**UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS**

**(Universidad del Perú, Decana de América)**

**Facultad de Ingeniería de Sistemas e Informática**

**Escuela Profesional de Ingeniería de Software**

**Logotipo

Descripción generada automáticamente**

**Curso:** Gestión de la Configuración de Software

**Sección:** 01

**Plan de la Gestión de la Configuración de Software**

**Integrantes del Grupo 8:**

* Benites Meza, Marco Fabricio
* Bryan Anthony, Velarde Huancahuari
* Cancio Bedon, Alex Antonio
* Castillo Carranza José Richard
* Chaco Flores, Jose Luis
* Cueva Mantura, Alvaro Bertilo
* Lipa Palacios, Renzo Alejandro

**Docente:**

* Mg. Lenis Wong Portillo.

**Lima, 2024**

**INDICE**

[1. INTRODUCCIÓN 2](#_Toc429049484)

[1.1. Situación de la empresa: 3](#_Toc1978413826)

[1.2. Problemática: 4](#_Toc1991095239)

[2. GESTIÓN DE DOCUMENTOS 6](#_Toc65479128)

[2.1. Definir los roles de la GCS (Rol / descripción, es decir, responsabilidades) 6](#_Toc2034506395)

[2.2. Lista de 6 herramientas de la Gestión de la Configuración. 8](#_Toc1776904401)

[2.2.1. Ventajas y desventajas de 6 herramientas de GCS: 11](#_Toc856029123)

[2.3. Ventajas y desventajas de la herramienta seleccionada: Bitbucket 15](#_Toc1212216615)

[2.3.1. Ventajas de Bitbucket: 16](#_Toc1195442717)

[2.3.2. Desventajas de Bitbucket: 16](#_Toc190998814)

[3. ACTIVIDADES DE LA GESTION 17](#_Toc23931738)

[3.1. Identificación 17](#_Toc2821397)

[3.1.1. Clasificación (Inventario de los ítems) 18](#_Toc2011254458)

[3.1.2. Nomenclatura 20](#_Toc909614029)

[3.1.3. Diseño de repositorio 22](#_Toc172081195)

[3.1.4. Línea Base 22](#_Toc1389419606)

[3.2. Control 25](#_Toc1709284922)

[3.2.1. Solicitudes de cambio 26](#_Toc706413412)

[3.2.2. Proceso de gestión de cambios 31](#_Toc1662447470)

[3.3. Estado de la configuración 31](#_Toc309569143)

[REFERENCIAS 31](#_Toc883623345)

# **INTRODUCCIÓN**

## **Situación de la empresa:**

SeguriTech, con más de dos décadas de trayectoria, se ha consolidado como un referente en el mercado del desarrollo de software especializado en asegurar la calidad de la práctica de Gestión de Configuración de Software. Durante estos años, hemos liderado numerosos proyectos, 20 en total, tanto en el sector público como en el privado, ofreciendo soluciones innovadoras que abordan eficazmente los desafíos de la seguridad en la ciudad de Lima.

Nuestro enfoque se distingue por la calidad y la facilidad de uso de nuestros productos, lo que nos ha permitido ganar la confianza de una amplia gama de clientes. Desde organismos gubernamentales hasta empresas privadas, nuestros clientes confían en nuestra capacidad para proporcionar herramientas que les permitan gestionar y comunicar de manera rápida y eficiente los problemas relacionados con la seguridad ciudadana.

En nuestro proceso de desarrollo, nos destacamos por adoptar buenas prácticas y metodologías híbridas que garantizan la calidad y la eficiencia en cada entrega. Además, hemos integrado tecnologías de vanguardia en nuestra infraestructura, como el sistema de almacenamiento en la nube de Microsoft, que no solo reduce nuestros costos operativos, sino que también permite a nuestro talentoso equipo de desarrolladores trabajar de manera remota, maximizando así nuestra flexibilidad y capacidad de respuesta ante las necesidades del mercado.

En el contexto actual, donde la demanda de soluciones de seguridad ciudadana está en constante aumento, nuestra experiencia y compromiso con la excelencia nos sitúan en una posición única para ofrecer un sólido y sintetizado Plan de Gestión de la Configuración del Software. Este plan no sólo refleja nuestra profunda comprensión de las complejidades de la gestión de la configuración del software, sino que también demuestra nuestro compromiso continuo con la satisfacción del cliente y la mejora constante de nuestros procesos

## **Problemática:**

El principal problema radica en el uso incorrecto del sistema de control de versiones GitHub, puesto que todos los trabajadores suelen trabajar en la rama Main, lo cual resulta tedioso porque deben hacer un *pull request*para poder realizar los cambios respectivos. Asimismo, gran parte de los desarrolladores no hacen los commits especificando los cambios que realizan. Por otro lado, resulta difícil para los desarrolladores y otros miembros del equipo encontrar y gestionar la información de manera eficiente. Cuando hay una gran cantidad de proyectos y documentación mezclados en un solo lugar, la navegación se vuelve confusa y la gestión del código y la documentación se vuelve más complicada. Por ejemplo, si un desarrollador necesita encontrar el código fuente de un proyecto específico o la documentación relacionada con ese proyecto, tendría que buscar a través de todo el repositorio central, lo que puede llevar mucho tiempo y resultar en una pérdida de productividad. Del mismo modo, la actualización y mantenimiento del repositorio se vuelve más complejo debido a la gran cantidad de contenido que debe ser gestionado.

**En síntesis:**

* **Uso inadecuado del sistema de control de versiones (GitHub):** El principal reto consiste en cambiar la cultura de trabajo en lo que respecta al control de versiones. Los trabajadores deben comprender la importancia de trabajar en ramas que se encuentren separadas y utilizar pull requests para discutir y revisar los cambios, previamente a fusionarlos con la rama principal. Esto requiere una capacitación adecuada y la creación de procedimientos claros y que sean sencillos de seguir.
* **Falta de documentación en los commits:** La falta de documentación en los commits dificulta la comprensión de los cambios realizados en el código. En ese sentido, resulta fundamental establecer pautas y estándares para que los desarrolladores incluyan descripciones claras y concisas de los cambios realizados en cada commit. Esto mejorará la trazabilidad y facilitará la identificación de errores o problemas en el código fuente.
* **Organización del repositorio:** La combinación de muchos proyectos con la documentación completa en un repositorio central dificulta la navegación y la gestión eficiente del código y la documentación. En ese sentido, resulta necesario reorganizar el repositorio para separar claramente los proyectos del contenido de documentación asociado. Esto mejorará la visibilidad y facilitará el acceso a la información relevante para cada uno de los proyectos.
* **Adopción de prácticas de revisión de código:** Es clave implementar prácticas de revisión de código sistemáticas y regulares con la finalidad de asegurar la calidad del código fuente y fomentar el intercambio de conocimientos entre los miembros de los diferentes equipos de desarrollo. Esto implica establecer criterios claros para la revisión de código y realizar la asignación de responsabilidades específicas que les corresponde a cada miembro del equipo.
* **Cambio cultural y capacitación:** Abordar estos desafíos requerirá un cambio cultural dentro del equipo de desarrollo. En ese sentido, es necesario fomentar una cultura de colaboración, transparencia y responsabilidad relacionados con el control de versiones y la gestión del código. La capacitación continua y la comunicación efectiva son clave para garantizar la adopción exitosa de nuevas prácticas y procesos.

**La empresa ha enfrentado desafíos entre las cuales incluye:**

* Dificultades para mantener la coherencia entre versiones del software.
* Problemas de seguimiento y control de cambios en el código fuente.
* Falta de documentación adecuada sobre las configuraciones y versiones del software.
* Riesgo de pérdida de datos debido a un sistema de almacenamiento no centralizado
  1. **Objetivos:**
* Facilitar el seguimiento y control de cambios durante el ciclo de vida del desarrollo del software.
* Mejorar la calidad y confiabilidad del software entregado a los clientes.
* Asegurar la integridad y coherencia del software en diferentes entornos y versiones.
* Optimizar el uso de recursos, incluyendo el almacenamiento en la nube, para reducir costos y aumentar la eficiencia operativa.

Este plan proporcionará un marco estructurado para administrar los cambios en el software, garantizando la integridad, la consistencia y la trazabilidad de los artefactos de software en todas las etapas del ciclo de vida del desarrollo. La gestión efectiva de la configuración del software es clave para garantizar la calidad, la confiabilidad y la eficiencia de los productos de SeguriTech, así como para mantener la satisfacción del cliente y la competitividad en el mercado. Este plan brindará el marco necesario para lograr estos objetivos y promover una cultura de mejora continua en las distintas áreas de la empresa.

# **GESTIÓN DE DOCUMENTOS**

## **Definir los roles de la GCS (Rol / descripción, es decir, responsabilidades)**

|  |  |  |
| --- | --- | --- |
|  | **Descripción** | **Responsabilidades** |
| **Gestor de Configuración** | El gestor de configuración es la persona responsable de definir y mantener los procesos, alcances y políticas de gestión de la configuración del software durante todo el ciclo de vida del proyecto | 1. Desarrollar y mantener políticas y procedimientos de gestión de la configuración. 2. Supervisar la implementación de las políticas de gestión de la configuración. 3. Coordinar con otros equipos de trabajo para garantizar la integridad de la configuración del software. 4. Gestionar herramientas y sistemas de gestión de la configuración. |
| **Bibliotecario de Configuración** | El bibliotecario de configuración es la persona responsable de almacenar y mantener el repositorio central o principal de activos de configuración del software, como por ejemplo el código fuente, los documentos y los artefactos relacionados. | 1. Almacenar y organizar los activos de configuración en un repositorio centralizado. 2. Mantener el control de versiones de los activos de configuración. 3. Facilitar el acceso controlado a los activos de configuración según sea necesario. 4. Garantizar la integridad y seguridad del repositorio de configuración. |
| **Auditor de Configuración** | El auditor de configuración es la persona responsable de realizar auditorías de forma periódica con la finalidad de asegurar el cumplimiento de los procedimientos y políticas de gestión de la configuración. | 1. Realizar auditorías de configuración para estudiar y evaluar el cumplimiento de políticas y procedimientos. 2. Identificar y reportar desviaciones o incumplimientos de las políticas de gestión de la configuración. 3. Proporcionar recomendaciones para mejorar los procesos de gestión de la configuración. 4. Colaborar con el equipo de gestión de la configuración a fin de implementar acciones correctivas. |
| **Especialista en Soporte de Configuración (CSS, por sus siglas en inglés: Configuration Support Specialist):** | El especialista en soporte de configuración es aquel que se encarga de proporcionar soporte técnico y administrativo para las actividades relacionadas con la gestión de la configuración del software. | 1. Brindar asistencia técnica para el uso de herramientas de gestión de la configuración. 2. Ayudar en la configuración y mantenimiento de sistemas de gestión de la configuración. 3. Proporcionar capacitación y orientación sobre políticas y procedimientos de gestión de la configuración a los equipos de trabajo. 4. Colaborar con otros equipos para resolver problemas relacionados con la configuración del software. |

## **Lista de 6 herramientas de la Gestión de la Configuración.**

|  |  |  |
| --- | --- | --- |
| **Herramienta** | **Fuente** | **Características** |
| GitLab | Open Source | * Soporte para la gestión de pipelines de CI/CD. * Funcionalidades avanzadas de seguridad, como escaneo de código estático y análisis de dependencias. * Integración con herramientas de gestión de proyectos como Jira y Trello. * Capacidades de monitoreo y rendimiento para aplicaciones. * Gestión avanzada de permisos y roles de usuario. |
| Perforce | Propietaria (Perforce Software) | * Control de acceso granular para archivos y carpetas. * Capacidades avanzadas de administración de configuración y distribución de artefactos. * Integración con una amplia gama de herramientas de desarrollo y flujo de trabajo. * Soporte para la gestión de grandes volúmenes de datos. * Herramientas de revisión de código y colaboración en equipo |
| Git Kraken | Propietaria (Axosoft) | * Interfaz de usuario visual e intuitiva para la gestión de repositorios Git. * Herramientas avanzadas de fusión y resolución de conflictos. * Capacidades de gestión de tareas y tableros Kanban integrados. * Integración con servicios de alojamiento de repositorios en la nube. * Funcionalidades de revisión de código y comentarios en línea. |
| Plastic SCM | Propietaria (Codice Software) | * Soporte para una amplia variedad de flujos de trabajo de desarrollo, incluidos ramificación y fusión avanzadas. * Capacidades de seguimiento y gestión de cambios granulares. * Integración con herramientas de terceros como Jira, Jenkins y Visual Studio. * Funcionalidades de control de versiones distribuido. * Herramientas de análisis de rendimiento y optimización del proceso de desarrollo. |
| Azure DevOps | Propietaria (Microsoft) | * Integración profunda con el ecosistema de Microsoft, incluidas herramientas como Visual Studio y Azure. * Capacidades completas de gestión de proyectos ágiles, incluyendo tableros Kanban y seguimiento de sprints. * Amplias funcionalidades de colaboración, incluida la integración de repositorios, seguimiento de problemas y wikis. * Herramientas de análisis e informes integrados para el seguimiento del rendimiento del equipo y del proyecto. * Soporte para la automatización de la entrega de software a través de pipelines de CI/CD. |
| Apache Subversion | Open Source | * Control de versiones centralizado con soporte para la gestión de archivos binarios y textuales. * Historial de cambios detallado y capacidad de revertir a versiones anteriores. * Integración con una variedad de herramientas de desarrollo y sistemas operativos. * Soporte para la definición de propiedades personalizadas en archivos y directorios. * Acceso remoto a través de los protocolos HTTP y SVN. |
| Bitbucket | Propietaria (Atlassian) | * Integración estrecha con el ecosistema de Atlassian, incluidas herramientas como Jira, Confluence y Trello. * Soporte para la gestión de repositorios Git y Mercurial. * Funcionalidades avanzadas de revisión de código con comentarios en línea y flujos de trabajo personalizados. * Capacidades de automatización a través de pipelines de CI/CD integrados. * Herramientas de seguimiento de problemas y gestión de proyectos ágiles. |

### **Ventajas y desventajas de 6 herramientas de GCS:**

#### **GitLab:**

##### **Ventajas:**

* GitLab ofrece una amplia gama de características, esto incluye la gestión de repositorios, seguimiento de problemas, integración continua y entrega continua (CI/CD), de modo que representa una solución efectiva e integral.
* Es de código abierto y ofrece tanto versiones de pago como también gratuitas, de modo que es accesible para diferentes tipos de usuarios y organizaciones.
* Su interfaz de usuario es intuitiva y fácil de utilizar, de modo que facilita la colaboración dentro de equipos de desarrollo.

##### **Desventajas:**

* La curva de aprendizaje puede ser pronunciada para aquellos nuevos en Git y sistemas de control de versiones distribuidos.
* Pese a que existe una versión gratuita, algunas características más avanzadas están disponibles solamente en las versiones de pago.
* En entornos empresariales muy grandes, puede requerir una configuración y mantenimiento más avanzados.

#### **Perforce:**

##### **Ventajas:**

* Es conocido por su rendimiento y escalabilidad, de modo que es ideal para grandes equipos y proyectos de software.
* Ofrece una amplia gama de características para la gestión de configuraciones, esto incluye control de versiones, seguimiento de problemas y colaboración.
* Permite una gestión avanzada de ramas y fusiones, de modo que facilita el desarrollo paralelo de características.

##### **Desventajas:**

* Su interfaz de usuario puede ser menos intuitiva en comparación con otras herramientas, es así como se necesita tiempo adicional para aprender a utilizarla.
* Tiene un costo más significativo, especialmente para empresas más pequeñas o equipos con presupuestos limitados.
* Se requiere una infraestructura de servidor más dedicada, es así como puede aumentar los costos y la complejidad de implementación.

#### **Git Kraken:**

##### **Ventajas:**

* Posee una interfaz de usuario visualmente atractiva y fácil de utilizar, que es especialmente apreciada por los nuevos usuarios de Git.
* Ofrece integración con servicios de terceros como GitHub, GitLab y Bitbucket, de modo que facilita la colaboración con diferentes plataformas en tiempo real.
* Proporciona características de productividad adicionales, como por ejemplo la gestión de tarjetas Kanban y la integración con aplicaciones de comunicación como Slack.

##### **Desventajas:**

* Existen algunas de sus características más avanzadas, como la gestión de ramas y fusiones, que pueden ser menos sofisticadas, en contraste con otras herramientas.
* La versión gratuita tiene limitaciones en cuanto al número de repositorios y colaboradores, lo cual lo hace no tan ideal para el desarrollo de proyectos complejos.
* No es tan ampliamente empleado en entornos empresariales como GitLab o Perforce, y por ende, limita la disponibilidad de recursos de soporte y documentación, es decir, tiene menos recursos de soporte disponibles, como comunidades en línea activas, foros de discusión o documentación detallada. Esto trae como consecuencia una mayor dificultad para la resolución de problemas o la búsqueda de información sobre cómo utilizar SVN en comparación con herramientas más populares.

#### **Plastic SCM:**

##### **Ventajas:**

* Ofrece una diversidad de características avanzadas, como por ejemplo una arquitectura distribuida, ramas y fusiones semánticas, y soporte para grandes archivos binarios, lo cual la convierte en una herramienta ideal para proyectos de desarrollo de videojuegos y multimedia.
* Posee una interfaz gráfica que es intuitiva y fácil de comprender, esto facilita la visualización y gestión de ramas y gestión de cambios.
* Proporciona integración con IDE populares como Visual Studio y Unity, de modo que ayuda a simplificar el flujo de trabajo para los desarrolladores.

##### **Desventajas:**

* Tiene curva de aprendizaje más pronunciada, especialmente para aquellos acostumbrados a herramientas más tradicionales como Git o SVN.
* Su adopción puede ser limitada, en contraste con herramientas más establecidas como GitLab o Perforce, lo que puede afectar la disponibilidad de recursos de soporte y comunidad.
* Algunas de sus características avanzadas pueden no ser necesarias para todos los proyectos, lo que puede hacer que la herramienta sea excesivamente compleja para algunos equipos.

#### **Azure DevOps:**

##### **Ventajas:**

* Proporciona una suite completa de herramientas para la gestión de proyectos de software, esto incluye control de versiones, seguimiento de problemas, integración continua y entrega continua (CI/CD).
* Brinda integración con otros servicios de Microsoft, como Azure Cloud, Visual Studio y Teams, de forma que facilita la colaboración y la gestión de proyectos dentro de un ecosistema integrado.
* Asimismo, es altamente adaptable y escalable. Por un lado, Azure DevOps ofrece una variedad de características y funcionalidades que pueden adaptarse y personalizarse para satisfacer las necesidades específicas de un proyecto o equipo. Por otro lado, la herramienta tiene la capacidad de manejar un aumento en la cantidad de usuarios, el tamaño del código base, la complejidad del proyecto y las demandas de procesamiento sin experimentar problemas en el rendimiento o la funcionalidad.

##### **Desventajas:**

* Tiene una curva de aprendizaje marcada, especialmente para aquellos que no están familiarizados con el entorno de Microsoft y su conjunto de herramientas.
* Algunas de sus características pueden resultar muy sobrecargadas para proyectos más pequeños o equipos que solo requieren funcionalidades básicas de gestión de configuración de software.
* Pese a que ofrece una capa gratuita con ciertas limitaciones, muchas de sus características avanzadas requieren una suscripción de paga, de modo que resulta siendo costoso para las organizaciones.

#### **Apache Subversion (SVN):**

##### **Ventajas:**

* Es una herramienta estable y madura que ha sido ampliamente empleada en la industria de software durante muchos años, lo que la hace en una opción confiable para proyectos de software.
* Tiene una curva de aprendizaje menos empinada, en contraste con sistemas de control de versiones distribuidos como Git, de modo que es óptima para equipos nuevos en la gestión de configuración de software.
* Ofrece características básicas de control de versiones, como control de archivos, ramas y fusiones, que son muy útiles para muchos proyectos pequeños o medianos.

##### **Desventajas:**

* No posee ciertas características avanzadas ofrecidas por herramientas modernas como GitLab o Perforce, como la gestión de ramas y fusiones más sofisticadas.
* Al tener una arquitectura centralizada esto puede resultar menos flexible puesto que todos los archivos y la historia del repositorio residen en un único servidor central. Esto puede resultar en limitaciones cuando se trata de equipos distribuidos geográficamente, dado que que cada una de las operaciones de control de versiones debe comunicarse con este servidor central. Esto puede ocasionar latencia en la comunicación y puede no ser óptimo para equipos que trabajan en diferentes ubicaciones geográficas, especialmente si la conectividad no es ideal.
* Su desarrollo y soporte pueden ser menos activos en comparación con herramientas más modernas. Esto significa que pueden pasar más tiempo entre las actualizaciones de software y es posible que no se implementen nuevas características tan rápidamente como en las herramientas más modernas y populares.

## **Ventajas y desventajas de la herramienta seleccionada: Bitbucket**

### **Ventajas de Bitbucket:**

* **Control de versiones:** Bitbucket facilita el control de versiones del código fuente, lo que permite a los equipos mantener un registro de los cambios realizados en el código y revertir a versiones anteriores si es necesario.
* **Colaboración:** Permite a los equipos colaborar de manera efectiva en el desarrollo de software al proporcionar características como ramificaciones (branches), fusiones (merges) y solicitudes de extracción (pull requests).
* **Integración con herramientas de desarrollo:** Bitbucket se integra fácilmente con otras herramientas de desarrollo como Jira y Trello, lo que facilita la gestión de proyectos completos desde el inicio hasta el despliegue.
* **Control de acceso:** Ofrece opciones de control de acceso granular, lo que permite a los equipos controlar quién puede ver, editar o fusionar el código en el repositorio.
* **Revisión de código:** Facilita la revisión de código mediante la funcionalidad de solicitudes de extracción, lo que permite a los desarrolladores revisar y comentar el código de otros antes de fusionarlo con la rama principal.

### **Desventajas de Bitbucket:**

* **Curva de aprendizaje:** Para los nuevos usuarios, Bitbucket puede tener una curva de aprendizaje empinada, especialmente si no están familiarizados con el control de versiones o el uso de herramientas similares.
* **Limitaciones en planes gratuitos:** Aunque Bitbucket ofrece un plan gratuito, tiene ciertas limitaciones en cuanto al número de colaboradores y almacenamiento, lo que puede ser restrictivo para equipos más grandes o proyectos a gran escala.
* **Dependencia de la conexión a internet:** Como es una herramienta basada en la nube, Bitbucket depende de una conexión a Internet estable, lo que puede ser problemático en entornos donde la conectividad es irregular o limitada.
* **Personalización limitada:** Aunque Bitbucket ofrece una variedad de características, la personalización puede ser limitada en comparación con otras herramientas más avanzadas de gestión de la configuración de software.

# **ACTIVIDADES DE LA GESTION**

En la Gestión de Configuración de Software, los diferentes tipos de items que se manejan suelen clasificarse en varias categorías según su naturaleza y función dentro del proceso de desarrollo y mantenimiento del software. Basándome en lo que proporcionaste, aquí tienes una breve descripción de cada tipo de ítem:

* **Ítems en Evolución:** Son documentos o archivos ejecutables que están sujetos a cambios y revisiones a lo largo del ciclo de vida del software. Estos pueden incluir especificaciones de requisitos, diseños, código fuente, archivos binarios, entre otros. Estos items son modificados y actualizados conforme avanza el proyecto de desarrollo de software.
* **Ítems Tipo Fuente:** Estos son los componentes fundamentales del software, como el código fuente, scripts, archivos de configuración, plantillas, entre otros. Los items tipo fuente son aquellos que son modificados directamente por los desarrolladores y forman la base del software que se está desarrollando.
* **Items de Soporte:** Son documentos que sirven como respaldo o apoyo en caso de que ocurran errores o problemas con el sistema en general. Estos pueden incluir manuales de usuario, guías de instalación, documentación técnica, procedimientos de recuperación ante fallos, entre otros. Los items de soporte son importantes para facilitar la operación y mantenimiento del software una vez que está en producción.

## **Identificación**

### **Clasificación (Inventario de los ítems)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipo (E: Evolución, F:Fuente, S:Soporte)** | **Item** | **EAT** | **Proyecto** |
| E | Documento de Project Charter | .DOCX | RP |
| E | Cronograma del proyecto | .XLSX | RP |
| E | Documento de Negocios | .DOCX | RP |
| E | Documento de Elicitación de Requisitos | .XLSX | RP |
| E | Documento de Especificación de Requisito 1 | .DOCX | RP |
| E | Documento de Especificación de Requisito 2 | .DOCX | RP |
| E | Documento de Especificación de Requisito 3 | .DOCX | RP |
| E | Documento de Especificación de Requisito 4 | .DOCX | RP |
| E | Documento de Especificación de Requisito 5 | .DOCX | RP |
| E | Documento de Especificación de Requisito 6 | .DOCX | RP |
| E | Documento de Especificación de Requisito 7 | .DOCX | RP |
| E | Documento de Especificación de Requisito 8 | .DOCX | RP |
| E | Documento de Diseño arquitectónico | .DOCX | RP |
| E | Documento de Diseño de Interfaz | .DOCX | RP |
| E | Documento de Especificación de la base de datos | .DOCX | RP |
| E | Documento de Diseño de interfaz | .DOCX | RP |
| E | Documento de Especificación de la base de datos | .DOCX | RP |
| E | Documento de Diseño de interfaz | .DOCX | RP |

### **Nomenclatura**

La nomenclatura de nuestro proyecto de Software “Reporta PE” hace referencia al conjunto de reglas o convenciones establecidas para nombrar los diferentes elementos dentro del proyecto, como carpetas, archivos, entre otros. Esta tiene como propósito proporcionar al equipo de desarrollo una mayor comprensión del proyecto y evitar confusiones o errores durante el desarrollo y la revisión del código.  
En este proyecto, seguimos una serie de convenciones de nomenclatura para garantizar la consistencia y la claridad en el código y la documentación. Estas convenciones se aplican a variables, funciones, clases, archivos y otros elementos del proyecto. A continuación, presentaremos un ejemplo de uso de la nomenclatura   
RP\_DN\_DOCX

Donde:

RP: Reporta Pe  
DN: Documento de Negocio  
DOCX: Extensión de archivo asociada a Microsoft Word

Dado el ejemplo se ha dado uso de la nomenclatura siguiendo una serie de reglas que presentaremos a continuación.

#### **Reglas** **:**

Documentos:

**Regla Nº1:** Para documentos específicos del proyecto

Identificador: **ACRÓNIMODELPROYECTO\_NOMBREDELDOCUMENTO.EXTENSIÓN**

Ejemplo: "RP\_RS.docx" para el documento de Reportar Suceso en el proyecto de Reporta PE.

**Regla Nº2:** Documentos no especificados

Identificador: **ACRÓNIMO DEL DOCUMENTO.EXTENSIÓN**

ejemplo: MI.docx(modelación de interfaz)

**Regla Nº3:** Para documentos relacionados con la misma temática y que requieran numeración:

Identificador:**ACRÓNIMODELPROYECTO\_NOMBREDELDOCUMENTO\_NÚMERO.EXTENSIÓN**

Ejemplo: "RP\_RS\_001.docx" para una versión específica del documento de Reportar Suceso.

**Regla Nº4 :** Para añadir estados de un documento agregado al repositorio como Borrador, Final, Revisado, Aprobado

Identificador: **ACRÓNIMODELPROYECTO\_NOMBREDELDOCUMENTO\_ESTADO.EXTENSIÓN**

Ejemplo: “RP\_RS\_Borrador.docx para una versión en borrador o avance del documento de Reportar Suceso.

Observaciones

Si existen documentos que coinciden con la nomenclatura, se agregará la siguiente letra de la palabra (por ejemplo, "RSR" para Reportar Suceso Revisado).

### **Diseño de repositorio**

|  |
| --- |
|  |

### **Línea Base**

**Hito 1: Definición y Planificación del Proyecto**

Este hito marca el inicio del proyecto "Reporta Pe" (RP) y se centra en establecer una base sólida para el desarrollo de la red social de denuncias. Se han creado varios documentos fundamentales, incluido el Documento de Project Charter (RP\_PC.docx), que establece la visión, los objetivos y el alcance del proyecto. También se ha elaborado un Cronograma del proyecto (RP\_CP.xlsx) para organizar las actividades y los plazos. El Documento de Negocios (RP\_DN.docx) proporciona una descripción detallada del contexto empresarial y las necesidades a abordar. Además, se ha llevado a cabo un exhaustivo proceso de Elicitación de Requisitos, reflejado en múltiples Documentos de Elicitación de Requisitos (RP\_DER.xlsx), así como en Documentos de Especificación de Requisito (RP\_DER1.docx, RP\_DER2.docx, RP\_DER3.docx, RP\_DER4.docx, RP\_DER5.docx, RP\_DER6.docx, RP\_DER7.docx, RP\_DER8.docx) que detallan los requisitos del sistema. También se han desarrollado documentos de Diseño arquitectónico (RP\_DDA.docx), Diseño de Interfaz (RP\_DDl.docx), y Especificación de la base de datos (RP\_DEBD.docx), sentando así las bases técnicas del proyecto.

**Hito 2: Preparación de la Estrategia de Pruebas**

En este hito, se concentra en la elaboración de la estrategia y protocolos de pruebas para garantizar la calidad del producto final. Se han desarrollado múltiples Documentos de Protocolo de Pruebas (RP\_DPP.docx) que establecen los procedimientos y criterios para realizar pruebas exhaustivas en diferentes aspectos del sistema, como funcionalidad, rendimiento y seguridad. Estos documentos son fundamentales para asegurar que el sistema cumpla con los requisitos y expectativas establecidos.

**Hito 3: Cierre del Proyecto**

Este hito marca el final del proyecto "Reporta Pe" (RP) y se centra en documentar y formalizar el cierre del proyecto. Se han elaborado Actas de cierre del proyecto (RP\_ACP.docx) que detallan las actividades realizadas, los resultados obtenidos y las lecciones aprendidas durante el desarrollo del proyecto. Estas actas son importantes para proporcionar una conclusión formal al proyecto y permitir una transición adecuada hacia la fase de operación y mantenimiento del sistema.

#### **Tabla de Líneas Base del Proyecto Red Social de Denuncias**

|  |  |  |
| --- | --- | --- |
| **N° de Línea Base** | **Intervalo de Fecha** | **Artefactos** |
| **PLANIFICACIÓN** | **Fecha de inicio: 01/04/2024**  **Fecha de cierre:**  **30/04/2024** | 1. Elaboración del Project Charter 2. Elaboración del cronograma del proyecto 3. Elaboración de documento de negocio 4. Definición de requisitos funcionales 5. Estructuración de proyecto en GitHub 6. Especificar Requisito N°1 - Reportar suceso 7. Especificar Requisito N°2 - Ver mis reportes 8. Especificar Requisito N°3 - Mostrar mapa de seguridad 9. Especificar Requisito N°4 - Visualizar reportes guardados 10. Especificar Requisito N°5 - Buscar reporte 11. Especificar Requisito N°6 - Mostrar contactos seguros 12. Especificar Requisito N°7 - Buscar contacto 13. Especificar Requisito N°8 - Iniciar Denuncia 14. Verificación y revisión de documentación 15. Refinamiento de los Requisitos |
| **LB2** | **Fecha de inicio: 01/05/2024**  **Fecha de cierre:**  **06/06/2024** | 1. Implementación de requisito N°1 - Reportar suceso 2. Implementación de requisito N°2 - Ver mis reportes 3. Implementación de requisito N°3 - Mostrar mapa de seguridad 4. Implementación de requisito N°4 - Visualizar reportes guardados 5. Implementación de requisito N°5 - Buscar reporte 6. Implementación de requisito N°6 - Mostrar contactos seguros 7. Implementación de requisito N°7 - Buscar contacto 8. Implementación de requisito N°8 - Iniciar denuncia 9. Sprint Retrospective 2 10. Generar Baseline 02 |
| **LB2** | **Fecha de inicio:06/06/2024**  **Fecha de cierre:**  **16/06/2024** | 1. Pruebas de Experiencia de Usuario Pruebas de Funcionalidades 2. Pruebas de Interfaz de usuarioPruebas de Base de Datos 3. Sprint Retrospective 3 4. Generar Baseline 03 |
|
| **LB3** | **Fecha de inicio: 16/06/2024**  **Fecha de cierre:**  **24/06/2024** | 1. Preparación para el despliegue 2. Realización de pruebas finales 3. Transferencia del producto 4. Entrega de documentación de cierre |
|

## **Control**

### **Solicitudes de cambio**

A continuación, se presentan 7 ejemplos de las solicitudes de cambio del proyecto “Reporta Pe”

#### **N° 1**

|  |  |
| --- | --- |
| **ID** | 01 |
| **Fecha** | 23/05/2024 |
| **Sistema** | Reporta Pe |
| **Stakeholder** | Usuario Final |
| **Dueño del proceso** | Jefe de Soporte |
| **Descripcion** | Integrar un botón “recuperar contraseña” el cual redirigirá al usuario a una ventaja emergente que solicitará el correo electrónico para la recuperación de contraseña. |
| **Justificacion** | Establecer una medida de mayor seguridad y la facilitar el acceso para los usuarios que olviden su contraseña |

#### **N° 2**

|  |  |
| --- | --- |
| **ID** | 02 |
| **Fecha** | 23/05/2024 |
| **Sistema** | Reporta Pe |
| **Stakeholder** | Usuario Final |
| **Dueño del proceso** | Jefe de Análisis |
| **Descripcion** | Añadir un módulo de “Seguimiento de denuncia” para que los denunciantes puedan ver el estado de sus denuncias en tiempo real. |
| **Justificacion** | El usuario pueda dar seguimiento y ver el estado actual en el que se encuentra se procesó de denuncia y generar más confianza. |

#### **N° 3**

|  |  |
| --- | --- |
| **ID** | 03 |
| **Fecha** | 23/05/2024 |
| **Sistema** | Reporta Pe |
| **Stakeholder** | Usuario Final |
| **Dueño del proceso** | Jefe de Operaciones |
| **Descripcion** | Incorporar una funcionalidad de chat en vivo para que los ciudadanos puedan recibir asistencia inmediata al realizar una denuncia |
| **Justificacion** | Proveer soporte en tiempo real y guiar a los usuarios durante el proceso de denuncia |

#### **N° 4**

|  |  |
| --- | --- |
| **ID** | 04 |
| **Fecha** | 23/05/2024 |
| **Sistema** | Reporta Pe |
| **Stakeholder** | Usuario final, frontend |
| **Dueño del proceso** | Diseñador UX/UI |
| **Descripcion** | Modificación en la Interfaz de Usuario  Se propone realizar ajustes en la interfaz de usuario para mejorar la usabilidad y la experiencia del usuario. Esto incluye reorganizar los elementos de la página de inicio |
| **Justificacion** | Destacar las funciones principales de denuncia y búsqueda de reportes, así como simplificar el proceso de registro y acceso a la plataforma. |

#### **N° 5**

|  |  |
| --- | --- |
| **ID** | 05 |
| **Fecha** | 23/05/2024 |
| **Sistema** | Reporta Pe |
| **Stakeholder** | Usuario Final |
| **Dueño del proceso** | Jefe de Análisis |
| **Descripción** | Implementar dentro del panel de Realizar denuncia un campo de “Localización actual” antes del botón enviar denuncia |
| **Justificación** | Identificar la ubicación exacta del incidente y mejorar el nivel de precisión al reportar el incidente |

#### **N° 6**

|  |  |
| --- | --- |
| **ID** | 06 |
| **Fecha** | 23/05/2024 |
| **Sistema** | Reporta Pe |
| **Stakeholder** | Usuario final ,diseñador UX/UI |
| **Dueño del proceso** | Equipo de desarrollo |
| **Descripcion** | Incorporación de Funcionalidad de Comentarios |
| **Justificacion** | Se propone agregar la capacidad para que los usuarios comenten en los reportes de denuncias. Esto permitirá una mayor interacción entre los usuarios, la posibilidad de proporcionar más detalles o aclaraciones sobre un reporte específico, y fomentará la participación en la comunidad de la red social. |

#### **N° 7**

|  |  |
| --- | --- |
| **ID** | 07 |
| **Fecha** | 23/05/2024 |
| **Sistema** | Reporta PE |
| **Stakeholder** | Equipo de Desarrollo, Usuarios Finales, Analista QA |
| **Dueño del proceso** | Equipo de Desarrollo |
| **Descripcion** | Se propone integrar un sistema de verificación de identidad para los usuarios que deseen denunciar en la plataforma Reporta PE. Este sistema requerirá que los usuarios verifiquen su identidad mediante correo electrónico, número de teléfono o verificación de identidad biométrica antes de presentar una denuncia. |
| **Justificacion** | La integración de un sistema de verificación de identidad es fundamental para garantizar la autenticidad y la fiabilidad de las denuncias presentadas en la plataforma Reporta PE. Al requerir que los usuarios verifiquen su identidad antes de presentar una denuncia, se reduce el riesgo de presentaciones falsas o fraudulentas |

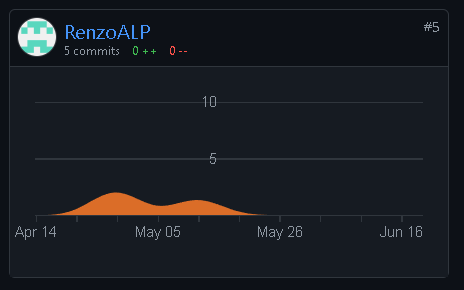
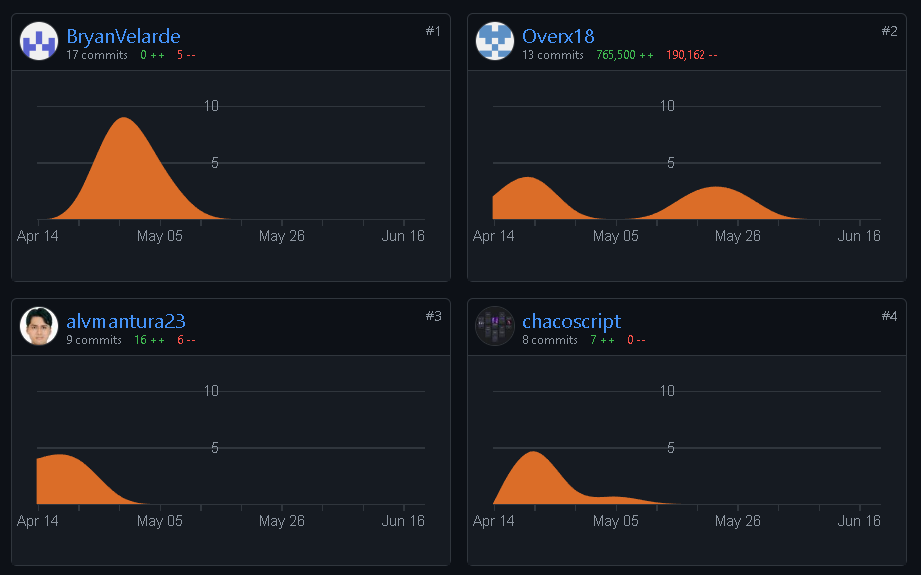
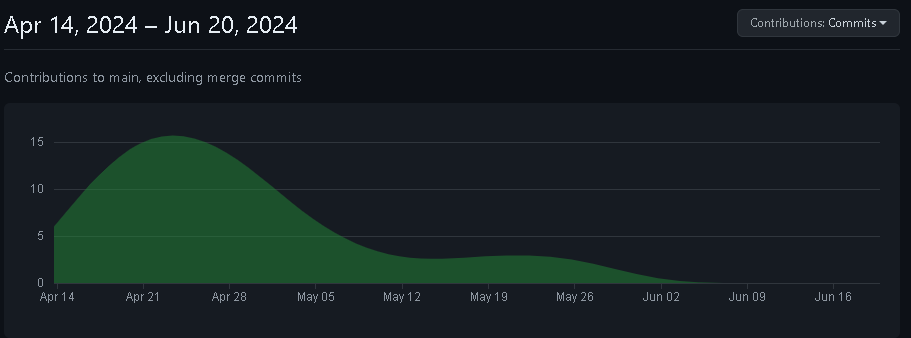
|  |  |
| --- | --- |
| **ID** | 08 |
| **Fecha** | 23/05/2024 |
| **Sistema** | Reporta PE |
| **Stakeholder** | Equipo de Desarrollo, Administradores del Sistema, Usuarios Finales |
| **Dueño del proceso** | Equipo de Desarrollo |
| **Descripcion** | Se propone la inclusión de un sistema de notificaciones en tiempo real en la plataforma Reporta PE. Este sistema enviará notificaciones automáticas a los usuarios cuando se realicen actualizaciones importantes en sus reportes de denuncias, como cambios en el estado de la denuncia, comentarios adicionales de otros usuarios o acciones tomadas por las autoridades competentes. |
| **Justificacion** | La inclusión de un sistema de notificaciones en tiempo real es esencial para mejorar la experiencia del usuario y la eficacia de la plataforma. Al proporcionar a los usuarios actualizaciones instantáneas sobre sus denuncias, se promueve una mayor confianza y satisfacción del usuario al demostrar un compromiso con la transparencia y la atención a sus preocupaciones. |

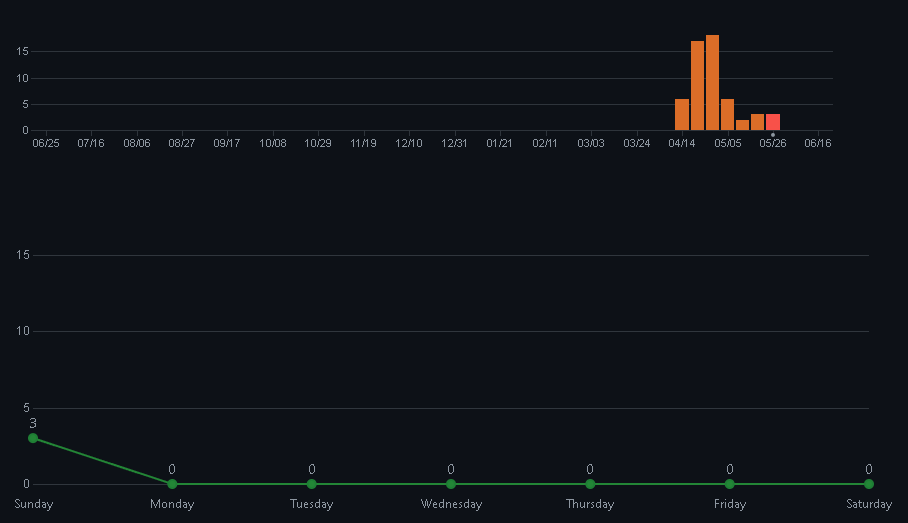
#### **N° 8**

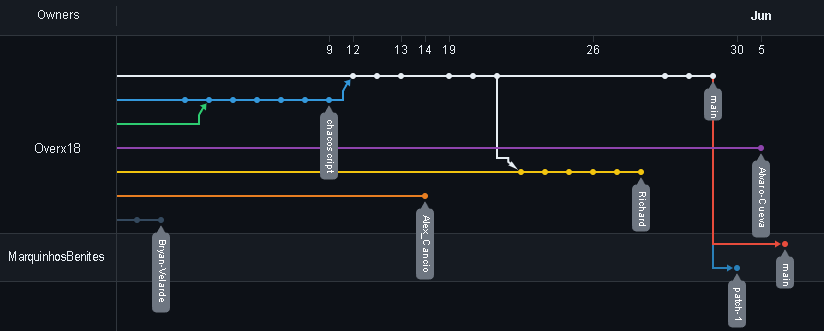
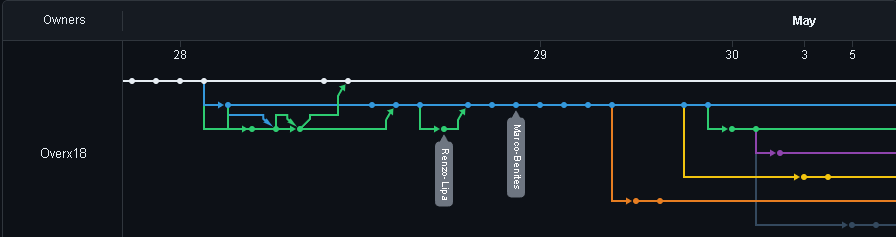
|  |  |
| --- | --- |
| **ID** | 09 |
| **Fecha** | 23/05/2024 |
| **Sistema** | Reporta Pe |
| **Stakeholder** | Equipo de Seguridad Interna |
| **Dueño del proceso** | Gerente de Seguridad |
| **Descripcion** | Implementar una funcionalidad de "Notificación de Incidentes" que permita a los usuarios reportar incidentes de seguridad y recibir actualizaciones automáticas sobre el progreso de la resolución. |
| **Justificacion** | Proporcionar una vía eficiente para que los empleados informen sobre incidentes de seguridad, así como para mantenerlos informados sobre las acciones tomadas para abordar dichos incidentes, aumentando así la transparencia y la confianza en los procedimientos de seguridad de la empresa. |

### **Proceso de gestión de cambios**

## **Estado de la configuración(Describir)**







## **REFERENCIAS**

Atlassian. (s/f). *Bitbucket*. Bitbucket. Recuperado el 13 de mayo de 2024, de <https://bitbucket.org/>

*GitHub Advanced Security para Azure DevOps*. (s/f). Microsoft.com. Recuperado el 13 de mayo de 2024, de <https://azure.microsoft.com/es-es/products/devops/github-advanced-security/>

*Plastic SCM - Documentation, technical articles, additional resources and support*. (s/f). Plasticscm.com. Recuperado el 13 de mayo de 2024, de <https://docs.plasticscm.com/>

Sync, G. W. P. (2024, abril 10). *GitKraken Client support & documentation*. GitKraken Help Center. <https://help.gitkraken.com/gitkraken-client/gitkraken-client-home/>

*Top version control system — Perforce Helix Core*. (s/f). Perforce.com. Recuperado el 13 de mayo de 2024, de <https://www.perforce.com/solutions/version-control>

*The most-comprehensive AI-powered DevSecOps platform*. (s/f). Gitlab.com. Recuperado el 13 de mayo de 2024, de <https://about.gitlab.com/>

*Apache Subversion*. (s/f). Apache.org. Recuperado el 13 de mayo de 2024, de <https://subversion.apache.org/>