|  |  |
| --- | --- |
| PROGRAMA DE FORMACIÓN | Tecnólogo en implementación y gestión de bases de datos |

|  |  |  |  |
| --- | --- | --- | --- |
| COMPETENCIA | 220501099 - Probar la solución del *software* de acuerdo con parámetros técnicos y modelos de referencia. | RESULTADOS DE APRENDIZAJE | 220501099 -01- Planificar pruebas de servicios de bases de datos de acuerdo a parámetros técnicos y modelos de referencia.  220501099 -02 Ejecutar pruebas a los servicios de bases de datos para verificar su funcionalidad. |

|  |  |
| --- | --- |
| NÚMERO DEL COMPONENTE FORMATIVO | CF20 |
| NOMBRE DEL COMPONENTE FORMATIVO | Diseño y ejecución de pruebas |
| BREVE DESCRIPCIÓN | El componente aborda el concepto de pruebas de forma general, el tipo de prueba que se pueden realizar a las bases de datos, qué herramientas tecnológicas existen en la actualidad que permiten realizar dichas pruebas. Además, se construye un plan de pruebas que a su vez contiene diferentes casos de prueba diseñados para su respectiva ejecución, logrando una documentación mediante ciertos estándares descritos. |
| PALABRAS CLAVE | Pruebas funcionales, plan de pruebas, TDD, modelos, *script*, entornos |
| ÁREA OCUPACIONAL | Ciencias naturales, aplicadas y relacionadas. |
| IDIOMA | Español |

# **Tabla de contenidos**

**Introducción**

**1.** **Pruebas**

**2. Herramientas de prueba**

**3. Modelos de referencia para pruebas**

**4. Casos de prueba**

**5. Plan de pruebas**

**6. Ejecución de pruebas**

6.1. Creación entorno

6.2. Desarrollo guiado por pruebas (TDD)

6.3. Codificación de casos

6.4. Documentación de pruebas

**Introducción**

|  |
| --- |
| Cuadro de texto |
| Estimado aprendiz, a través del presente componente formativo conocerá sobre las pruebas, herramientas de las mismas y modelos de referencia que le aportan al diseño y a la construcción del plan de pruebas que se le aplica a una base de datos mediante la implementación de herramientas tecnológicas definidas en los casos de este tipo.  Lo invitamos a ver el siguiente video para documentarse sobre cada una de las temáticas comprendidas.  ¡Muchos éxitos en este proceso de aprendizaje! |

**Guion de video introductorio**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tipo de recurso** | Video spot animado | | | |
| **NOTA** | **La totalidad del texto locutado para el video no debe superar las 500 palabras aproximadamente** | | | |
| **Título** | Introducción diseño y ejecución de pruebas | | | |
| **Escena** | **Imagen** | **Sonido** | **Narración** | **Texto** |
| **1** | Presentar una imagen de una empresa en la era digital  Imagen imaginativa de los empresarios y del personal de las firmas financieras. Concepto de recursos humanos, planificación de los recursos institucionales y tecnología digital.  Imagen de referente  <https://image.shutterstock.com/image-photo/imaginative-visual-business-people-financial-600w-1903039975.jpg>  **Imagen** 228130\_i1  Seguidamente las imágenes que se presentan:  Empresario que utiliza un ordenador para el concepto de administración de documentos, base de datos de documentación en línea y sistema o software de almacenamiento de archivos digitales, mantenimiento de registros, tecnología de bases de datos, acceso a archivos, uso compartido de documentos.  Imagen de referente  <https://image.shutterstock.com/image-photo/businessman-using-computer-document-management-600w-2016196394.jpg>  **Imagen** 228130\_i2      **Imagen** de referente  <https://image.shutterstock.com/image-vector/customer-research-marketing-study-user-600w-2170950639.jpg> **Imagen** 228130\_i3 | N/A | El almacenamiento de información es importante en las empresas ya integradas en la era digital, debido a que se dispone de una cantidad significativa de datos. Los especialistas en análisis de datos elaboran estrategias que pueden ayudar a identificar mejor las necesidades de clientes y prever riesgos o el comportamiento del mercado. Es por esto que las organizaciones saben el grado de relevancia de gestionar la información de sus clientes y del mercado. | Almacenamiento de información  Era digital  Estrategias  Comportamiento del mercado |
| **2** | Presentar imágenes de gran cantidad de información  Trabajo de oficina no organizado. Papel contable pilas de documentos, desorden en los archivos de la tabla contable. Papel rutinario vectorial escritorio de trabajo imprimir hojas desordenadas mucha persona, concepto sobrecargado  Imagen de referente  <https://image.shutterstock.com/image-vector/unorganized-office-work-accounting-paper-600w-1458923129.jpg>  **Imagen** 228130\_i4  Proceso de certificación y normalización, garantía de calidad y conformidad con las normas internacionales. Concepto con iconos para el certificado, aprobación, cumplimiento de los requisitos. Persona que utiliza el equipo  Imagen de referente  <https://image.shutterstock.com/image-photo/certification-standardization-process-quality-assurance-600w-2102219899.jpg>  **Imagen** 228130\_i5 | N/A | Al aparecer nuevas necesidades la organización debe estar preparada, ya que se hace referencia a la administración de enormes volúmenes de datos, la gran mayoría de esfuerzos se enfocan en procesar información con características de gran volumen, dejando de lado otros aspectos de los datos que, en el caso de las bases de datos tradicionales, se concentran más en la integridad de los mismos. Por otra parte, se requiere la implementación de pruebas a la base de datos para validar su perfecto funcionamiento. | Necesidades  Volumen de datos  Implementación de pruebas |
| **3** | En la medida que sale el texto visualizar las imágenes que se presentan  Desarrollo de bases de datos, concepto de administración. Los ingenieros de software trabajan con big data. Ordenadores que desarrollan sistemas con hojas de cálculo, tablas. Ilustración vectorial gráfica plana aislada en fondo blanco  Imagen de referente:  [https://image.shutterstock.com/image-vector/database-development-administration-concept-*software*-600w-2169430401.jpg](https://image.shutterstock.com/image-vector/database-development-administration-concept-software-600w-2169430401.jpg)  **Imagen** 228130\_i6  empresario que utiliza la interfaz de tecnología  Imagen de referente  <https://image.shutterstock.com/image-photo/businessman-using-technology-interface-600w-310012835.jpg>  **Imagen** 228130\_i7  Proceso de certificación y normalización, negocio certificado por la iso, conformidad con las normas internacionales y concepto de garantía de calidad. Persona tocando el icono del certificado.  Imagen de referente  <https://image.shutterstock.com/image-photo/certification-standardization-process-iso-certified-600w-2103334220.jpg>  **Imagen** 228130\_i8 | N/A | En este componente, se abordan varios autores conocidos en pruebas de bases de datos, de quienes se han citado y referenciado conceptos y ejemplos para los fines educativos, en el entendido que el conocimiento es social y, por lo tanto, es para usarlo por quienes necesitan adquirirlo. Además, contiene temas cruciales refiriéndose a la construcción del plan de pruebas, es decir, cómo está compuesto el plan correspondiente detallando cada uno de sus componentes. Por otra parte, es pertinente desarrollar los conocimientos en las herramientas tecnológicas que permiten la implementación de pruebas en las bases de datos; actualmente varias de las pruebas que se le aplican a un determinado *software* se pueden personalizar para implementarlas en la validación de una base de datos. | Construcción del plan de pruebas.  Herramientas tecnológicas  Implementación de pruebas |
| **4** | En la medida que sale el texto visualizar las imágenes que se presentan  Concepto de pruebas de aplicaciones de software - Proceso de pruebas de aplicaciones de software con herramientas y marcos de automatización para identificar errores - Ilustración 3D  Imagen de referente  [https://image.shutterstock.com/image-illustration/*software*-application-testing-concept-process-600w-2181263961.jpg](https://image.shutterstock.com/image-illustration/software-application-testing-concept-process-600w-2181263961.jpg)  **Imagen** 228130\_i9  Resolución de problemas Icono de cristal y engranajes de aumento de símbolos en el fondo moderno Ilustración de vectores de diseño plano  Imagen de referente  <https://image.shutterstock.com/image-vector/troubleshooting-symbol-magnifying-glass-gears-600w-198791339.jpg>  Imagen 228130\_i10  concepto de infraestructura de Internet. Resumen de los antecedentes tecnológicos. Representación 3D.  Imagen de referente  <https://image.shutterstock.com/image-illustration/cloud-architecture-platform-internet-infrastructure-600w-1748437547.jpg>  **Imagen** 228130\_i11 | N/A | El tipo de pruebas que se realizan a una base de datos está comprendido en tres grandes grupos, las pruebas funcionales que se centran en el comportamiento del sistema, o cierta parte del componente del sistema, descrito en las especificaciones de los requerimientos, las pruebas no funcionales que incluyen las pruebas rendimiento, carga, estrés, usabilidad, mantenibilidad, fiabilidad o portabilidad, entre otras; por ende, se centran en características del *software* que establecen “cómo trabaja el sistema”. Por último, las pruebas estructurales que se pueden ejecutar en todos los niveles de las pruebas, y son especialmente efectivas para medir estructura o arquitectura de la base de datos. | Tipo de pruebas  Pruebas funcionales  Pruebas no funcionales  Pruebas estructurales |
| **5** | Presentar las imágenes en la medida en que se expone la descripción.  Tecnología de desarrollo de aplicaciones, depuración y prueba de código de software para errores y fallos de búsqueda. Concepto de datos de programación, análisis y codificación. Ilustración vectorial.  Imagen de referente:  [https://image.shutterstock.com/image-vector/*software*-developer-freelancer-coding-this-600w-343443329.jpg](https://image.shutterstock.com/image-vector/software-developer-freelancer-coding-this-600w-343443329.jpg)  **Imagen** 228130\_i12  Desarrollador de software o programador independiente es Codificación. Esto también representa los requisitos de recopilación de analistas de negocios, código de prueba de probadores, ingeniero que proporciona soporte de aplicaciones, ilustración vectorial, conjunto de iconos.  Imagen de referente:  <https://image.shutterstock.com/image-illustration/update-concept-neon-word-lights-600w-2158360563.jpg>  Imagen 228130\_i13  Joven empresario en concepto de cloud computing  Imagen de referente:  <https://image.shutterstock.com/image-photo/young-businessman-cloud-computing-concept-600w-533935501.jpg>  **Imagen** 228130\_i14 | N/A | En conclusión, una de las tareas más importantes es el plan de pruebas de las bases de datos, que abarca desde el modelamiento de los casos de prueba, hasta la definición de las estructuras del plan de pruebas, para lo cual se debe tener total claridad y entero conocimiento de los requerimientos del sistema, usabilidad de la información y de los recursos e infraestructura con la que cuenta la organización tanto para el almacenamiento como para su procesamiento.  Se espera que este componente formativo sea útil para todos, aprendices y lectores en general, que estén interesados en acercarse a los conceptos básicos de la programación de aplicaciones y gestión de pruebas de bases de datos, por medio de herramientas manuales y herramientas tecnológicas. | Plan de pruebas  Recursos e infraestructura |
| **Nombre del archivo** | 228130\_v1 | | | |

**Desarrollo de contenidos**

1. **Pruebas**

|  |
| --- |
| Cuadro de texto |
| Las pruebas de base de datos son aquellas que se realizan para comprobar los diferentes métodos y procesos implementados en la gestión de la base de datos, para asegurarse que cumplan las diferentes reglas de los datos como se espera, y durante la manipulación de los mismos, que aquellos no difieran, o sean borrados, creados o modificados de manera inesperada. También las pruebas abordan el concepto de calidad de los servicios en cuanto a la efectividad de los datos y la concurrencia de los usuarios a dicha bases.  Además, las pruebas son una técnica de detención de defectos que se caracteriza por construir fallas o errores de manera planificada. El objetivo es detectar fallas antes del paso a producción. |

**Diseño de casos de prueba**

|  |  |
| --- | --- |
| Tipo de recurso | Cajón de texto de color |
| El diseño de casos de prueba es una parte fundamental en las diferentes pruebas por componentes y sistemas de bases de datos, en donde se diseñan casos de prueba (entradas y salidas esperadas) para la comprobación del sistema de BD.  El objetivo crucial en el diseño de casos de prueba es construir un conjunto de ellos que sea eficiente, descubriendo posibles defectos en los programas y como resultado final se demuestra que el sistema de bases de datos cumple o satisface los requerimientos (Blanquicett, et al. 2018).  Cómo se diseña un caso de prueba:   * Selecciona una característica de la base de datos o el componente que se desea probar. * Seleccionar un conjunto de entradas que ejecutarán las características pertinentes. * Documentar las salidas obtenidas o los diferentes rangos de salida.   Desarrollo de Negocios Virtuales: 1.3 Base de Datos de un sitio Web  Imagen de referente: <http://negociosvirtualesit.blogspot.com/2015/10/base-de-datos-de-un-sitio-web.html>  **Imagen** 228130\_i15 | |

|  |  |  |
| --- | --- | --- |
| **Tipo de recurso** | Infografía interactiva Punto caliente | |
| **Texto introductorio** | En la actualidad existen una gran variedad de aproximaciones que sirven como guía para generar diseños de casos de prueba: pruebas basadas en requerimientos, pruebas de particiones y pruebas estructurales. | |
|  | | |
| **Código de la imagen** | **Imagen** 228130\_i16 | |
| **Punto caliente 1** | Se diseñan puntualmente para probar los requerimientos del sistema; esta aproximación generalmente se implementa en la etapa de pruebas del sistema.  Para cada requerimiento registrado, se identifican diferentes casos de prueba que demuestran que el sistema satisface dicho requerimiento.   * Demostrar que el sistema implementó adecuadamente sus requerimientos. * Probar atributos no funcionales como la fiabilidad o la usabilidad. | **Imagen** 228130\_i17 |
| **Punto caliente 2** | Este tipo de pruebas consiste en identificar **particiones** de entradas y salidas, y se diseñan las respectivas pruebas para que el sistema ejecute las entradas correspondientes a todas esas particiones y genere sus salidas.  Para tener en cuenta: las particiones son datos agrupados según características comunes.  Una buena práctica para seleccionar caso de pruebas consiste en la elección de casos de pruebas en los límites de las particiones junto con casos de pruebas cercanos al punto medio de las particiones, se eligen de acuerdo con las especificaciones del programa. | **Imagen** 228130\_i18 |
| **Punto caliente 3** | Conocer la estructura de la base de datos para diseñar las respectivas pruebas con el objetivo de ejecutar todas las partes de la base de datos.  Este tipo de aproximaciones son denominadas pruebas de caja blanca. | **Imagen** 228130\_i19 |

**Herramientas tecnológicas para pruebas**

|  |
| --- |
| Cuadro de texto |
| En la actualidad existen muchas herramientas dedicadas a testear (probar) determinados aspectos de un sistema, las cuales se clasifican según el objetivo en que se centran para dichas pruebas.  **Características**   * + Capacidad de admitir el marco de pruebas.   + Control de revisión.   + Gestión de configuración de prueba.   + Seguimiento de problemas.   + Generación de informes.   + Capacidad de configurar formatos de informes. |

**Tipo de herramientas tecnológicas para pruebas**

|  |  |  |
| --- | --- | --- |
| **Tipo de recurso** | Pestañas o tabs horizontales | |
| **Introducción** | Dentro de las herramientas tecnológicas para prueba se encuentran algunos tipos como: estructural, funcional y no funcional. A continuación, se presenta información sobre estos. | |
| **Estructural** | El tipo de pruebas estructurales se puede ejecutar en todos los niveles de las pruebas, siendo especialmente efectivo para medir estructura o arquitectura de la base de datos. Para expresar el alcance como un conjunto de pruebas se emplea el concepto de cobertura.  Es habitual en muchos casos, al implementar este tipo de pruebas estructurales, acompañarlo con herramientas de apoyo para validar la cobertura del código, como es el caso de pruebas de integración o pruebas de componentes, este tipo son llamadas también pruebas de caja blanca. | Imagen de referencia  [https://www.panel.es/*software*-qa-cuales-son-los-tipos-de-pruebas-*software*/](https://www.panel.es/software-qa-cuales-son-los-tipos-de-pruebas-software/)  **Imagen:** 228130\_i20 |
| **Funcional** | Se centran en el comportamiento del sistema, o cierta parte del componente del sistema, descrito en las especificaciones de los requerimientos.  Por ende, se definen a partir de funciones o características y su respectiva interoperabilidad con diferentes partes del sistema, lográndose ejecutar en todos los niveles de pruebas integración, componentes, sistemas, etc.  Se consideran pruebas de caja negra (“*black-box testing*”) puesto que se valora el comportamiento externo del sistema. Las pruebas de seguridad o las pruebas de interoperabilidad entre sistemas o componentes son casos especializados de las pruebas funcionales (Ramos et al 2019). | Imagen de referencia  [https://www.panel.es/*software*-qa-cuales-son-los-tipos-de-pruebas-*software*/](https://www.panel.es/software-qa-cuales-son-los-tipos-de-pruebas-software/)  **Imagen:** 228130\_i21 |
| **No funcional** | Las pruebas no funcionales incluyen las de rendimiento, carga, estrés, usabilidad, mantenibilidad, fiabilidad o portabilidad, entre otras. Por ende, se centran en características del *software* que establecen “cómo trabaja el sistema” (Reyes et al, 2020).  También se pueden ejecutar en todos los niveles de pruebas, se puede testear de diversas formas, por ejemplo, por tiempos de respuesta en el caso de pruebas de rendimiento o por número máximo de clientes conectados en pruebas de estrés. | Pruebas no funcionales  Imagen de referencia  <https://static.wixstatic.com/media/a59c1e_37d972411a124358b094baa271b49c32~mv2.png/v1/fit/w_971%2Ch_605%2Cal_c/file.png>  **Imagen:** 228130\_i22 |

1. **Herramientas de pruebas**

|  |
| --- |
| Cuadro de texto |
| Las herramientas implementadas en pruebas del sistema y bases de datos automatizadas, aportan un gran beneficio a la ejecución de pruebas funcionales y de regresión en el respectivo sistema. Estas deben aportar unos resultados significativos y consistentes respecto a los datos de entrada proporcionados.  **Características:**   * La prueba de base de datos implica la prueba de diferentes capas, como la capa de interfaz de usuario, la capa de acceso, la capa empresarial y finalmente la base de datos. * La prueba de base de datos se utiliza para verificar la integridad de los datos, el mapeo de datos y las propiedades ACID (*Atomicity, Consistency, Isolation y Durability*) de la base de datos. * Los resultados obtenidos en determinada herramienta son fundamentales como guía para realizar cambios o como base de documentación de las funcionalidades de la base de datos. * Cada una de las herramientas tiene su objetivo determinado en pro de generar los mejores resultados de las pruebas a realizar. * Las herramientas disponen de un manual técnico para realizar detalladamente las pruebas, desde su página oficial. |

**Tipos**

|  |  |
| --- | --- |
| **Tipo de recurso** | Rutas / Pasos. Verticales 1 |
| **Introducción** | Las herramientas de prueba se pueden clasificar en los siguientes tipos: |
| Imagen de referencia: <https://image.shutterstock.com/image-vector/browser-based-cache-vector-icon-600w-2087619277.jpg>  **Imagen:** 228130\_i23 | |
| **Botón 1** | **Pruebas unitarias**  Son pruebas de muy bajo nivel y son ejecutadas cercar de la fuente de la aplicación. Su funcionamiento consiste en probar métodos y funciones de forma individual, y los diferentes módulos o componentes del sistema. Este tipo de pruebas son en realidad muy económicas al momento de automatizarlas y se puede ejecutar mediante un servidor de integración continua. |
| **Botón 2** | **Pruebas de integración**  Tienen por objetivo verificar que los diferentes módulos y componentes funcionan bien en conjunto. Este tipo de pruebas son un poco más costosas al momento de automatizarlas, ya que se requiere que todos los componentes marchen a la perfección. |
| **Botón 3** | **Pruebas funcionales**  Se centran en los requerimientos del sistema, solo verifican el resultado de una acción y no comprueban los diferentes estados intermedios para realizar dicha acción. |
| **Botón 4** | **Pruebas integrales**  Se encargan de replicar el comportamiento de usuario con el sistema. Estas pruebas son muy útiles para realizar seguimiento y test a las bases de datos, pero resultan muy difíciles de mantener y muy costosas de llevar a cabo cuando se encuentran automatizadas. |
| **Botón 5** | **Pruebas de aceptación**  Son pruebas formales que verifican si el sistema en cuestión satisface los requerimientos empresariales. Es requisito fundamental que se esté ejecutando la aplicación durante las respectivas pruebas y se centren en replicar el comportamiento o conductas de los usuarios. También se puede medir el rendimiento del sistema y rechazar o aceptar posibles cambios según los objetivos trazados. |
| **Botón 6** | **Pruebas de rendimiento**  Son las encargadas de evaluar el rendimiento de un sistema con una carga de trabajo determinado. Ayudan a medir la fiabilidad, la velocidad, la escalabilidad y la capacidad de respuesta de una aplicación. |
| **Botón 7** | **Pruebas de humo**  Las pruebas de humo son una clase de pruebas básicas que sirven para evaluar el funcionamiento básico del sistema. Son pruebas para ejecutar de forma rápida y su objetivo primordial es garantizar que las principales funciones del sistema operen de forma eficiente y según lo previsto. |
| **Botón 8** | **Pruebas de comunicación**  Son las encargadas de medir si hay comunicación con los diferentes dispositivos *hardware*, con la respectiva base de datos. |
| **Botón 9** | **Pruebas de volumen**  Simula la carga de muchos datos, para la medición del desempeño de la base de datos frente a una carga considerable de datos a ser procesados. |
| **Botón 10** | **Pruebas de sobrecarga**  Estas pruebas se utilizan para calcular la capacidad límite dándole cargas excesivas y diferentes peticiones, muchas veces con el objetivo de crear cuellos de botella para que determine comportamientos. |
| **Botón 11** | **Prueba de disponibilidad de datos**  Se enfoca en la respuesta de los daros desde la base de datos, con el objetivo de realizar la medición o posibles fallos inesperados. |
| **Botón 12** | **Pruebas de facilidad de uso**  Eevalúa la adaptación por parte de los usuarios en cuanto a ingreso y presentación de los datos, si se familiarizan con la forma de trabajo habitual. |
| **Botón 13** | **Pruebas de operación**  Consiste en la validación del funcionamiento del sistema con sus respectivos componentes, procedimientos de operación, incluyendo su respectivo arranque y disposición de la base de datos. |
| **Botón 14** | **Pruebas de entorno**  Se encargan de la validación de cómo interacciona el sistema respecto a otros sistemas. |

|  |
| --- |
| Cuadro de texto |
| Para conocer sobre herramientas automatizadas que realizan pruebas a las bases de datos, se invita a consultar el documento anexo. |

**Instalación y ejecución de pruebas de rendimiento de una base de datos *MySQL***

|  |
| --- |
| Cuadro de texto |
| Instalación y ejecución de pruebas de rendimiento de una base de datos *MySQL* por medio de una herramienta tecnológica llamada *Jmeter*, donde se puede revisar y exportar un resumen del resultado de las pruebas y exportar, además, un reporte gráfico donde se evidencian todos los ítems que se tienen en cuenta para la validación del rendimiento de una base de datos.    **Video tutorial Instalación y ejecución de pruebas de rendimiento de una base de datos MySQL por medio de una herramienta tecnológica llamada *Jmeter.***  **CF20** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tipo de recurso** | Video clase/tutorial | | | |
| **NOTA** | **La totalidad del texto locutado para el video no debe superar las 500 palabras aproximadamente** | | | |
| **Título** | Instalación y ejecución de pruebas de rendimiento de una base de datos *MySQL* por medio de una herramienta tecnológica llamada *Jmeter*. | | | |
| **Escena** | **Imagen** | **Sonido** | **Narración** | **Texto** |
|  |  | N/A |  | Ya está grabado por el experto, ubicado en la carpeta anexos, 228130\_v2.mp4 |
| **Nombre del archivo** | **228130\_v2** | | | |
| **Datos del narrador** | Joaquín Fernando Aguilar Camacho | | | |

**3. Modelos de referencia para pruebas**

|  |
| --- |
| Cuadro de texto |
| Los modelos de referencia para pruebas son una base fundamental para tener como guía al momento de empezar a construir un plan de pruebas, permite visualizar la metodología a implementar y cómo está compuesto cada uno de sus artefactos con el objetivo de llevar un control en cada una de las etapas y sus correspondientes documentos diligenciados como productos y/o entregables. |

**Pruebas de *software* basadas en modelos**

|  |  |  |
| --- | --- | --- |
| **Tipo de recurso** | Infografía interactiva Punto caliente | |
| **Texto introductorio** | A continuación, se presenta información sobre las pruebas de *software* como concepto, características, ventajas y modelos.  **Pruebas de *software* basadas en modelos** | |
| Concepto  Características  Ventajas  Modelos | | |
| **Código de la imagen** | **Imagen** 228130\_i24 | |
| **Punto caliente 1** | Este método de pruebas tiene como objetivo generar casos ejecutables basándose en el modelo y diseño de alto nivel del sistema. | Imagen que contiene Diagrama  Descripción generada automáticamente  **Imagen** 228130\_i25 |
| **Punto caliente 2** | 1. La construcción de un modelo donde se documenten los aspectos más relevantes del comportamiento del sistema.  2. Generación de casos de pruebas a partir del modelo. Estas pruebas que se van a planificar son de alto nivel, representan la parte lógica de los casos de prueba y no especifican si están descritas en un lenguaje de programación.  3. Corroboración de los casos de prueba. Las pruebas de alto nivel se concretizan por medio de un código o script en alguna tecnología elegida para que sean ejecutadas dichos casos de pruebas.  4. La ejecución de los casos de pruebas sobre el sistema que se está evaluando.  5. La obtención del reporte de los resultados producto de la ejecución de los casos de pruebas  6. Analizar los resultados obtenidos. | Diagrama  Descripción generada automáticamente con confianza baja  **Imagen** 228130\_i26 |
| **Punto caliente 3** | * Ubicación de errores en el sistema que se está evaluando. * Reducción en los tiempos del proceso y costos en la gestión de pruebas. * Mejoramiento de la calidad de las pruebas a ejecutar. * Detección de anomalías en ciertos requerimientos. * Mejoramiento en la trazabilidad. * Requerimientos actualizados y evolucionados. | Diagrama  Descripción generada automáticamente  **Imagen** 228130\_i27 |
| **Punto caliente 4** | Modelos más implementados:   * Modelo de *McCall* * Modelo de *Boehm* * Modelo de *Gilb* * Modelo ISO 9126 | Imagen que contiene Diagrama  Descripción generada automáticamente  **Imagen** 228130\_i28 |

**Modelos para realizar las pruebas de *software***

|  |  |  |
| --- | --- | --- |
| **Tipo de recurso** | Slider Presentación | |
| **Introducción** | Los modelos más reconocidos para realizar las pruebas de *software* son los siguientes: | |
| **Modelo *McCall***  El modelo *McCall* se enfoca en la opinión de los usuarios y desarrolladores en tres puntos de vistas cruciales para la medición de calidad del sistema:   * Operación * Revisión * Transición   Cada uno de esos parámetros se subdividen en diferentes características que son de gran ayuda para determinar la respectiva validación. | | **Figura 1**  *Modelo McCall*    **Imagen:** 228130\_i29 |
| **Modelo *Boehm***  El modelo de *Boehm* es definido por un ciclo de vida del tipo espiral, creado por Barry Boehm en el año de 1988, donde incluye el análisis de riesgos. Las cuatro (4) fases primordiales en este modelo son:   * Planificación * Análisis de riesgo * Ingeniería * Evaluación del cliente   Las versiones del sistema a evaluar están dirigidas por el número de iteraciones que va a ser representada en forma de espiral de adentro hacia a fuera, determinando el riesgo y la posible solución. | | **Figura 2**  Modelo Boehm    *Nota (*Wikipedia*, 2018).* Fuente:  <https://es.m.wikipedia.org/wiki/Archivo:Modelo_Espiral_Boehm.svg>  **Imagen:** 228130\_i30 |
| **Paradigma *GQM* (*Goal-Question-Metric*)**  Este modelo se enfoca en los componentes de mejora en los procesos, creado por Basili y Rombaches por medio de un paradigma que se requiere evaluar mediante preguntas y establecer el cumplimiento de los objetivos.  Los puntos relevantes en este modelo son:  **Metas:** que son los objetivos, las intenciones específicas y generales que va a tener dicha organización.  **Preguntas:** se establecen los cuestionarios para cada uno de los objetivos con su correspondiente evaluación.  **Métricas:** son la medición con base en las respuestas de las preguntas; cabe indicar que se realiza un análisis de si esa métrica ayudó al proceso cumpliendo el objetivo deseado. | | Imagen de referencia  <https://image.shutterstock.com/image-photo/goal-question-metric-business-concept-600w-1337433434.jpg>  **Imagen:** 228130\_i31 |
| **Modelo *Gilb***  Mediante este modelo de *Gilb* se puede construir una serie de especificaciones las mismas que se describen para determinar la calidad del proyecto. Este tipo de requerimientos deben ser construidos tanto por el investigador como por el usuario.  En realidad, lo que se busca es el cumplimiento de los requerimientos definidos, y obteniendo a su vez una satisfacción por parte del usuario. Cada una de las características definidas son medibles a otros tipos de sub características donde se debe especificar:   * + - Nombre de la característica     - Unidad de medición     - Prueba o recogida de datos     - El valor estimado     - El valor óptimo     - El valor obtenido en el sistema     - Observaciones | | Metrics concept illustration. Data chart with calculator.  Imagen de referencia  <https://image.shutterstock.com/image-illustration/metrics-concept-illustration-data-chart-600w-1155774370.jpg>  **Imagen:** 228130\_i32 |
| **Marco ISO 9126**  Es un estándar en la ingeniería del *software* que se fundamenta en el nivel de satisfacción a nivel de producto. El objetivo principal en este tipo de modelo es que el producto cumpla con el requerimiento en un contenido específico de uso.  La norma está compuesta por seis factores:   * + - Funcionabilidad     - Fiabilidad     - Usabilidad     - Eficiencia     - Mantenibilidad     - Portabilidad   La calidad interna y externa expuesta en ISO9126-1 (2001) se declara “la totalidad de atributos de un producto que determina su capacidad de satisfacer necesidades explícitas e implícitas cuando es usadas bajo condiciones específicas”. Se define como calidad externa “el grado en la que un producto satisface necesidades explícitas e implícitas cuando se utiliza bajo condiciones especificadas. | | ISO 9126 - Métricas para la calidad de tu web - GrupoDigital360  Imagen de referencia  <https://grupodigital360.com/wp-content/uploads/2018/05/ISO9126-1024x672.png>  **Imagen:** 228130\_i33 |

**4. Casos de prueba**

|  |  |  |
| --- | --- | --- |
| **Tipo de recurso** | Pestañas o tabs horizontales | |
| **Introducción** | A continuación, se aborda lo referente a los casos de prueba: conceptos, composición, tipos, formatos, entre otros. | |
| **Conceptos** | Un caso de prueba es un conjunto de variables o condiciones donde se especificará que un determinado sistema, parte del componente del sistema, o una base de datos, o determinada característica cumple o no cumple con los requerimientos.  Un dato importante al realizar casos de pruebas es que está destinado a probar una tarea en específico o una variable. Esto beneficia al momento de implementar una herramienta de *testing*, ya que cuenta con más flexibilidad al momento de realizar las pruebas. | Trasero detrás de la vista retrato de su muy hábil y ocupado hombre geek que tipea creación de base de datos base de datos en moderno interior estilo industrial pared de hormigón estación interior  Imagen de referencia  <https://image.shutterstock.com/image-photo/rear-back-behind-view-portrait-600w-1836307210.jpg>  **Imagen:** 228130\_i34 |
| **Script de prueba vs. caso de prueba** | Un *script* de prueba es un programa corto cuyo objetivo es evaluar determinadas funciones. Un caso de prueba es un documento con una serie de pasos planificados que se debe completar para cumplir un objetivo propuesto. | **Figura 3**  Script de prueba vs. caso de prueba  Gráfico que enumera diferentes tipos de propósitos de casos de prueba: la columna de la izquierda muestra la funcionalidad, la unidad, el rendimiento, la base de datos; la columna de la derecha muestra la interfaz de usuario, la integración, la seguridad, la usabilidad  *Nota (*Mc Mullin*, 2021).* Fuente: [https://es.parasoft.com/blog/how-to-write-test-cases-for-*software*-examples-tutorial/](https://es.parasoft.com/blog/how-to-write-test-cases-for-software-examples-tutorial/)  **Imagen:** 228130\_i35 |
| **Composición** | * + Identificador: puede ser numérico o alfanumérico, como su nombre lo dice permite identificar un caso de prueba según su código.   + Nombre del caso de prueba: se debe implementar una nomenclatura predefina o estándar. Lo más importante es emplear un nombre significativo en base a lo que se va a probar.   + Descripción: en este ítem se realiza una descripción a detalle, plasmando el componente o la sección que se va a probar, reflejando el ambiente de pruebas, la data que se va a utilizar y las respectivas precondiciones.   + Pasos: son las etapas de forma secuencial que se van a ejecutar en el caso de prueba.   + Resultados esperados: obtenido de forma exitosa por la herramienta implementada en la ejecución de las pruebas, y por último concluir o determinar si el respectivo test fue exitoso. | Resultado de imagen para casos de pruebas  Imagen de referencia  <https://www.diariodeqa.com/post/buenas-pr%C3%A1cticas-al-escribir-casos-de-prueba>  **Imagen:** 228130\_i36 |
| **Tipos** | Los casos de prueba tienen la habilidad de medir muchos aspectos del sistema. Los pasos planificados en el caso de pruebas tienen la intención de incluir un resultado de falla en lugar de un resultado exitoso. Algunos ejemplos de casos de pruebas más conocidos son los siguientes:  **Tabla 1**  *Tipos de caso de prueba*  Tabla que muestra ejemplos de casos de prueba comunes para funcionalidad, seguridad y usabilidad  *Nota. Reproducida de Diferentes tipos de casos de prueba.* Mc Mullin*, (2021).* [https://es.parasoft.com/blog/how-to-write-test-cases-for-*software*-examples-tutorial/](https://es.parasoft.com/blog/how-to-write-test-cases-for-software-examples-tutorial/) | How to Write Test Cases for Software: Examples & Tutorial  Imagen de referencia  <https://www.parasoft.com/wp-content/uploads/2021/05/image-2-test-cases-collage-300x148.png>  **Imagen:** 228130\_i37 |
| **Formato** | Cada componente de una prueba unitaria definirá aspectos cruciales que incluyen:   * Funciones a realizar para prueba plasmada. * La data implementada para la prueba. * Resultados esperados en la ejecución de la prueba. * Aseguramiento de la prueba, que la ejecución sea independiente de los demás componentes del sistema.   Es importante tener en cuenta que el formato estándar de pruebas como mínimo debe contener:   * Un nombre relevante de la prueba. * Datos estandarizados para la prueba. * Nombre del método, unidad o componente bajo prueba. * Aplicar un estado positivo de la prueba. * Ejecución de la prueba unitaria. | icono de análisis o análisis como la métrica kpi. diseño de trazo de contorno o elemento simple de validación gráfica de logotipo de doc para negocios o web. concepto de violación del proceso de búsqueda o regulación o modelo financiero  Imagen de referencia  <https://image.shutterstock.com/image-vector/insight-assesment-icon-like-kpi-600w-2182852107.jpg>  **Imagen:** 228130\_i38 |
| **Mejores prácticas** | La forma de redactar casos de pruebas de forma eficiente se va logrando con el paso del tiempo y la práctica. Se debe realizar o mantener un título descriptivo, realizar una descripción detallada del caso de prueba y mantener un lenguaje claro y conciso.  También se deben plasmar aquellos detalles como los resultados esperados y las diferentes condiciones específicas. Toda esta información es crucial para la ejecución de la prueba por medio de la herramienta tecnológica.  Una lista de aspectos a tener en cuenta para construir casos de pruebas eficientes:   * Mantener los datos e información de forma transparente. * Construir los casos de pruebas de forma reutilizables. * Mantener los identificadores de los casos de prueba de forma única. * Implementar revisión por pares es muy relevante. * Los casos de prueba construidos deben tener en cuenta los requerimientos definidos. * Detallar los resultados esperados. | Imagen de referencia  <https://qalified.com/wp-content/uploads/2022/02/6460946-1-scaled.jpg>  **Imagen:** 228130\_i39 |
| **Ejecución** | En esta fase se procede a probar la base de datos con los casos de prueba, obteniendo así los resultados reales; se realiza preferiblemente de forma automática. | Concepto de Monitoreo de Rendimiento de Red - NPM - Herramientas y Procesos para Monitorear la Calidad de Servicio de una Red Digital - Observabilidad y Monitoreo en Nube - Ilustración 3D  Imagen de referencia  <https://image.shutterstock.com/image-illustration/network-performance-monitoring-concept-npm-600w-2181428889.jpg>  **Imagen:** 228130\_i40 |
| **Evaluación** | En la evaluación de resultados obtenidos se determina si la prueba fue exitosa o no, realizando una comparación sistemática de los resultados obtenidos en la etapa de ejecución de los casos de prueba y los resultados esperados plasmados dentro del caso de prueba. Este proceso se puede realizar de manera manual o automática. | Company process development structure. Business organization. Digital communication. Data set,process,classification,database,data analytic and evaluation.The working process.Multitasking.Web  Imagen de referencia  <https://image.shutterstock.com/image-vector/company-process-development-structure-business-600w-1968222760.jpg>  **Imagen:** 228130\_i41 |

**5. Plan de pruebas**

|  |  |
| --- | --- |
| Tipo de recurso | Cajón de texto de color |
| En la temática de pruebas de *software* resaltan dos términos que vale la pena profundizar y conocer.  **Suite de pruebas**  Es un conjunto de pruebas de forma organizadas, donde se destinan los diferentes casos de pruebas en lo que se refiere a su código fuente, el conjunto de pruebas o la colección de dependencias o el conjunto de las respectivas pruebas que se le aplicarán al sistema.  El conjunto de pruebas es muy similar al tener un archivador y dentro de él se ingresa cada uno de los casos de pruebas.  **Plan de pruebas**  La respectiva configuración que lleva un plan de pruebas va en términos de pruebas manuales, pruebas automatizadas acompañada de un formato de forma general describiendo cada uno de los pasos como se van a ejecutar dichas pruebas. Iniciaran con la evaluación del sistema implementado los conjuntos de pruebas y los casos de pruebas propuestos, antes de realizar cambios o adicionar nuevas funciones (Pérez et al. 2020).  Además, describe el objetivo de la prueba general y brinda un registro de la gestión de planificación de pruebas. Este plan de pruebas es muy flexible permitiendo ajustar a las necesidades del equipo de trabajo.  Tecnología de desarrollo de aplicaciones, depuración y prueba de código de software para errores y fallos de búsqueda. Concepto de datos de programación, análisis y codificación. Ilustración vectorial.  Imagen de referente: <https://image.shutterstock.com/image-vector/technology-application-development-debugging-code-600w-1868752312.jpg>  **Imagen** 228130\_i42 | |

**Componentes del plan de pruebas**

|  |  |
| --- | --- |
| **Tipo de recurso** | Rutas / Pasos. Verticales 1 |
| **Introducción** | El plan de pruebas contiene unos componentes los cuales se describen a continuación: |
| Imagen de referente: [https://mundotesting.com/plan-de-pruebas-de-*software*/](https://mundotesting.com/plan-de-pruebas-de-software/)  **Imagen** 228130\_i43 | |
| **Botón 1** | **Identificador**  Permite la identificación del plan de pruebas con respecto a otro existente, debe ser único e irrepetible. No debe existir otro plan con este identificador, el objetivo crucial de este ítem es llevar un control riguroso de todos los planes de pruebas.  Ejemplo: Plan 001 – (Nombre del plan) |
| **Botón 2** | **Introducción**  Es la parte inicial del documento, el objetivo principal es detallar el alcance de las pruebas, generando una breve explicación o resumen de todas las secciones que integran el plan de pruebas. |
| **Botón 3** | **Objetivo**  Expresa la finalidad del plan de pruebas, no corresponde destacar resultados esperados o medibles mediante indicadores. |
| **Botón 4** | **Estrategia**  Expresa la metodología que se va a implementar, las acciones que se ejecutaran para permitir alcanzar el objetivo general del plan de pruebas. |
| **Botón 5** | **Alcance**  Corresponde en trazar los límites de la totalidad del trabajo que se necesita para dar por culminado el plan de pruebas. |
| **Botón 6** | **Propósito**  Determinación de lo que se espera al momento de terminar la implementación del plan de pruebas |

|  |  |
| --- | --- |
| **Tipo de recurso** | Acordeón tipo 1 |
| **Introducción** | Continuando con la temática del plan de pruebas se aborda información sobre: documentación, características, criterios, tareas, modificadores. |
| Hoja de papel de aplicación CV, mano de hombre de negocios con currículum, concepto de entrevista de trabajo buscar, investigación de habilidades de datos personales, resultados de formación analizar diseño moderno plano aislado en verde  Imagen de referente: <https://image.shutterstock.com/image-vector/cv-application-paper-sheet-business-600w-393287248.jpg>  **Imagen:** 228130\_i44 | |
| **Documentación o entregables**  En esta fase corresponde al alistamiento de todos los documentos que se entregaran como parte de la implementación del plan de pruebas, especificando los siguientes datos:   * + - Nombre del documento     - Persona encargada de la entrega del documento     - Nombre de la persona que recibe este documento     - Fecha planificada     - Fecha de entrega de cada uno de los entregables   Se recomienda la implementación de una tabla donde el tipo de letra, el tamaño de la letra y los colores de la tabla sean de forma concisa y entendible para que todas las personas involucradas en el plan de pruebas, puedan identificar de forma eficiente las características de los entregables.  Una lista de los documentos que se deben entregar en esta fase:   * + - Casos de pruebas     - Especificación del diseño del caso de prueba     - Reportes de defectos o novedades     - Evidencias de las pruebas     - Reportes reflejados por alguna herramienta de gestión de pruebas     - otro documento que se considere relevante entregar y/o adjuntar | |
| **Características a ser probadas**  En este ítem se realiza una lista de todas las características que serán objetos de pruebas. Es de vital importancia realizar la lista con los interesados del proyecto, para determinar con exactitud las características que serán probadas en el plan.  Algunas características a tener en cuenta son:   * + Características funcionales   + Características no funcionales   + Interfaz gráfica, etc.   Estas características se deben declarar de forma clara y concisa para no tener inconvenientes al momento de ejecutar las pruebas. | |
| **Características que no van a ser probadas**  Se debe realiza una lista de las características que no van a ser probadas, justificando por qué no serán parte del plan de pruebas y el riesgo que se pueden generar al no contemplarse. | |
| **Criterios de aprobación y fallo**  En esta sección se describen los criterios de aceptación que se corroboran al ejecutar cada una de las respectivas pruebas. Todo error reportado debe ir acompañado de su respectivo mensaje de validación. Cuando un criterio de aprobación es rechazado se toman las respectivas correcciones por el criterio de fallo. Todo criterio de aceptación debe ir siempre acompañado con su respectivo criterio de fallo, donde se especifica la acción que se ejecutará sobre el plan de pruebas. | |
| **Tareas de prueba**  Describen cada una de las tareas y su respectiva fecha de ejecución, el objetivo es mantener un buen control de cada una de las tareas incluidas en el plan. Se recomienda tener en cuenta las siguientes indicaciones:   * Establecer el nombre de la tarea * Descripción de la tarea * Fecha de inicio * Fecha de finalización * Duración de la tarea * Persona responsable de la tarea * Rol que desempeña la tarea   Para determinar la duración de cada una de las tareas es crucial implementar un WBS (Estructura de Descomposición del Trabajo), donde dentro de él se describe cada una de las tareas de forma desglosada para obtener dicha duración, o se puede pedir opinión a expertos para tener una visión más específica. | |
| **Modificadores**  Hay dos tipos de modificaciones o ampliaciones dentro de un plan de pruebas, que han de tomarse en cuenta para que se pueda cubrir toda la funcionalidad existente:  Funcionalidades de usuario final:cuando se agregan nuevos módulos, pantallas o flujos en las funcionalidades que son utilizadas por el cliente o se evidencian visualmente.  Funcionalidades internas*:* son cambios internos que mantienen todos los elementos visuales de la misma manera, por lo tanto, el cliente final no observa ninguna modificación, pero al modificar componentes o flujos internos, como accesos a base de datos o a capas de lógica de negocio, deben cubrirse con casos de pruebas dentro del plan de pruebas e incluir toda la funcionalidad de este tipo. | |

**6. Ejecución de pruebas**

|  |  |
| --- | --- |
| Tipo de recurso | Cajón de texto de color |
| La ejecución de pruebas es el proceso mediante el cual se implementa una prueba sobre el componente o el sistema que se está probando.  Cómo mejorar la ejecución y las pruebas de sus controles internos  Imagen de referente: <https://blog.softexpert.com/es/ejecucion-pruebas-controles-internos/>  **Imagen** 228130\_i45  La preparación de la ejecución de las pruebas comprende la variedad de las pruebas, los flujos de trabajo de la ejecución de los casos de pruebas o conjunto de pruebas, así como la personalización del comportamiento en la ejecución de las pruebas en el entorno. Las siguientes son fases para tener en cuenta en la ejecución de las pruebas:   * + Visión general de la ejecución de pruebas. Existe una gran variedad en la ejecución de las pruebas, se deben elegir la que mejor se adapten al equipo de prueba.   + Flujo de trabajo de ejecución de caso de prueba. Las necesidades de ejecución de pruebas a veces pueden ser algo complejas, existen varias funcionalidades que se adaptan al flujo de trabajo para la ejecución de dichas pruebas.   + Flujo de trabajo de ejecución de conjunto de pruebas Un conjunto de pruebas es una colección de pruebas que fueron agrupada. La gran mayoría de conjuntos de pruebas incluyen pruebas manuales y automatizadas, es de gran beneficio cuando se pueden planificar pruebas automatizadas sin ninguna supervisión. Cuando se procede a ejecutar un conjunto de pruebas, se puede especificar los casos de pruebas de forma secuencial o en paralelo.   + Preferencias de ejecución de pruebas, Puede personalizar el comportamiento de la ejecución de prueba en su entorno configurando las preferencias del usuario y del sistema relacionadas con dicha ejecución de prueba. | |

**6.1 Creación del entorno**

|  |
| --- |
| Cuadro de texto |
| Un entorno de prueba es la respectiva configuración de tipos de entornos para determina prueba. Por ejemplo, un entorno de prueba que implementan *Chrome, Windows 10 y Apache Tomcat* es una configuración de estos tipos de entorno: navegador, sistema operativo y servidor de aplicaciones.  Cada tipo de entorno puede tener uno o más valores. Una plataforma de plan de pruebas es el conjunto de todos los tipos de entorno y sus respectivos valores para determinado plan de prueba. La plataforma está encargada del soporte del producto y de la prueba que está realizando el equipo.  Los entornos de pruebas se construyen lo más similar posible al entorno de producción, de esta forma al sistema o a la base de datos se pueden realizar dichas pruebas, teniendo en cuenta los cambios sustanciales y el comportamiento no esperado.  Los tipos de entornos pueden variar según su comportamiento y objetivo en función al desarrollo del sistema.  Existen tres tipos de desarrollo:   * Desarrollo *web* * Servicios *web* * Publicación de *Wikis* |

|  |  |
| --- | --- |
| **Tipo de recurso** | Acordeón tipo 2 |
| **Introducción** | Se presenta información sobre la creación de entorno en cuanto, a los tipos de entorno de pruebas, elementos claves, *script* de pruebas, ejemplo de *script* de pruebas. |
| Concepto de diseño plano, un programador está asegurando el sistema  Imagen de referente: <https://image.shutterstock.com/image-vector/flat-design-concept-programmer-securing-600w-1851058906.jpg>  **Imagen** 228130\_i46 | |
| **Tipos de entornos de pruebas**  Los entornos de pruebas pueden ser construidos por la empresa que está desarrollando el sistema o por organizaciones que brindan ese tipo de servicio, (Loor et al. 2020).  Los tipos de entorno son:  Estándar:pueden ser creados por los administradores o por personas que tengan permisos por parte de los administradores para crearlos. De esta forma es controlado qué persona sube o realiza cambios en el sistema.  Basados en suscripción*:* este tipo de entorno se enfoca en desarrollo de sistemas grandes, con varios módulos y usuarios. | |
| **Elementos claves para los entornos de pruebas**  Construir un entorno de pruebas tiene como objetivo principal simular el entorno de producción; se deben tener en cuenta algunos elementos básicos:   * Navegador. * Sistema operativo por parte del cliente. * Configuración de red. * Configuración del servidor de bases de daros. * Configuración del servidor del sistema. * Datos para realizar las pruebas.   Además, se debe construir un mecanismo que genere informes de errores para recopilar los resultados de las pruebas y, mediante este registro, trabajar en las mejoras a realizar. También es importante resaltar puntos claves:   * Qué se busca con las pruebas a realizar. * Verificar que las condiciones para la ejecución de la prueba estén creadas. * Planificar en qué momentos se realizará cada etapa de prueba, involucrando a los miembros correctos del equipo. | |
| **Instalación del *script***  El *script* de prueba es un conjunto de instrucciones para realizar automáticamente las pruebas al sistema. Estas pruebas son realizadas en el entorno de pruebas automatizadas, se ejecutan según el formato del *script*, por medio del *Script Testing* podemos implementar diferentes comandos para desarrollar un *script*. | |
| **Cómo escribir un *script* de pruebas**  Es una secuencia de instrucciones para realzar la prueba automáticamente.  Hay tres maneras para escribir una secuencia de comandos de pruebas:   * Grabar y reproducir: en este método en particular la persona que realiza el *testing* debe escribir cualquier código en lugar de registrar las acciones del usuario, además, se debe tener en cuenta el comportamiento esperado de la prueba. Este método es sencillo de construir, se utiliza principalmente en un lenguaje de programación simplificado como *VBScript*. * Secuencia de comandos basadas en palabra clave/datos: en este método hay una independencia entre evaluadores y desarrolladores. En la secuencia de comandos basada en datos, el evaluador define la prueba implementando palabras claves. El trabajo de los desarrolladores consiste en la implementación del código del *script* de prueba para las palabras claves y actualizar el código cuando sea pertinente. * Escribir código usando el lenguaje de programación: el evaluador efectivamente tendrá que ir más allá de la grabación / reproducción y aprender a construir *scripts* simples. Se pueden escribir los *scripts* de prueba en un lenguaje más sencillo como J*avaScript* o *Ruby*. | |
| **Ejemplo de un *script* de prueba**  Serie de instrucciones para escribir una secuencia de comando para un inicio de sesión de un sitio *web*:   * + - Especificar en la herramienta de automatización que pueda localizar los campos nombre de usuario y contraseña.     - Actualizar la página de inicio del sitio *web*, dar *click* en iniciar sesión, verificar los campos de nombre de usuario y contraseña.     - Continuar escribiendo el nombre de usuario y la contraseña, oprimir el botón ingresar.     - Determinar que se evidencia el mensaje de bienvenida de inicio de sesión, el cual es visible después de ingresar y diligenciar los campos respectivos de nombre de usuario y contraseña, de acuerdo con los estilos CSS.     - Ingresar el título de bienvenida.     - Si el título es lo esperado del resultado de la prueba, evidencia el resultado en el informe pertinente. | |
| **Plantilla de *script* de prueba**  *Test Script* es un documento con formato reutilizable que contiene información preseleccionada y que es importante para crear un *script* de prueba utilizable. Este documento determina qué tan detalladas son sus pruebas y qué información debe incluirse en cada caso de prueba. | |

**6.2. Desarrollo guiado por pruebas (TDD)**

|  |  |  |
| --- | --- | --- |
| **Tipo de recurso** | | Pestañas o tabs Verticales |
| **Introducción** | | Se presenta información sobre el desarrollo dirigido por pruebas, sus características e implementación. |
| Test Driven Development (Tdd)  Imagen de referencia: <https://www.digite.com/es/agile/desarrollo-dirigido-por-pruebas-tdd/>  **Imagen** 228130\_i47 | | |
| **Concepto** | El desarrollo dirigido por pruebas TDD, es un proceso en el que se construyen primero las pruebas antes de escribir el código, y cuando las pruebas son exitosas se limpia el código para realizar los preparativos del paso a producción.  La premisa de TDD según Kent Beck es que todo código desarrollado debe ser probado y refactorizado constantemente. | |
| **Características** | Además del gran beneficio de TDD de mantener el código probado constantemente, también logra la notificación de errores de forma temprana y un diagnóstico sencillo de los errores donde se puede evidenciar lo que ha fallado.  Características principales o beneficios de TDD:   * Mejora la flexibilidad del desarrollo del código, facilita la incorporación de nuevas contrataciones. * Reducción de costo en mejoras, se mantiene el código limpio y se minimizan los riesgos. * Desarrollo entorno a la integración, entregas oportunas y despliegues continuos. * Disminución en número de errores. * Reducción en los costos de soporte. | |
| **Pruebas (Prácticas)** | La metodología de TDD es muy sencilla de implementar, al escribir un test que pruebe el código que se quiere desarrollar. TDD es un mecanismo de comunicación fluida, se implementa un ciclo de RED – GREEN – REFACTOR, una vez se encuentre en la fase de verde, se debe quitar duplicación y refactorizar, esta se basa en tres leyes:   * Se construye primero el test y luego se procede a construir código * Es suficiente una prueba unitaria para probar el código * No exceder en el tamaño del código que se requiere pasar por una prueba unitaria.   Hacer clic en el documento anexo para conocer más sobre estas pruebas. | |

**6.3. Codificación de casos**

|  |  |
| --- | --- |
| Tipo de recurso | Cajón de texto de color |
| Un caso de prueba es una serie de acciones de codificación que los programadores ejecutan para verificar diferentes funciones del *software*. Los casos de prueba contienen pasos de análisis, condiciones de comportamiento y resultados de las acciones.  Cómo hacer casos de prueba  Imagen de referente: <https://www.diariodeqa.com/post/buenas-pr%C3%A1cticas-al-escribir-casos-de-prueba>  **Imagen** 228130\_i48  Características   * Solo prueba una cosa en un caso de prueba * Debe tener un propósito exacto * Debe estar escrito en un lenguaje claro y fácil de entender * Debe ser relativamente pequeño * Debe ser independiente * No debe tener pasos o palabras innecesarias * Debe ser repetible * Debe usar terminología consistente e identificación de funcionalidad | |

**6.4. Documentación de pruebas**

|  |  |  |
| --- | --- | --- |
| **Tipo de recurso** | Slider Presentación | |
| **Introducción** | presenta algunas pruebas que se pueden documentar como: unitarias, estrés, integración. | |
| **Unitarias**  Documentar el código de un programa es adicionar información importante que permita explicar lo que realiza esa determinada acción, paso a paso, para que al momento de ser observada por otro desarrollador lo pueda comprender de forma rápida.  Las pruebas unitarias son una forma de comprobar que cierto fragmento de código funciona perfectamente, es un procedimiento más que se lleva a cabo durante la implementación de metodologías ágiles.  Características de las pruebas unitarias:   * + Las pruebas unitarias evidencias que la lógica del código funciona como se espera o cumple con el requerimiento.   + Maximiza la legibilidad del código, facilita realizar cambios de forma rápida   + Los test unitarios bien realizados sirven como documentación del proyecto.   + Se realizan de forma rápida, donde se puede implementar en todo el desarrollo del proyecto.   + Permite la refactorización del código   + Como las pruebas unitarias dividen el código en pequeños fragmentos, es posible probar distintas partes del proyecto sin tener que esperar a que otras estén completadas. | | No abandonemos las pruebas unitarias  Imagen de referente: <https://www.qalovers.com/2018/01/unit-testing.html>  **Imagen** 228130\_i49 |
| **Estrés**  Es una prueba de carga que se implementa para determinar los límites del sistema. Normalmente se requiere realizar prueba de estrés a una API o de un sitio web para determinar:   * El comportamiento del sistema en condiciones extremas * La capacidad máxima del sistema en pro de su rendimiento * El punto de quiebre del sistema y sus respectivos fallos * La recuperación del sistema ante posibles fallas de forma inesperadas   Hay que tener en cuenta que una prueba de estrés no significa que el sistema se va a comportar siempre de esa forma, es una prueba de exposición a posibles picos de carga para evaluar su rendimiento.  Una vez culminadas todas las pruebas, se obtendrán las respectivas conclusiones sobre el rendimiento de la aplicación según los resultados obtenidos. Finalmente, el cliente dispondrá de un informe detallado con toda la referencia de las pruebas realizadas, junto con las conclusiones finales y las propuestas de mejoras y medidas correctoras correspondientes. | | Imagen de referente: <http://www.expertscoding.es/pruebas-de-carga-y-stress/>  **Imagen** 228130\_i50 |
| **Integración**  Este tipo de pruebas se encargan de validar la respectiva comunicación entre diferentes componentes, interfaces o interacciones con diferentes partes del sistema.  Dentro de la práctica de realizar las pruebas de integración se tienen diferentes enfoques:   * + En las pruebas de integración *Bin Bang*, todos los componentes son integrados y se realiza las pruebas como un todo.   + Las pruebas de integración descendente se realizan de arriba a abajo siguiendo el flujo arquitectónico   + Las pruebas de integración ascendente, se llevan a cabo desde la parte inferior del flujo de control hacia arriba. Los componentes o sistemas se sustituyen por controladores.   + En las pruebas de integración incremental, las partes de desarrollo realizado por el programador se integran para ejecutar una sola prueba. | | Imagen de referente: <https://qawerk.es/process/integration-testing/>  **Imagen** 228130\_i51 |

**Documentación de estándares**

|  |  |  |
| --- | --- | --- |
| **Tipo de recurso** | Pestañas o tabs horizontales | |
| **Introducción** | Se presenta información sobre documentos. | |
| **IEEE 1012** | ~~El estándar IEEE 1012 consiste en la verificación y validación de un~~ *~~software~~*~~, es un procedimiento basado en normas de calidad en algunos modelos de vida de un~~ *~~software~~*~~.~~  ~~Define cuatro niveles de integridad para describir la importancia del~~ *~~software~~*~~, varían desde alta integridad hasta baja integridad, para el usuario.~~  ~~Los niveles de integridad del~~ *~~software~~* ~~son un rango de valores que representan la complejidad, criticidad, riesgo, nivel de seguridad, nivel de protección del~~ *~~software~~* ~~desempeño deseado, confiabilidad y otras características del proyecto en sí, que definen la importancia del~~ *~~software~~* ~~para el usuario o adquisidor. Las características suelen determinar el nivel de integridad dependiendo del objetivo de la aplicación y uso del sistema.~~  ~~El nivel de integridad determina el número de tareas a efectuar, entre más alto mayor número de tareas se llevan a cabo y son más rigurosas.~~  El IEEE 1012 es estándar se sustenta en la verificación y validación de *software*, este procedimiento se basa en normas de calidad de modelos de vida de un software.  Contiene cuatro niveles de integridad que describen la relevancia del *software*, variando desde baja hasta alta integridad para el usuario.  Estos niveles son valores que representan complejidad, criticidad, nivel de seguridad, entre otros, que precisan la trascendencia del *software* para el usuario final. Estas características determinan el nivel de integridad teniendo en cuenta el objetivo y uso del sistema.  El nivel de integridad permite identificar el número de tareas a desarrollarse, entre más alto es mayor el número de tareas a realizar, siendo estas más rigurosas.  Componentes de la documentación IEEE   * El alcance * El propósito * Campo de aplicación * Objetivos * Anexo (gráficas y tablas) | Concepto de collage de control de calidad estándar  Imagen de referente: <https://qawerk.es/process/integration-testing/>  **Imagen** 228130\_i52 |
| **IEEE 829** | La documentación de las pruebas de *software* o bases de datos es muy importante para llevar el control de las diferentes actividades que se han ejecutado. El estándar 829 para la documentación de prueba de *software* y sistema pertenece a un estándar de IEEE que especifica la manera de implementar los documentos para su uso en las ocho fases definidas de prueba de *software*. En cada una de las fases se especifica qué documento se debe diligenciar para llevar una documentación completa.  Documentos requeridos por la norma IEEE 829   * + Plan de prueba maestro (MTP), es el plan de pruebas que proporciona un documento general para llevar una planificación y gestión de las pruebas para la implementación de múltiples pruebas.   + Plan de pruebas de nivel (LTP). En este documento se plasma el alcance, los objetivos, el enfoque, los recursos, y el cronograma de las actividades de la ejecución de las pruebas.   + Diseño de prueba de nivel (LTD), detalla cada caso de prueba y los resultados esperados.   + Caso de prueba de nivel (LTL), documento donde se especifican los datos de pruebas para implementarlos en los casos de prueba.   + Procedimiento de prueba de nivel (LTPr), detalla específicamente como se debe ejecutar cada una de las pruebas, las condiciones y los pasos que se deben seguir.   + Registro de prueba de nivel (LTL), documento que registra al detalle con su respectiva fecha cada una de la ejecución de las pruebas y sus respectivas observaciones, por ejemplo, que caso de prueba se ejecuta, persona responsable de la prueba, el orden de la ejecución y si fueron exitosas las pruebas.   + Informe de anomalías (AR), registra cada evento que ocurra durante la ejecución de las pruebas.   + Informe de estado de prueba intermedia de nivel (LITSR), se registran los resultados provisionales obtenidos en las pruebas para proporcionar evaluaciones y diferentes tomas de acciones basada en los resultados.   + Informe de prueba de nivel (LTR), señala los resultados de las actividades de prueba designadas y proporcionar evaluaciones y recomendaciones basadas en los resultados después de que la ejecución de la prueba haya finalizado para el nivel de prueba específico.   + Informe de prueba maestro (MTR). Se registra cada una las fases realizadas en la ejecución de las pruebas, también contiene cada una de las evaluaciones de calidad realizadas, las acciones de corrección ejecutadas. Además, se registra qué pruebas se realizaron y cuánto fue su duración, para mejoras en futuras planificaciones. Este documento final se utiliza para indicar si el sistema de *software* bajo prueba es apto para el propósito de acuerdo con el cumplimiento de los criterios de aceptación. | Concepto IEEE 829 con placa base para ordenador. Texto IEEE 829 escrito en  Tecnología Placa Base Tecnología Antecedentes de tecnología digital. IEEE  829 con círculo impreso Fotografía de stock - Alamy  Imagen de referente: <https://c8.alamy.com/compes/2h2ypfp/concepto-ieee-829-con-placa-base-para-ordenador-texto-ieee-829-escrito-en-tecnologia-placa-base-tecnologia-antecedentes-de-tecnologia-digital-ieee-829-con-circulo-impreso-2h2ypfp.jpg>  **Imagen** 228130\_i53 |
| **ISO** | Las normas ISO son implementadas para mejorar los procesos en las empresas, establecen un sistema para gestionar la calidad de los productos o servicios. Contiene 4 principios fundamentales:   * Responsabilidad de la dirección * Gestión de recursos * Realización de producto o servicio * Medición, análisis y mejora   La documentación obligatoria que exige la norma ISO es la siguientes:   * + El alcance, del respectivo sistema de gestión de calidad   + Información detallada y necesaria para apoyar todos los procesos de ejecución y evaluación de pruebas   + La política de calidad   + Los objetivos de calidad   + Información documentada requerida | Concepto de tecnología empresarial de garantía de garantía de control de calidad de los estándares ISO.  Imagen de referente: <https://image.shutterstock.com/image-photo/iso-standards-quality-control-assurance-600w-1253334961.jpg>  **Imagen** 228130\_i54 |

|  |
| --- |
| Cuadro de texto |
| Tenga en cuenta que el componente formativo dispone de más recursos que se ubican en este, para ello diríjase al menú principal en donde encontrará entre otros, la síntesis, una actividad didáctica, material complementario. |

**Síntesis**

|  |  |
| --- | --- |
| **Tipo de recurso** | Síntesis |
| Tecnólogo en implementación y gestión de bases de datos  Síntesis: diseño y ejecución de pruebas | |
| **Introducción** | Estimado Aprendiz le invitamos a revisar el resumen de las temáticas abordados en el desarrollo del componente formativo. |
| **Figura 4**  *Síntesis diseño y ejecución de pruebas*  Diseño y ejecución de pruebas  Conceptos  Tipos de pruebas  Plan de pruebas  *Suit* de pruebas  Casos de pruebas  Ejecución de pruebas  Creación del entorno  TDD  Codificación  Documentación  Herramientas  Documentación  **Imagen** 228130\_i55 | |

**Actividad didáctica**

La actividad didáctica se encuentra en anexos / Actividad didáctica

**Material complementario**

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo de recurso | Material complementario | | |
| Tema | Referencia APA del material | tipo | Enlace |
| Casos de Prueba | Gómez, J. (2020). *Cómo diseñar y mantener casos de prueba.* video]. Youtube. | Vídeo | <https://www.youtube.com/watch?v=ZEqwbR1YyJU> |
| Construcción de un test | Muñiz, J., & Fonseca, E. (2019). Diez pasos para la construcción de un test. *Psicothema*, 31(1), p. 7-16. | Artículo | <https://investigacion.unirioja.es/documentos/5c55196234ada657a18c95dd> |
| Pruebas de Validación | SENA. (2021). *Pruebas de validación de bases de datos: introducción.* [video]. Youtube | vídeo | <https://www.youtube.com/watch?v=U3RWZi2CwQo> |
| Pruebas de carga | Garrido, A., Borja, Y., & Gutiérrez, G. (2021). Rendimiento de MariaDB y PostgreSQL. *Revista Científica y Tecnológica UPSE*, 7(2), p. 9-16. | Artículo | <https://repositorio.upse.edu.ec/handle/46000/7315> |

**Glosario**

|  |  |
| --- | --- |
| **Tipo de recurso** | Glosario |
| Caso de prueba: | es una serie de acciones de codificación que los programadores ejecutan para verificar diferentes funciones del *software*. |
| Entorno: | es la respectiva configuración de tipos de ambiente de despliegue para determinar una prueba. |
| IEEE: | Es el estándar IEEE 1012 consiste en la verificación y validación de un *software*, es un procedimiento basado en normas de calidad en algunos modelos de vida de un *software*. |
| ISO: | ISO son implementadas para mejorar los procesos en las empresas, esta norma establece un sistema para gestionar la calidad de los productos o servicios. |
| Plan de pruebas: | contiene la respectiva configuración que lleva un plan de pruebas, va en términos de pruebas manuales, pruebas automatizadas acompañada de un formato de forma general describiendo cada uno de los pasos como se van a ejecutar dichas pruebas. |
| Prueba de integración: | su objetivo es verificar que los diferentes módulos y componentes funcionan bien en conjunto. Este tipo de pruebas son un poco más costosas al momento de automatizarlas, ya que se requiere que todos los componentes marchen a la perfección. |
| *Script*: | es un conjunto de instrucciones para realizar automáticamente las pruebas al sistema. |
| *Suit* de pruebas: | es un conjunto de pruebas de forma organizadas, donde se destinada los diferentes casos de pruebas en lo que se refiere a su código fuente, el conjunto de pruebas o la colección de dependencias o el conjunto de las respectivas pruebas que se la aplicaran al sistema. |
| TDD: | desarrollo dirigido por pruebas. |
| XML: | sigla en inglés de *eXtensible Markup Language*, traducido como “Lenguaje de Marcado Extensible” o “Lenguaje de Marcas Extensible”; es un metalenguaje que permite definir lenguajes de marcas, desarrollado por el *World Wide Web Consortium* (W3C), utilizado para almacenar datos en forma legible. |

**Referencias bibliográficas**

|  |  |
| --- | --- |
| **Tipo de recurso** | Bibliografía |
| Blanquicett, L., Bonfante, M., & Acosta, J. (2018). Prácticas de pruebas desde la Industria de *Software*. *La Plataforma ASISTO como Caso de Estudio*. Información Tecnológica. <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1l2l4cd/TN_cdi_scielo_journals_S0718_07642018000100011> | |
| Hidalgo, N., Costa, D., Marin, A. & Trujillo, Y. (2020). Elementos para selección de herramientas para la automatización de Pruebas no funcionales. *Serie científica de la Universidad de las Ciencias Informáticas*, 13(1), p. 104-112. <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1l2l4cd/TN_cdi_doaj_primary_oai_doaj_org_article_347cf9d7fc6d441aaa26cd8308dfde6e> | |
| Loor, J., Delgado, M., & Fernández, P. (2020). Priorización de casos de prueba en entornos de desarrollo ágil. *Ingeniería Industrial*, 41(2). <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1l2l4cd/TN_cdi_gale_infotracmisc_A678804944> | |
| Meléndez, M. (2018). *Calidad de los datos: la base para la modernización de sistemas de información* [Video, *Software* Guru]. Youtube. <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1fqo2uc/sena_virtualprovpro26558> | |
| Morales, M., Durán, J., Tandazo, E., & Morales, S. (2019). Rendimiento de bases de datos columnares. *Ingenius. Revista de Ciencia y Tecnología*, núm. 22, 47-58. <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1i756fj/TN_cdi_doaj_primary_oai_doaj_org_article_628e06c68e64447f95ac88bf44252c83> | |
|  | |
| Pérez, Y., Gallegos, J., Zapata, S., Ccama, D., & Choque, R. (2020). Design thinking en la planificación de pruebas de *software. Innovación y Software*, 1(2), p. 40-51. <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1l2l4cd/TN_cdi_doaj_primary_oai_doaj_org_article_49cd1beda5b84284be236b081b7bd5cf> | |
| Ramos, Y., Morales, Z., & Trujillo, Y. (2019). Propuesta de automatización de pruebas funcionales durante el ciclo de vida del *software* en *Desoft. Serie científica de la Universidad de las Ciencias Informáticas*, 12(9), p. 112-127. <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1l2l4cd/TN_cdi_doaj_primary_oai_doaj_org_article_dd1e5ca5d67a41d59e07fed1d2837503> | |
|  | |
| Toll, Y. & Ril, Y. (2019). Propuesta de manual de procedimiento para pruebas de sistema. *Serie Científica De La Universidad De Las Ciencias Informáticas,* *4*(2). <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1l2l4cd/TN_cdi_doaj_primary_oai_doaj_org_article_59038fc9254142ec941dee961c4a25f5> | |
| Torres, I. & Diaz, H. (2011). Pruebas de aceptación y piloto. *Serie científica de la Universidad de las Ciencias Informáticas, 4(1).* <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1l2l4cd/TN_cdi_doaj_primary_oai_doaj_org_article_10a1e2ea318043d694afa97ec2d20806> | |
|  | |
| Velázquez, Y., Velázquez, A., & Collado, L. (2020). Diseño de herramienta para casos de pruebas funcionales en la Universidad de las Ciencias Informáticas. *Serie científica de la Universidad de las Ciencias Informáticas*. [https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1l2l4cd/TN\_cdi\_doaj\_primary\_oai\_doaj\_org\_article\_b29579d39c5242d4ab9eab0bc385568 b](https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1l2l4cd/TN_cdi_doaj_primary_oai_doaj_org_article_b29579d39c5242d4ab9eab0bc385568%20b) | |