuada. El principal beneficio económico del software es el ahorro de mano de obra. El cálculo del ahorro de mano de obra debe tener en cuenta tanto las tarifas por hora como los beneficios adicionales de ahorro de mano de obra para todos los turnos. La inversión en el sistema vale la pena en la mayoría de las aplicaciones, los clientes pueden monitorear o administrar instalaciones complejas a largas distancias desde estaciones de trabajo simples con recursos mínimos de hardware y conexión a Internet estable.

Según la investigación realizada por Carhuamaca (2017) en su tesis "Un sistema que reemplaza las funciones humanas en la validación de documentos digitales en una empresa digital" que permite lograr resultados positivos, como la reducción de costos y tiempos aumentando la eficiencia. El resultado del proceso de validación de documentos sube un dieciséis por ciento.

El proceso de desarrollo y entrega todavía está respaldado por la automatización, las pruebas y la detección oportuna de posibles errores antes de la implementación. Se refiere a los cambios organizacionales que alteran la gestión del ciclo de vida de un sistema. En concreto, se trata de un conjunto de técnicas destinadas a reducir el costo de despliegue y el tiempo transcurrido entre la introducción de cambios en el software y su puesta en producción en palabras de Nebel (2010)

4.3 Definición de términos básicos

Framework de Automatización: Es un marco de trabajo o herramienta de codificación y programación que mediante scripts estructurados bajo las normas y Core del negocio o el Sistema a revisar, permite ejecutar pruebas funcionales, pruebas de regresión de manera automática sin la intervención del humano, ahorrando tiempo y ayudando en la escalabilidad del sistema mediante encuentro de fallos a tiempo.

Rendimiento: La capacidad de soportar scripts de automatización mediante un lenguaje de programación de preferencia uno que use la compañía u organización en la cual se realiza el proceso de pruebas automatizadas.

Diseño: Es la capacidad de organizar los casos de prueba de manera eficiente para desarrollar el proceso de Testing de un proyecto o sistema en desde las pruebas funcionales continuando con las pruebas de regresión y terminando con las pruebas automatizadas.

Proceso: El proceso de Testing demanda mucha precisión y análisis en la capacidad de crear los casos de pruebas, los cuales deben ser ejecutados como fueron planificados y contar con un conjunto de actividades secuenciales que cumplen un objetivo.

Pruebas Funcionales: son aquellas pruebas que se realizan mediante el análisis de los requerimientos de un sistema o proyecto, estas requieren que se revise el cumplimiento de las expectativas o necesidades del cliente en usabilidad del sistema.

Detección de errores: es la capacidad de encontrar fallos del sistema que se revisa pero de manera oportuna de preferencia en la etapa de diseño o pruebas exploratorias. Adicionando la prevención de estos en producción cuando el software es usado por el cliente.

Tiempo: Es la unidad de medición de las pruebas mediante el flujo de pruebas automatizadas abarcando la mayor característica en comparación de las pruebas funcionales de regresión realizadas de la manera tradicional y manual.

Costo: Es el beneficio de ahorro en términos monetarios de la realización de pruebas automatizados en comparación de las pruebas manuales pro el área de Testing en el proceso de calidad de software.

5. Propuesta de Solución

5.1. Metodología de la solución

El desarrollo de software es una de las nuevas tendencias en el siglo 21, generando valor agregado para los clientes que lo usan desde una App hasta sistemas web. La empresa de facturación electrónica brinda soluciones tecnológicas basadas en las normativas de Sunat, ofreciendo sistemas de emisión, declaración y validación de comprobantes de pago y administrativos, las soluciones tecnológicas pasan por un proceso certificado por la compañía el cual abarca desde la reunión con los interesados hasta la subida a producción con las funcionales requeridas.

El proceso de un proyecto o desarrollo de soluciones informáticas de solución electrónica para clientes a nivel nacional demanda de un proceso de gestión de proyectos y ciclo de desarrollo de vida del software. El proceso empieza con la reunión de los Product owner con el área de proyectos, se plantean los requerimientos nuevos o ajustes que se desean para el software, seguidamente después de tener claro los cambio y/o ajustes se le comunica al área de desarrollo y se le consulta el tiempo que demandarían en tener lista la solución. Después de tener los cambios listos el equipo de desarrollo se reúne con el área de Testing para pasarle los ajustes a revisar, el área de Testing se encarga de armar el diseño de pruebas, ejecutarlas de manera manual muchas veces con poco tiempo de entrega. Encontrar bug o errores del sistema para que desarrollo los corrija, luego de terminar la revisión se pasa al ambiente de UAT (User Acceptance Testing) donde el usuario realiza pruebas para verificar que se cumplió con lo solicitado. Finalmente cuando el usuario da la conformidad se realiza el despliegue o pase a producción, culminando el proceso.

En el proceso de Testing muchas veces los pases que brinda desarrollo son con poco tiempo para realizar las pruebas, y al realizar de manera manual pocas veces se puede realizar pruebas de

no impacto o pruebas de regresión. Lo que a veces se tiene que realizar test solo en el flujo principal del requerimiento

La solución que se plantea desarrollar en la investigación para que ayude con la realización de las pruebas automatizadas es generar un framework de automatización para sistemas web con Selenium y los pasos a seguir con los siguientes:

- Seleccionar un lenguaje de programación: Selenium es compatible con varios lenguajes de programación, como Java, C#, Python, Ruby, etc. Es importante elegir un lenguaje con el que se tenga experiencia o se desee aprender.
- Instalar las dependencias necesarias: Es necesario tener instalado Selenium y el controlador del navegador web que se desea utilizar (como ChromeDriver, FirefoxDriver, etc.).
- Diseñar la estructura del framework: Se puede diseñar una estructura de carpetas para organizar los scripts de prueba, los datos de prueba y los recursos necesarios.
- Crear un script de inicio: Crear un script que inicialice la configuración necesaria para las pruebas automatizadas, como la inicialización de una instancia de Selenium y la configuración de las opciones de navegador.
- Crear scripts de prueba: Crear scripts de prueba que automatizan las acciones necesarias para probar la funcionalidad del sistema web, utilizando las funciones y métodos de Selenium.
- Ejecutar las pruebas: Ejecutar las pruebas automatizadas utilizando una herramienta de automatización de pruebas (como JUnit, TestNG, etc.) o mediante la ejecución de los scripts de prueba desde la línea de comandos.
- Generar informes: Generar informes automatizados para analizar los resultados de las pruebas.
- Mejorar y mantener el framework: Continuar mejorando y manteniendo el framework a medida que surjan nuevos requisitos o se detecten problemas.

Ciclo de vida tradicional – cascada

El ciclo de vida tradicional de software es un enfoque para el desarrollo de software que sigue un proceso estructurado y secuencial. Se divide en varias fases, como análisis de requisitos, diseño, codificación, pruebas y mantenimiento. Cada fase es completada antes de pasar a la siguiente, y el enfoque es en la planificación y el control riguroso del proyecto. Algunos ejemplos de metodologías tradicionales son el ciclo de vida del software de cascada, el proceso de desarrollo de sistemas clásicos y el proceso de desarrollo de sistemas estructurados.

La metodología cascada de software es una de las metodologías tradicionales de desarrollo de software. Se caracteriza por seguir un proceso secuencial y estrictamente lineal, donde cada fase del proyecto debe ser completada antes de avanzar a la siguiente. El proceso se divide en varias fases, como análisis de requisitos, diseño, codificación, pruebas y mantenimiento.

La modelo cascada se divide en las siguientes fases:

- Análisis de requisitos: en esta fase se recopilan, analizan y especifican los requisitos del sistema.
- Diseño: en esta fase se diseña la arquitectura, la interfaz y el algoritmo del sistema.
- Codificación: en esta fase se escribe el código fuente del sistema.
- Pruebas: en esta fase se realizan pruebas para verificar que el sistema cumple con los requisitos especificados y se detectan y corrigen los errores.
- Mantenimiento: en esta fase se realizan cambios y actualizaciones para mejorar el sistema y corregir problemas

Figura 4

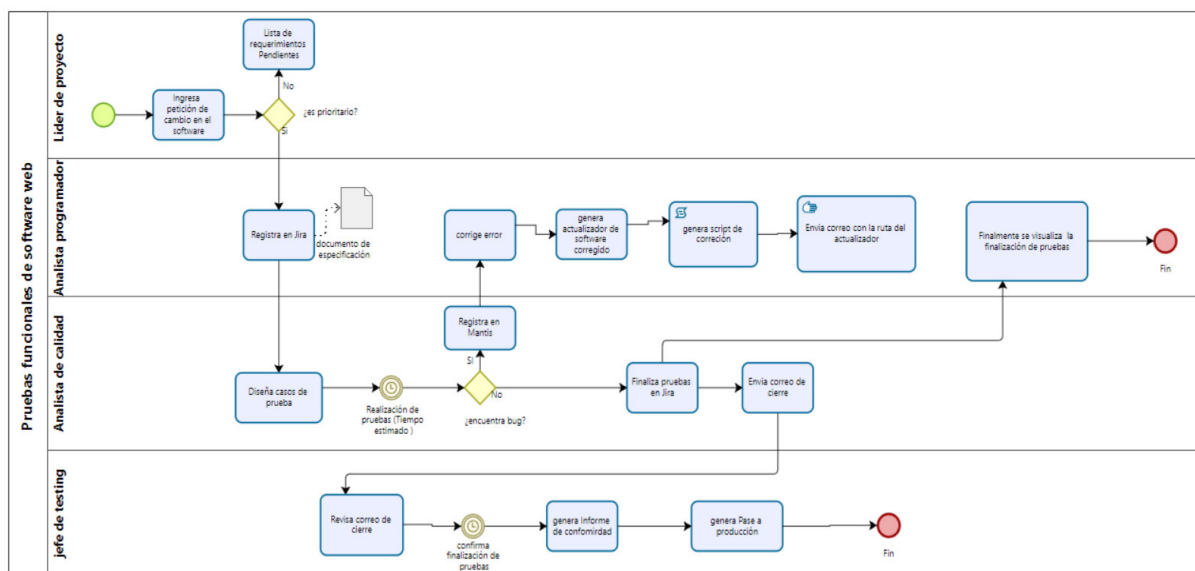
Estructura Metodología Cascada



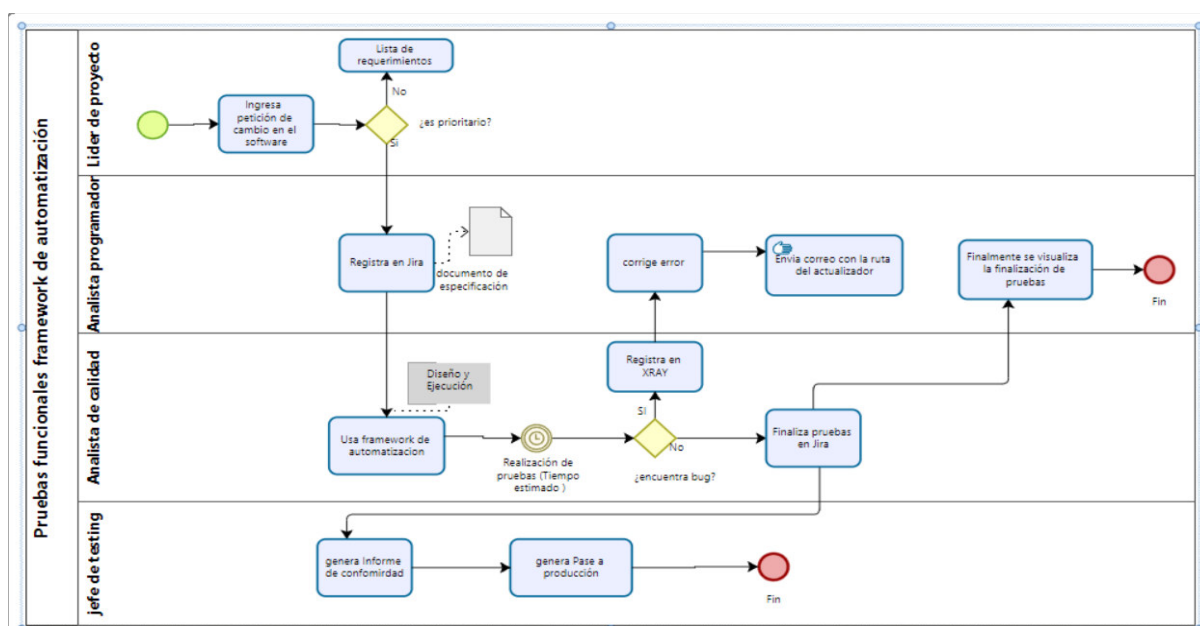
Nota: Elaboración Propia

La metodología cascada es aplicable para proyectos de software de tamaño pequeño o mediano, con requisitos bien definidos y estables. Sin embargo, esta metodología tiene sus desventajas, ya que no se adapta bien a los cambios en los requisitos del cliente, y no se presta a proyectos complejos y con alta incertidumbre

Diagrama de proceso actual vs diagrama de proceso con sistema

Figura 5*Proceso Actual de Testing de software*

Nota: Elaboración Propia

Figura 6*Proceso de Testing con la solución*

Nota: Elaboración Propia

Roles del proyecto

Los roles en un proyecto son importantes para identificar las actividades encargadas a cada colaborador con la finalidad de lograr los objetivos y mantener un orden, para la investigación se consideraron los siguientes colaboradores

Tabla 1

Roles del Proyecto

Rol	Cantidad	Perfil del puesto
Líder de proyecto	1	es responsable de planificar, dirigir y supervisar el proyecto de automatización de pruebas, establecer objetivos y metas, asignar recursos y asegurar que el proyecto se ejecute dentro del presupuesto y del plazo establecido.
Analista de pruebas automatizadas	1	es responsable de diseñar, crear y ejecutar scripts de prueba automatizados, evaluar el rendimiento de las pruebas, analizar los resultados de las pruebas y preparar informes de calidad
Desarrollador de automatización	1	es responsable de diseñar, crear y mantener el framework de automatización, y trabajar con el equipo de pruebas para asegurar que el framework se integre y ejecute correctamente.
Tester manual	1	es responsable de realizar pruebas manuales, identificar problemas y errores, y proporcionar retroalimentación al equipo de desarrollo.
Arquitecto de sistemas	1	es responsable de diseñar la arquitectura del sistema y asegurar que el sistema se integre correctamente con el framework de automatización.
Administrador de configuración	1	es responsable de configurar y administrar las herramientas y recursos necesarios para el proyecto de automatización de pruebas.
Gerente de calidad	1	es responsable de asegurar que el proyecto se ejecute de acuerdo con las normas y estándares de calidad establecidos.

Analista de negocios	1	es responsable de comunicar los requisitos del negocio al equipo de automatización y asegurar que las pruebas automatizadas cubren los requisitos del negocio.
----------------------	---	----------------------------------------------------------------------------------------------------------------------------------------------------------------

Nota: Elaboración Propia

Aspectos tecnológicos de la solución

Framework de Automatización

Un framework de automatización es un conjunto de herramientas, librerías, plantillas, convenciones y mejores prácticas que se utilizan para automatizar pruebas de software. El objetivo principal de un framework de automatización es proporcionar una estructura sólida y reutilizable para automatizar pruebas, lo que permite a los equipos de pruebas automatizar pruebas de manera eficiente y efectiva.

Un framework de automatización puede incluir:

- Una estructura de directorios y archivos para organizar los scripts de prueba y los recursos necesarios.
- Un conjunto de librerías y herramientas para automatizar acciones comunes, como la ejecución de scripts, la captura de pantalla, la generación de informes, entre otros.
- Un conjunto de convenciones y mejores prácticas para escribir scripts de prueba y mantener el código.
- Un mecanismo de configuración para personalizar el framework según las necesidades del proyecto.
- Un mecanismo de ejecución para ejecutar los scripts de prueba automatizados.

- Un buen framework de automatización debe ser fácil de usar y mantener, y también debe ser escalable y adaptable a los cambios en el sistema y en los requisitos del negocio.

Tipo de framework

Framework de pruebas funcionales: Se utilizan para automatizar pruebas que evalúan la funcionalidad del sistema, es decir, pruebas que se enfocan en el comportamiento del sistema desde el punto de vista del usuario. Ejemplos de herramientas de pruebas funcionales son Selenium, Appium, etc.

SOAP UI

Soap UI es una herramienta de pruebas de software para servicios web SOAP y REST. Es una aplicación de escritorio de código abierto que permite a los usuarios crear, probar, depurar y monitorear servicios web de manera eficiente. SOAP UI se utiliza para crear casos de prueba automatizados y manuales para servicios web SOAP y REST. Permite a los usuarios crear y enviar peticiones SOAP y REST, analizar y comparar las respuestas, y generar informes de pruebas. También proporciona una interfaz gráfica de usuario (GUI) para crear y editar peticiones SOAP y REST, y una consola de línea de comandos para automatizar tareas de prueba. SOAP UI también incluye características avanzadas como la automatización de pruebas, la integración con otras herramientas de pruebas, la simulación de servicios web y la monitorización de servicios web en tiempo real. En resumen, SOAP UI es una herramienta muy completa para automatizar pruebas de

servicios web SOAP y REST, que permite a los usuarios crear, probar, depurar y monitorear servicios web de manera eficiente.

Selenium Webdriver

Selenium WebDriver es una de las partes de la suite de herramientas de automatización de pruebas de software Selenium. Es una interfaz de programación de aplicaciones (API) que permite a los desarrolladores escribir scripts de prueba automatizados para automatizar la interacción con un navegador web. WebDriver es una implementación de una especificación denominada WebDriver, que es un estándar abierto para la automatización de pruebas de navegadores web. La especificación WebDriver se encarga de establecer una interfaz común para la automatización de pruebas de navegadores web, independientemente del lenguaje de programación o el sistema operativo utilizado. WebDriver permite a los desarrolladores escribir scripts de prueba automatizados en varios lenguajes de programación, como Java, Python, C#, Ruby, etc. y los scripts se ejecutan en un navegador web real, lo que permite a los equipos de pruebas automatizar pruebas funcionales y de aceptación. En resumen, Selenium WebDriver es una interfaz de programación de aplicaciones (API) que permite a los desarrolladores escribir scripts de prueba automatizados para automatizar la interacción con un navegador web, y es una parte esencial de la suite de herramientas Selenium.

Java

Java es un lenguaje de programación de propósito general ampliamente utilizado para desarrollar aplicaciones de software. Es uno de los lenguajes de programación más populares para automatizar pruebas de software debido a su fiabilidad, escalabilidad y facilidad de uso. Java se utiliza para escribir scripts de prueba automatizados utilizando una variedad de herramientas y

marcos de automatización de pruebas, como Selenium WebDriver, JUnit, TestNG, etc. Además, existen una gran cantidad de librerías y frameworks en Java específicamente diseñados para automatizar pruebas, como Serenity, Cucumber, Appium, etc. Java es un lenguaje de programación orientado a objetos, lo que significa que se basa en la creación de objetos que interactúan entre sí para llevar a cabo una tarea. Esto facilita la creación de scripts de prueba modularizados y reutilizables

Page Of Model (POM)

Page Object Model (POM) es un patrón de diseño de automatización de pruebas de software que se utiliza para crear scripts de prueba automatizados más mantenibles y reutilizables. Se basa en la idea de crear objetos que representan las diferentes páginas o elementos de una aplicación web, y utilizarlos para interactuar con la página en lugar de escribir código para interactuar directamente con los elementos de la página. La implementación de POM implica la creación de una clase para cada página o elemento de la aplicación web, donde se especifica la lógica de la interacción con esa página o elemento. Esto permite a los desarrolladores crear scripts de prueba más fáciles de mantener y reutilizar, ya que el código de interacción con la página se encuentra en un solo lugar. Además, POM ayuda a mejorar la organización del código y aumenta la claridad en el código, ya que permite identificar fácilmente qué partes del código se relacionan con qué página o elemento de la aplicación. También ayuda a reducir el acoplamiento entre los scripts de prueba y el código de la aplicación, lo que facilita la realización de cambios en la aplicación sin tener que cambiar los scripts de prueba.

Cucumber

Cucumber es un framework de automatización de pruebas de software que se utiliza para escribir pruebas de aceptación en lenguaje natural. Se basa en el concepto de "pruebas de aceptación de comportamiento" (Behaviour-Driven Development - BDD), que permite a los desarrolladores y los usuarios comunicarse mediante el lenguaje común del negocio para definir los comportamientos esperados del sistema. Cucumber permite escribir escenarios de prueba en un formato de "dado-cuando-entonces", donde se describen los pasos necesarios para alcanzar un determinado comportamiento, y se especifica el resultado esperado. Estos escenarios se escriben en un archivo de lenguaje natural llamado "fichero de especificación" o "fichero de características", y luego se traducen a código automatizado. Cucumber se utiliza principalmente para automatizar pruebas de aceptación, ya que permite a los desarrolladores y los usuarios trabajar juntos para definir los requisitos del negocio y asegurar que el sistema cumple con estos requisitos. Además, permite a los equipos de pruebas automatizar pruebas de aceptación de manera eficiente y efectiva, y mejora la comunicación y la colaboración entre los diferentes miembros del equipo.

Behavior driven development (BDD)

Behavior Driven Development (BDD) es un enfoque de desarrollo de software que se basa en la comunicación entre los desarrolladores, los usuarios y los miembros del equipo de pruebas mediante el lenguaje natural del negocio. El objetivo principal de BDD es asegurar que el sistema cumple con los requisitos del negocio y proporciona un valor real al usuario final. BDD se basa en la idea de escribir escenarios de prueba en lenguaje natural, utilizando un formato de "dado-cuando-entonces" para describir el comportamiento esperado del sistema. Estos escenarios se utilizan para guiar el desarrollo del sistema y asegurar que el sistema cumple con los requisitos

del negocio. BDD se apoya en una metodología de desarrollo ágil, y se enfoca en la colaboración entre los diferentes miembros del equipo, asegurando que todos los miembros del equipo comprendan el objetivo de las pruebas.

Web browser Testing

Web browser Testing es el proceso de automatizar pruebas de software para evaluar el comportamiento de una aplicación web en diferentes navegadores y plataformas. Estas pruebas se utilizan para asegurar que una aplicación web se comporta de manera consistente y correcta en diferentes navegadores y plataformas. El objetivo de las pruebas de navegador es asegurar la compatibilidad de la aplicación web con los diferentes navegadores y plataformas. Estas pruebas pueden incluir tareas como simular interacciones con la página, verificar la visualización de la página, validar la funcionalidad y la estabilidad de la aplicación. Para automatizar las pruebas de navegadores, se pueden utilizar herramientas de automatización de pruebas como Selenium WebDriver, que permite escribir scripts de prueba automatizados que se ejecutan en un navegador real. También se pueden utilizar servicios de pruebas en la nube como Sauce Labs o BrowserStack, que ofrecen una plataforma para ejecutar pruebas automatizadas en diferentes navegadores y plataformas.

Maven Allure

Maven Allure es una combinación de dos herramientas de automatización de pruebas: Maven y Allure. Maven es una herramienta de gestión de proyectos de software para Java que permite a los desarrolladores automatizar varias tareas como la construcción, documentación y gestión de dependencias. Allure es una herramienta de generación de informes de pruebas

automatizadas que proporciona una interfaz web para ver y analizar los resultados de las pruebas. Permite generar informes detallados, interactivos y atractivos que facilitan la comprensión de los resultados de las pruebas. Al combinar estas dos herramientas, Maven Allure permite a los equipos de desarrollo automatizar las pruebas y generar informes detallados e intuitivos sobre los resultados de estas, lo cual facilita la toma de decisiones y el seguimiento del proyecto.

5.2. Desarrollo de la solución

Tipo de Framework Usado

Selenium Webdriver

Selenium WebDriver es un framework de automatización de pruebas para aplicaciones web. Permite a los desarrolladores automatizar las pruebas de navegador utilizando lenguajes de programación como Java, C#, Python, Ruby, entre otros. WebDriver es compatible con una variedad de navegadores, incluyendo Chrome, Firefox, Safari y Edge. Con Selenium WebDriver, se pueden crear scripts para automatizar tareas como llenar formularios, hacer clic en botones y verificar el contenido de las páginas web. Es una herramienta popular en la industria de pruebas automatizadas. La figura siguiente muestra el código para establecer una conexión con Selenium WebDriver y Java utilizando el navegador Chrome.

Figura 7

conexión de selenium webdriver con chromedriver

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class ChromeTest {
    public static void main(String[] args) {
        // Establecer la ruta al ejecutable de ChromeDriver
        System.setProperty("webdriver.chrome.driver",
            "ruta/al/ejecutable/chromedriver");

        // Crear una instancia de ChromeDriver
        WebDriver driver = new ChromeDriver();

        // Navegar a una página web específica
        driver.get("https://www.google.com");

        // Cerrar el navegador
        driver.quit();
    }
}
```

Nota: Elaboración propia

Método Page off Model

El método "Page Object Model" es un patrón de diseño que se utiliza para la automatización de pruebas con Selenium WebDriver. Este patrón consiste en crear una clase para cada página web que se va a probar, y en estas clases se describen los elementos de la página y las acciones que se pueden realizar en ellos.

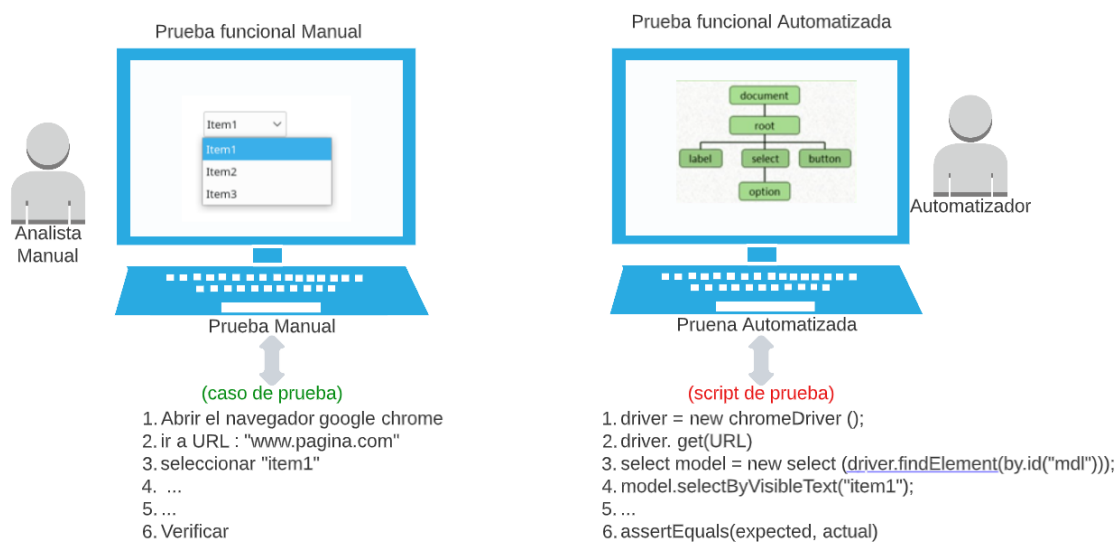
Por ejemplo, si estamos automatizando una página de inicio de sesión, podríamos crear una clase "LoginPage" que contenga los elementos de la página, como el campo de nombre de usuario y el botón de inicio de sesión. También se pueden crear métodos para realizar acciones específicas

en la página, como ingresar un nombre de usuario y contraseña y hacer clic en el botón de inicio de sesión.

De esta forma, al usar el Page Object Model, se separa la lógica de automatización de las pruebas de la lógica de la aplicación, lo que facilita la lectura y mantenimiento del código. Además, si cambian los elementos de la página, solo se tiene que modificar la clase correspondiente en lugar de tener que modificar todas las pruebas que interactúan con esa página.

Figura 8

page of model VS Analista



Nota: Elaboración propia

Figura 9

Interfaz login aplicación de Facturación electrónica

Portal de Facturación Electrónica


RUC: (*)

USUARIO: (*)

CONTRASEÑA: (*)

Olvidé mi Contraseña

CÓDIGO: (*)



Cambiar Imagen

INICIAR SESIÓN

REGÍSTRESE

Nota: Elaboración Propia

Figura 10

Script de automatización de interfaz login



```

32 public void tearDown() {
33     driver.quit();
34 }
35
36 @Test
37 // dirige a la url que señalamos
38 driver.get("http://192.168.2.107/appefacturacion/pages/login.jsf");
39 driver.manage().window().setSize(new Dimension(1616, 886));
40 driver.findElement(By.id("login:ruc")).click();
41 driver.findElement(By.id("login:ruc")).sendKeys("20386475855");
42 driver.findElement(By.id("login:username")).click();
43 driver.findElement(By.id("login:username")).sendKeys("sandra.benites@gmail.com");
44 driver.findElement(By.id("login:password")).sendKeys("sandr8638934");
45 driver.findElement(By.id("login:idLoginBuscar")).click();
46 driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
47
48 {
49     WebElement element = driver.findElement(By.id("frmPrincipal:pnListado"));
50     Actions builder = new Actions(driver);
51     builder.moveToElement(element).clickAndHold().perform();
52 }
53 {
54     WebElement element = driver.findElement(By.id("frmPrincipal:pnListado"));
55     Actions builder = new Actions(driver);
56     builder.moveToElement(element).perform();
57 }
58 {
59     WebElement element = driver.findElement(By.id("frmPrincipal:pnListado"));
60     Actions builder = new Actions(driver);
61     builder.moveToElement(element).release().perform();
62 }
63 }
  
```

Nota: Elaboración Propia

Estructura del Framework de Automatización

Figura 11*Estructura del Código en paquetes del Framework de Automatización**Nota:* Elaboración Propia

Script de automatización

Estructura de un script de automatización de pruebas utilizando Selenium WebDriver en Java , los códigos de las bandejas generales se encuentra en los anexos.

Figura 12

Script de automatización con selenium webdriver y java

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.By;

public class SeleniumTest {
    public static void main(String[] args) {

        // Establecer la ruta del controlador del navegador
        System.setProperty("webdriver.chrome.driver", "path/to/chromedriver");

        // Crear una instancia del WebDriver
        WebDriver driver = new ChromeDriver();

        // Navegar a una página web específica
        driver.get("http://www.google.com");

        // Encontrar un elemento en la página (en este caso el campo de
        búsqueda)
        driver.findElement(By.name("q")).sendKeys("Selenium WebDriver");

        // Enviar el formulario (simular clic en el botón de búsqueda)
        driver.findElement(By.name("btnK")).click();

        // Esperar a que la página cargue
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

        // Verificar que el título de la página contenga el término buscado
        assert driver.getTitle().contains("Selenium WebDriver");

        // Cerrar el navegador
        driver.quit();
    }
}
```

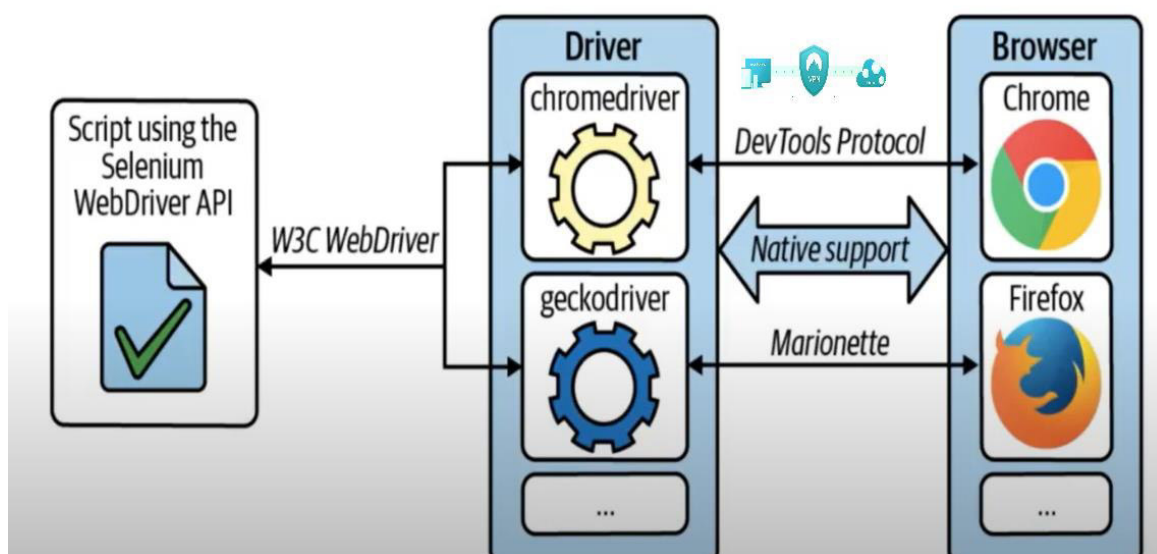
Nota: Elaboración Propia

Arquitectura de Framework de Automatización

Estructura del framework de automatización de pruebas utilizando Selenium WebDriver

Figura 13

Arquitectura de Framework de Automatización para la OSE



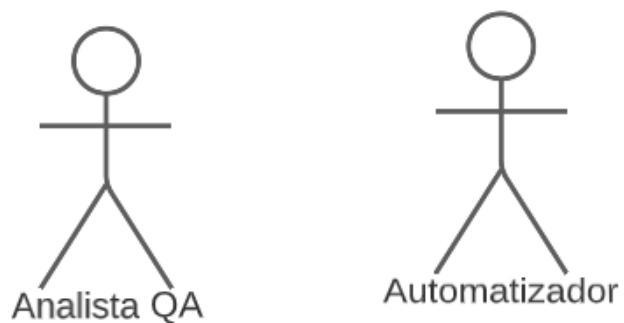
Nota: Elaboración Propia

Análisis del negocio

Actores del sistema (QA – software)

Figura 14

Actores del sistema

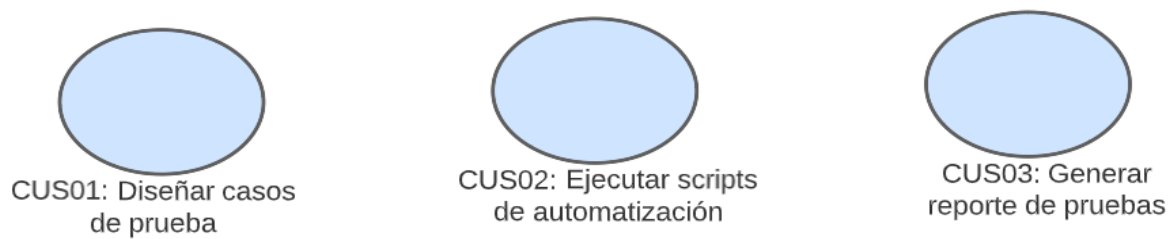


Nota: Elaboración Propia

Casos de uso

Figura 15

Casos de Uso del sistema

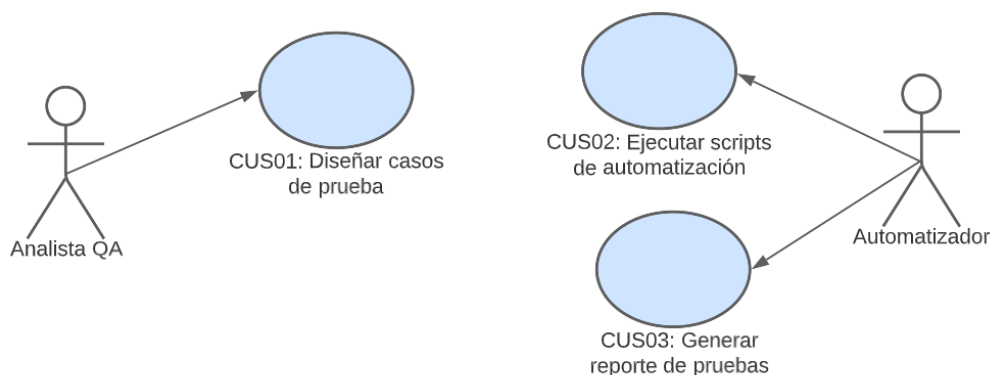


Nota: Elaboración Propia

Diagrama CUS

Figura 16

Diagrama Caso de Uso del Sistema



Nota: Elaboración Propia

Requerimientos funcionales

Tabla 2

Requerimientos funcionales del Framework de automatización

Requerimiento	Descripción
RF1	El framework debe soportar el lenguaje de programación Java para que los desarrolladores puedan escribir scripts de prueba en el lenguaje que prefieran.
RF2	El framework debe ser capaz de automatizar pruebas en el navegador Google Chrome para asegurar la compatibilidad de la aplicación web.
RF3	El framework debe permitir la integración con otras herramientas de automatización de pruebas, como Selenium WebDriver
RF4	El framework debe ser fácil de usar para los desarrolladores y los Analistas QA
RF5	El framework debe generar informes detallados de las pruebas
RF6	El framework debe permitir capturar evidencia de los bugs
RF7	El framework debe soportar diferentes tipos de pruebas
RF8	El framework debe tener la capacidad de soportar los scripts de automatización

RF9	El framework debe poder integrarse con herramientas de seguimiento de errores y control de versiones para poder rastrear y solucionar los errores detectados en las pruebas automatizadas.
RF10	El framework debe tener la capacidad de automatizar pruebas de seguridad para detectar vulnerabilidades en la aplicación.

Nota: Elaboración Propia

Requerimiento no funcional

Tabla 3

Requerimientos no funcionales

Requerimiento	Descripción
RNF1	El framework debe ser escalable para adaptarse a diferentes tamaños de proyectos y aumentar o disminuir el número de pruebas automatizadas según sea necesario.
RNF2	El framework debe contar con un equipo de soporte técnico disponible para resolver cualquier problema o pregunta que surja durante el uso del mismo.
RNF3	El framework debe permitir la personalización de pruebas y scripts según las necesidades específicas del proyecto.
RNF4	El framework debe ser sostenible a largo plazo, con una arquitectura bien diseñada y una estrategia de mantenimiento y actualización clara.
RNF5	El framework debe ser fiable y producir resultados precisos y consistentes cada vez que se ejecutan las pruebas.

Nota: Elaboración propia

Matriz requisitos vs CUS

Tabla

4

Matriz Requisitos VS CUS

	Requisitos Funcionales	Casos de uso	Actores	Descripción
RF1	El framework debe soportar el lenguaje de programación Java para que los desarrolladores puedan escribir scripts de prueba en el lenguaje que prefieran.	CUS02	Ejecutar scripts de automatización	Automatizador
RF2	El framework debe ser capaz de automatizar pruebas en el navegador Google Chrome para asegurar la compatibilidad de la aplicación web.			Este caso permitirá al Automatizador usar lenguaje java en sus scripts de automatización. Este caso debe permitir usar el navegador Google Chrome para las pruebas
RF3	El framework debe permitir la integración con otras herramientas de automatización de pruebas, como Selenium WebDriver			Este caso permitirá integrar herramientas de automatización como selenium, Maven
RF4	El framework debe ser fácil de usar para los desarrolladores y los Analistas QA	CUS01	Diseñar casos de prueba	Analista QA
RF5	El framework debe generar informes detallados de las pruebas	CUS03	Generar reportes de pruebas	Automatizador
RF6	El framework debe permitir capturar evidencia de los bugs			Este caso permitirá obtener reportes de las pruebas ejecutadas Este caso permitirá obtener evidencia de las pruebas ejecutadas
RF7	El framework debe soportar diferentes tipos de pruebas	CUS02	Ejecutar scripts de automatización	Este caso permitirá usar el framework para las distintas pruebas que se necesiten.
RF8	El framework debe tener la capacidad de soportar los scripts de automatización			Este caso permitirá contar con distintos modelos de scripts de automatización
RF9	El framework debe poder integrarse con herramientas de seguimiento de errores y control de versiones para poder rastrear y solucionar los errores detectados en las pruebas automatizadas.			Este caso permitirá tener un control de versiones del framework y de los errores encontrados para ser reportados en herramientas de gestión de errores con mantis o xray
RF10	El framework debe tener la capacidad de automatizar pruebas de seguridad para detectar vulnerabilidades en la aplicación.	CUS01	Diseñar casos de prueba	Analista QA
				Este caso permitirá cubrir posibles vulnerabilidades de seguridad cubiertas en las pruebas.

Nota: Elaboración propia

5.3. Factibilidad técnica – operativa

Factibilidad Técnica

La factibilidad técnica de una investigación se refiere a la capacidad de llevar a cabo un estudio utilizando los recursos y herramientas disponibles. Esto incluye consideraciones como el acceso a sujetos de estudio, la disponibilidad de tecnologías y equipos necesarios, y la capacidad de analizar e interpretar los datos recolectados. Una investigación que es técnicamente factible tiene una mayor probabilidad de éxito en la obtención de resultados significativos y confiables.

La factibilidad técnica para crear un framework de automatización con Selenium WebDriver y Java es alta. Selenium WebDriver es una herramienta ampliamente utilizada y soportada para automatizar pruebas de navegadores web, y Java es un lenguaje de programación popular y potente que se puede utilizar para desarrollar aplicaciones de automatización. Además, existen una gran cantidad de recursos y documentación disponibles para ayudar en el desarrollo de un framework de automatización con Selenium WebDriver y Java. Sin embargo, hay algunas consideraciones a tener en cuenta al desarrollar un framework de automatización con Selenium WebDriver y Java. Por ejemplo, se debe tener en cuenta la compatibilidad del framework con diferentes versiones de navegadores y sistemas operativos, y se deben implementar mecanismos adecuados para manejar problemas como la detección de elementos dinámicos en la página web.

Los recursos mínimos necesarios que se utilizan para el desarrollo de un framework de automatización de pruebas funcionales son:

Tabla 5*Especificaciones del Hardware del servidor de pruebas*

Especificaciones de Hardware servidor de pruebas	
Procesador	Intel Core 7 @ 2.6 GHZ
Memoria RAM	16 GB
Disco duro	1 TB

Nota: Elaboración Propia

Los analistas funcionales para realizar las pruebas automatizadas necesitan las siguientes especificaciones en sus equipos

Tabla 6*Especificaciones del hardware para equipos de Analistas QA*

Especificaciones del Hardware para equipos de Analistas QA	
Procesador	Intel core i9 11th gen
Memoria Ram	16 GB
Tipo de Disco	SSD
Capacidad de Disco	1 TB

Nota: Elaboración Propia

Las características del software a necesitar son las siguientes

Tabla 7*Especificaciones del sistema software*

Especificaciones de Software del servidor	
Sistema Operativo	Windows server 2008
IDE	Eclipse o IntelliJ IDEA
Antivirus	Symantec
Gestor de DB	MySQL

Nota: Elaboración Propia

Es importante mencionar que existen herramientas de automatización de pruebas en la nube que ofrecen una plataforma de automatización de pruebas y una interfaz de programación de aplicaciones (API) para automatizar pruebas de aplicaciones web y móviles, estas herramientas incluyen un costo mensual dependiendo el uso, pero son una opción para reducir costos en infraestructura y personal.

Factibilidad Operativa

La factibilidad operativa para crear un framework de automatización con Selenium WebDriver y Java es alta. Selenium WebDriver es una herramienta ampliamente utilizada y soportada para automatizar pruebas de navegadores web, y Java es un lenguaje de programación popular y potente que se puede utilizar para desarrollar aplicaciones de automatización.

Además, existen una gran cantidad de recursos y documentación disponibles para ayudar en el desarrollo de un framework de automatización con Selenium WebDriver y Java, incluyendo tutoriales, ejemplos y bibliotecas de terceros. Con un equipo de desarrolladores con experiencia en Java y automatización de pruebas, debería ser posible desarrollar un framework sólido y escalable.

Sin embargo, hay algunas consideraciones a tener en cuenta en cuanto a la factibilidad operativa al desarrollar un framework de automatización con Selenium WebDriver y Java. Por ejemplo, es importante tener en cuenta la infraestructura existente y las necesidades específicas del negocio para asegurar que el framework se ajuste a los requerimientos operativos. También se debe tener en cuenta la capacidad de soporte y mantenimiento del framework una vez implementado.

En cuanto a los recursos necesarios para implementar y operar el framework, es importante contar con un equipo de desarrolladores con experiencia en Java y automatización de pruebas, así

como un ambiente adecuado de pruebas y una infraestructura de automatización escalable. Además, se deben implementar mecanismos para monitorear y mantener el framework, como un sistema de seguimiento de errores y un plan de contingencia en caso de fallas.

Tabla 8*Características del Analista QA Funcional*

Tipo de Usuario	Analista QA
Formación	Ingeniería de sistemas y/o afines
Habilidades	Diseño, creación y ejecución de pruebas
Actividades	Diseño de casos de pruebas

Nota: Elaboración Propia

Tabla 9*Características de Automatizador*

Tipo de usuario	Automatizador de pruebas
Formación	Desarrollador
Habilidades	Especialista en Java y selenium
Actividades	Crear script de automatización

Nota: Elaboración Propia

5.4. Cuadro de inversión

Un cuadro de inversión para un framework de automatización con Java podría incluir los siguientes costos:

Tabla 10*Costos en Recursos humanos*

Recursos Humanos					
Item	Personal	Cantidad	Meses	sueldo	Total
1	Líder del Proyecto	1	4	S/ 5,500.00	S/ 22,000.00
2	Analista de pruebas A.	1	3	S/ 3,150.00	S/ 9,450.00
3	Desarrollador de A.	1	2	S/ 2,500.00	S/ 5,000.00
4	Tester manual	1	2	S/ 2,200.00	S/ 4,400.00
5	Arquitecto de sistemas	1	1	S/ 4,000.00	S/ 4,000.00
6	Analista de Negocio	1	3	S/ 3,500.00	S/ 10,500.00
Total					S/ 55,350.00

Nota: Elaboración Propia

Tabla 11

Recursos de software

Software					
ítem	Descripción	Cantidad	Meses	Subtotal	Total
1	Licencia IDE	1		S/ 1,500.00	S/ 1,500.00
1	Licencia symantec	1		S/ 87.00	S/ 87.00
1	Licencia windows server	1		S/ 450.00	S/ 450.00
Total					S/ 2,037.00

Nota: Elaboración Propia

Tabla 12

Recursos de hardware

Hardware					
ítem	descripción	Cantidad	Meses	Subtotal	Total
1	Laptop	6		S/ 3,500.00	S/ 21,000.00
2	Servidor	1		S/ 2,500.00	S/ 2,500.00
Total					S/ 23,500.00

Nota: Elaboración Propia

Tabla 13

Costos totales

ítem	descripción	Total
1	Recursos Humanos	S/ 55,350.00
2	Software	S/ 2,037.00
3	Hardware	S/ 23,500.00
Total		S/ 80,887.00

Nota: Elaboración Propia

El costo total de la implementación de un Framework de automatización es S/ 80,887.00

6. Análisis de Resultados

6.1. Análisis costos – beneficios

La implementación de un framework de Automatización de pruebas funcionales para una Ose, brinda una cantidad de beneficios a la compañía de los cuales se pueden diferenciar en dos grupos, los no financieros y los Financieros

Beneficio no Financiero

La implementación de un framework de software muestra los siguiente beneficios no Financieros:

- Mejora de la eficiencia: Automatizando tareas repetitivas y tediosas, se libera tiempo para que los empleados se enfoquen en tareas más importantes y de mayor valor agregado.
- Mayor precisión: La automatización reduce el margen de error humano al realizar tareas, lo que aumenta la precisión y la confiabilidad de los resultados.
- Escalabilidad: El framework de automatización permite a las empresas escalar sus operaciones y procesos de manera eficiente.
- Mejora en la toma de decisiones: La automatización puede recopilar y analizar datos de manera más rápida y precisa, lo que permite a las empresas tomar decisiones informadas.
- Mejora en la seguridad: La automatización puede ayudar a las empresas a mejorar la seguridad de sus sistemas y datos al monitorear y responder a posibles amenazas de manera automatizada.
- Mejora en la satisfacción del cliente: La automatización puede ayudar a las empresas a mejorar la atención al cliente al procesar solicitudes y resolver problemas de manera más rápida y eficiente.

Beneficio financiero

Los beneficios financieros que brinda implementar un framework de Automatización se realiza a través de un análisis económico donde se comparen los gastos con los ingresos, evaluando el VAN y el TIR del Proyecto.

VAN (Valor Actual Neto) y TIR (Tasa Interna de Retorno) son dos indicadores financieros utilizados para medir la viabilidad económica de un proyecto. VAN: El VAN es una medida del valor económico de un proyecto, que indica si el proyecto genera suficientes flujos de efectivo para cubrir los costos iniciales del mismo y generar un retorno adicional para los inversionistas. El VAN se calcula sumando los flujos de efectivo futuros de un proyecto, descontándolos a una tasa de descuento determinada, y restando los costos iniciales del proyecto. Si el VAN es positivo, el proyecto es considerado viable económicamente.

TIR: La TIR es una medida de la rentabilidad de un proyecto, que indica el porcentaje de retorno que el proyecto genera sobre la inversión inicial. Se calcula como la tasa de descuento que hace que el VAN sea igual a cero. Es decir, es la tasa que hace que el valor actual de los flujos de caja futuros de un proyecto sea igual al costo de inversión. Una TIR mayor al costo de oportunidad de los fondos de inversión, es considerado un buen indicador de viabilidad del proyecto.

En resumen, el VAN se utiliza para determinar el valor económico de un proyecto, mientras que la TIR se utiliza para determinar la rentabilidad de un proyecto.

A continuación se muestra el Flujo económico del proyecto de implementación de framework de automatización de pruebas.

Resultado del Análisis financiero del Flujo de caja de la implementación de framework de automatización de pruebas funcionales en una OSE

Tabla 14*Van y TIR de la Implementación de un Framework de automatización*

Inversión	S/ 10,125.00
COK	10%
VAN	S/ 54,074.49
TIR	45%

Nota: Elaboración Propia

Con los valores obtenidos de Van y TIR se puede visualizar que el proyecto es viable y se obtienen los beneficios esperados.

Tabla 15*Flujo de caja del proyecto*

	Mes 0	Mes 1	Mes 2	Mes 3	Mes 4	Mes 5	Mes 6	Mes 7	Mes 8	Mes 9	Mes 10	Mes 11	Mes 12
INGRESOS													
Incremento de Transacciones	-	S/ 10,000	S/ 10,000	S/ 9,000	S/ 9,000	S/ 11,000	S/ 11,000	S/ 11,000	S/ 15,000	S/ 15,000	S/ 15,000	S/ 22,000	S/ 22,000
Aumento de flujo de comprobantes	-	S/ 4,000	S/ 4,000	S/ 4,000	S/ 4,000	S/ 4,000	S/ 4,000	S/ 4,000	S/ 4,000	S/ 4,000	S/ 4,000	S/ 4,000	S/ 4,000
Reducción de consumo de recursos		S/ 800	S/ 800	S/ 800	S/ 800	S/ 800	S/ 800	S/ 800	S/ 800	S/ 800	S/ 800	S/ 800	S/ 800
Reducción de costo por Horas hombre	-	S/ 500.0	S/ 500.0	S/ 500.0	S/ 500.0	S/ 500.0	S/ 500.0	S/ 500.0	S/ 500.0	S/ 500.0	S/ 500.0	S/ 500.0	S/ 500.0
TOTAL DE INGRESOS	0	S/ 15,300	S/ 15,300	S/ 14,300	S/ 14,300	S/ 16,300	S/ 16,300	S/ 16,300	S/ 20,300	S/ 20,300	S/ 20,300	S/ 27,300	S/ 27,300
EGRESOS													
Gasto en Personal	S/ 5,000.00	S/ 5,000.00	S/ 16,300.00	S/ 16,300.00	5000	S/ 5,000.00	S/ 5,000.00	2000	2000	2000	2000	2000	20000
Software	S/ 3,500.00	-	S/ 1,125.00	S/ 1,125.00	-	-	-	-	-	-	-	-	-
Hardware	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00	S/ 1,625.00
TOTAL DE EGRESO	S/ 10,125.00	S/ 6,625.00	S/ 19,050.00	S/ 19,050.00	S/ 6,625.00	S/ 6,625.00	S/ 6,625.00	S/ 3,625.00	S/ 3,625.00	S/ 3,625.00	S/ 3,625.00	S/ 3,625.00	S/ 21,625.00
UTILIDAD	- S/10,125.00	S/ 8,675.00	-S/ 3,750.00	-S/ 4,750.00	S/ 7,675.00	S/ 9,675.00	S/ 9,675.00	S/ 12,675.00	S/ 16,675.00	S/ 16,675.00	S/ 16,675.00	S/ 23,675.00	S/ 5,675.00
Impuesto	-1822.50	S/ 1,561.50	-S/ 675.00	-S/ 855.00	S/ 1,381.50	S/ 1,741.50	S/ 1,741.50	S/ 2,281.50	S/ 3,001.50	S/ 3,001.50	S/ 3,001.50	S/ 4,261.50	S/ 1,021.50
FLUJO NETO	-S/ 8,302.50	S/ 7,113.50	-S/ 3,075.00	-S/ 3,895.00	S/ 6,293.50	S/ 7,933.50	S/ 7,933.50	S/ 10,393.50	S/ 13,673.50	S/ 13,673.50	S/ 13,673.50	S/ 19,413.50	S/ 4,653.50

Nota: Elaboración Propia

Análisis de Resultados

La implementación de un framework de automatización para pruebas funcionales ha logrado los siguientes resultados en base a los objetivos planteados. Comprendiendo que los objetivos alcanzados son un pilar importante en el desarrollo del proyecto, por eso se realizara un análisis en cada objetivo planteado

Determinar si la implementación de un framework de automatización mejora el proceso de pruebas funcionales del sistema web de una OSE

Tabla 16

Determinación de un Framework de Automatización para las pruebas funcionales

	Antes de la implementación	Después de la implementación	Diferencia Después - Antes
Descripción	Cantidad Promedio	Cantidad Promedio	Cantidad
CT: Casos Totales de pruebas ejecutados	120	300	180
CPNI: Casos de pruebas no impacto	20	80	60
CPA: Casos de pruebas Automatizados	0	280	-280

Nota: Elaboración propia

Según los resultados obtenidos se observa que la implementación de un framework de automatización mejoro el proceso de pruebas funcionales a través del número de casos totales de pruebas ejecutados aumentaron de 120 en promedio a 300 en promedio , adicionalmente también aumento el número de casos de no impacto ejecutados en cada requerimiento de 20 a 80 ayudando a mitigar los errores futuros en los módulos del sistema , finalmente el número de pruebas automatizados aumento creando una nueva practica en el área de Testing, innovando en la automatización de pruebas bajo la curva de aprendizaje correspondiente con las herramientas necesarias y acorde a las necesidades de sistema web de facturación electrónica.

Identificación de errores

Tabla 17

Verificar el proceso de identificación de errores.

	Antes de la implementación	Después de la implementación	Diferencia Después - Antes
Descripción	Cantidad Promedio	Cantidad Promedio	Cantidad
Numero de errores encontrados en las pruebas de Humo	2	5	3
Cantidad de errores en las pruebas funcionales	6	13	7
Numero de fallos post Testing	3	0	-3

Nota: Elaboración Propia

La observación de los resultados obtenidos mediante los indicadores de la dimensión de identificación de errores de las pruebas funcionales para un framework de automatización indican que el número de errores hallados en las pruebas iniciales o de humo han aumentado detectando a tiempo fallos potenciales , después la cantidad de errores encontrados en las pruebas funcionales aumento permitiendo abarcar la mayor cantidad de módulos y/o características del sistema, finalmente el número de errores post Testing , es decir el número de errores encontrados en producción después de una certificación de Testing se lograron reducir al máximo. En conclusión se verifica que se logró aumentar la identificación de fallos en el sistema a tiempo.

Tiempo empleado en las pruebas

Tabla 18*Verificar el tiempo empleado en las pruebas*

	Antes de la implementación	Después de la implementación	Diferencia antes - Después
Descripción	Tiempo (días)	Tiempo (días)	Tiempo (días)
Tiempo empleado en las pruebas funcionales	7	4	3
Tiempo empleado en la elaboración de casos de prueba	2	1	1
Tiempo empleado en las pruebas de no impacto y regresión	0	2	-2

Nota: Elaboración Propia

En base a los resultados obtenidos de la evaluación y comparación de los tiempos empleados antes y después del framework de automatización se concluye lo siguientes, el tiempo empleado en la realización de las pruebas funcionales se redujo de 7 a 4 días lo que permite tener a los analistas de calidad disponibles para futuros requerimientos, adicional a ellos el tiempo de elaboración de casos de pruebas se redujo al 50% del tiempo, permitiendo empezar las pruebas antes del tiempo anterior, finalmente se agregó tiempo para las pruebas de no impacto y regresión las cuales permiten considerar escenarios extras a las pruebas funcionales y brindar mayor seguridad en el pase a producción. Se puede concluir que el tiempo se aprovecha de manera más eficiente con el apoyo del framework de automatización de pruebas, el cual es un apoyo para los analistas de calidad, ya que les permite aprovechar el tiempo ahorrado en aprender nuevo conocimiento y mejorar el framework de automatización para obtener mejores resultados.

Costos de solución de errores

Tabla 19*Costos de la implementación de solución*

	Antes de la implementación	Después de la implementación	Diferencia antes - Después
Descripción	soles	soles	soles
Costo por diseño de pruebas	150	100	50
Costo por analista programador en corrección de issues	400	200	200
Costo de pruebas post despliegue a UAT	500	150	350

Nota: Elaboración Propia

En base a los resultados obtenidos por el análisis comparativo de la situación actual con referencia a la situación antes de la solución se concluye lo siguiente, el costo por diseño de pruebas se redujo esto debido a que ya se cuenta con una plantilla reutilizables para los casos de pruebas que solo necesita mantenimiento en periodos largos de tiempo, además el costo por analista programador disminuyo debido a que los errores se encuentran a tiempo antes del pase a producción evitan sobre costos de corrección de issues por la parte de desarrollo, adicional a ello el costo de pases al ambiente de Testing para usuarios es menor debido a que los despliegues ahora son limpios de errores y a tiempo.

7. Aportes Destacables a la Empresa

La solución planteada proporciona varios beneficios a la empresa, entre los más destacables podemos mencionar:

- Ahorro de tiempo y costos: La automatización de pruebas puede reducir significativamente el tiempo y los costos necesarios para realizar pruebas manuales, permitiendo que los equipos se enfoquen en tareas más importantes.
- Mayor precisión: Las pruebas automatizadas son menos propensas a errores humanos, lo que aumenta la precisión y la confiabilidad de las pruebas.
- Mayor cobertura de pruebas: El uso de un framework permite la creación de scripts de pruebas que pueden ser ejecutados de manera automática y repetitiva, lo que aumenta la cobertura de pruebas y permite detectar errores de manera temprana.
- Mejora en la calidad del software: Al detectar y corregir errores tempranamente, se puede mejorar la calidad del software entregado a los clientes.
- Facilidad de mantenimiento: Un framework bien diseñado permite la reutilización de código y la fácil actualización de los scripts de pruebas.
- Mejora de la eficiencia y velocidad de pruebas funcionales del sistema de facturación electrónica.
- Mayor consistencia en la ejecución de pruebas sobre todo en las pruebas de no impacto del sistema de facturación electrónica
- Facilidad para mantener y actualizar scripts de prueba en los distintos módulos del sistema.
- Reducción de errores humanos (tester) al automatizar tareas repetitivas del sistema.
- Mayor cobertura de pruebas al ejecutar múltiples pruebas de manera simultánea como las funcionales, no impacto y regresión.

- Integración con otras herramientas de Testing y automatización como Jira, xray o un gestor de Bugs.

8. Conclusiones

En el presente informe se determinó como la implementación de un framework de automatización mejora el proceso de pruebas funcionales del sistema web de una OSE, Lima, 2023. En palabras de Spillner (2014), manifestó que La prueba de regresión funcional implica revisar un programa previamente probado después de haber sido alterado. Se lleva a cabo para verificar que los cambios realizados no hayan causado problemas en el sistema. Además, Hernández (2021), indica que el desarrollo de un framework gráfico permitió realizar pruebas automatizadas en el área de Testing. Al momento de buscar nuevas soluciones para las pruebas, hay muchas incertidumbres sobre qué lenguaje utilizar, qué tipo de extensión es adecuada y basada en qué navegador. Además, se requiere un framework que permita la transición de las pruebas funcionales manuales a automatizadas de manera efectiva, con una curva de aprendizaje no muy pronunciada. La mayoría de los testers no son expertos en programación y su fortaleza está en el análisis, pero aun así es necesario innovar en las pruebas a través de la automatización. Por lo tanto, se concluyó La implementación de un framework de automatización mejora el proceso de pruebas funcionales en un sistema web de una OSE. La prueba de regresión funcional implica revisar un programa previamente probado para verificar que los cambios no hayan causado problemas en el sistema. El desarrollo de un framework gráfico permitió realizar pruebas automatizadas en el área de Testing, reduciendo la incertidumbre sobre qué lenguaje utilizar y permitiendo una transición efectiva a las pruebas automatizadas con una curva de aprendizaje moderada. A pesar de que la mayoría de los testers no son expertos en programación, es necesario innovar en las pruebas a través de la automatización.

En el presente informe se verifico si la implementación de un framework de automatización mejora el proceso de pruebas funcionales en base a la dimensión de identificación de errores del sistema web de una OSE, Lima, 2023. En palabras de Pantaleo (2011) dice que muchas empresas especialistas en pruebas de software son contratadas para realizar pruebas en proyectos, lo que lleva a los responsables a creer que así están mejorando la calidad de su producto. Pero en la mayoría de los casos, solo pagan para saber cuántos errores hay en su software. Estas pruebas no mejoran la calidad, solo detectan errores y en el mejor de los casos, brindan posibles soluciones, los proyectos suelen ser evaluados por compañías especializadas en Testing, pero esto no garantiza una mejora en la calidad del software. En la mayoría de los casos, solo se les informa a los responsables de los errores en su software, y el dinero pagado no contribuye a la mejora de la calidad, solo a la identificación de errores y, en el mejor de los casos, a la dirección de soluciones potenciales. Además, Gutierrez (2022), indica que se busca solucionar las constantes incidencias reportadas por los clientes al área de soporte, las cuales impactan negativamente en la organización, causando demoras y costos en el proceso de revisión y pruebas de funcionalidad para corregir los errores encontrados. La optimización se lleva a cabo a través de una plataforma de automatización que utiliza Selenium Web Controlador, con el objetivo de reducir el malestar de los clientes y asegurarse de que el sistema cumpla con sus especificaciones y requisitos, en resumen se concluye que la implementación de un framework de automatización mejora el proceso de pruebas funcionales en una OSE. Se menciona que muchas empresas especializadas en pruebas de software son contratadas para realizar pruebas, pero esto no garantiza una mejora en la calidad del software y solo detecta errores y brinda posibles soluciones. Además, se menciona que la optimización se lleva a cabo para solucionar las incidencias reportadas por los clientes, reducir el malestar de los clientes y asegurarse de que el sistema cumpla con sus especificaciones y

requisitos. La solución se implementa a través de una plataforma de automatización utilizando Selenium Web Controlador.

En el presente informe se verifico si la implementación de un framework de automatización mejora el proceso de pruebas funcionales en base a la dimensión de tiempo empleado en las pruebas del sistema web de una OSE, Lima, 2023. Según la SSTQB (2008) menciona al tiempo de pruebas como herramienta que brinda datos sobre el rendimiento del código de software en tiempo real. Se utiliza comúnmente para detectar punteros sin asignar, verificar cálculos de punteros y monitorear el uso, la asignación y la liberación de memoria, y para detectar fugas de memoria. Además, Capcha (2018) manifiesta que el objetivo de la implementación de un Framework de Automatización en el área de Testing en un proyecto de Desarrollo es solucionar el problema de retrasos en la ejecución de los casos de pruebas de regresión. A pesar de contar con 5 analistas de calidad enfocados en estas pruebas, el proceso de verificación todavía requería hasta tres días de tiempo, causando un cuello de botella en el proceso de desarrollo de software. La realización manual de las pruebas, que dependía de la intervención humana en todo momento, resultaba en un aumento de tiempo y colaboradores necesarios si la cantidad de sprint a atender aumentaba. se concluye que la implementación de un framework de automatización en el área de Testing en un proyecto de desarrollo es necesaria para mejorar el proceso de pruebas funcionales en términos de tiempo empleado. La herramienta de tiempo de pruebas brinda información en tiempo real sobre el rendimiento del código de software y la implementación de un framework de automatización busca solucionar el problema de retrasos en la ejecución de pruebas de regresión. A pesar de contar con suficientes analistas de calidad, la realización manual de las pruebas requería mucho tiempo y colaboradores, lo que generaba cuellos de botella en el proceso de desarrollo de software.

En el presente informe se verifico si la implementación de un framework de automatización mejora el proceso de pruebas funcionales en base a la dimensión de costos de solución de errores en las pruebas del sistema web de una OSE, Lima, 2023. Según Cortes (2020), manifiesta que la implementación de un marco de automatización de pruebas de regresión en Selenium es la solución a un problema presente en la empresa Surecomp, dedicada a desarrollar software financiero. La empresa carece de un proceso formal de Testing, lo que resulta en una falta de documentación clara para la comunicación con otras áreas, retrasos en el diseño de los casos de pruebas y extensiones en las entregas de informes de certificación de software. Esto genera pérdidas de tiempo para el área de Testing y para las áreas que tienen procesos compartidos con ella, así como también desorden y descontento entre clientes internos y externos de la compañía. Además, Martínez Orozco et al., (2014), indican que la evaluación económica de un proyecto se basa en la comparación de los gastos e ingresos a lo largo del tiempo, teniendo en cuenta el descuento correspondiente. La principal ventaja económica de un software es la reducción de costos laborales. El cálculo de la reducción de mano de obra debe considerar tanto la tasa horaria como los ahorros adicionales. En la mayoría de las aplicaciones, una inversión en un sistema resulta económicamente justificada, permitiendo realizar tareas de monitoreo o control de instalaciones complejas desde estaciones de computación simples con requisitos mínimos de hardware y una conexión a Internet confiable.

9. Recomendaciones

Se recomienda la implementación de un framework de automatización para mejorar el proceso de pruebas funcionales en un sistema web de una OSE, permitiendo realizar pruebas automatizadas con reducción de incertidumbres y una curva de aprendizaje moderada.

se recomienda considerar la implementación de un framework de automatización para mejorar el proceso de pruebas funcionales y reducir el impacto de las incidencias reportadas por los clientes en la organización. La utilización de una plataforma de automatización como Selenium Web Controlador puede ser una solución efectiva para asegurarse de que el sistema cumpla con las especificaciones y requisitos requeridos por los clientes, reduciendo el malestar y mejorando la calidad del software.

Se recomienda implementar un framework de automatización en el área de Testing para mejorar el proceso de pruebas funcionales y reducir el tiempo empleado en las mismas, lo que a su vez permitirá resolver los problemas de retrasos y ser más eficiente en el proceso de desarrollo de software.

Se recomienda implementar un framework de automatización de pruebas para mejorar el proceso de pruebas funcionales en una OSE. La implementación de un marco de automatización de pruebas de regresión en Selenium puede solucionar los problemas actuales de falta de documentación clara y retrasos en la entrega de informes de certificación de software. La implementación puede resultar económicamente justificada, ya que puede reducir los costos laborales y permitir tareas de monitoreo o control de instalaciones complejas desde estaciones de computación simples.

10. Referencias

- Blasco Alexandra. (2012, January 10). *Automatización de pruebas: Un paso fundamental para mejorar la calidad del software*. <https://www.clavei.es/blog/automatizacion-de-pruebas-un-paso-fundamental-para-mejorar-la-calidad-del-software/>
- Capcha Coronado Edwin. (2018). *Implementación del Framework de automatización de proceso de QA en un proyecto de diseño de software en una consultora* [Tesis para optar el título profesional de ingeniero de sistemas, Universidad Nacional Mayor de San Marcos]. <https://hdl.handle.net/20.500.12672/10210>
- Carhuamaca, G. (2017). *Sistema que reemplaza funciones de un operador humano durante la validación de documentos digitales en Core Andina Group*. <https://hdl.handle.net/20.500.12692/26911>
- Cortés, A. (2020). *Automatización de pruebas de regresión para reducción de tiempo de entrega de nuevas versiones de software* [Tesis para optar el grado de Magister en tecnologías de la información, Universidad de Chile]. <https://repositorio.uchile.cl/handle/2250/177640>
- Cumbreras Aguaded, C., & Conesa Fuentes, M. C. (2006). *Usabilidad En Las Páginas Web: Distintas Metodologías, Creación De Una Guía De Evaluación Heurística Para Analizar Un Sitio Web, Aplicación En Enfermería*. 5(2), 1–17. <http://www.redalyc.org/articulo.oa?id=365834731037>
- Friedenberg, D., Hamburg, M., McKay, J., Posthuma, M., Schaefer, H., Smilgin, R., Smith, M., Toms, S., Ulrich, S., Walsh, M., Zakaria, E., Müller, T., Beer, A., Klonk, M., Verma, R., Graham, D., van Veenendaal, E., Black, R., Eldh, S., ... Thompson, G. (2018). *Certified Tester Foundation Level Syllabus*.

- Graham, D., & Fewster, M. (2012). *Praise for Experiences of Test Automation*.
<https://ptgmedia.pearsoncmg.com/images/9780321754066/samplepages/0321754069.pdf>
- Gutiérrez, D. (2022). *Implementación De Plataforma De Automatización De Procesos Usando “Selenium Web Driver” Para Optimizar Las Pruebas De Regresión En San Isidro, 2021*.
http://repositorio.ulasamericas.edu.pe/bitstream/handle/upa/2338/01_TESIS_GUTIERREZ%20ZAPATA%20DANIELA%20LORENZ.pdf?sequence=1&isAllowed=y
- Hernández, D. S. (2021). *Desarrollo de un framework de pruebas automatizadas para la verificación de aplicaciones web*. [Tesis para obtener el grado de Maestro en Sistemas Computacionales, Universidad Autónoma de Querétaro]. <http://ri-ng.uaq.mx/handle/123456789/3384>
- International Software Testing Qualifications Board. (2018). *Probador Certificado Programa de Estudio de Nivel Avanzado*.
- Jorgensen, P. C. (2013). *Software Testing Fourth Edition A Craftsman’s Approach* (4th ed.). 2013.
<https://malenezi.github.io/malenezi/SE401/Books/Software-Testing-A-Craftsman-s-Approach-Fourth-Edition-Paul-C-Jorgensen.pdf>
- Madi Rawane. (2013). *Learning Software Testing with Test Studio*.
- McKay Judy, & Graham Bath. (2014). *The Software Test Engineer’s Handbook, 2nd Edition, 2nd Edition* (2nd ed.). Rocky Nook.
- Orozco, J. M., Felipe, D., Toro, H., Andrés, E., & Salazar, Q. (2014). Aplicativo para el Acceso vía Web a Observatorios Astronómicos Web Application for Remote Control of Astronomical Observatories. *Scientia et Technica Año XIX, 19(1)*, 77–83.
<http://www.utp.edu.co/observatorioastronomico/observatorio->

OVHcloud. (2023, January 15). *Certificaciones e informes ISO/IEC 27001, 27017 y 27018*.

<https://www.ovhcloud.com/es/enterprise/certification-conformity/iso-27001-27017-27018/>

Pantaleo Guillermo. (2011). *Calidad En El Desarrollo De Software* (1st ed.).

Qalified Building quality. (2022, September 19). *¿Qué es un Framework de Automatización de*

Pruebas? Todo lo que debes saber. <https://qalified.com/es/framework-automatizacion-pruebas/>

Rivera Martínez, C. A. (2018). *Automatización de pruebas de regresión* [Tesis para optar al Grado

de Magister en Tecnologías de la información., Universidad de Chile].

<https://repositorio.uchile.cl/handle/2250/165608>

Spillner, Andreas., Linz, Tilo., & Schaefer, H. (Hans). (2014). *Software testing foundations : a*

study guide for the certified tester exam. Rocky Nook.

TCI. (2020a). *Misión, visión de la empresa de facturación electrónica*.

TCI. (2020b). *Política De Seguridad De La Información De La Empresa De Facturación*

Electrónica. <https://www.tci.net.pe/mision-vision-y-valores/>

11. Anexos

Anexo 1

Matriz operacionalización de variables

Titulo		Implementación De Un Framework De Automatización Para Mejorar El Proceso De Pruebas Manuales Del Sistema Web De Una Ose, Lima, 2023		
variables	definición Conceptual	definición Operacional	Dimensiones	Escala de medición
Independiente				
Framework de automatización	“La automatización consiste en la construcción de un conjunto de scripts reutilizables, con los que podemos aumentar drásticamente la capacidad de testear software. Permite reducir los costes de cualquier organización que necesite probar sucesivas versiones de un mismo producto.” (Blasco, 2012)	La automatización es un Proceso por el cual se permite obtener mayor cantidad de pruebas a distintos niveles desde las unitarias hasta las integrales abarcando todo el sistema, además de escalar en nuevas características de este.	Rendimiento	ordinal
			Diseño	
			Proceso	
Dependiente				
Pruebas Funcionales	"La prueba funcional de un sistema incluye pruebas que evalúan las funciones que el sistema debe realizar. Los requisitos funcionales pueden estar descritos en productos de trabajo tales como especificaciones de requisitos de negocio, épicas, historias de usuario, casos de uso, o especificaciones funcionales, o pueden estar sin documentar"(ISQTB, 2018, p.51)	Proceso que ejecuta un analista QA para revisar y verificar las características funcionales de un sistema a través de un proyecto o requerimiento.	identificación de errores, fallos, defectos y hallazgo del software	ordinal
			Tiempo empleado en pruebas	
			Costo en solucionar errores, fallos, defectos y hallazgos del software	

Elaboración Propia

Anexo 2

Código de Bandeja emisor

```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class BandejaEmisorInputsTest {
    private WebDriver driver;
    private Map<String, Object> vars;
    JavascriptExecutor js;
    @Before
    public void setUp() {
        driver = new ChromeDriver();
        js = (JavascriptExecutor) driver;
        vars = new HashMap<String, Object>();
    }
    @After
    public void tearDown() {
        driver.quit();
    }
    @Test
    public void bandejaEmisorInputs() {
        // Test name: BandejaEmisorInputs
        // Step # | name | target | value
        // 1 | open | http://192.168.2.107/appefacturacion/pages/login.jsf#
|
driver.get("http://192.168.2.107/appefacturacion/pages/login.jsf#");
        // 2 | setWindowSize | 1616x886 |
        driver.manage().window().setSize(new Dimension(1616, 886));
        // 3 | click | id=login:ruc |
        driver.findElement(By.id("login:ruc")).click();
        // 4 | type | id=login:ruc | 20112811096
```

```

        driver.findElement(By.id("login:ruc")).sendKeys("20112811096");
        // 5 | click | css=.form-login > .row-table:nth-child(3) |
        driver.findElement(By.cssSelector(".form-login > .row-table:nth-
child(3)")).click();
        // 6 | click | id=login:username |
        driver.findElement(By.id("login:username")).click();
        // 7 | type | id=login:username | sandra.benites@tci.net.pe

driver.findElement(By.id("login:username")).sendKeys("sandra.benites@tci.net
.pe");
        // 8 | type | id=login:password | Sandra201020

driver.findElement(By.id("login:password")).sendKeys("Sandra201020");
        // 9 | click | id=login:idLoginBuscar |
        driver.findElement(By.id("login:idLoginBuscar")).click();
        // 10 | click | id=iconid_frm_menu:j_id11 |
        driver.findElement(By.id("iconid_frm_menu:j_id11")).click();
        // 11 | click | name=frmPrincipal:txtOrdenCompra |

driver.findElement(By.name("frmPrincipal:txtOrdenCompra")).click();
        // 12 | type | id=frmPrincipal:txtOrdenCompra | $# =&

driver.findElement(By.id("frmPrincipal:txtOrdenCompra")).sendKeys("$# =&");
        // 13 | click | id=frmPrincipal:idBuscarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idBuscarBandejaEmisor")).click();
        // 14 | assertElementPresent | css=.rich-messages-label |
        {
            List<WebElement>                      elements
driver.findElements(By.cssSelector(".rich-messages-label"));
            assert(elements.size() > 0);
        }
        // 15 | assertText | css=.rich-messages-label | (*) Ingrese una
orden de compra válida.
        assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("(*) Ingrese una orden de compra válida.));
        // 16 | click | id=frmPrincipal:idLOimpiiarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idLOimpiiarBandejaEmisor")).click();
        // 17 | click | id=frmPrincipal:txtOrdenCompra |
        driver.findElement(By.id("frmPrincipal:txtOrdenCompra")).click();
        // 18 | type | id=frmPrincipal:txtOrdenCompra | 123-456

driver.findElement(By.id("frmPrincipal:txtOrdenCompra")).sendKeys("123-
456");
        // 19 | click | id=frmPrincipal:idBuscarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idBuscarBandejaEmisor")).click();
        // 20 | click | id=frmPrincipal:idLOimpiiarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idLOimpiiarBandejaEmisor")).click();
        // 21 | click | id=frmPrincipal:txtCorreoEmi |
        driver.findElement(By.id("frmPrincipal:txtCorreoEmi")).click();
        // 22 | click | id=frmPrincipal:idBuscarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idBuscarBandejaEmisor")).click();

```

```

// 23 | type | id=frmPrincipal:txtCorreoEmi | $# =&
driver.findElement(By.id("frmPrincipal:txtCorreoEmi")).sendKeys("#$ =&");
// 24 | click | id=frmPrincipal:idBuscarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idBuscarBandejaEmisor")).click();
// 25 | assertElementPresent | css=.rich-messages-label |
{
    List<WebElement>                elements                =
driver.findElements(By.cssSelector(".rich-messages-label"));
    assert(elements.size() > 0);
}
// 26 | assertText | css=.rich-messages-label | (*) Ingrese un
correo electrónico válido.
    assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("(*) Ingrese un correo electrónico válido.));
// 27 | click | id=frmPrincipal:idLOimpiarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idLOimpiarBandejaEmisor")).click();
// 28 | click | id=frmPrincipal:txtCorreoEmi |
    driver.findElement(By.id("frmPrincipal:txtCorreoEmi")).click();
// 29 | type | id=frmPrincipal:txtCorreoEmi |
CorreoEmisro.123_456@tci.net.pe

driver.findElement(By.id("frmPrincipal:txtCorreoEmi")).sendKeys("CorreoEmisr
o.123_456@tci.net.pe");
// 30 | click | id=frmPrincipal:idBuscarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idBuscarBandejaEmisor")).click();
// 31 | click | id=frmPrincipal:idLOimpiarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idLOimpiarBandejaEmisor")).click();
// 32 | click | id=frmPrincipal:txtNumeroInterno |

driver.findElement(By.id("frmPrincipal:txtNumeroInterno")).click();
// 33 | click | id=frmPrincipal:txtNumeroInterno |

driver.findElement(By.id("frmPrincipal:txtNumeroInterno")).click();
// 34 | type | id=frmPrincipal:txtNumeroInterno | $# =&

driver.findElement(By.id("frmPrincipal:txtNumeroInterno")).sendKeys("#$
=&");
// 35 | click | id=frmPrincipal:idBuscarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idBuscarBandejaEmisor")).click();
// 36 | assertElementPresent | css=.rich-messages-label |
{
    List<WebElement>                elements                =
driver.findElements(By.cssSelector(".rich-messages-label"));
    assert(elements.size() > 0);
}
// 37 | assertText | css=.rich-messages-label | (*) Ingrese un
número interno válido.
    assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("(*) Ingrese un número interno válido.));
// 38 | click | id=frmPrincipal:idLOimpiarBandejaEmisor |

```

```

driver.findElement(By.id("frmPrincipal:idLOimpiarBandejaEmisor")).click();
// 39 | click | css=.form-body:nth-child(2) > .row-table:nth-
child(5) |
    driver.findElement(By.cssSelector(".form-body:nth-child(2) > .row-
table:nth-child(5)")).click();
// 40 | click | id=frmPrincipal:cboDocumento |
driver.findElement(By.id("frmPrincipal:cboDocumento")).click();
// 41 | select | id=frmPrincipal:cboDocumento | label=Factura
{
    WebElement                                dropdown                                =
driver.findElement(By.id("frmPrincipal:cboDocumento"));
    dropdown.findElement(By.xpath("//option[.
'Factura']")).click();
}
// 42 | click | id=frmPrincipal:documento |
driver.findElement(By.id("frmPrincipal:documento")).click();
// 43 | type | id=frmPrincipal:documento | =&

driver.findElement(By.id("frmPrincipal:documento")).sendKeys("=&");
// 44 | doubleClick | id=frmPrincipal:idBuscarBandejaEmisor |
{
    WebElement                                element                                =
driver.findElement(By.id("frmPrincipal:idBuscarBandejaEmisor"));
    Actions builder = new Actions(driver);
    builder.doubleClick(element).perform();
}
// 45 | assertElementPresent | css=.rich-messages-label |
{
    List<WebElement>                            elements                            =
driver.findElements(By.cssSelector(".rich-messages-label"));
    assert(elements.size() > 0);
}
// 46 | assertText | css=.rich-messages-label | (*) Ingrese un
N°comprobante válido.
    assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("(*) Ingrese un N°comprobante válido.");
// 47 | click | id=frmPrincipal:idLOimpiarBandejaEmisor |

driver.findElement(By.id("frmPrincipal:idLOimpiarBandejaEmisor")).click();
// 48 | click | id=frmPrincipal:cboDocumento |
driver.findElement(By.id("frmPrincipal:cboDocumento")).click();
// 49 | select | id=frmPrincipal:cboDocumento | label=Factura
{
    WebElement                                dropdown                                =
driver.findElement(By.id("frmPrincipal:cboDocumento"));
    dropdown.findElement(By.xpath("//option[.
'Factura']")).click();
}
// 50 | click | id=frmPrincipal:documento |
driver.findElement(By.id("frmPrincipal:documento")).click();
// 51 | type | id=frmPrincipal:documento | T001-74

driver.findElement(By.id("frmPrincipal:documento")).sendKeys("T001-74");
// 52 | click | id=frmPrincipal:idBuscarBandejaEmisor |

```

```
driver.findElement(By.id("frmPrincipal:idBuscarBandejaEmisor")).click();  
    // 53 | click | id=frmPrincipal:idLOimpiarBandejaEmisor |  
driver.findElement(By.id("frmPrincipal:idLOimpiarBandejaEmisor")).click();  
    }  
}
```

Anexo 3

Código bandeja Receptor

```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class BandejaReceptorInputsTest {
    private WebDriver driver;
    private Map<String, Object> vars;
    JavascriptExecutor js;
    @Before
    public void setUp() {
        driver = new ChromeDriver();
        js = (JavascriptExecutor) driver;
        vars = new HashMap<String, Object>();
    }
    @After
    public void tearDown() {
        driver.quit();
    }
    @Test
    public void bandejaReceptorInputs() {
        // Test name: BandejaReceptorInputs
        // Step # | name | target | value
        // 1 | open | http://192.168.2.107/appefacturacion/pages/login.jsf#
|
driver.get("http://192.168.2.107/appefacturacion/pages/login.jsf#");
        // 2 | setWindowSize | 1616x886 |
driver.manage().window().setSize(new Dimension(1616, 886));
        // 3 | click | id=login:ruc |
driver.findElement(By.id("login:ruc")).click();
        // 4 | type | id=login:ruc | 20112811096
```



```

        driver.findElement(By.id("login:ruc")).sendKeys("20112811096");
        // 5 | click | id=login:username |
        driver.findElement(By.id("login:username")).click();
        // 6 | type | id=login:username | sandra.benites@tci.net.pe

driver.findElement(By.id("login:username")).sendKeys("sandra.benites@tci.net
.pe");
        // 7 | type | id=login:password | Sandra201020

driver.findElement(By.id("login:password")).sendKeys("Sandra201020");
        // 8 | click | id=login:idLoginBuscar |
        driver.findElement(By.id("login:idLoginBuscar")).click();
        // 9 | click | id=iconid_frm_menu:j_id14 |
        driver.findElement(By.id("iconid_frm_menu:j_id14")).click();
        // 10 | doubleClick | name=frmPrincipal:j_id91 |
        {
            WebElement                element
driver.findElement(By.name("frmPrincipal:j_id91"));
            Actions builder = new Actions(driver);
            builder.doubleClick(element).perform();
        }
        // 11 | type | name=frmPrincipal:j_id91 | ! + =
        driver.findElement(By.name("frmPrincipal:j_id91")).sendKeys("! +
=");
        // 12 | click | id=frmPrincipal:idBuscarBandejaRecepor |

driver.findElement(By.id("frmPrincipal:idBuscarBandejaRecepor")).click();
        // 13 | assertElementPresent | css=.rich-messages-label |
        {
            List<WebElement>                elements
driver.findElements(By.cssSelector(".rich-messages-label"));
            assert(elements.size() > 0);
        }
        // 14 | assertText | css=.rich-messages-label | (*) Ingrese una
orden de compra válida
            assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("(*) Ingrese una orden de compra válida"));
        // 15 | click | id=frmPrincipal:idLimpiarBandejaRecepor |

driver.findElement(By.id("frmPrincipal:idLimpiarBandejaRecepor")).click();
        // 16 | click | name=frmPrincipal:j_id91 |
        driver.findElement(By.name("frmPrincipal:j_id91")).click();
        // 17 | click | name=frmPrincipal:j_id91 |
        driver.findElement(By.name("frmPrincipal:j_id91")).click();
        // 18 | type | name=frmPrincipal:j_id91 | 11102034-__777

driver.findElement(By.name("frmPrincipal:j_id91")).sendKeys("11102034-
__777");
        // 19 | click | id=frmPrincipal:idBuscarBandejaRecepor |

driver.findElement(By.id("frmPrincipal:idBuscarBandejaRecepor")).click();
        // 20 | click | id=frmPrincipal:idLimpiarBandejaRecepor |

driver.findElement(By.id("frmPrincipal:idLimpiarBandejaRecepor")).click();
        // 21 | click | name=frmPrincipal:j_id110 |
        driver.findElement(By.name("frmPrincipal:j_id110")).click();

```

```

// 22 | type | name=frmPrincipal:j_id110 | ! + =
driver.findElement(By.name("frmPrincipal:j_id110")).sendKeys("! +
=");

// 23 | click | id=frmPrincipal:idBuscarBandejaRecepor |
driver.findElement(By.id("frmPrincipal:idBuscarBandejaRecepor")).click();
// 24 | assertElementPresent | css=.rich-messages-label |
{
    List<WebElement>                elements
driver.findElements(By.cssSelector(".rich-messages-label"));
    assert(elements.size() > 0);
}
// 25 | assertText | css=.rich-messages-label | (*) Ingrese correo
un correo válido
    assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("(*) Ingrese correo un correo válido"));
// 26 | click | id=frmPrincipal:idLimpiarBandejaRecepor |
driver.findElement(By.id("frmPrincipal:idLimpiarBandejaRecepor")).click();
// 27 | click | name=frmPrincipal:j_id110 |
driver.findElement(By.name("frmPrincipal:j_id110")).click();
// 28 | type | name=frmPrincipal:j_id110 | correo_12.34@tci.net.pe
driver.findElement(By.name("frmPrincipal:j_id110")).sendKeys("correo_12.34@t
ci.net.pe");
// 29 | click | id=frmPrincipal:idBuscarBandejaRecepor |
driver.findElement(By.id("frmPrincipal:idBuscarBandejaRecepor")).click();
// 30 | click | id=frmPrincipal:idLimpiarBandejaRecepor |
driver.findElement(By.id("frmPrincipal:idLimpiarBandejaRecepor")).click();
// 31 | click | id=frmPrincipal:cboDocumento |
driver.findElement(By.id("frmPrincipal:cboDocumento")).click();
// 32 | select | id=frmPrincipal:cboDocumento | label=Factura
{
    WebElement                dropdown
driver.findElement(By.id("frmPrincipal:cboDocumento"));
    dropdown.findElement(By.xpath("//option[.
'Factura']")).click();
}
// 33 | click | id=frmPrincipal:documento |
driver.findElement(By.id("frmPrincipal:documento")).click();
// 34 | type | id=frmPrincipal:documento | ! + =
driver.findElement(By.id("frmPrincipal:documento")).sendKeys("! +
=");

// 35 | doubleClick | id=frmPrincipal:idBuscarBandejaRecepor |
{
    WebElement                element
driver.findElement(By.id("frmPrincipal:idBuscarBandejaRecepor"));
    Actions builder = new Actions(driver);
    builder.doubleClick(element).perform();
}
// 36 | assertElementPresent | css=.rich-messages-label |
{
    List<WebElement>                elements
driver.findElements(By.cssSelector(".rich-messages-label"));

```

```

        assert(elements.size() > 0);
    }
    // 37 | assertText | css=.rich-messages-label | (*) Ingrese N°
comprobante válido
        assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("(*) Ingrese N° comprobante válido"));
    // 38 | click | id=frmPrincipal:idLimpiarBandejaRecepor |

driver.findElement(By.id("frmPrincipal:idLimpiarBandejaRecepor")).click();
    // 39 | click | id=frmPrincipal:cboDocumento |
        driver.findElement(By.id("frmPrincipal:cboDocumento")).click();
    // 40 | select | id=frmPrincipal:cboDocumento | label=Factura
    {
        WebElement                                dropdown                                =
driver.findElement(By.id("frmPrincipal:cboDocumento"));
        dropdown.findElement(By.xpath("//option[.
'Factura']")).click();                                =
    }
    // 41 | click | id=frmPrincipal:documento |
        driver.findElement(By.id("frmPrincipal:documento")).click();
    // 42 | type | id=frmPrincipal:documento | F00-123
        driver.findElement(By.id("frmPrincipal:documento")).sendKeys("F00-
123");
    // 43 | click | id=frmPrincipal:idBuscarBandejaRecepor |

driver.findElement(By.id("frmPrincipal:idBuscarBandejaRecepor")).click();
    // 44 | click | id=frmPrincipal:idLimpiarBandejaRecepor |

driver.findElement(By.id("frmPrincipal:idLimpiarBandejaRecepor")).click();
    }
}

```

Anexo 4

Código Bandeja mercadería

```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class RecepciondeMercaderiaInputsTest {
    private WebDriver driver;
    private Map<String, Object> vars;
    JavascriptExecutor js;
    @Before
    public void setUp() {
        driver = new ChromeDriver();
        js = (JavascriptExecutor) driver;
        vars = new HashMap<String, Object>();
    }
    @After
    public void tearDown() {
        driver.quit();
    }
    @Test
    public void recepciondeMercaderiaInputs() {
        // Test name: RecepciondeMercaderiaInputs
        // Step # | name | target | value
        // 1 | open | http://192.168.2.107/appefacturacion/pages/login.jsf#
|
driver.get("http://192.168.2.107/appefacturacion/pages/login.jsf#");
        // 2 | setWindowSize | 1616x886 |
        driver.manage().window().setSize(new Dimension(1616, 886));
        // 3 | click | id=login:ruc |
        driver.findElement(By.id("login:ruc")).click();
        // 4 | type | id=login:ruc | 20112811096
```

```

        driver.findElement(By.id("login:ruc")).sendKeys("20112811096");
        // 5 | click | id=login:username |
        driver.findElement(By.id("login:username")).click();
        // 6 | type | id=login:username | sandra.benites@tci.net.pe

driver.findElement(By.id("login:username")).sendKeys("sandra.benites@tci.net
.pe");
        // 7 | type | id=login:password | Sandra201020

driver.findElement(By.id("login:password")).sendKeys("Sandra201020");
        // 8 | click | id=login:idLoginBuscar |
        driver.findElement(By.id("login:idLoginBuscar")).click();
        // 9 | click | id=iconid_frm_menu:j_id15 |
        driver.findElement(By.id("iconid_frm_menu:j_id15")).click();
        // 10 | click | id=frmPrincipal:txtNroManifiesto |

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).click();
        // 11 | type | id=frmPrincipal:txtNroManifiesto | =&.

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).sendKeys("=&.");
        // 12 | click | id=frmPrincipal:idBuscarGuiaRemision |

driver.findElement(By.id("frmPrincipal:idBuscarGuiaRemision")).click();
        // 13 | assertElementPresent | css=.rich-messages-label |
        {
            List<WebElement>                elements                =
driver.findElements(By.cssSelector(".rich-messages-label"));
            assert(elements.size() > 0);
        }
        // 14 | assertText | css=.rich-messages-label | Ingrese un N° de
comprobante válido.
        assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("Ingrese un N° de comprobante válido."));
        // 15 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
        // 16 | click | id=frmPrincipal:txtNroManifiesto |

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).click();
        // 17 | type | id=frmPrincipal:txtNroManifiesto | F00-1

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).sendKeys("F00-
1");
        // 18 | click | id=frmPrincipal:idBuscarGuiaRemision |

driver.findElement(By.id("frmPrincipal:idBuscarGuiaRemision")).click();
        // 19 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
        // 20 | click | id=frmPrincipal:txtNroTransporte |

driver.findElement(By.id("frmPrincipal:txtNroTransporte")).click();
        // 21 | type | id=frmPrincipal:txtNroTransporte | =&.

driver.findElement(By.id("frmPrincipal:txtNroTransporte")).sendKeys("=&.");
        // 22 | click | id=frmPrincipal:idBuscarGuiaRemision |

```

```

driver.findElement(By.id("frmPrincipal:idBuscarGuiaRemision")).click();
    // 23 | click | css=.rich-messages-label |
    driver.findElement(By.cssSelector(".rich-messages-
label")).click();
    // 24 | click | css=.rich-messages-label | Ingrese un número de
transporte válido
    driver.findElement(By.cssSelector(".rich-messages-
label")).click();
    // 25 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
    // 26 | click | id=frmPrincipal:txtNroTransporte |

driver.findElement(By.id("frmPrincipal:txtNroTransporte")).click();
    // 27 | click | id=frmPrincipal:txtNroTransporte |

driver.findElement(By.id("frmPrincipal:txtNroTransporte")).click();
    // 28 | type | id=frmPrincipal:txtNroTransporte | nul2ñMm-óÜ

driver.findElement(By.id("frmPrincipal:txtNroTransporte")).sendKeys("nul2ñMm
-óÜ");
    // 29 | click | id=frmPrincipal:idBuscarGuiaRemision |

driver.findElement(By.id("frmPrincipal:idBuscarGuiaRemision")).click();
    // 30 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
    }
}

```

Anexo 5

Código Bandeja Despacho

```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class BandejaDespachoInputsTest {
    private WebDriver driver;
    private Map<String, Object> vars;
    JavascriptExecutor js;
    @Before
    public void setUp() {
        driver = new ChromeDriver();
        js = (JavascriptExecutor) driver;
        vars = new HashMap<String, Object>();
    }
    @After
    public void tearDown() {
        driver.quit();
    }
    @Test
    public void bandejaDespachoInputs() {
        // Test name: BandejaDespachoInputs
        // Step # | name | target | value
        // 1 | open | http://192.168.2.107/appefacturacion/pages/login.jsf#
|
driver.get("http://192.168.2.107/appefacturacion/pages/login.jsf#");
        // 2 | setWindowSize | 1616x886 |
        driver.manage().window().setSize(new Dimension(1616, 886));
        // 3 | click | id=login:ruc |
```

```

        driver.findElement(By.id("login:ruc")).click();
        // 4 | type | id=login:ruc | 20112811096
        driver.findElement(By.id("login:ruc")).sendKeys("20112811096");
        // 5 | click | id=login:username |
        driver.findElement(By.id("login:username")).click();
        // 6 | type | id=login:username | sandra.benites@tci.net.pe

driver.findElement(By.id("login:username")).sendKeys("sandra.benites@tci.net
.pe");
        // 7 | type | id=login:password | Sandra201020

driver.findElement(By.id("login:password")).sendKeys("Sandra201020");
        // 8 | click | id=login:idLoginBuscar |
        driver.findElement(By.id("login:idLoginBuscar")).click();
        // 9 | click | id=iconid_frm_menu:j_id12 |
        driver.findElement(By.id("iconid_frm_menu:j_id12")).click();
        // 10 | click | id=frmPrincipal:txtNroManifiesto |

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).click();
        // 11 | type | id=frmPrincipal:txtNroManifiesto | =&.

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).sendKeys("=&.");
        // 12 | click | id=frmPrincipal:idBuscarGuiaRemision |

driver.findElement(By.id("frmPrincipal:idBuscarGuiaRemision")).click();
        // 13 | assertElementPresent | css=.rich-messages-label |
        {
            List<WebElement>                elements                =
driver.findElements(By.cssSelector(".rich-messages-label"));
            assert(elements.size() > 0);
        }
        // 14 | assertText | css=.rich-messages-label | Ingrese un N°
comprobante válido
            assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("Ingrese un N° comprobante válido"));
        // 15 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
        // 16 | click | id=frmPrincipal:txtNroManifiesto |

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).click();
        // 17 | type | id=frmPrincipal:txtNroManifiesto | F00-1

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).sendKeys("F00-
1");
        // 18 | click | id=frmPrincipal:idBuscarGuiaRemision |

driver.findElement(By.id("frmPrincipal:idBuscarGuiaRemision")).click();
        // 19 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
        // 20 | click | id=frmPrincipal:txtNroTransporte |

driver.findElement(By.id("frmPrincipal:txtNroTransporte")).click();
        // 21 | type | id=frmPrincipal:txtNroTransporte | =&.

```



```

driver.findElement(By.id("frmPrincipal:txtNroTransporte")).sendKeys("="&.");
// 22 | click | id=frmPrincipal:idBuscarGuiaRemision |

driver.findElement(By.id("frmPrincipal:idBuscarGuiaRemision")).click();
// 23 | click | css=.rich-messages-label |
driver.findElement(By.cssSelector(".rich-messages-
label")).click();
// 24 | click | css=.rich-messages-label | Ingrese un número de
transporte válido
driver.findElement(By.cssSelector(".rich-messages-
label")).click();
// 25 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
// 26 | click | id=frmPrincipal:txtNroTransporte |

driver.findElement(By.id("frmPrincipal:txtNroTransporte")).click();
// 27 | click | id=frmPrincipal:txtNroTransporte |

driver.findElement(By.id("frmPrincipal:txtNroTransporte")).click();
// 28 | type | id=frmPrincipal:txtNroTransporte | nul2ñMm-óÜ

driver.findElement(By.id("frmPrincipal:txtNroTransporte")).sendKeys("nul2ñMm-óÜ");
// 29 | click | id=frmPrincipal:idBuscarGuiaRemision |

driver.findElement(By.id("frmPrincipal:idBuscarGuiaRemision")).click();
// 30 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
}
}

```

Anexo 6

Archivo POM

```
<project                                xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>SeleniumAutomation</groupId>
  <artifactId>AplicativoEportal</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <dependencies>
    <dependency>
      <groupId>org.seleniumhq.selenium</groupId>
      <artifactId>selenium-java</artifactId>
      <version>4.0.0</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.13.2</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

Anexo 7

Código Bandeja Transacciones

```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class TransaccionesInputsTest {
    private WebDriver driver;
    private Map<String, Object> vars;
    JavascriptExecutor js;
    @Before
    public void setUp() {
        driver = new ChromeDriver();
        js = (JavascriptExecutor) driver;
        vars = new HashMap<String, Object>();
    }
    @After
    public void tearDown() {
        driver.quit();
    }
    @Test
    public void transaccionesInputs() {
        // Test name: TransaccionesInputs
        // Step # | name | target | value
        // 1 | open | http://192.168.2.107/appefacturacion/pages/login.jsf#
|
driver.get("http://192.168.2.107/appefacturacion/pages/login.jsf#");
        // 2 | setWindowSize | 1616x886 |
        driver.manage().window().setSize(new Dimension(1616, 886));
        // 3 | click | id=login:ruc |
```

```

        driver.findElement(By.id("login:ruc")).click();
        // 4 | type | id=login:ruc | 20112811096
        driver.findElement(By.id("login:ruc")).sendKeys("20112811096");
        // 5 | click | id=login:username |
        driver.findElement(By.id("login:username")).click();
        // 6 | type | id=login:username | sandra.benites@tci.net.pe

driver.findElement(By.id("login:username")).sendKeys("sandra.benites@tci.net
.pe");
        // 7 | type | id=login:password | Sandra201020

driver.findElement(By.id("login:password")).sendKeys("Sandra201020");
        // 8 | click | id=login:idLoginBuscar |
        driver.findElement(By.id("login:idLoginBuscar")).click();
        // 9 | click | id=iconid_frm_menu:j_id9 |
        driver.findElement(By.id("iconid_frm_menu:j_id9")).click();
        // 10 | click | id=frmPrincipal:txtIdTransaccion |

driver.findElement(By.id("frmPrincipal:txtIdTransaccion")).click();
        // 11 | type | id=frmPrincipal:txtIdTransaccion | *+ '=

driver.findElement(By.id("frmPrincipal:txtIdTransaccion")).sendKeys("*+
'=");
        // 12 | click | id=frmPrincipal:idBuscarTransaccion |

driver.findElement(By.id("frmPrincipal:idBuscarTransaccion")).click();
        // 13 | assertElementPresent | css=.rich-messages-label |
        {
            List<WebElement>                elements                =
driver.findElements(By.cssSelector(".rich-messages-label"));
            assert(elements.size() > 0);
        }
        // 14 | assertText | css=.rich-messages-label | Ingrese un ID de
transacción válido
        assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("Ingrese un ID de transacción válido"));
        // 15 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
        // 16 | click | id=frmPrincipal:txtIdTransaccion |

driver.findElement(By.id("frmPrincipal:txtIdTransaccion")).click();
        // 17 | type | id=frmPrincipal:txtIdTransaccion | 1234,345

driver.findElement(By.id("frmPrincipal:txtIdTransaccion")).sendKeys("1234,34
5");
        // 18 | click | id=frmPrincipal:idBuscarTransaccion |

driver.findElement(By.id("frmPrincipal:idBuscarTransaccion")).click();
        // 19 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
        // 20 | click | id=frmPrincipal:cbNroTransaccion |

driver.findElement(By.id("frmPrincipal:cbNroTransaccion")).click();
        // 21 | select | id=frmPrincipal:cbNroTransaccion | label=Factura

```

```

        {
            WebElement                                dropdown                                =
driver.findElement(By.id("frmPrincipal:cbNroTransaccion"));
            dropdown.findElement(By.xpath("//option[.
'Factura']")).click();
        }
        // 22 | click | id=frmPrincipal:txtNroManifiesto |

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).click();
        // 23 | type | id=frmPrincipal:txtNroManifiesto | *+ '=

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).sendKeys("*+
'=");
        // 24 | click | id=frmPrincipal:idBuscarTransaccion |

driver.findElement(By.id("frmPrincipal:idBuscarTransaccion")).click();
        // 25 | assertElementPresent | css=.rich-messages-label |
        {
            List<WebElement>                            elements                            =
driver.findElements(By.cssSelector(".rich-messages-label"));
            assert(elements.size() > 0);
        }
        // 26 | assertText | css=.rich-messages-label | Ingrese un N°
comprobante válido
            assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("Ingrese un N° comprobante válido"));
        // 27 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
        // 28 | click | id=frmPrincipal:cbNroTransaccion |

driver.findElement(By.id("frmPrincipal:cbNroTransaccion")).click();
        // 29 | select | id=frmPrincipal:cbNroTransaccion | label=Factura
        {
            WebElement                                dropdown                                =
driver.findElement(By.id("frmPrincipal:cbNroTransaccion"));
            dropdown.findElement(By.xpath("//option[.
'Factura']")).click();
        }
        // 30 | click | id=frmPrincipal:txtNroManifiesto |

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).click();
        // 31 | type | id=frmPrincipal:txtNroManifiesto | F00-223

driver.findElement(By.id("frmPrincipal:txtNroManifiesto")).sendKeys("F00-
223");
        // 32 | click | id=frmPrincipal:idBuscarTransaccion |

driver.findElement(By.id("frmPrincipal:idBuscarTransaccion")).click();
        // 33 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
        // 34 | click | id=frmPrincipal:txtNombreComprobante |

driver.findElement(By.id("frmPrincipal:txtNombreComprobante")).click();
        // 35 | type | id=frmPrincipal:txtNombreComprobante | *+ '=

```

```

driver.findElement(By.id("frmPrincipal:txtNombreComprobante")).sendKeys("*+
'=");
    // 36 | click | id=frmPrincipal:idBuscarTransaccion |

driver.findElement(By.id("frmPrincipal:idBuscarTransaccion")).click();
    // 37 | assertElementPresent | css=.rich-messages-label |
    {
        List<WebElement>                elements                =
driver.findElements(By.cssSelector(".rich-messages-label"));
        assert(elements.size() > 0);
    }
    // 38 | assertText | css=.rich-messages-label | Ingrese un nombre
comprobante válido
        assertThat(driver.findElement(By.cssSelector(".rich-messages-
label")).getText(), is("Ingrese un nombre comprobante válido"));
    // 39 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();
    // 40 | click | id=frmPrincipal:txtNombreComprobante |

driver.findElement(By.id("frmPrincipal:txtNombreComprobante")).click();
    // 41 | type | id=frmPrincipal:txtNombreComprobante | nu21ÜÑ'MNn-
r_G.123á

driver.findElement(By.id("frmPrincipal:txtNombreComprobante")).sendKeys("nu2
1ÜÑ\MNn-r_G.123á");
    // 42 | click | id=frmPrincipal:idBuscarTransaccion |

driver.findElement(By.id("frmPrincipal:idBuscarTransaccion")).click();
    // 43 | click | id=frmPrincipal:idLimpiarTransaccion |

driver.findElement(By.id("frmPrincipal:idLimpiarTransaccion")).click();

```