Universidad Continental

FACULTAD DE INGENIERÍA

Escuela Académico Profesional de Ingeniería de Sistemas e Informática

Trabajo de Suficiencia Profesional

Competencias desarrolladas en el proceso de optimización de pruebas de software en la empresa Globant

Pamela Felisa Espinoza Calderon

Para optar el Título Profesional de
Ingeniero de Sistemas e Informática

Lima, 2025

INFORME DE CONFORMIDAD DE ORIGINALIDAD DE TRABAJO DE INVESTIGACIÓN

A	:	Decano de la Facultad de Ingeniería
DE	:	CESAR HERNAN PATRICIO PERALTA Asesor de trabajo de investigación
ASUNTO	:	Remito resultado de evaluación de originalidad de trabajo de investigación
FECHA	:	22 de Marzo de 2025

Con sumo agrado me dirijo a vuestro despacho para informar que, en mi condición de asesor del trabajo de investigación:

Título:
Competencias Desarrolladas en el Proceso de Optimización de Pruebas de Software en la Empresa GLOBANT

Autor:
PAMELA FELISA ESPINOZA CALDERON – EAP. Ingeniería de Sistemas e Informática

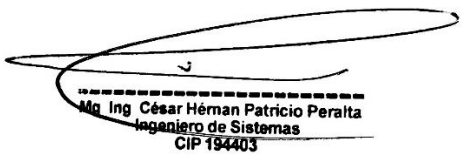
Se procedió con la carga del documento a la plataforma "Turnitin" y se realizó la verificación completa de las coincidencias resaltadas por el software dando por resultado 15 % de similitud sin encontrarse hallazgos relacionados a plagio. Se utilizaron los siguientes filtros:

- Filtro de exclusión de bibliografía SI [X] NO [ ]
- Filtro de exclusión de grupos de palabras menores SI [X] NO [ ]
  N° de palabras excluidas **(en caso de elegir "SI")**: 20
- Exclusión de fuente por trabajo anterior del mismo estudiante SI [ ] NO [X]

En consecuencia, se determina que el trabajo de investigación constituye un documento original al presentar similitud de otros autores (citas) por debajo del porcentaje establecido por la Universidad Continental.

Recae toda responsabilidad del contenido del trabajo de investigación sobre el autor y asesor, en concordancia a los principios expresados en el Reglamento del Registro Nacional de Trabajos conducentes a Grados y Títulos – RENATI y en la normativa de la Universidad Continental.

Atentamente,



Asesor de trabajo de investigación

AGRADECIMIENTOS

A mi familia, por la confianza depositada en mí y por creer siempre que mis sueños podían hacerse realidad.

A aquellas personas que ahora se encuentran en el cielo, cuyo apoyo siento siempre presente.

Y a la Universidad Continental, por haberme guiado y acompañado a lo largo de este camino lleno de aprendizajes.

DEDICATORIA

Dedico este trabajo a Dios, por haberme dado salud y fortaleza para alcanzar mis objetivos académicos.

A mi mamá, mi papá, mi madrina y mi hermana, quienes me brindan el amor y la motivación necesarios para seguir adelante y superar cada desafío.

ÍNDICE

CAPÍTULO I. DESCRIPCIÓN DE LA EMPRESA ……. 8

1.1 Datos Generales de la Empresa …… 8

1.2 Actividades Principales de la Empresa ……. 9

1.3 Reseña Histórica de la Empresa ……. 10

1.4 Organigrama de la Empresa …… 12

1.5 Visión y Misión …….. 12

1.6 Bases Legales y Documentos Administrativos ……. 13

1.7 Descripción del Área Donde se Realizan Actividades Profesionales …… 13

1.8 Descripción del Cargo y Responsabilidades del Bachiller en la Empresa …… 15

CAPÍTULO II. PLANTEAMIENTO DEL ESTUDIO ……... 16

2.1 Antecedentes y Diagnóstico Situacional ……... 16

2.2 Identificación de Oportunidad o Necesidad en el Área de Actividad Profesional ……. 18

2.3 Objetivos de la Actividad Profesional ……... 19

2.4 Justificación de la Actividad Profesional ……... 20

2.5 Resultados Esperados ……... 20

CAPÍTULO III. MARCO TEÓRICO ……... 24

3.1 Integración Continua / Despliegue Continuo ……... 24

3.2 DevSecOps …… 26

3.3 DevOps …… 27

3.4 SDLC (Ciclo de Vida del Desarrollo de Software) …… 31

3.5 ISTQB …… 32

3.6 Testing …… 33

3.7 Jira 34

3.8 Release 36

3.9 UAT 38

3.10 Plugins 40

CAPÍTULO IV. IMPLEMENTACIÓN 42

4.1 Descripción de Actividades Técnicas Profesionales 42

4.1.1 Enfoque de las Actividades Profesionales 42

4.1.2 Alcance de las Actividades Profesionales 44

4.1.3 Entregables de las Actividades Profesionales 45

4.2 Aspectos Técnicos de la Actividad Profesional 49

4.2.1 Metodologías 49

4.2.2 Técnicas 53

4.2.3 Instrumentos 54

4.3 Ejecución de las Actividades Profesionales 56

4.3.1 Cronograma de Actividades Realizadas 56

CAPÍTULO V. RESULTADOS 57

5.1 Resultados Finales de las Actividades Realizadas 57

5.1.1 Metodología SCRUM 57

5.1.2 Definición de Política de Versionado 58

5.1.3 Resultados del Desarrollo de Funcionalidades 58

5.2 Logros Alcanzados 60

5.2.1 Mejora en el Proceso de Desarrollo, Pruebas y Entrega del Producto 60

5.2.2 Mejora en el Proceso de Versionado 60

5.2.3 Mejora en la Disponibilidad de Información Analítica …… 60

5.3 Dificultades Encontradas …… 61

5.3.1 Ausencia de Versionamiento en la Documentación …… 61

5.3.2 Falta de Equipo de Calidad …… 62

5.3.3 Ausencia de Roles en el Equipo …… 63

5.3.4 Falta de Ceremonias …… 63

5.3.5 Ausencia de Herramientas para el Equipo de Calidad …… 64

5.4 Planteamiento de Mejoras …… 64

5.4.1 Metodologías Propuestas …… 65

5.4.2 Descripción de la Implementación …… 65

5.5 Análisis …… 66

5.6 Aporte del Bachiller en la Empresa y/o Institución …… 68

REFERENCIAS BIBLIOGRÁFICAS …… 70

RESUMEN

Globant Perú es parte de Globant, una empresa global de servicios tecnológicos y soluciones digitales con un enfoque en la transformación digital de empresas. Su esencia radica en combinar tecnología de vanguardia con creatividad para ayudar a las empresas a innovar y adaptarse a los cambios del mercado.

Globant Perú, como filial local, se especializa en ofrecer soluciones personalizadas en áreas como desarrollo de software, inteligencia artificial, automatización y diseño de experiencias digitales. La empresa tiene un fuerte enfoque en la agilidad y la innovación, trabajando con grandes empresas en diversos sectores como finanzas, retail, telecomunicaciones y más.

Además, Globant se destaca por su compromiso con el talento local, promoviendo un ambiente de trabajo inclusivo y colaborativo, y apostando por el crecimiento profesional de sus empleados. Su cultura organizacional resalta la creatividad, el trabajo en equipo y el desarrollo constante de habilidades tecnológicas, lo que les permite mantenerse a la vanguardia en la industria de la tecnología.

El core de Globant Perú se basa en la innovación tecnológica, la transformación digital, el talento humano y la creación de soluciones disruptivas que impulsen el éxito de sus clientes en un mundo cada vez más digitalizado.

Palabras claves: Desarrollo, software, ambiente de prueba, testing, release, gestión, integración continua, despliegue continuo, prueba de aceptación del usuario, ciclo de vida del desarrollo de software, metodología, ágil.

CAPÍTULO I

DESCRIPCIÓN DE LA EMPRESA

1.1 Datos Generales de la Empresa

Razón Social

Globant Perú S.A.C.

RUC: 20514492825

Dirección Fiscal

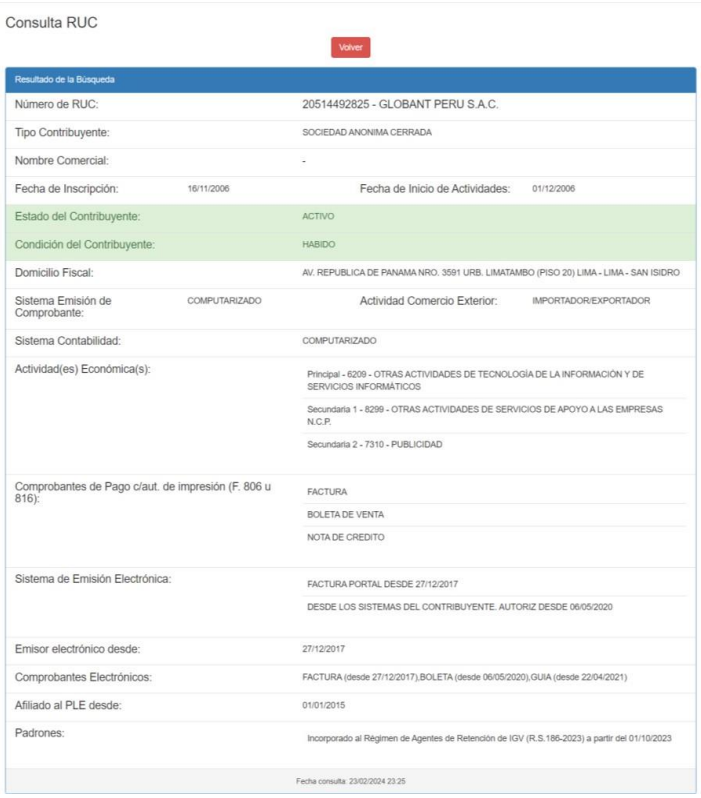
Av. República de Panamá Nro. 3591, Urb. Limatambo (Piso 20), San Isidro, Lima.

Registro Único de Contribuyente

A continuación, se consigna la Ficha RUC de la empresa.

Figura 1

Ficha RUC



Consulta RUC

Volver

Resultado de la Búsqueda			
Número de RUC:	20514492825 - GLOBANT PERU S.A.C.		
Tipo Contribuyente:	SOCIEDAD ANONIMA CERRADA		
Nombre Comercial:	-		
Fecha de Inscripción:	16/11/2006	Fecha de Inicio de Actividades:	01/12/2006
Estado del Contribuyente:	ACTIVO		
Condición del Contribuyente:	HABIDO		
Domicilio Fiscal:	AV. REPUBLICA DE PANAMA NRO. 3591 URB. LIMATAMBO (PISO 20) LIMA - LIMA - SAN ISIDRO		
Sistema Emisión de Comprobante:	COMPUTARIZADO	Actividad Comercio Exterior:	IMPORTADOR/EXPORTADOR
Sistema Contabilidad:	COMPUTARIZADO		
Actividad(es) Económica(s):	Principal - 6209 - OTRAS ACTIVIDADES DE TECNOLOGÍA DE LA INFORMACIÓN Y DE SERVICIOS INFORMÁTICOS Secundaria 1 - 8299 - OTRAS ACTIVIDADES DE SERVICIOS DE APOYO A LAS EMPRESAS N.C.P. Secundaria 2 - 7310 - PUBLICIDAD		
Comprobantes de Pago c/aut. de impresión (F. 806 u 816):	FACTURA BOLETA DE VENTA NOTA DE CREDITO		
Sistema de Emisión Electrónica:	FACTURA PORTAL DESDE 27/12/2017 DESDE LOS SISTEMAS DEL CONTRIBUYENTE. AUTORIZ DESDE 06/05/2020		
Emisor electrónico desde:	27/12/2017		
Comprobantes Electrónicos:	FACTURA (desde 27/12/2017),BOLETA (desde 06/05/2020),GUIA (desde 22/04/2021)		
Afiliado al PLE desde:	01/01/2015		
Padrones:	Incorporado al Régimen de Agentes de Retención de IGV (R.S.186-2023) a partir del 01/10/2023		

Fecha consulta: 23/02/2024 23:25

1.2 Actividades Principales de la Empresa

Globant S.A. es una empresa internacional cuya oficina central se encuentra en Buenos Aires. Se especializa en el sector de Tecnología de la Información (TI), ofreciendo una amplia cartera de soluciones de negocio, que incluyen:

- Desarrollo de software
- Infraestructura tecnológica
- Servicios creativos
- Aplicaciones móviles
- Administración de contenido
- Comercio electrónico

La empresa cuenta con una sólida presencia a nivel global, con un amplio equipo de empleados distribuidos en 25 países, liderando la transformación digital y cognitiva. Globant clasifica sus actividades en dos grandes grupos, según las necesidades de sus clientes y del mercado:

Estudios

Globant se centra en brindar valor a la tecnología, la disciplina y diversas industrias, impulsando la innovación y la transformación digital a través de sus principales áreas de estudio:

- Reinvent
- Create
- Digital
- Enterprise
- Productos y plataformas

Comprometida con el crecimiento y la evolución tecnológica, Globant desarrolla productos y plataformas diseñadas para maximizar el potencial de sus clientes, ayudándolos a alcanzar su mejor versión:

- Augoor
- BeHealthy
- FluentLab
- GeneXus
- MagnifAI
- Navigate
- StarMeUp
- WaaSabi
- Walmeric

1.3 Reseña Histórica de la Empresa

Hace 21 años, en un bar de Argentina, cuatro amigos compartieron un sueño en común: crear una empresa digital que ofreciera las mejores experiencias, brindando soluciones tecnológicas y desarrollo de software a otras organizaciones, al mismo tiempo que impulsara el crecimiento profesional de quienes quisieran formar parte del mundo de Tecnología de la Información (TI). Fue así como, en 2003, Martín Migoya, Guibert Englebienne, Martín Umarán y Néstor Nocetti fundaron Globant.

Gracias a su rápido crecimiento, la compañía decidió expandirse a más de 10 países, consolidándose como un referente en el sector. A lo largo de los años, ha llevado a cabo diversas adquisiciones estratégicas:

- 2008: Adquisición de Accendra.
- 2011: Adquisición de Nextive, especializada en aplicaciones para dispositivos móviles.
- 2013: Adquisición del 86,25% del grupo Huddle.
- 2014: Ingreso a la Bolsa de Nueva York (NYSE).
- 2015: Adquisición de la empresa india Clarice Technologies.
- 2016: Adquisición de WAE.
- 2017: Adquisición de PointSource y Ratio.
- 2018: Adquisición de Small Footprint.
- 2019: Adquisición de Avanxo (Colombia) y Belatrix (Argentina).
- 2020: Adquisición de BiLive.
- 2021: Adquisición de la empresa inglesa CloudShift.
- 2022: Adquisición de la empresa uruguaya GeneXus.
- 2023: Adquisición de la empresa estadounidense ExperienceIT.

Portafolio de Clientes

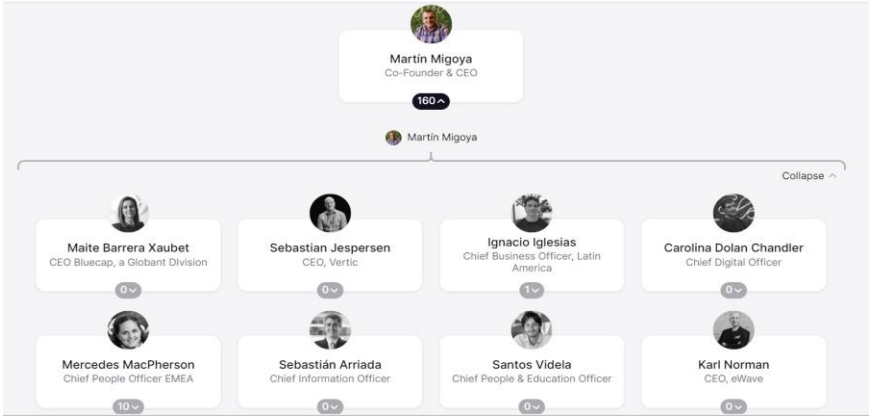
Globant ha trabajado con una amplia variedad de empresas a nivel mundial, destacándose marcas reconocidas de distintos sectores: 3g Smart Group, AEP Energy, Amadeus, American Express, Aon, Bally, BBVA, BR Petrobras, Boehringer Ingelheim, Cars.com, Cisco, Cloudera, Coca-Cola, Dell, EA, Embraer, F.biz, FOX, Gree, GroupM, Hortonworks, Interbank, Iseatz, LATAM Airlines, LinkedIn, Mach, Mercado Libre, MoneyGram, National Geographic, Nespresso, Osram, Prisma, Puma, Sabre, Santander, Travelocity, ViajaNet, Wobi y YPF.

1.4 Organigrama de la Empresa

Se estructura de la siguiente forma:

Figura 2

Organigrama de Globant Perú



1.5 Visión y Misión

Visión

Ser la empresa líder en desarrollo de software, ofreciendo soluciones tecnológicas de vanguardia que optimicen los servicios y eleven la experiencia del usuario a un nivel superior. Nuestro objetivo es consolidar un posicionamiento estratégico en el mundo de la Tecnología de la Información (TI).

Misión

Transformar el mundo y las realidades de las empresas dentro del sector TI, impactando positivamente en el talento global y en el desarrollo de las personas a través de la innovación y la tecnología.

1.6 Bases Legales y Documentos Administrativos

El presente Código de Ética establece cómo se implementan los valores de Globant en el trabajo diario, proporcionando un conocimiento compartido no solo sobre lo que hacemos, sino también sobre cómo lo hacemos.

Los principios fundamentales que rigen nuestra conducta se estructuran en los siguientes compromisos:

- **Compromiso personal**: Servir con honestidad y rectitud en todas las actividades dentro de Globant.
- **Compromiso con los demás**: Promover un trato respetuoso, fomentando la empatía y la colaboración en el ambiente de trabajo.
- **Compromiso con Globant**: Lograr el éxito empresarial a través de prácticas comerciales éticas, justas y legales.
- **Compromiso de Globant con sus empleados**: Asegurar que todos los Globers sean tratados con respeto y justicia, garantizando un entorno laboral seguro y libre de acoso o discriminación.
- **Compromiso con nuestros socios comerciales**: Conducir las transacciones de manera honesta, responsable y transparente, para construir relaciones de confianza con nuestros socios.

1.7 Descripción del Área Donde se Realizan Actividades Profesionales

El área se enfoca en roles relacionados con Quality Control (Control de Calidad). Las actividades desarrolladas están directamente vinculadas con la garantía de calidad en el desarrollo de software, asegurando que los productos cumplan con los estándares esperados.

Visión

Garantizar la calidad en el desarrollo de software para cumplir con los resultados esperados, satisfaciendo las necesidades tanto del usuario como de los stakeholders. Se busca que los clientes y las áreas involucradas reconozcan el valor de los distintos tipos de pruebas, con el objetivo de entregar un producto de la más alta calidad.

Promesa

Seguir innovando en metodologías de prueba y nuevas funcionalidades para garantizar el cumplimiento de los objetivos establecidos y la entrega de desarrollos de calidad.

Objetivos

- Identificar y examinar las necesidades del usuario.
- Evaluar y mejorar el funcionamiento de los sistemas existentes.
- Realizar un análisis de brechas y proponer soluciones adecuadas.
- Diseñar objetivos de cambio basados en los requerimientos identificados.
- Facilitar la comunicación de la configuración.
- Redactar los cambios realizados y crear manuales de capacitación.
- Trabajar en conjunto con otros equipos para garantizar una implementación eficiente de los cambios.
- Ejecutar simulaciones y pruebas para evaluar el desempeño y la calidad del software.
- Elaborar manuales para el uso del usuario con una descripción clara y comprensible.
- Analizar, instalar y resolver enigmas en los sistemas de software.
- Verificar que el producto final funcione correctamente antes de su entrega.

1.8 Descripción del Cargo y Responsabilidades del Bachiller en la Empresa

El Analista Funcional de Calidad es un profesional encargado de analizar los procesos de negocio con el objetivo de identificar y comprender las necesidades de información del cliente. Actúa como nexo entre el usuario o cliente y el equipo de desarrollo de software, asegurando que los requerimientos sean correctamente interpretados y plasmados en la aplicación. A partir de su análisis, define las funcionalidades y especificaciones que deberá cumplir el software para satisfacer eficientemente las necesidades del cliente. A continuación, se detallan algunas de sus principales funciones:

- Elaborar casos de prueba para validar el correcto desarrollo del software.
- Preparar el ambiente en el que se realizaran las pruebas para la ejecución de los test.
- Ejecutar casos de prueba de forma manual para validar el comportamiento del sistema.
- Registrar y documentar los defectos encontrados durante la ejecución de pruebas.
- Realizar retesting de los casos de prueba para verificar la corrección de errores.
- Mantener un consolidado de los casos de prueba ejecutados.
- Elaborar el reporte final de pruebas con los hallazgos y resultados obtenidos.
- Enviar un informe de conformidad con el resumen de las pruebas realizadas.

CAPÍTULO II

PLANTEAMIENTO DEL ESTUDIO

2.1 Antecedentes y Diagnóstico Situacional

Globant nació en 2003 con el propósito de brindar servicios tecnológicos especializados en el desarrollo de software, arquitectura informática, soluciones tecnológicas, aseguramiento de calidad de productos y mantenimiento continuo de procesos. Su objetivo ha sido proporcionar soluciones eficientes a empresas de gran alcance, facilitando la implementación de tecnología en sus operaciones.

Gracias a su enfoque innovador, la empresa logró establecerse rápidamente en el mercado de TI, consolidando un alto estándar en la planificación, desarrollo, calidad e implementación de software, basándose en un ciclo de vida de proyecto estructurado.

En respuesta a su rápido crecimiento, Globant identificó la necesidad de expandir sus áreas de oportunidad y mejora para mantenerse a la vanguardia tecnológica. Como parte de este proceso, el área de Quality Engineering fue optimizada para reducir los riesgos comerciales de los clientes, proporcionando servicios de pruebas innovadoras que garantizan productos de alta calidad y satisfacen las necesidades de los usuarios.

Servicios de Quality Engineering

Dentro de los servicios especializados en pruebas de software, Globant ofrece soluciones diseñadas para garantizar la funcionalidad, estabilidad y eficiencia de los productos tecnológicos. Algunos de estos servicios incluyen:

- Agile Testing
- Game Testing
- Test Automation

- Mobile Testing
- Accessibility Testing
- Big Data Testing
- Load & Performance Testing
- Mobile Performance Testing
- AI Testing

Figura 3

Modelo de Actividades del Testing en la Empresa

Para evitar que las pruebas sean tratadas como una fase aislada, los procesos de prueba y la estrategia se adaptan para ejecutarlas lo antes posible, manteniendo siempre el enfoque en la entrega de un producto bajo altos estándares de calidad y alineado con las necesidades del negocio.

La implementación del concepto de pruebas continuas, junto con la adopción de Integración Continua y Despliegue Continuo (CI/CD) en coordinación con los desarrolladores, permite ampliar el alcance de la calidad tanto en el desarrollo como en la

producción. Esto se logra mediante la automatización de pruebas, garantizando una distribución continua eficiente y confiable.

2.2 Identificación de Oportunidad o Necesidad en el Área de Actividad Profesional

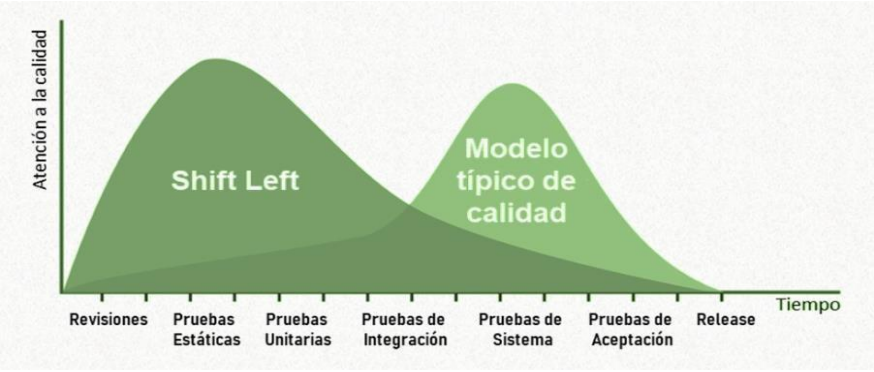
Dentro del proceso de calidad en la empresa Globant, se ha identificado una oportunidad de optimización en el proceso de pruebas recurrentes en entornos de desarrollo, con el objetivo de reducir el sobretrabajo y evitar la re-ejecución innecesaria de pruebas en ambientes que pueden ser manipulados por los desarrolladores. Estos entornos, al ser configurables según sus necesidades, no siempre son confiables para la validación de calidad.

Por ello, se busca reformar, estructurar y priorizar los entornos de prueba en tres niveles, alineándose con uno de los principios del testing establecidos por el ISTQB, que señala que las pruebas exhaustivas son imposibles, ya que nunca se puede tener control total sobre un producto en desarrollo.

La implementación del concepto de pruebas tempranas permitirá optimizar el proceso de testing, asegurando que la calidad del software se valide de manera eficiente y adaptada a las características y complejidad de cada entorno.

Figura 4

Modelo de Calidad en Etapas Tardías en la Empresa



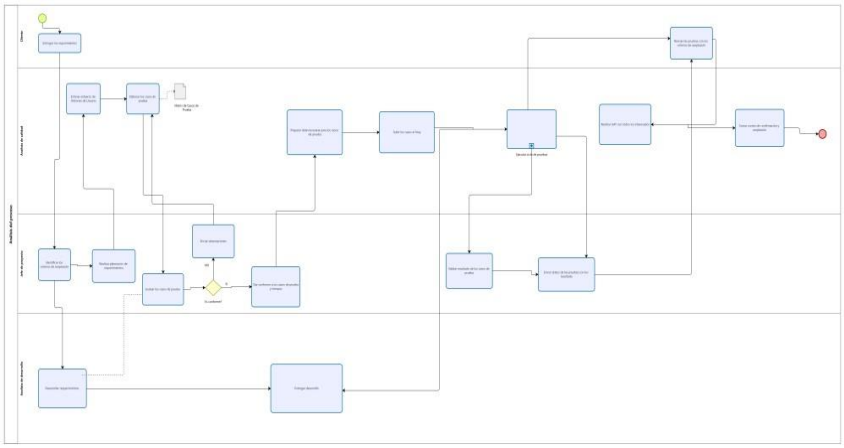
2.3 Objetivos de la Actividad Profesional

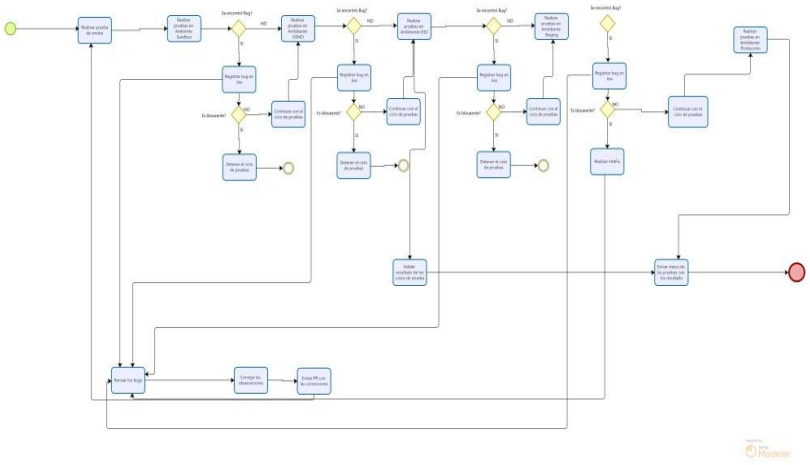
El objetivo principal de la actividad profesional del Bachiller es implementar pruebas tempranas con el propósito de prevenir defectos en lugar de detectarlos en etapas avanzadas del desarrollo, cuando el ciclo de desarrollo de software ya está próximo a finalizar. Este enfoque se orienta a la prevención de errores en lugar de su corrección tardía. Para lograrlo, es fundamental fomentar la colaboración y aplicar prácticas y técnicas que promuevan un entendimiento compartido dentro del equipo de trabajo.

Un enfoque moderno del testing implica realizar pruebas de manera continua a lo largo del ciclo de desarrollo, así como realizar un análisis de riesgos que puedan afectar el valor del producto de software. Por ello, es esencial definir qué probar, cómo probar, cuándo probar y dónde probar. Asimismo, el proceso de pruebas debe iniciarse desde la ideación de los requerimientos y la definición de los criterios de aceptación, garantizando así una validación temprana y efectiva del producto.

Figura 5

Diagrama del Proceso de Calidad de Software en la Empresa





2.4 Justificación de la Actividad Profesional

El presente trabajo de suficiencia profesional tiene como finalidad proponer una mejora en el proceso de calidad, con el objetivo de agilizar, optimizar y reducir el reproceso en las fases de testing.

La mejora planteada consiste en aplicar el concepto de pruebas tempranas en un ambiente de pruebas controlado, eliminando dos fases de testing en entornos manejados por los desarrolladores. De este modo, se evita la duplicación innecesaria de casos de prueba, optimizando el uso de recursos y mejorando la eficiencia del proceso de aseguramiento de calidad.

2.5 Resultados Esperados

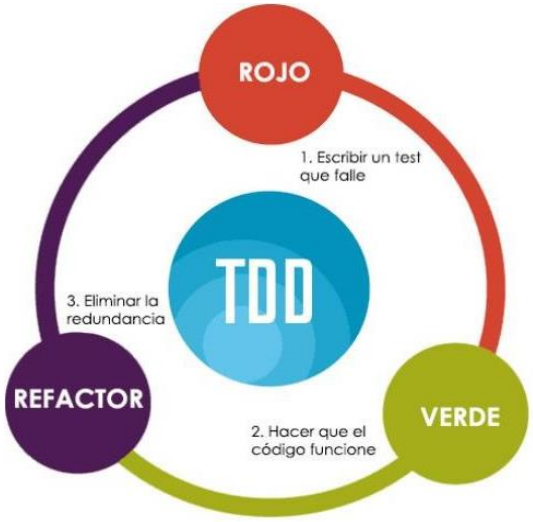
Aplicación de la Metodología Ágil

Para mejorar el proceso de calidad del software y garantizar que cumpla con los requisitos necesarios, es fundamental asegurarnos de que el enfoque implementado sea el adecuado. Esto incluye la utilización de prácticas como el Desarrollo Guiado por Pruebas (TDD).

El modelo de desarrollo guiado por pruebas utilizado en Globant Perú se basa en realizar pruebas unitarias antes de escribir el código, asegurando un proceso estructurado y confiable desde el inicio. Este enfoque permite garantizar que el software cumpla con los estándares de calidad y se alinee con los objetivos del proyecto (Figura 6).

Figura 6

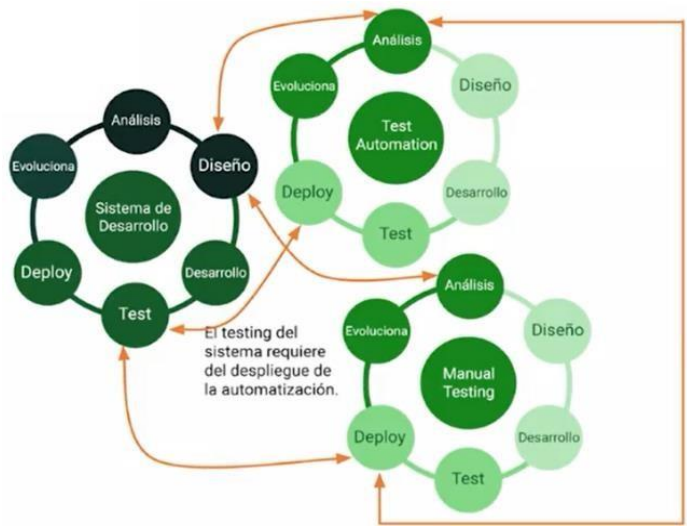
Modelo de Desarrollo Guiado por Pruebas de Software en la Empresa



Asimismo, se promueve el uso de pruebas automatizadas como una mejora sobre el testing manual. Según el *Curso de Introducción a la Automatización de Pruebas*, el modelo de automatización muestra una clara ventaja al reducir tiempos de ejecución, minimizar errores humanos y garantizar la repetibilidad de los casos de prueba. Este enfoque ofrece una diferencia significativa frente al Manual Testing, que depende más de la intervención directa y tiende a ser menos eficiente (Figura 7).

Figura 7

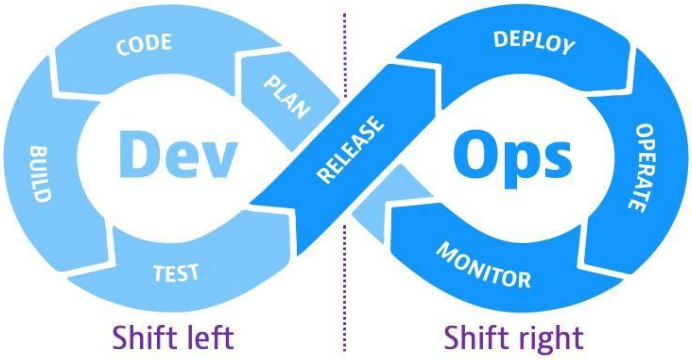
Modelo de Automatización de Pruebas de Software



El enfoque Shift-Left, clave en la metodología ágil DevOps, destaca la importancia de incorporar las pruebas desde las primeras etapas del desarrollo. Este concepto, según Saif Gunja, permite identificar y corregir defectos antes de que avancen a fases más complejas, ahorrando costos y mejorando la calidad del producto desde sus cimientos (Figura 8).

Figura 8

Enfoque Shift-Left



Por otro lado, el Enfoque Cero Defectos considera la calidad como un resultado esperado basado en un proceso de mejora continua. Este concepto, respaldado por el marco

de *Calidad Aplicada en la Gestión Empresarial*, enfatiza la necesidad de establecer estándares estrictos y un compromiso con la excelencia en cada etapa del ciclo de vida del software (Figura 9).

Figura 9

Enfoque Cero Defectos

Finalmente, un conjunto de herramientas específicas facilita la implementación de pruebas en diversas etapas del proceso de desarrollo, asegurando que el testing sea integral y efectivo. Estas herramientas permiten un control más riguroso y una mejor adaptabilidad del sistema a las necesidades del negocio, reforzando la calidad del producto final (Figura 10).

Figura 10

Herramientas para Pruebas de Software

CAPÍTULO III

MARCO TEÓRICO

A continuación, se describirán las bases teóricas utilizacdas en el ejercicio profesional de la empresa.

3.1 Integración Continua / Despliegue Continuo

La Integración Continua (CI) es una práctica que implica agregar de manera automática y frecuente los cambios de código a un repositorio común, permitiendo que los desarrolladores trabajen con una base de código actualizada y consistente.

El Despliegue Continuo (CD), también conocido como implementación continua, es un proceso en el que los cambios de código se integran, prueban y distribuyen. Este proceso tiene dos enfoques principales:

- En la distribución continua, los cambios se prueban y están listos para ser implementados, pero no se lanzan automáticamente al entorno de producción.
- En la implementación continua, las actualizaciones se despliegan de forma automatizada directamente en producción, agilizando la entrega de nuevas funciones y correcciones.

Figura 11

Integración Continua / Despliegue Continuo

El uso de CI/CD permite evitar errores y fallas de código en las empresas sin perjudicar el ciclo continuo de desarrollo y actualizaciones de software. Además, esta

metodología ayuda a mitigar la complejidad, aumentar la eficacia y optimizar los flujos de trabajo, especialmente a medida que las aplicaciones crecen.

La CI/CD automatiza las tareas manuales necesarias para que el código pase de la etapa de desarrollo a la producción. Esto no solo reduce el tiempo de inactividad, sino que también acelera los lanzamientos de código. Gracias a la rapidez con la que se integran las actualizaciones, los comentarios de los usuarios pueden incorporarse con mayor frecuencia y eficacia, lo que genera resultados positivos para los usuarios y aumenta la satisfacción del cliente.

Integración Continua (CI)

La integración continua, representada por la parte "CI" en la sigla CI/CD, es un flujo de automatización que otorga a los desarrolladores fusionar cambios de código en un repositorio compartido con mayor facilidad y frecuencia. Cada vez que se realiza una actualización, se activan pruebas automatizadas para verificar que los cambios sean confiables.

La CI resuelve el problema de que múltiples desarrolladores trabajen simultáneamente en diferentes ramas de una aplicación, lo que podría generar conflictos al unificar el código. Para garantizar su éxito, las modificaciones incorporadas deben validarse mediante:

- El flujo de compilación dentro de la aplicación de forma automática.
- La ejecución de pruebas automatizadas (generalmente de unidad e integración) para asegurar que los cambios no introduzcan fallas.

Este proceso implica probar todos los componentes, desde las clases individuales hasta los módulos que conforman toda la aplicación. En caso de que las pruebas detecten una

incompatibilidad entre el código nuevo y el existente, la CI facilita una rápida resolución de errores.

CI/CD y su relación con DevOps

La CI/CD es esencial dentro de la metodología DevOps, se encarga de promover la relación entre los equipos de desarrollo y operaciones. Ambas prácticas se centran en la automatización de los procesos de integración y entrega de código, acelerando la transición de una idea (como una nueva función, mejora o corrección de errores) desde el desarrollo hasta su implementación en producción, donde genera valor para el usuario final.

En el marco colaborativo de DevOps, la seguridad se considera una responsabilidad compartida e integrada en cada etapa del proceso. Esta filosofía de trabajo ha dado origen al término DevSecOps, que enfatiza la necesidad de diseñar una base sólida de seguridad desde el inicio.

3.2 DevSecOps

DevSecOps significa desarrollo, seguridad y operaciones y representa una extensión de la práctica de DevOps. Cada término en DevSecOps define funciones y responsabilidades específicas para los equipos de software durante la creación de aplicaciones.

Componentes de DevSecOps

Desarrollo. El desarrollo abarca la planificación, codificación, creación y prueba de la aplicación. Este proceso se centra en garantizar que la aplicación cumpla con los requisitos funcionales y técnicos esperados.

Seguridad. La seguridad implica incorporar medidas de protección desde las primeras etapas del ciclo de desarrollo de software. Los programadores aseguran que el

código esté libre de vulnerabilidades, mientras que los profesionales de seguridad realizan pruebas exhaustivas antes del lanzamiento del software, mitigando riesgos y amenazas.

Operaciones. El equipo de operaciones es responsable de la publicación del software, su monitoreo continuo y la resolución de problemas que puedan surgir una vez implementado.

Objetivo de DevSecOps

El principal objetivo de DevSecOps es permitir que los equipos de desarrollo aborden los problemas de seguridad de manera rápida y eficiente. A diferencia de las prácticas tradicionales de seguridad, que a menudo no se adaptaban a ciclos de desarrollo más cortos ni a actualizaciones rápidas, DevSecOps ofrece una solución que armoniza la seguridad con la velocidad del desarrollo moderno.

Implementación de DevSecOps

Para implementar DevSecOps con éxito, las empresas deben promover un cambio cultural liderado por la alta dirección. Los líderes séniores son los encargados de explicar al equipo de DevOps La relevancia y las ventajas de implementar prácticas de seguridad.

Además, es crucial que los desarrolladores de software y los equipos de operaciones dispongan de las herramientas, sistemas y motivación necesarios para integrar DevSecOps en su flujo de trabajo.

3.3 DevOps

DevOps es un marco de trabajo y una filosofía en constante evolución que busca mejorar el desarrollo de aplicaciones, reducir los tiempos de entrega y acelerar la publicación de nuevas funciones o productos de software para los clientes.

Esta metodología fomenta una comunicación fluida y continua, así como la colaboración, integración, visibilidad y transparencia entre los equipos de desarrollo de aplicaciones (Dev) y sus homólogos en operaciones tecnológicas (Ops).

La conexión entre Dev y Ops abarca todas las etapas del ciclo de vida de DevOps, desde la planificación inicial del software hasta las fases de codificación, construcción, pruebas, despliegue, operación y monitoreo continuo. Este enfoque permite establecer un bucle de retroalimentación constante con los clientes, incorporando sus comentarios para mejorar, desarrollar, probar y desplegar nuevas características de manera más ágil. Como resultado, se logran lanzamientos continuos y rápidos de actualizaciones o cambios funcionales necesarios.

Los pilares de DevOps: CAMS

DevOps organiza sus objetivos en cuatro categorías principales:

- Cultura
- Automatización
- Medición
- Uso compartido (CAMS, por sus siglas en inglés)

Los instrumentos de DevOps juegan un papel fundamental en estas categorías, ya que permiten optimizar, acortar y automatizar las tareas manuales, estáticas o complejas que son de suma importancia para la integración, el desarrollo, las pruebas, la puesta en marcha y la supervisión. Estos instrumentos también dan iniciativa a los principios clave de DevOps, como la automatización, la colaboración y la integración entre los equipos de desarrollo y operaciones.

Herramientas en las etapas del ciclo de vida de DevOps

A continuación, se describen las principales fases del ciclo de vida de DevOps y algunas herramientas empleadas en cada una:

Planificación. En esta fase se establecen los requisitos y objetivos empresariales. Herramientas como Jira o Git facilitan el seguimiento de problemas conocidos y la gestión de proyectos.

Codificación. Esta fase incluye el diseño del software y la creación del código. Herramientas utilizadas:

- GitHub
- GitLab
- Bitbucket
- Stash
- Compilación

Se gestionan las versiones y compilaciones del software mediante herramientas que automatizan la creación de paquetes listos para la producción. Ejemplos:

- Docker
- Ansible
- Puppet
- Chef
- Gradle
- Maven
- JFrog Artifactory

Pruebas. En esta etapa se realizan pruebas continuas (manuales o automatizadas) para garantizar la calidad del software. Herramientas de muestra:

- JUnit
- Codeception
- Selenium
- Vagrant
- TestNG
- BlazeMeter

Puesta en marcha. Se utilizan herramientas para administrar y automatizar las tareas de producción relacionadas con el lanzamiento de nuevas versiones. Ejemplos:

- Puppet
- Chef
- Ansible
- Jenkins
- Kubernetes
- OpenShift
- OpenStack
- Docker
- Jira

Funcionamiento. Esta etapa se centra en la administración del software durante su producción. Herramientas utilizadas:

- Ansible
- Puppet

- PowerShell
- Chef
- Salt
- Otter

Supervisión. Aquí se identifican y recopilan datos sobre dudas que surgen en una versión específica del software en producción. Herramientas de muestra:

- New Relic
- Datadog
- Grafana
- Wireshark
- Splunk
- Nagios
- Slack

3.4 SDLC (Ciclo de Vida del Desarrollo de Software)

El Ciclo de Vida del Desarrollo de Software (SDLC) es un proceso organizado que orienta a los equipos de software en la creación de aplicaciones de alta calidad. Este marco de trabajo ayuda a reducir costos, minimizar errores y garantizar que el software se mantenga alineado con los objetivos del proyecto en cada etapa del desarrollo.

El SDLC lleva a los equipos de software a través de las siguientes etapas clave:

Análisis de Requisitos

En esta fase, se identifican y documentan las necesidades del cliente y los objetivos del proyecto, asegurando una comprensión clara de lo que debe lograrse.

Planificación

Se elabora un plan detallado que define los recursos, el cronograma y las tareas necesarias para completar el proyecto de manera eficiente.

Diseño Arquitectónico

Se diseña la arquitectura del software, incluyendo la estructura del sistema y las especificaciones técnicas necesarias para cumplir con los requisitos.

Desarrollo de Software

En esta etapa, los desarrolladores codifican el software según los requisitos y diseños definidos previamente.

Pruebas

Se realizan pruebas para verificar y validar que el software funcione correctamente, identificando y solucionando cualquier error o defecto antes de la implementación.

Implementación

El software se despliega en el entorno de producción, donde está disponible para los usuarios finales.

3.5 ISTQB

El *International Software Testing Qualifications Board* (ISTQB) es una organización internacional dedicada a la certificación de calidad en software. Fundada en noviembre de 2002 en Edimburgo y registrada legalmente en Bélgica, el ISTQB se encarga de definir y mantener una organización de certificación a nivel mundial para profesionales del testing de software.

Esta organización suministra un esquema de estudios y un glosario que establecen las bases para la acreditación y evaluación de los profesionales del testing, labor que es gestionada por los comités de cada país. Según el propio ISTQB, han creado el esquema más exitoso del mundo para la certificación de probadores de software, siendo una de las certificaciones más importantes a nivel personal en el campo de la calidad del software.

Aunque existen certificaciones específicas para organizaciones, el ISTQB destaca como una de las pocas certificaciones de alto valor reconocidas internacionalmente para testers e ingenieros de calidad, especialmente en Europa (Testeando Software, 2014).

Niveles de Certificación ISTQB

- ISTQB Foundation Level
- ISTQB Advanced Level
- Advanced Level
- Expert Level

3.6 Testing

El testing de software o Software QA (Aseguramiento de la Calidad del Software) es una rama de la ingeniería de software que consiste en ejecutar programas o aplicaciones siguiendo una metodología estructurada para identificar errores y garantizar su calidad. Este proceso puede definirse también como la validación y verificación de un programa de software o aplicación.

Es fundamental destacar que el testing debe ejecutarse de forma paralela al desarrollo del software. A medida que se construye el producto, es necesario realizar tareas de testing

para prevenir problemas de funcionalidad y corregir desviaciones antes del lanzamiento. En términos generales, las pruebas de software son esenciales para:

- Detectar errores en el software.
- Verificar que el software cumple con los requisitos del cliente.
- Esto permite al equipo de desarrollo corregir errores y entregar un producto de alta calidad.

Importancia del Testing en el Desarrollo de Software

El testing es crucial porque, a lo largo del proceso de desarrollo, existen múltiples puntos donde errores humanos pueden derivar en un software que no cumpla con las expectativas o requisitos del cliente. Ante la creciente competencia en el sector tecnológico, realizar pruebas adecuadas se ha transformado en un factor fundamental para el éxito de cualquier desarrollo. Para verificar la calidad de una aplicación, existen diferentes tipos de pruebas que permiten evaluar diversos aspectos del software. A continuación, se detallan algunas de ellas:

Pruebas Funcionales. Evalúan el comportamiento del sistema, subsistema o componentes de software, asegurándose de que cumplan con los requisitos establecidos. Incluyen controles como la seguridad y la interoperabilidad.

Pruebas No Funcionales. Examinan el comportamiento externo del sistema, incluyendo aspectos como el rendimiento, la usabilidad y la confiabilidad.

Pruebas Estructurales. Se centran en la comprobación de los componentes internos del software y su integración.

Pruebas de Regresión. Estas pruebas se realizan después de corregir errores, para verificar que el problema ha sido solucionado y que no ha afectado otras partes del sistema (Lupita, 2021).

3.7 Jira

Jira es una herramienta web que se ha establecido como el estándar en el mercado en áreas como la administración de proyectos, tareas y gestión de errores. Diseñado especialmente para el desarrollo de software, Jira es una herramienta versátil que facilita significativamente los flujos de trabajo y la colaboración, ya sea en equipos pequeños o grandes.

El software fue desarrollado por la empresa australiana Atlassian y está disponible en el mercado desde 2002. Curiosamente, el nombre *Jira* deriva del término japonés *Gojira*, que significa Godzilla, pero se adoptó su versión abreviada y coloquial.

Características Principales de Jira

Jira ofrece una flexibilidad excepcional, permitiendo su uso en diferentes áreas según las necesidades del equipo. Aunque es ampliamente utilizada en el desarrollo de software, también es aplicable a otros ámbitos no técnicos. La herramienta actúa como un panel de visión general y como una herramienta de planificación, ayudando a optimizar el flujo de trabajo en un equipo. Entre sus principales funciones destacan:

Asignación de Tareas. Las tareas individuales se crean en forma de tickets, los cuales se distribuyen y procesan dentro del equipo.

Seguimiento del Progreso. Permite compartir estados intermedios y realizar un seguimiento de los avances del proyecto.

Identificación y Resolución de Errores: Jira facilita la detección y solución rápida de problemas potenciales.

Adecuación a Metodologías Ágiles

Jira es especialmente adecuado para metodologías de gestión de proyectos ágiles, como Scrum y Kanban. Gracias a su diseño, la herramienta puede gestionar proyectos de diferente envergadura, desde pequeños hasta grandes proyectos complejos, de manera eficiente. Además, Jira incluye una función de documentación detallada que cumple dos propósitos importantes:

Registro Completo del Proyecto. Proporciona toda la documentación necesaria para mantener un historial claro.

Integración de Nuevos Miembros. Permite que los nuevos integrantes del equipo se adapten de manera rápida y sencilla al flujo de trabajo existente.

Personalización y extensibilidad

Además de su producto principal, Atlassian ofrece numerosos plugins y soluciones complementarias que pueden integrarse fácilmente en Jira. Estas extensiones permiten personalizar y optimizar los flujos de trabajo para ajustarse a las necesidades específicas de uno o varios proyectos.

Jira, con su enfoque integral y versátil, es una herramienta indispensable para equipos que buscan mejorar su productividad y colaboración, tanto en proyectos técnicos como en no técnicos (IONOS Digital Guide, 2022).

3.8 Release

La gestión de entregas de software es el proceso de entregar nuevas versiones de software o actualizaciones. Este proceso abarca mucho más que simplemente crear o actualizar un programa; incluye una serie de pasos necesarios para garantizar que la nueva entrega cumpla con los requisitos y expectativas.

Proceso de Gestión de Entregas

Cuando se requiere una nueva entrega, se deben seguir varios pasos:

Recopilación de Requisitos. Se identifican las nuevas exigencias y las dependencias con los componentes existentes.

Producción de la Nueva Versión. Se desarrolla la actualización o nueva versión del software.

Pruebas. Se somete el software a pruebas para validar su funcionalidad y calidad.

Preparación de la Entrega. Se prepara un archivo listo para instalar, acompañado de manuales de uso.

Documentación. Se entregan los documentos de diseño y pruebas a la organización.

El propósito de este proceso estructurado es adaptar el software a nuevos requisitos, corregir fallas y responder a la evolución de las tecnologías.

Beneficios de la Gestión de Entregas de Software

Implementar un *Release Management* estructurado ofrece ventajas significativas frente a un enfoque intuitivo o improvisado. Entre los principales beneficios destacan:

Planificación de Recursos. Facilita la asignación eficiente de los recursos necesarios para el desarrollo y entrega.

Estructura y Eficiencia. Garantiza un proceso organizado y efectivo.

Gestión de Cambios. Introduce los cambios de manera controlada, minimizando el impacto sobre los usuarios.

Validación Previa a la Entrega. Permite verificar la funcionalidad y el uso adecuado del software mediante pruebas exhaustivas antes de su implementación.

Control de Versiones. Asegura el uso de la versión correcta gracias a un almacenamiento centralizado y la gestión de versiones.

Importancia del Rol de Gestor de Entregas

La gestión de entregas requiere la supervisión de un profesional responsable de coordinar las actividades de desarrollo, pruebas, implementación y soporte al usuario, especialmente en proyectos complejos. Este profesional debe contar con:

- Un conocimiento integral del ciclo de vida del desarrollo de software.
- Experiencia en diversos sistemas operativos y plataformas de aplicaciones.
- Una comprensión de las funciones y aspectos económicos relacionados con el desarrollo.

La gestión de entregas responde a la necesidad de garantizar que los proyectos se ejecuten de manera estructurada, eficiente y con un enfoque en la calidad.

3.9 UAT

La *Prueba de Aceptación del Usuario* (UAT) es una etapa crucial en el ciclo de vida del desarrollo de software, donde el cliente o usuario evalúa el software para verificar si funciona adecuadamente y cumple con los requisitos establecidos y cumple con las expectativas planteadas durante la fase de análisis.

La UAT se realiza como la última prueba de software, después de completar otras pruebas como las pruebas unitarias, pruebas del sistema, pruebas funcionales y pruebas de regresión.

Propósito de la UAT

Cada software se desarrolla con base en requisitos o necesidades específicas del cliente o usuario. Por ello, el propósito principal de la UAT es validar que el software cumpla con esos requisitos.

Momento Clave. Es la última oportunidad para que el usuario o cliente pruebe el software antes de su lanzamiento.

Validación Realista. Permite comprobar si el software puede realizar de manera eficiente y sin errores las tareas para las que fue diseñado, en un entorno real.

Retroalimentación del Cliente. Los comentarios del cliente o usuario durante esta fase son fundamentales para realizar mejoras finales y asegurar que el producto sea tanto de alta calidad como relevante para sus necesidades.

Importancia de la UAT en el ciclo de desarrollo

La UAT valida que el software cumple con los requisitos del cliente o usuario final, por lo que se realiza después de haber completado todas las pruebas técnicas. Estas incluyen:

- Pruebas unitarias
- Pruebas de integración
- Pruebas del sistema
- Pruebas funcionales

A pesar de que los desarrolladores y testers pueden validar el software según las especificaciones funcionales, la perspectiva del usuario final puede revelar inconsistencias,

problemas de usabilidad o requisitos que no fueron claramente definidos. Algunas razones comunes de estas discrepancias incluyen:

Requisitos Poco Claros. Los desarrolladores pueden haber recibido especificaciones incompletas o ambiguas.

Cambios en el Alcance. Durante el proyecto, los requisitos o prioridades pueden haber cambiado sin una adecuada comunicación.

Mejoras Necesarias. El cliente puede identificar ajustes que no se contemplaron inicialmente.

La *Prueba de Aceptación del Usuario* (UAT) es esencial para garantizar que el software no solo sea técnicamente funcional, sino que también cumpla con las expectativas y necesidades reales de los usuarios. Al ofrecer la última oportunidad para realizar ajustes y mejoras antes del lanzamiento, la UAT asegura que el producto final sea eficiente, libre de errores y completamente alineado con los objetivos del cliente (Prasad, 2024).

3.10 Plugins

Un plugin es un componente de código diseñado para añadir funciones a un programa o herramienta existente. No funciona de manera independiente, sino que se integra a un software principal (también conocido como host) para dar una mayor experiencia al usuario y brindar acciones adicionales. Por ejemplo, en un gestor de contenidos, un plugin puede agregar funcionalidades que ayuden a analizar el desempeño de un sitio web y sus páginas.

Los plugins no se limitan únicamente a los gestores de contenidos. De hecho, cualquier software que permita agregar aplicaciones o extensiones para optimizar su uso puede ofrecer soporte para plugins. Cuando una aplicación o software es muy demandado,

los usuarios suelen solicitar nuevas funciones al desarrollador. Para satisfacer estas necesidades, el software principal proporciona un entorno adecuado que permite a los plugins:

- Registrarse de manera autónoma.
- Intercambiar datos con el host.

Esto es posible gracias a las API (Interfaces de Programación de Aplicaciones), que ofrecen una interfaz estándar. Las API facilitan que tanto los desarrolladores originales como terceros puedan crear plugins para mejorar el software principal. Además, los usuarios tienen la flexibilidad de añadir o eliminar estos complementos sin alterar el programa base ni comprometer su estabilidad.

Ventajas y adopción de los plugins

Cuando un plugin se vuelve popular, suele estar disponible en los sitios web del software anfitrión o en plataformas que recopilan extensiones diseñadas para mejorar programas existentes. En algunos casos, las propias empresas fomentan la creación de plugins o permiten que terceros los integren en sus aplicaciones, aunque no hayan sido desarrollados por ellas mismas. Los plugins son una herramienta clave para:

- Personalizar la experiencia del usuario.
- Ampliar las funcionalidades del software principal.
- Facilitar el uso y optimización del programa.

Este enfoque promueve la colaboración entre desarrolladores y usuarios, y asegura que el software pueda adaptarse a nuevas necesidades de manera eficiente (Coppola, 2023).

CAPÍTULO IV

IMPLEMENTACIÓN

4.1 Descripción de Actividades Técnicas Profesionales

Este capítulo del Trabajo de Suficiencia Profesional está enfocado en detallar el proceso llevado a cabo por el bachiller en su rol como Analista de Calidad durante su carrera profesional en Ingeniería de Sistemas e Informática, específicamente en el ámbito de la calidad de software.

4.1.1 Enfoque de las Actividades Profesionales

Para garantizar la calidad del software y cumplir con las expectativas de los usuarios, el analista de calidad debe realizar diversas funciones que abarcan todas las fases del ciclo de vida del desarrollo de software (SDLC). A continuación, se describen las principales actividades técnicas y profesionales:

Refinamiento del Producto. El refinamiento del producto consiste en detallar y dividir los elementos del backlog en unidades más pequeñas y concretas. Esto incluye añadir descripciones, criterios de aceptación, estimaciones, actualizaciones y ordenar los elementos del backlog.

- El refinamiento no se limita a reuniones específicas, sino que puede realizarse durante el Sprint Planning, el Daily Scrum o de manera informal entre los miembros del equipo.
- El Product Owner es responsable de garantizar que los elementos del backlog sean transparentes y comprendidos por todo el equipo Scrum.
- Generalmente, las reuniones de refinamiento se centran en: Recortar y ajustar los elementos del backlog; estimar los esfuerzos necesarios; redactar criterios de aceptación para los Product Backlog Items (PBIs) (Rodríguez, 2023).

Análisis de Requerimientos. El análisis de requisitos es una etapa crucial en el desarrollo de software. Implica identificar, documentar y analizar las necesidades del cliente o usuario para garantizar que el proyecto cumpla con sus objetivos. Este proceso incluye:

- Recopilación de requisitos de las partes interesadas.
- Documentación y validación de los requisitos.
- Creación de un documento de especificación que sirve como guía para el diseño, desarrollo y pruebas.

El análisis de requisitos es un proceso iterativo que debe ser flexible para adaptarse a cambios en el alcance del proyecto (Walker, 2023).

Planteamiento de Estrategias de Pruebas. Una estrategia de pruebas del software es un mapa detallado que integra métodos de diseño de pruebas en una secuencia bien planificada. Permite definir: Los pasos que se realizarán; el tiempo, esfuerzo y recursos necesarios; la secuencia de pruebas, desde las unitarias hasta las de regresión (Toledo, 2022).

Diseño de Casos de Prueba. Los casos de prueba son conjuntos de condiciones y variables que se utilizan para verificar si una característica o funcionalidad del software funciona correctamente. Un caso de prueba incluye: ID del caso; Descripción; Condiciones previas; Gravedad; Entorno; Pasos y datos; Resultados esperados y reales (QAwerk, 2021).

Gestión y Ejecución de Pruebas. La gestión de pruebas utiliza herramientas para planificar, preparar y ejecutar pruebas de manera eficiente. Beneficios:

- Creación de entornos de prueba controlados.
- Mejora de la trazabilidad y la colaboración entre equipos.
- Identificación y resolución temprana de problemas.

Durante esta fase, los gestores de pruebas supervisan el progreso, gestionan problemas y generan informes para las partes interesadas (Gomstyn y Jonker, 2023).

Seguimiento y Reporte de Errores. El seguimiento de errores implica registrar y monitorear fallos detectados durante las pruebas. Los errores se clasifican por prioridad y gravedad para garantizar su resolución eficiente. El reporte debe incluir: descripción detallada del defecto; condiciones y contexto del fallo; y severidad asignada.

Automatización de Pruebas. La automatización utiliza herramientas para ejecutar pruebas y analizar resultados de manera más rápida y precisa que las pruebas manuales. Beneficios: reducción de costos y tiempo; mayor precisión y repetibilidad; detección eficiente de errores (Zap Chernyak, 2023).

Informe sobre Resultados. El reporte de pruebas contiene información sobre las discrepancias entre los resultados esperados y los obtenidos durante la ejecución de pruebas. Debe ser: Claro, transparente y trazable; completo y preciso para facilitar la comprensión del equipo. Y debe incluir detalles como: Descripción del defecto; Momento y lugar de detección; Severidad asignada para priorizar la resolución.

4.1.2 Alcance de las Actividades Profesionales

El alcance de las actividades profesionales realizadas por el bachiller se detalla de la siguiente forma:

- Estimar el esfuerzo de las Historias de Usuario.
- Elaborar los casos de prueba.
- Preparar la data necesaria para la ejecución de los casos de prueba.
- Subir los casos de prueba a la herramienta Xray.
- Validar la entrega del desarrollo.

- Realizar pruebas en el ambiente menor (Sandbox).
- Concretar reuniones con la participación del Arquitecto y el Visual Designer.
- Ejecutar pruebas en el ambiente de Demo.
- Organizar reuniones de UAT con el Product Owner (PO).
- Realizar pruebas en el ambiente de desarrollo (Dev).
- Coordinar reuniones de UAT con la participación del PO, el Business Analyst (BA), los desarrolladores y el Visual Designer (VD).
- Ejecutar pruebas en el ambiente de Staging (Stg).
- Organizar reuniones de UAT con la participación del PO, BA, desarrolladores, VD y Stakeholders.
- Elaborar un reporte detallado sobre los resultados de las pruebas.
- Enviar un correo de confirmación y aceptación.

4.1.3 Entregables de las Actividades Profesionales

El propósito del presente informe de Trabajo de Suficiencia Profesional es presentar los resultados obtenidos en aspectos tanto técnicos como de gestión y ejecución de proyectos, mediante la aplicación de la metodología SCRUM.

Términos y Definiciones. La siguiente lista presenta las descripciones de algunos términos clave que se utilizarán a lo largo del presente documento, los cuales hacen referencia a los aspectos técnicos y funcionales de Salesforce.

BU. Business Units.

SFMC. Salesforce Marketing Cloud.

CDP - Customer Data Platform. Solución tecnológica que permite a las empresas recoger, unificar y administrar datos de clientes provenientes de diversas fuentes.

MKT. Marketing.

CR. Conversion Rate. Es un valor porcentual que se calcula dividiendo el número de conversiones exitosas (compras, descargas, registros, etc.) entre el número total de conversiones potenciales.

HD. Home Delivery (Entrega a domicilio).

Pick Up. Servicio que permite a los clientes comprar en línea y recoger sus artículos en persona en un punto de venta designado.

C&C. Click and Collect. Método de envío utilizado en negocios digitales, donde el comprador puede recoger en la tienda, desde su auto, las compras realizadas en línea.

CC. Commerce Cloud

Tabla 1

Historia de Usuario

ID JIRA	ACCIÓN	DESCRIPCIÓN	ALCANCE	FUERA DE ALCANCE	SISTEMAS IMPACTADOS
EMC-9	Configuración del sender profile	Como usuario de Marketing Cloud, necesito el sender profile para envío de correos promocionales y transaccionales.	Sender profile para correos promocionales y transaccionales.	No se realizarán cambios o adecuaciones adicionales.	Marketing Cloud
EMC-7	Creación de documento y configuración de Einstein	Requiere un documento para solicitar la configuración de Einstein para la nueva BU de Palacio de Hierro.	Documento con especificaciones para la configuración de Einstein.		Salesforce Cloud/Einstein, Marketing Cloud
EMC-12	Configuración de Einstein	Configuración de collect codes en Einstein desde Salesforce Cloud.	Configuración de collect codes en Einstein.		Salesforce Cloud/Einstein, Marketing Cloud
EMC-15	Validación de configuración (Einstein)	Validar configuración realizada en SFCC para comprobar la correcta recepción de información.	Especificaciones de los tracking collect codes.		Salesforce Cloud/Einstein, Marketing Cloud
EMC-13	Documento para cambio de Fluent	Crear un documento con detalles técnicos para cambios y nuevos campos en Fluent.	Análisis técnico para especificaciones, cambios y nuevos campos requeridos.		Fluent
EMC-14	Implementación de cambios en Fluent	Implementar cambios en Fluent con tareas divididas por prioridad de entrega.	Funcionalidades impactadas: Journey Pick Up, Journey C&C, Journey de devoluciones.		Fluent
EMC-34	Migración de carrito abandonado	Enviar correo de carrito abandonado usando inteligencia de compra configurada en Einstein.	Journey de automatización de marketing para carrito abandonado.		Salesforce Cloud, Einstein, Marketing Cloud
EMC-35	Recomendaciones	Envío de recomendaciones de productos según el comportamiento de compra.	Journey de automatización de marketing para recomendaciones personalizadas.		Salesforce Cloud, Einstein, Marketing Cloud
EMC-36	Cupón Pimera Compra	Enviar correo con cupón de descuento para la primera compra.	Journey de automatización de marketing para envío de cupones.		Salesforce Cloud, Einstein, Marketing Cloud
EMC-37	Cambio de Precio	Avisar sobre cambios de precio mediante correo electrónico.	Journey de automatización de marketing para informar cambios de precio.		Salesforce Cloud, Einstein, Marketing Cloud
EPH-ECC-03	**CORREOS TRANSACCIONALES TRACKING DE PEDIDOS**				
EMC-21	Correo de confirmación de compra	Ejecutar el trigger para enviar correos de confirmación de compra.	Correos de confirmación para HD, C&C, Pick Up, Celebra y Noches Palacio.		Marketing Cloud, Salesforce Cloud
EMC-16	Journey Tracking de Pedido (Envio/Entrega)	Notificar envío y confirmación de entrega a través de correo.	Notificación de envío: Shipping update y HD. Confirmación de entrega HD.	Noches Palacio, Celebra, C&C, Pick Up.	Marketing Cloud
EMC-22	Journey de C&C	Enviar correo de confirmación para modalidad C&C.	C&C confirmación. C&C recordatorio 1er día. C&C recordatorio 2do día.		Marketing Cloud

EMC-23	Journey de Pick Up	Enviar correo de confirmación para modalidad Pick Up.	Pick Up confirmación. Pick Up recordatorio 1er día. Pick Up recordatorio 2do día.	Marketing Cloud
EPH-ECC-04	**CORREOS TRANSACCIONALES CANCELACIÓN**			
EMC-11	Correo de cancelación por falta de inventario	Configurar correos de cancelación por falta de inventario.	HD, C&C, Pick Up, correo de cancelación.	Marketing Cloud
EMC-30	Correo de cancelación por solicitud del cliente	Enviar correo de cancelación solicitado por el cliente.	HD, C&C, Pick Up, correo de cancelación.	Marketing Cloud
EMC-20	Correo de cancelación por fraude	Enviar correo de cancelación por fraude.	HD, C&C, Pick Up, Celebra, Noches Palacio, correo de cancelación.	Marketing Cloud
EPH-ECC-05	**CORREOS TRANSACCIONALES DEVOLUCIÓN**			
EMC-24	Journey de devoluciones	Informar al cliente sobre la aprobación de una devolución.	HD, solicitud y aprobación de devolución.	Marketing Cloud
EPH-ECC-06	**CORREOS TRANSACCIONALES ACCOUNT**			
EMC-25	Confirmación cambio de contraseña	Enviar correo de confirmación por cambio de contraseña.	My Account.	Marketing Cloud, Salesforce Cloud
EMC-26	Correo de recuperación de contraseña	Enviar correo de recuperación de contraseña.	My Account.	Marketing Cloud, Salesforce Cloud
EMC-27	Correo de creación de cuenta	Confirmar la creación de una cuenta con un correo.	My Account.	Marketing Cloud, Salesforce Cloud
EMC-28	Correo de Actualización de datos	Notificar sobre actualizaciones de datos en la cuenta.	My Account.	Marketing Cloud, Salesforce Cloud
EMC-31	Correo de suscripción al newsletter	Confirmar suscripción al newsletter.	My Account.	Marketing Cloud, Salesforce Cloud
EPH-ECC-07	**CORREOS TRANSACCIONALES STOCK**			
EMC-38	Journey de out y back stock	Enviar correo cuando un producto esté nuevamente en stock.	HD, C&C, back in stock, Pick Up.	Marketing Cloud, Salesforce Cloud
EPH-ECC-08	**CORREOS TRANSACCIONALES NOCHES PALACIO**			
EMC-32	Generate coins	Enviar correo con ticket de monedas adquiridas en Noches Palacio.	Noches Palacio.	Marketing Cloud, Salesforce Cloud
EMC-33	Journey Coins Expiration reminder	Recordar al cliente la expiración de sus monedas de Noches Palacio.	Noches Palacio.	Marketing Cloud, Salesforce Cloud

4.2 Aspectos Técnicos de la Actividad Profesional

4.2.1 Metodologías

Metodología Scrum. Es un enfoque ágil y cooperativo utilizado por equipos para lograr resultados satisfactorios en proyectos. Se caracteriza por entregas parciales que priorizan el beneficio del cliente. Este marco de trabajo está compuesto por tres fases principales:

- Organización del Product Backlog
- Ejecución del Sprint
- Control del proceso

Beneficios de la Metodología Scrum.

- Transparencia. Aumenta la visibilidad sobre el progreso del trabajo.
- Entrega rápida. Resultados disponibles en periodos cortos.
- Mejora en la comunicación. Facilita la interacción fluida entre los miembros del equipo.
- Colaboración. Promueve el trabajo en equipo y la autogestión.
- Productividad y calidad. Mejora los resultados al enfocarse en objetivos claros.
- Adaptabilidad. Responde rápidamente a cambios en los requisitos del proyecto.

Roles en Scrum.

- Scrum Master. Es responsable de garantizar que el equipo entienda y aplique correctamente los principios de Scrum. Tareas principales: Eliminar obstáculos que afecten la productividad; Facilitar la comunicación entre los miembros del equipo.

- Product Owner. Representa los intereses del cliente y del negocio en el proyecto. Responsabilidades: Gestionar y priorizar el Product Backlog; Asegurar que el equipo entienda los requisitos y expectativas del cliente.
- Equipo de Desarrollo. Grupo multidisciplinario encargado de completar las tareas necesarias para cumplir con las Historias de Usuario dentro del Sprint. Características del equipo: Trabajan de forma colaborativa y autogestionada; Toman decisiones en conjunto para alcanzar los objetivos del proyecto.

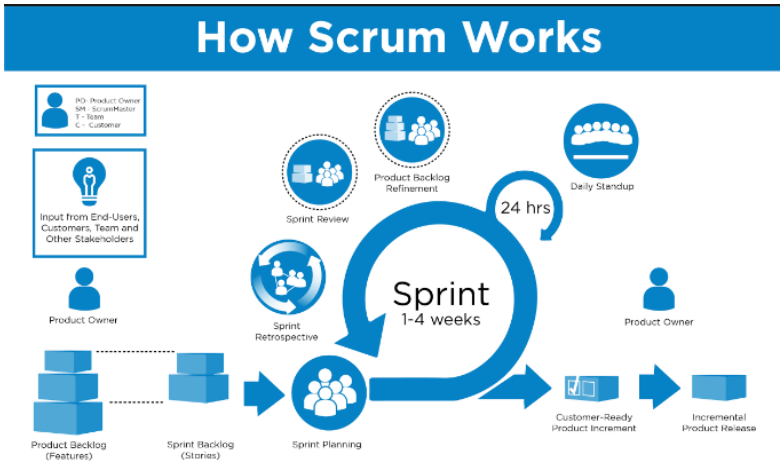
Actividades y Procesos en Scrum.

- Reuniones Diarias (Daily Scrum). Breves reuniones que se realizan diariamente para mantener al equipo informado sobre el progreso y los posibles bloqueos. Durante esta reunión, cada miembro responde a tres preguntas: ¿Qué hice ayer? ¿Qué haré hoy? ¿Qué obstáculos enfrento? (Rodríguez, 2023).
- Sesión de Planificación del Sprint. Marca el inicio de cada Sprint.
    - Propósito. Definir los elementos del Product Backlog a trabajar durante el Sprint.
    - Participantes. El Product Owner presenta las tareas prioritarias. El equipo de desarrollo estima el esfuerzo necesario y se compromete a completar las tareas seleccionadas (Rodríguez, 2023).
- Revisión del Sprint. Al finalizar el Sprint, se realiza una reunión para: Mostrar el trabajo completado al Product Owner y otros stakeholders; Recopilar comentarios y retroalimentación para mejorar en futuras iteraciones (Rodríguez, 2023).
- Eventos en Scrum: Estos eventos son reuniones clave que aseguran la colaboración y el progreso hacia la entrega del producto final. Se llevan a cabo en

momentos estratégicos y permiten: Garantizar la transparencia; Promover una comunicación efectiva entre todos los miembros involucrados (Rodríguez, 2023).

Figura 12

Metodología Scrum



Metodología Kanban. La metodología Kanban es una subcategoría de la gestión ágil de proyectos que promueve la planificación adaptativa, el desarrollo iterativo y la mejora continua. Este enfoque ayuda a los equipos a cumplir con los estándares establecidos y entregar resultados de alta calidad.

El método Kanban permite visualizar el progreso de las tareas a través de un tablero dividido en columnas, que representan diferentes estados del trabajo. Las columnas más comunes son:

- **Hacer.** Tareas por realizar.
- **En curso.** Tareas que se están trabajando.
- **A probar.** Tareas pendientes de validación por parte del cliente.
- **Completadas.** Tareas finalizadas.

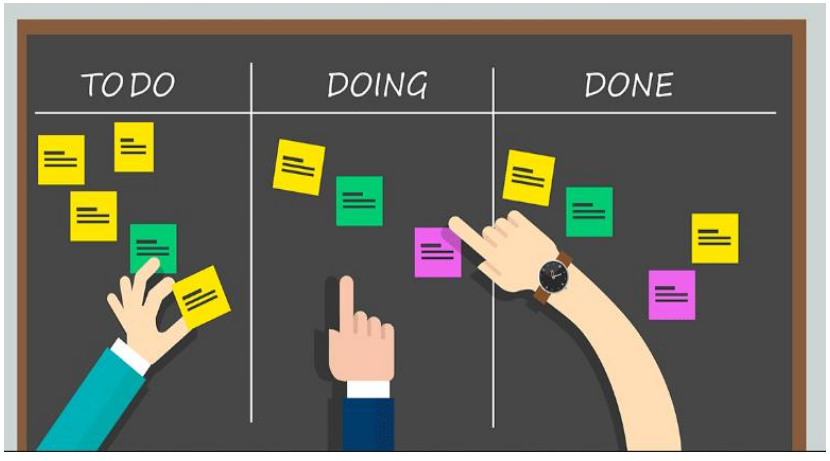
Las tareas se representan mediante tarjetas, etiquetas o notas post-it que se desplazan de una columna a otra según el estado del trabajo. Esto proporciona una visión clara del flujo de trabajo y del progreso del equipo.

Principios de la Metodología Kanban. Kanban se distingue por una serie de principios clave que lo diferencian de otras metodologías ágiles:

- **Calidad garantizada.** Todo lo que se hace debe salir bien a la primera. No hay margen de error, lo que refuerza el enfoque en la precisión.
- **Reducción del desperdicio.** Se centra en hacer solo lo necesario y evitar actividades superfluas. Promueve la eficiencia al eliminar elementos secundarios o irrelevantes.
- **Mejora continua.** No es solo un método de gestión, sino también una herramienta para mejorar continuamente los procesos y alcanzar los objetivos del proyecto.
- **Flexibilidad.** Las tareas se seleccionan desde el backlog en función de las prioridades y necesidades actuales. Se pueden ajustar según cambien los requisitos del cliente o del proyecto (Rodríguez, 2023).

Figura 13

Proceso de la Metodología Kanban



4.2.2 Técnicas

Las técnicas son un conjunto de reglas y protocolos diseñados para cumplir con los objetivos planteados durante la ejecución del Trabajo de Suficiencia Profesional. A continuación, se describen las principales técnicas empleadas:

Versionado. Es el proceso de asignar un nombre, código o número único a un software. Este sistema facilita la identificación, control y seguimiento de las diferentes versiones del producto a lo largo de su desarrollo.

Reuniones Diarias (Daily Meetings). Son reuniones de corta duración que permiten llevar un control sobre las actividades del equipo día a día. Durante estas reuniones, los miembros responden tres preguntas clave: ¿Qué hice ayer? ¿Qué haré hoy? ¿Tengo algún impedimento que me bloquee el avance?

Iteraciones. Este proceso está diseñado para completar los requisitos del cliente previamente definidos en el backlog. Las tareas se priorizan para su resolución durante cada iteración, permitiendo avanzar de manera eficiente y garantizar que el producto final cumpla con las expectativas del cliente.

Tabla 2

Aplicaciones

Aplicación	Descripción	Consideraciones adicionales
Slack	Herramienta interna de comunicación informal entre los miembros del equipo de trabajo de Globant.	Comunicaciones generalmente para todo el equipo.
Google Meet	Herramienta para la programación de reuniones internas o informales.	
Teams	Herramienta definida para la comunicación con el cliente (chat y reuniones).	- Sprint Review. - Sesiones de refinamiento o interacción con el cliente.

Email	Herramienta definida para la comunicación formal entre los miembros del equipo y el cliente.	Usamos el email de Globant para la interacción formal con el cliente.
VPN	Herramienta para acceder a la red privada.	
Drive	Repositorio interno compartido donde se almacena la información generada en el proyecto.	Requiere autorización previa para acceder al repositorio.
Jira	Gestión de backlog. - Estructura: - Épicas - Historias de Usuario - Spikes - Tareas	Requiere una cuenta por usuario. Gestionado por el cliente.
Confluence	Herramienta para la gestión de la documentación del proyecto: - Documentación funcional. - Documentación técnica. - Documentación de configuración. - Way of Work.	Requiere una cuenta por usuario. Gestionado por el cliente.
Salesforce	Incluye Salesforce Commerce Cloud (SFCC) y Salesforce Marketing Cloud (SFMC).	Requiere una cuenta por usuario. Gestionado por el cliente.
Figma	Herramienta para la gestión de los diseños del producto.	Requiere una cuenta por usuario. Gestionado por el cliente.
Bitbucket	Repositorio de código (DevTeam).	Requiere una cuenta por usuario. Gestionado por el cliente.
PlanIT Poker	Herramienta web para apoyar el proceso de estimación de esfuerzo de las Historias de Usuario.	Cada integrante del equipo se conecta a la sala habilitada por el BA para discutir y votar las Historias de Usuario.
Draw.io	Herramienta para realizar diagramas de entendimiento o bosquejos rápidos.	Los diagramas se pueden documentar en Confluence para referencia futura.
Bizagi	Aplicación para diagramar, documentar y presentar gráficamente procesos usando BPMN (Business Process Model and Notation).	La aplicación puede descargarse de forma gratuita.

4.2.3 Instrumentos

Los equipos y herramientas utilizados durante el desarrollo del Trabajo de Suficiencia Profesional son los siguientes:

Cliente. El cliente es la persona responsable de comunicar las necesidades de la entidad al equipo de desarrollo. Este rol es fundamental para garantizar que los entregables cumplan con las expectativas y requisitos establecidos.

Equipo de Cómputo. Laptop Lenovo con Procesador: AMD RYZEN 7 PRO 5850U with Radeon Graphics 1.90 GHz, Windows 10 Enterprise, Memoria RAM de 16 GB.

Ambientes de Pruebas. Se utilizaron diversos ambientes de pruebas, denominados como Sandbox, Demo, Developer y Staging, donde se realizaron diferentes tipos de pruebas de software:

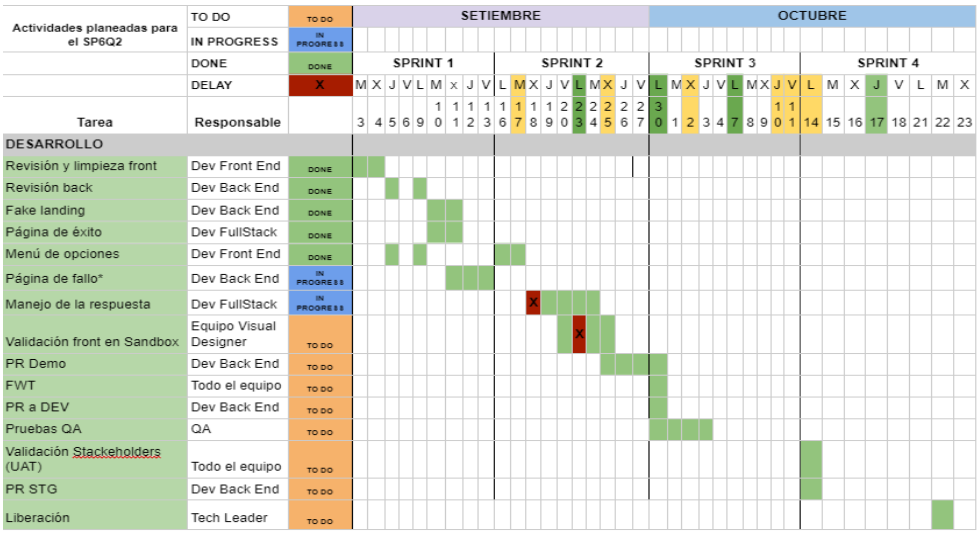
- ***Pruebas Manuales.*** Ejecución paso a paso realizada por personas. Navegación e interacción con la interfaz en diferentes módulos y resoluciones.
- ***Pruebas Automatizadas.*** Ejecutadas mediante programas que corren test scripts previamente escritos.
- ***Pruebas de Integración.*** Verificación de que los módulos independientes puedan integrarse y trabajar en conjunto.
- ***Pruebas Funcionales.*** Validan que los requisitos del negocio se cumplan, centrándose en los resultados de los procesos.
- ***Pruebas End-to-End.*** Simulan el proceso completo que realiza un usuario común al interactuar con el software.
- ***Pruebas de Regresión.*** Verifican que las nuevas implementaciones no afecten negativamente los escenarios previamente validados.
- ***Pruebas de Smoke.*** Ejecuciones rápidas para verificar aspectos clave del software sin entrar en detalles profundos.

4.3 Ejecución de las Actividades Profesionales

4.3.1 Cronograma de Actividades Realizadas

Figura 14

Cronograma de Actividades Realizadas



CAPÍTULO V

RESULTADOS

5.1 Resultados Finales de las Actividades Realizadas

En este apartado se analiza el resultado obtenido con el nuevo esquema implementado para la optimización de pruebas de software, el cual se desarrolló siguiendo las metodologías establecidas en el proyecto. Los principales resultados son los siguientes:

5.1.1 Metodología SCRUM

Se completaron cuatro sprints en las fechas comprometidas con los integrantes del equipo y aceptadas por el cliente. Cada sprint incluyó los siguientes entregables:

Product Backlog (Lista de Producto). Registro detallado de los requisitos y tareas a realizar durante el proyecto.

Sprint Planning. Planificación colaborativa entre el equipo y el cliente para establecer objetivos claros y alcanzables durante el sprint.

Sprint. Durante esta fase se realizaron las siguientes actividades y entregables:

- ***Definiciones y Requerimientos.*** Clarificación de los objetivos y especificaciones funcionales.
- ***Estructura de Tablas de la Base de Datos.*** Diseño de las tablas necesarias para el almacenamiento de datos.
- ***Esquema de Clases.*** Estructura lógica de las clases para el desarrollo del software.
- ***Librerías BackEnd***. Desarrollo y configuración de las librerías necesarias para el funcionamiento del servidor.
- ***Librerías FrontEnd.*** Desarrollo y configuración de las librerías para la interfaz de usuario.

- ***Código BackEnd.*** Implementación del código correspondiente al lado del servidor.
- ***Código FrontEnd.*** Desarrollo de la interfaz de usuario interactiva y funcional.
- ***Manual de Usuario.*** Documentación que detalla el uso del producto final para los usuarios finales.
- ***Entregable o Compilado.*** Producto funcional listo para revisión.

Sprint Review. Presentación y evaluación de los resultados obtenidos durante el sprint. Y retroalimentación del cliente para ajustes y mejoras continuas.

5.1.2 Definición de Política de Versionado

En este caso, se llevará un historial del versionado durante el primer Sprint, siendo esta una etapa de desarrollo inicial. Las entregas se controlarán utilizando la siguiente notación:

- Sprint 01: Versión 00.00.01
- Sprint 02: Versión 00.00.02
- Sprint 03: Versión 00.00.03
- Sprint 04: Versión 00.00.04
- Sprint 05: Versión 00.00.05
- Sprint 06: Versión 00.00.06
- Sprint 07: Versión 00.00.07
- Sprint 08: Versión 00.00.08

5.1.3 Resultados del Desarrollo de Funcionalidades

El desarrollo de la funcionalidad, con la optimización del ambiente de pruebas, permitió entregar un producto con calidad y dentro del tiempo pactado con el equipo y el cliente. Las funcionalidades entregadas incluyen los siguientes módulos:

Módulo de PDP (Producto Detalle por Página). Como cliente, necesito que en la página de e-commerce, en PDP, pueda ver el detalle del producto que he seleccionado para la compra del alimento para mi mascota a través de la suscripción.

Módulo de PLP (Producto Lista Página). Como cliente, necesito que en la página de e-commerce, en PLP, pueda ver los productos disponibles para la suscripción.

Módulo de Shipping. Como cliente, necesito que en la página de e-commerce, en Shipping, pueda seleccionar la dirección de envío para recibir el alimento de manera mensual.

Módulo de Billing & Finish. Como cliente, necesito que en la página de e-commerce, en Billing, pueda seleccionar la tarjeta para realizar la compra y programar el envío mensual del producto.

Módulo de Thank You Page. Como cliente, necesito que en la página de *Thank You* de e-commerce, al finalizar la compra, aparezca información detallada de la suscripción, la dirección de envío y el método de pago utilizado.

Módulo de Cancelación. Como cliente, necesito que en el menú de bienvenida exista un apartado de Suscripciones donde pueda consultar el estado de mis pedidos y cancelar mi suscripción.

Módulo de Cambio de Método de Pago. Como cliente, necesito que en el menú de bienvenida exista un apartado de Suscripciones donde pueda cambiar el método de pago de mi suscripción.

Módulo de Cambio de Dirección de Envío. Como cliente, necesito que en el menú de bienvenida exista un apartado de Suscripciones donde pueda seleccionar o cambiar la dirección de envío de mi suscripción.

Módulo de Reactivación. Como cliente, necesito que en el menú de bienvenida, en la sección *Mis Suscripciones*, pueda reactivar mi suscripción.

Módulo de Notificaciones. Como cliente, necesito recibir notificaciones que me mantengan informado sobre el estado de mi suscripción.

5.2 Logros Alcanzados

En este apartado se presentan los logros obtenidos tras la implementación de la optimización de pruebas de software, destacando los siguientes:

5.2.1 Mejora en el Proceso de Desarrollo, Pruebas y Entrega del Producto

- Se completaron exitosamente los sprints comprometidos para el desarrollo del producto.
- Los ambientes de prueba se optimizaron mediante la selección basada en criticidad y priorización de pruebas.
- La implementación de la metodología Scrum permitió trabajar y desarrollar la calidad en los ambientes correctos; y evitar la duplicidad de pruebas en entornos innecesarios.

5.2.2 Mejora en el Proceso de Versionado

- La generación de documentación versionada permitió un mejor seguimiento de los incrementos realizados en cada iteración.
- Este enfoque previno la pérdida de información importante, redujo el retrabajo y minimizó la documentación excesiva.

5.2.3 Mejora en la Disponibilidad de Información Analítica

Al finalizar cada fase de ejecución en los diferentes ambientes, se obtuvieron:

- **Métricas.** Indicadores clave que reflejan el rendimiento y los resultados del proceso.
- **KPI (Key Performance Indicators).** Medidas críticas para evaluar el progreso y los entregables.

- **Bitácoras.** Registros detallados que permiten un control más estable sobre los entregables y el avance del producto.

5.3 Dificultades Encontradas

5.3.1 Ausencia de Versionamiento en la Documentación

El control de versiones en la documentación es una herramienta esencial que permite registrar y gestionar las modificaciones realizadas en uno o más documentos a lo largo del tiempo. Este proceso es fundamental para individuos y equipos, ya que asegura la consistencia, precisión y responsabilidad en la gestión de la información.

Impactos de la Falta de Versionamiento

Duplicación de Esfuerzos. Sin un sistema de versionamiento adecuado, diferentes miembros del equipo pueden trabajar en versiones desactualizadas del mismo documento, generando redundancias y pérdida de tiempo.

Confusión entre Versiones. La coexistencia de múltiples versiones no organizadas puede generar malentendidos y errores en la implementación de tareas o decisiones.

Pérdida de Información Crítica. La ausencia de control de versiones aumenta el riesgo de eliminar o modificar accidentalmente datos importantes, lo que puede ser perjudicial para el proyecto.

Beneficios de Implementar un Sistema de Versionamiento

Colaboración Eficiente. Garantiza que todos los miembros del equipo trabajen sobre la versión más actualizada del documento, evitando conflictos y mejorando la productividad.

Protección de la Integridad del Trabajo. Al registrar cada cambio realizado, se puede restaurar rápidamente cualquier versión anterior en caso de errores o modificaciones no deseadas.

Prevención de Ineficiencias. Mantener un único documento actualizado continuamente ayuda a evitar confusiones y asegura que el proyecto fluya sin interrupciones.

Resguardo de Información. En entornos donde los documentos son editados y compartidos frecuentemente, el versionamiento actúa como una capa adicional de seguridad para los datos importantes.

5.3.2 Falta de Equipo de Calidad

La ausencia de un equipo dedicado a la calidad en el desarrollo de software puede tener graves consecuencias, entre ellas:

Pérdida de Valor Económico. Un software defectuoso disminuye la percepción de calidad y afecta negativamente el retorno de inversión.

Usuarios Insatisfechos. Los errores en el producto pueden generar frustración y desconfianza en los clientes.

Riesgos Graves. En casos críticos, un software de baja calidad puede poner en peligro la vida de las personas.

Las aplicaciones de software, creadas e implementadas por humanos, están naturalmente expuestas a errores debido a factores como:

Naturaleza Imperfecta. Es imposible anticipar todas las combinaciones de entradas en un software.

Presión por plazos. Los tiempos ajustados afectan la calidad del trabajo y el juicio de los desarrolladores.

Falta de experiencia. Los desarrolladores con poca experiencia pueden cometer errores involuntarios.

Complejidad del sistema. Los sistemas complicados dificultan la identificación y corrección de fallas.

Si estos errores no son detectados y solucionados, el producto final será defectuoso y no cumplirá con las expectativas del cliente.

5.3.3 Ausencia de Roles en el Equipo

La ausencia de ciertos roles esenciales dentro de un equipo, especialmente en las pruebas de software, puede generar obstáculos difíciles de superar. Los impactos de la Falta de Liderazgo son:

Pruebas de Software Inadecuadas. Sin un líder que gestione el proyecto, es difícil coordinar y optimizar las actividades de prueba.

Repercusiones en los Sistemas Digitales. La calidad de los sistemas se ve comprometida, afectando la operatividad y funcionalidad.

Impacto Económico. Los problemas de calidad repercuten directamente en la sostenibilidad económica de la organización.

Sin embargo, un liderazgo adecuado garantiza:

Coordinación Efectiva. Optimización de los recursos y actividades del equipo.

Calidad en los sistemas. Mejores procesos de pruebas y entrega de productos que cumplan con las expectativas del cliente.

Sostenibilidad Económica. Sistemas confiables que respalden el crecimiento y la estabilidad a largo plazo de la empresa.

5.3.4 Falta de Ceremonias

Las ceremonias ágiles son fundamentales para garantizar la mejora continua en los procesos y productos desarrollados por los equipos ágiles. Su implementación tiene un impacto positivo en diversas áreas:

Incremento de la Productividad. Al establecer objetivos claros para cada sprint, las ceremonias ágiles ayudan al equipo a trabajar de manera más enfocada y eficiente.

Comunicación Eficaz con el Cliente. Proveen un canal directo para resolver rápidamente dudas o ajustar detalles críticos según las necesidades del cliente.

Mayor Flexibilidad del Proyecto. Permiten a los clientes comunicar de forma proactiva sus preferencias o cambios en los requisitos, lo que facilita la adaptación a nuevas demandas o condiciones del entorno.

5.3.5 Ausencia de Herramientas para el Equipo de Calidad

La falta de herramientas adecuadas para el equipo de calidad puede tener efectos negativos significativos, entre ellos:

Reducción de la Productividad. Sin las herramientas necesarias, los miembros del equipo no pueden realizar sus tareas de manera eficiente.

Disminución de la Calidad del Trabajo. La ausencia de recursos esenciales dificulta la implementación de controles de calidad efectivos, lo que se traduce en un producto final de menor calidad.

Aumento del Riesgo para la Seguridad. Las limitaciones en herramientas y recursos pueden comprometer la seguridad de la organización, especialmente en sistemas críticos.

Desorganización del Equipo. La falta de soporte adecuado puede generar desorden y una disminución significativa en la colaboración y el trabajo en equipo.

5.4 Planteamiento de Mejoras

Como resultado de las dificultades identificadas anteriormente, se proponen metodologías, políticas y herramientas específicas para superarlas. Estas son las propuestas planteadas:

5.4.1 Metodologías Propuestas

Política de Versionamiento. El versionamiento es un método que permite llevar un control estructurado, bajo un esquema numérico, de las actualizaciones realizadas en documentos, despliegues e instalaciones. Este proceso facilita la identificación de los cambios efectuados a lo largo del tiempo. Se propone utilizar Confluence para gestionar las versiones obtenidas, asegurando un registro claro y accesible de las modificaciones realizadas.

Metodología Scrum. Scrum es un marco de trabajo ágil que implementa un conjunto de buenas prácticas destinadas a promover la colaboración en equipo y alcanzar los mejores resultados posibles en un proyecto. Se caracteriza por obtener entregas parciales y periódicas del producto final; se priorizan aquellas entregas que aporten mayor beneficio al cliente o destinatario del proyecto.

Es especialmente adecuado para entornos complejos, donde los requisitos son cambiantes o poco claros, Además, fomenta la innovación, competitividad, flexibilidad y productividad; permite obtener resultados rápidamente mediante la implementación de ciclos iterativos y adaptativos.

Scrum es ideal para proyectos en los que se requiere adaptarse a nuevos desafíos, con equipos altamente productivos que trabajan de manera colaborativa y constante para alcanzar los objetivos planteados.

5.4.2 Descripción de la Implementación

La implementación de la metodología Scrum se desarrolla en cinco pasos clave que garantizan la organización, comunicación y mejora continua del equipo. Estos pasos son:

A. Refinamiento del Backlog. Este proceso permite a todos los miembros del equipo un espacio para compartir iniciativas, propuestas y preocupaciones. Se detallan las tareas

pendientes, se priorizan según su impacto y se asegura que todos comprendan los objetivos del proyecto.

B. Planificación de las Fases. Cada iteración comienza con una reunión de planificación en la que el Product Owner discute las prioridades con el equipo. Las ideas y propuestas se transforman en tareas específicas, organizadas por orden de importancia y factibilidad.

C. Scrum Diario. Se realiza una reunión diaria para revisar el progreso del proyecto, fortalecer la comunicación dentro del equipo y asegurar que todos estén alineados con los objetivos. Durante esta reunión, cada miembro comparte los avances realizados desde el último encuentro; expone las tareas planeadas para el día siguiente e identifica posibles obstáculos y se plantean soluciones colaborativas.

D. Reunión de Revisión de Etapa. Al concluir cada fase, el equipo presenta su trabajo al Product Owner. El Product Owner evalúa si los resultados cumplen con las expectativas y decide si se aceptan o se requieren ajustes.

E. Reunión Retrospectiva de Fase. Se analiza el desempeño del equipo, destacando fortalezas y áreas de mejora. En esta reunión participan todos los integrantes del equipo, incluyendo el Product Owner y el Scrum Master, para: revisar lo que funcionó correctamente; identificar las tareas pendientes o no completadas; proponer estrategias y conclusiones para optimizar el rendimiento en la siguiente etapa.

5.5 Análisis

Cuando se creó el área de calidad en la empresa Globant, no se contaba con un equipo completo ni con una guía establecida sobre los principios del testing. Esto resultó, de manera involuntaria, en la formación de un cuello de botella que dificultaba garantizar que el

desarrollo cumpliera con la calidad del producto adecuada. La optimización de los ambientes de prueba en la calidad del software es fundamental por los siguientes motivos:

Eficiencia en la Detección de Errores

Un ambiente de prueba bien optimizado permite identificar y corregir errores de manera más rápida y efectiva, reduciendo significativamente el tiempo total de desarrollo.

Reproducción de Condiciones Reales

Al simular de manera precisa el entorno de producción, se pueden identificar problemas que, de otro modo, pasarían desapercibidos en ambientes de prueba inadecuados.

Mejora en la Cobertura de Pruebas

Un ambiente optimizado facilita la ejecución de una variedad más amplia de pruebas, asegurando que diferentes aspectos del software sean evaluados y que los fallos potenciales sean abordados a tiempo.

Reducción de Costos

Detectar errores en etapas tempranas del desarrollo es considerablemente menos costoso que corregirlos una vez que el software ha sido lanzado.

Aumento de la Confianza

Un ambiente de prueba bien configurado brinda mayor confianza en la calidad del software, lo que asegura a los equipos y clientes que el producto cumplirá con los requisitos establecidos.

Facilitación de la Colaboración

Ambientes de prueba optimizados promueven una colaboración más efectiva entre desarrolladores y testers, mejorando tanto la comunicación como la comprensión de los requerimientos.

Flexibilidad ante Cambios

Un entorno bien diseñado puede adaptarse con facilidad a modificaciones en los requisitos o en las tecnologías utilizadas, lo cual es esencial en el contexto de un desarrollo ágil.

En conclusión, la optimización de los ambientes de prueba no solo mejora la eficiencia y efectividad del desarrollo, sino que también contribuye a garantizar un proceso más ágil y una mayor calidad del producto, beneficiando tanto al equipo de desarrollo como a los usuarios finales.

5.6 Aporte del Bachiller en la Empresa y/o Institución

La implementación de la metodología Scrum en los procesos de pruebas de software en la empresa Globant constituye un aporte significativo, ya que esta metodología ha permitido:

Agilidad y Transparencia en los Procesos

La incorporación de todas las ceremonias de Scrum ha optimizado la forma de trabajo, aumentando la agilidad y mejorando la interacción en cada célula del proyecto.

Transformación Cultural en la Organización

La eliminación de jerarquías internas ha fomentado una comunicación más fluida y efectiva entre los miembros del equipo, eliminando barreras para la interacción y colaboración.

Facilitación de la Adopción de Nuevas Metodologías

Este cambio cultural en el equipo, especialmente en el área de Investigación y Desarrollo, prepara a la empresa para adoptar nuevas metodologías, políticas y herramientas tecnológicas en el futuro.

Optimización del Versionamiento

La implementación de un enfoque de desarrollo paralelo basado en ramas ha acelerado la liberación de productos en ciclos bimensuales. Este modelo organiza el empaquetado y la liberación de versiones para los usuarios finales, garantizando un proceso más eficiente y ordenado.

En resumen, el aporte del bachiller ha sido clave para impulsar una transformación organizacional y técnica, que ha mejorado significativamente la calidad y eficiencia de los procesos de desarrollo en la empresa.

REFERENCIAS BIBLIOGRÁFICAS

Coppola, M. (19 de enero de 2023). Qué es un plugin, para qué sirve y cómo instalarlo. *Blog Hubspot.* https://blog.hubspot.es/website/que-es-plugin

Gomstyn, A. y Jonker, A. (29 de diciembre de 2023). ¿Qué es la gestión de pruebas? *Blog IBM.* https://www.ibm.com/es-es/topics/test-management

IONOS Digital Guide. (4 de noviembre de 2022). ¿Qué es Jira? Todo lo que necesitas saber sobre el software de gestión de proyectos. *Blog IONOS.* https://www.ionos.com/es-us/digitalguide/paginas-web/desarrollo-web/que-es-jira/

Lupita. (26 de noviembre de 2021). ¿Qué es el testing de software y por qué es tan importante? *Blog Programa en Línea.* https://www.programaenlinea.net/que-es-el-testing-de-software-y-por-que-es-tan-importante/

Prasad, D. (14 de mayo de 2024). Pruebas de aceptación del usuario (UAT): ¿Qué es? *Blog Geekflare.* https://geekflare.com/es/user-acceptance-testing/

QAwerk. (20 de octubre 2021). Cómo redactar casos de prueba: Guía completa de Qawerk. *Blog QAwerk.* https://qawerk.es/blog/como-redactar-casos-de-prueba/

Rodríguez, S. (18 de setiembre de 2023). ¿Qué es el refinamiento del Backlog? *Blog Scrumio.* https://www.scrumio.com/blog/refinamiento-backlog

Testeando Software. (14 de enero de 2014). ISTQB. ¿Qué es? ¿Cuales son los niveles de certificación? *Blog Testeando Software.* https://testeandosoftware.com/istqb-que-es-cuales-son-los-niveles-de-certificacion/

Toledo, F. (7 de marzo de 2022). Estrategia de pruebas de software: ¿Cómo crear la adecuada para tu proyecto? *Blog Abstracta.* https://es.abstracta.us/blog/guia-crear-estrategia-pruebas-software-adecuada/

Walker, R. (24 de enero de 2023). Análisis de requisitos de software. *Blog AppMaster.* https://appmaster.io/es/blog/analisis-de-requisitos-de-software

Zap Chernyak, A. (29 de setiembre de 2023). ¿Qué es la automatización de pruebas? Una guía sencilla y sin jerga. *Blog ZapTest.* https://www.zaptest.com/es/que-es-la-automatizacion-de-pruebas-una-guia-sencilla-y-sin-jerga