Plan de SQA

Muebles BIFU

Gestión de Calidad

Gleadell Carla

Jerez Emiliano



Tabla de contenido

[Propósito 4](#_Toc170302486)

[Referencias 4](#_Toc170302487)

[Gestión 5](#_Toc170302488)

[Organización 5](#_Toc170302489)

[Actividades 5](#_Toc170302490)

[Actividades de calidad a realizarse 5](#_Toc170302491)

[Gestión de configuraciones 5](#_Toc170302492)

[Gestión de riesgos 5](#_Toc170302493)

[Realizar Revisión Técnica Formal (RTF) 5](#_Toc170302494)

[Documentación 6](#_Toc170302495)

[Propósito 6](#_Toc170302496)

[Documentación mínima requerida 6](#_Toc170302497)

[Especificación de requerimientos del software 6](#_Toc170302498)

[Descripción del diseño del software 7](#_Toc170302499)

[Documentación de usuario 7](#_Toc170302500)

[Plan de Gestión de configuración 7](#_Toc170302501)

[Propósito 7](#_Toc170302502)

[Resumen 8](#_Toc170302503)

[Organización, Responsabilidades 8](#_Toc170302504)

[Herramientas, Entorno, e Infraestructura 8](#_Toc170302505)

[Forma de trabajo 9](#_Toc170302506)

[Control de Cambios 9](#_Toc170302507)

[Estándares, prácticas, convenciones y métricas 10](#_Toc170302508)

[Objetivos de las Métricas 10](#_Toc170302509)

[Métricas de Proceso 11](#_Toc170302510)

[Métricas de Proyecto 11](#_Toc170302511)

[Métricas de Producto 11](#_Toc170302512)

[Estándar de documentación 11](#_Toc170302513)

[Estándar de codificación 11](#_Toc170302514)

[Diseño de interfaces gráficas de usuario 12](#_Toc170302515)

[Estándar de verificación y prácticas 12](#_Toc170302516)

[Revisiones y auditorías 12](#_Toc170302517)

[Objetivo 13](#_Toc170302518)

[Requerimientos mínimos 13](#_Toc170302519)

[Revisión de requerimientos 13](#_Toc170302520)

[Revisión de diseño preliminar 13](#_Toc170302521)

[Revisión de diseño crítico 13](#_Toc170302522)

[Auditoría funcional 13](#_Toc170302523)

[Auditoría física 13](#_Toc170302524)

[Auditorías internas al proceso 13](#_Toc170302525)

[Revisiones de gestión 13](#_Toc170302526)

[Revisión del Plan de gestión de configuración 13](#_Toc170302527)

[Revisión Post Mortem 13](#_Toc170302528)

[Agenda 13](#_Toc170302529)

[Otras revisiones 12](#_Toc170302530)

[Revisión de documentación de usuario 12](#_Toc170302531)

[Verificación 13](#_Toc170302532)

[Reporte de problemas y acciones correctivas 13](#_Toc170302533)

[Herramientas, técnicas y metodologías 14](#_Toc170302534)

[Gestión de riesgos 14](#_Toc170302535)

[Anexos 15](#_Toc170302536)

Plan de SQA

# Propósito

La calidad del software se refiere a recibir un producto o servicio que cumpla con el costo, las especificaciones y las expectativas establecidas. El propósito de este plan de aseguramiento de la calidad del software (SQA) es definir los estándares y actividades que deben aplicarse en todos los proyectos de la organización para garantizar la elaboración de productos de software de alta calidad.

Este plan incluye el propósito y alcance del Plan de Calidad. La tarea principal del Plan de Calidad es garantizar que se cumplan todas las normas y estándares establecidos para asegurar el éxito del proyecto. Es fundamental que todos los miembros del equipo comprendan la importancia de las tareas de SQA y se comprometan a aceptar y corregir los errores que se detecten.

## Referencias

[ANSI/IEEE Std 730.1-1989, IEEE Standard for Software Quality Assurance

Documento Plantilla gestión de riesgos

Documento plan de gestión de riesgos

Documento plantilla de revisión de SQA

Documento Informe Final de SQA

Documento plantilla de Revisión técnica formal

DocumentoManejo del Ambiente Controlado

Documento Plan de gestión de Configuración

SQuaRE, ISO 25000:2005,

Quality management systems – Requirements ISO 9001:2008]

SwBook

Heurísticas de Nielsen

# Gestión

## Organización

La organización de este proyecto está conformada por dos miembros, quienes comparten las siguientes responsabilidades: gestión del proyecto, gestión de calidad, gestión de configuraciones y cambios, gestión de riesgos, y gestión de validación y verificación.

## Actividades

### Actividades de calidad a realizarse

Las actividades de calidad que se realizan son las siguientes:

### Gestión de configuraciones

Las actividades de gestión de configuraciones incluyen la documentación de la funcionalidad y las características de los ítems de configuración, así como el registro de la información necesaria para administrar estos ítems de manera efectiva. Además, se encargan del control de las solicitudes de cambios y de los cambios aprobados, permitiendo deshacer e identificar los cambios y asegurando la persistencia de versiones.

Los responsables de la gestión de configuraciones son los programadores y el diseñador del equipo, compuesto por Gleadell Carla y Jerez Emiliano.

### Gestión de riesgos

Las actividades de gestión de riesgos incluyen la identificación de posibles riesgos, la evaluación de su gravedad y la planificación de medidas para supervisarlos y mitigarlos adecuadamente. Esto permite evitar retrasos u otros problemas que puedan afectar la calidad del producto.

Los responsables de la gestión de riesgos son Gleadell Carla y Jerez Emiliano.

### Realizar Revisión Técnica Formal (RTF)

Las revisiones técnicas formales (RTF) tienen como objetivo descubrir errores en la funcionalidad, la lógica o la implementación del producto de software. Además, verifican que el producto cumpla con sus especificaciones, se ajuste a los estándares establecidos y señalan las posibles desviaciones detectadas.

Los responsables de la gestión de riesgos son Gleadell Carla y Jerez Emiliano.

# Documentación

## Propósito

La documentación del proyecto es esencial en el desarrollo de cualquier producto de software. Los proyectos desarrollados en la organización seguirán la metodología de desarrollo PSI, basada en el Proceso Unificado. Esta metodología proporciona un marco de trabajo y plantillas indispensables para la correcta documentación de un producto de software a lo largo de todo el proceso de desarrollo.

## Documentación mínima requerida

### Especificación de requerimientos del software

El documento de especificación de requerimientos debe describir, de forma clara y precisa, cada uno de los requerimientos esenciales del software, así como las interfaces externas. El cliente deberá recibir una especificación que se ajuste a sus necesidades, cubriendo el alcance del proyecto acordado inicialmente y los cambios realizados durante el desarrollo.

**La especificación debe cumplir con los siguientes criterios:**

* **Ser completa:**
  + Externamente, en relación con el alcance acordado.
  + Internamente, sin elementos sin especificar.
* **Ser consistente:** No debe haber elementos contradictorios.
* **Ser no ambigua:** Todos los términos referidos al área de aplicación deben estar definidos en un glosario.
* **Ser verificable:** Debe ser posible verificar, siguiendo un método definido, si el producto final cumple con cada requerimiento.
* **Incluir procedimientos de verificación:** Detallar los procedimientos adecuados para verificar si el producto cumple con los requerimientos.
* **Incluir requerimientos de calidad:** Considerar los atributos específicos del software que afectan la 'calidad en el uso'.

**Los requerimientos de calidad del producto a construir deben incluir:**

* **Funcionalidad:**
  + Adecuación a las necesidades.
  + Precisión de los resultados.
  + Interoperabilidad.
  + Seguridad de los datos.
* **Confiabilidad:**
  + Madurez.
  + Tolerancia a fallos.
  + Recuperabilidad.
* **Usabilidad:**
  + Comprensible.
  + Aprendible.
  + Operable.
  + Atractivo.
* **Mantenibilidad:**
  + Analizable.
  + Modificable.
  + Estable: no deben producirse efectos inesperados tras modificaciones.
  + Verificable.

Cada uno de estos atributos debe cumplir con las normas y regulaciones aplicables.

### Descripción del diseño del software

El documento de diseño especifica cómo se construirá el software para satisfacer los requerimientos establecidos. Debe describir los componentes y subcomponentes del diseño del software, incluidas las interfaces internas. El cliente deberá recibir un diseño que cubra los aspectos acordados con él, en función de su importancia y sus conexiones lógicas.

**El diseño debe cumplir con los siguientes criterios:**

* **Corresponder a los requerimientos a incorporar:**
  + Cada elemento del diseño debe contribuir a algún requerimiento.
  + La implementación de cada requerimiento debe estar contemplada en al menos un elemento del diseño.
  + Ser consistente con la calidad del producto.

### Documentación de usuario

La documentación de usuario debe especificar y describir los datos y entradas de control requeridos, así como la secuencia de entradas, opciones, limitaciones del programa y otros ítems necesarios para la ejecución exitosa del software. Todos los errores deben ser identificados y las acciones correctivas descritas.

Como resultado del proyecto, el cliente recibirá una documentación de usuario que cumpla con los requerimientos específicos del proyecto.

## Plan de Gestión de configuración

### Propósito

Este documento detalla las actividades de gestión de configuración de software que se llevarán a cabo durante el desarrollo del proyecto Muebles Bifu. El propósito es controlar la entrega y el cambio de los elementos a lo largo del ciclo de vida del sistema, así como mantener un registro del estado de los elementos y las solicitudes de cambio.

### Resumen

La Gestión de Configuración identifica los elementos del proyecto Muebles Bifu, proporciona control sobre ellos y genera informes de estado de la configuración. Este control y reporte aseguran una gestión clara y una asignación de responsabilidades precisa a lo largo del ciclo de vida del producto.

### Organización, Responsabilidades

Se designarán roles específicos para la gestión de configuración en el proyecto Muebles Bifu:

* **Gestor de Configuración:** Responsable de identificar, registrar y controlar los elementos de configuración.
* **Arquitecto:** Encargado de definir la estructura del repositorio y la organización de los elementos.
* **Cliente:** Participa en la evaluación y aprobación de cambios.
* **Programador:** Implementa los cambios aprobados.
* **Líder del Proyecto:** Supervisa todas las actividades de gestión de configuración.
* **Co-líder del Proyecto:** Colabora en la supervisión y asume responsabilidades específicas.
* **Analista:** Identifica y registra los elementos de configuración y gestiona los cambios propuestos.
* **Diseñador:** Define la estructura y organización de los elementos de configuración.
* **Documentador:** Crea y mantiene la documentación del proyecto.
* **Comité de Control de Configuración:** Evalúa y aprueba los cambios propuestos.
* **Ingeniero de Pruebas:** Diseña y ejecuta casos de prueba.

### Herramientas, Entorno, e Infraestructura

Las herramientas, entorno e infraestructura para la gestión de configuración en el proyecto Muebles Bifu son las siguientes:

* **Herramientas:**
  + GitHub como repositorio de código.
  + Git como control de versiones.
  + Google Drive como plataforma de documentación.
* **Hardware:**
  + Los miembros del equipo utilizarán sus computadoras personales.
* **Ubicación física de servidores y clientes:**
  + En las computadoras personales de cada miembro del equipo.
* **Ubicación física de documentos y líneas base:**
  + Almacenados en Google Drive, organizados por etapas del ciclo de vida del proyecto según la metodología PSI.

### Forma de trabajo

Durante el proceso de gestión de configuración, se utilizará GitHub como repositorio principal y Git para el control de versiones del producto. Los miembros del equipo accederán al servidor para almacenar y actualizar las partes modificadas utilizando las herramientas adecuadas.

### Control de Cambios

El proceso de control de cambios se realizará a través de un Formulario de Pedido y Detección de Cambio, el cual especificará los detalles de la solicitud, incluyendo el nombre y versión del elemento de configuración, el peticionario, la fecha de petición, la necesidad del cambio, la descripción del cambio, la prioridad, el estado y la fecha del cambio. El Comité de Control de Configuración evaluará y aprobará los cambios propuestos, y estos se implementarán una vez aprobados. Se realizarán informes y auditorías periódicas para revisar la evolución de los cambios y asegurar la calidad del producto.

~~Reportes y Auditorias~~

~~Se realizará las siguientes auditorias:~~ **~~Auditoria Funcional:~~** ~~Cuyo objetivo es comprobar que se han completado todas las pruebas necesarias para el / los ECS auditados, y que, teniendo en cuenta los resultados de los tests, se puede afirmar que el / los ECS satisfacen los requisitos que se impusieron sobre él.~~

**~~Revisión formal de certificación~~**~~: Cuyo objetivo es certificar que el / los ECS se comportan correctamente en su entorno operativo. ACA NO SE SI PONER EL ALGO DEL TP DE AUDITORIAS QUE HICIMOS PORQUE NO LO APLICAMOS AL PROYECTO PERO SI LO VIMOS, MAS ABAJO CASI AL FINAL TAMBIEN HAY UN APARTADO COMPLETO DE AUDITORIA PERO NO LO APLICAMOS NUNCA AL PROYECTO POR ESO NO SE SI PONERLO AUNQUE SEA CONCEPTUALMENTE~~

# Estándares, prácticas, convenciones y métricas

Esta sección tiene como objetivo identificar los estándares, prácticas, convenciones y métricas que serán aplicadas durante el proyecto, así como cómo se monitoreará y asegurará el cumplimiento de estos ítems.

**Estándares y Prácticas**

Durante el desarrollo del proyecto, se seguirán los siguientes estándares y prácticas para garantizar la calidad del software:

* **Estándares de Codificación:** Se adoptará el estándar de codificación de PHP para asegurar coherencia y legibilidad en el código fuente. Esto incluye el uso de Composer para la autoloading según el estándar PSR-4.
* **Prácticas de Documentación:** Se utilizarán plantillas definidas para la elaboración de entregables, siguiendo la metodología PSI, que incluye especificaciones de encabezados, pies de página, fuentes y datos mínimos.
* **Revisiones Técnicas Formales (RTF):** Se llevarán a cabo revisiones técnicas formales en varias etapas del desarrollo para identificar errores y asegurar que el software cumpla con sus especificaciones.

**Convenciones**

* **Control de Versiones:** Se utilizará Git como herramienta de control de versiones y GitHub como plataforma para gestionar el repositorio de código.
* **Gestión de Problemas:** GitHub Issues será la herramienta principal para el reporte y seguimiento de problemas y desviaciones.

**Métricas**

Aunque en este proyecto no se han aplicado métricas específicas, reconocemos la importancia de las métricas en el aseguramiento de la calidad del software. Según el IEEE “Standard Glossary of Software Engineering Terms”, una métrica es una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado.

Las métricas son una herramienta poderosa y fundamental para el trabajo en SQA. Su principal aporte radica en las medidas preventivas que pueden surgir a raíz de su estudio. Aportan información valiosa para hacer controles estadísticos de la calidad y mejorar la eficiencia del grupo de trabajo y la calidad de los productos.

## Objetivos de las Métricas

Los objetivos importantes dentro de un programa de métricas son:

* **Documentar las metas:** Establecer qué debe lograrse antes de invertir recursos en un programa de métricas.
* **Identificar la información necesaria:** Determinar las métricas necesarias para lograr estas metas y el marco de referencia de donde pueden obtenerse.

## Métricas de Proceso

Estas métricas se recopilan de todos los proyectos y durante un largo período de tiempo. Caracterizadas por:

* **Control y ejecución del proyecto.**
* **Medición de tiempos de las fases.**

## Métricas de Proyecto

Estas métricas permiten evaluar el estado del proyecto y seguir la pista de los riesgos. Caracterizadas por:

* **Cantidad de puntos de función liberados por unidad de tiempo.**
* **Costo del desarrollo y soporte.**
* **Horas trabajadas y tiempo transcurrido.**
* **Distribución del esfuerzo por fase.**
* **Cambios sobre requerimientos durante el desarrollo y en operación.**
* **Cronograma vs estimado.**
* **Porcentaje de requerimientos implementados por unidad de tiempo.**

## Métricas de Producto

Estas métricas se centran en las características del software y no en cómo fue producido. Caracterizadas por:

* **Tamaño, calidad, totalidad, volatilidad y esfuerzo.**
* **Puntos de caso de uso y puntos de función.**
* **Complejidad de diseño y código.**
* **Densidad de defectos y tiempo requerido para reparar.**

## Estándar de documentación

Para la escritura de documentos, se han definido plantillas que deben ser utilizadas en la elaboración de entregables, siguiendo y respetando la estructura propuesta por la metodología de desarrollo PSI. Estas plantillas especifican:

* **Encabezado y pie de página.**
* **Fuente y tamaño de fuente para el estilo normal.**
* **Fuente y tamaño de fuente para los títulos.**
* **Datos mínimos que se deben incluir:**
  + Fecha
  + Versión
  + Responsables

## Estándar de codificación

En el desarrollo de este proyecto de software, se adoptó el estándar de codificación de PHP para asegurar la coherencia y legibilidad del código fuente. Este estándar sigue las directrices y prácticas recomendadas por la comunidad de PHP, incluyendo el uso de Composer para el autoloading según el estándar PSR-4, mejorando la mantenibilidad y facilitando la colaboración entre desarrolladores.

## Diseño de interfaces gráficas de usuario

Para la implementación de las interfaces gráficas de usuario (GUI), se han utilizado las bibliotecas y componentes de Bootstrap. Bootstrap proporciona un conjunto de herramientas frontend que incluyen estilos CSS predefinidos, componentes interactivos y un diseño responsivo, lo cual facilita el desarrollo de interfaces consistentes y visualmente atractivas.

Durante el diseño y desarrollo de las interfaces de usuario, se han seguido las heurísticas de Nielsen. Estas heurísticas, propuestas por Jakob Nielsen, son principios generales para el diseño de interfaces efectivas y usables. Se basan en años de investigación en experiencia de usuario (UX) y están diseñadas para identificar problemas comunes que afectan la usabilidad de un sistema.

Las heurísticas de Nielsen proporcionan un marco para evaluar la calidad de la interfaz de usuario, asegurando que sea intuitiva, eficiente y satisfactoria para los usuarios finales. Al aplicar estas heurísticas durante el diseño y la evaluación de las interfaces, se busca mejorar la experiencia del usuario y reducir la posibilidad de errores de diseño que puedan afectar la usabilidad del software.

## Estándar de verificación y prácticas

Para la ejecución de casos de prueba y corrección de errores, se utilizan pruebas unitarias aplicadas con PHPUnit. Esto consiste en la selección de un conjunto de entradas que ejecutan un caso de prueba, documentando las salidas esperadas y las salidas obtenidas para asegurar el correcto funcionamiento del software. En caso de encontrar errores, se deben reportar para analizar el problema y utilizar medidas de corrección que resuelvan el conflicto. Para realizar esto, se definen casos de prueba que sean efectivos para descubrir defectos en los programas.

También se utiliza PHPUnit para realizar el mocking que simula entornos controlados y XDebug para depuración del código para identificar y resolver errores de manera efectiva. Estas herramientas ayudan a asegurar la calidad del software al verificar el funcionamiento correcto de las unidades de código y facilitar la corrección de problemas encontrados durante las pruebas.

# Revisiones y auditorías

## Objetivo

El objetivo de esta sección es definir las revisiones y auditorías técnicas y de gestión que se realizarán durante el proyecto. Especifica cómo se llevarán a cabo estas revisiones y auditorías para asegurar que el producto de software cumple con los requisitos y estándares de calidad.

## Requerimientos mínimos

Se especifican las revisiones y auditorías que deben realizarse como mínimo, así como la agenda para la realización de las mismas.

## Revisión de requerimientos

Esta revisión se realiza para asegurar que se han cumplido con los requerimientos especificados por el Cliente. Se verifican todos los requisitos documentados y se asegura que estén claros, completos y sin ambigüedades.

## Revisión de diseño preliminar

[Esta revisión se realiza para asegurar la consistencia y suficiencia técnica del diseño preliminar del software.]

## Revisión de diseño crítico

[Esta revisión se realiza para asegurar la consistencia del diseño detallado con la especificación de requerimientos.]

## Auditoría funcional

[Esta auditoría se realiza previa a la liberación del software, para verificar que todos los requerimientos especificados en el documento de requerimientos fueron cumplidos.]

## Auditoría física

[Esta revisión se realiza para verificar que el software y la documentación son consistentes y están aptos para la liberación.]

## Auditorías internas al proceso

[Estas auditorías sirven para verificar la consistencia: del código versus el documento de diseño, especificaciones de interfase, implementaciones de diseño versus requerimientos funcionales, requerimientos funcionales versus descripciones de testeo.]

## Revisiones de gestión

[Estas revisiones se realizan periódicamente para asegurar la ejecución de todas las actividades identificadas en este Plan. Deben realizarse por una persona ajena al grupo de trabajo.]

## Revisión del Plan de gestión de configuración

[Esta revisión se realiza para asegurar la consistencia y completitud de los métodos especificados en el Plan de gestión de configuración.]

## Revisión Post Mortem

[Esta revisión se realiza al concluir el proyecto para especificar las actividades de desarrollo implementadas durante el proyecto y para proveer recomendaciones.]

## Agenda

[En esta sección se deberá especificar la agenda para las revisiones y auditorías detalladas anteriormente.]

# Otras revisiones

## Revisión de documentación de usuario

[Se revisa la completitud, claridad y aplicación de uso.]

# Verificación

[Se debe identificar todas las verificaciones que no fueron identificadas en el Plan de V & V para el software y debe especificar los métodos a ser usados.]

# Reporte de problemas y acciones correctivas

En nuestra metodología de seguimiento de desviaciones, utilizamos GitHub Issues como herramienta principal para la gestión de problemas, tareas y seguimiento de proyectos de desarrollo de software. GitHub Issues es una plataforma integrada en GitHub que ofrece diversas características y funcionalidades que facilitan el proceso de reporte, seguimiento y resolución de problemas. A continuación, detallamos las prácticas y procedimientos que seguiremos para este fin, así como los responsables comprometidos con la implementación de acciones correctivas:

**Prácticas y Procedimientos**

Utilizamos GitHub Issues como nuestra principal herramienta para el reporte y seguimiento de problemas y desviaciones en el desarrollo de software. Los pasos para reportar una desviación y llevar a cabo su seguimiento son los siguientes:

**Reporte de Desviaciones**

* Los miembros del equipo pueden reportar desviaciones directamente en GitHub Issues.
* Se detallan las características de la desviación, incluyendo una descripción clara del problema, pasos para reproducirlo (si es aplicable) y cualquier otra información relevante.
* Se asigna la desviación a un responsable del equipo para su resolución.

**Seguimiento y Resolución**

* El responsable asignado se encarga de analizar y resolver la desviación.
* Se realizan las acciones correctivas necesarias para abordar la desviación.
* Se actualiza el estado de la desviación en GitHub Issues a medida que avanza su resolución.
* Una vez resuelta, se marca la desviación como completada y se cierra el issue correspondiente.

**Responsables**

* **Desarrolladores del equipo:** Responsables de reportar desviaciones y de implementar acciones correctivas para resolverlas.
* **Líder de Proyecto:** Encargado de supervisar el proceso de seguimiento de desviaciones y asegurar que se tomen las acciones correctivas necesarias de manera oportuna.

# Herramientas, técnicas y metodologías

La herramienta principal que utilizamos para el seguimiento de desviaciones es GitHub Issues. A través de esta, gestionamos de manera efectiva los problemas identificados durante el desarrollo de software. Además, empleamos las siguientes técnicas y metodologías de soporte para nuestras actividades de aseguramiento de calidad:

* **GitHub Issues:** Utilizamos GitHub Issues como nuestra herramienta central para el seguimiento de desviaciones, aprovechando sus características integradas y su integración directa con GitHub para un flujo de trabajo eficiente.
* **Procedimientos definidos:** Contamos con procedimientos establecidos para el reporte, seguimiento y resolución de desviaciones, lo que garantiza una respuesta rápida y eficaz ante cualquier problema que surja durante el desarrollo de software.

Mediante el uso de estas herramientas, técnicas y metodologías, aseguramos un proceso robusto para el manejo de desviaciones, lo que contribuye a la calidad y fiabilidad de nuestros productos de software.

# Gestión de riesgos

La gestión de riesgos es un proceso dentro de un proyecto que tiene como objetivo identificar, evaluar y gestionar los riesgos que puedan afectar su éxito. Esta sección establece los métodos y procedimientos utilizados para especificar, monitorear y controlar las áreas de riesgo durante el proyecto.

**Métodos y Procedimientos**

En esta sección se detallan los métodos y procedimientos específicos que se utilizarán para gestionar los riesgos del proyecto.

**Procedimiento de Identificación de Riesgos:** El primer paso en la gestión de riesgos es identificar todos los posibles riesgos del proyecto. Para esto, se ha utilizado un proceso sistemático de análisis que permite identificar y documentar los riesgos de manera exhaustiva.

**Método de Tabla de Gestión de Riesgos:** Se ha empleado una tabla de gestión de riesgos donde se listan los riesgos identificados y se les asignan una serie de atributos para su evaluación y seguimiento. Estos atributos incluyen:

* **Descripción:** Una descripción breve y clara del riesgo que se está evaluando.
* **Categoría:** La categoría a la que pertenece el riesgo, como externo, interno, dirección de proyecto, de la organización, entre otros.
* **Impacto:** Una evaluación del impacto del riesgo sobre el proyecto, utilizando una escala del 1 al 5, donde 5 representa el impacto más alto y 1 el más bajo.
* **Probabilidad:** La probabilidad estimada de que el riesgo ocurra, también evaluada en una escala del 1 al 100.
* **Exposición al Riesgo (ER):** Calculada como la multiplicación del impacto por la probabilidad.
* **Magnitud:** La magnitud del riesgo, obtenida al multiplicar la exposición al riesgo por cien. Esta medida ayuda a priorizar los riesgos según su importancia relativa.
* **Consecuencias Asociadas:** Los posibles resultados o impactos que podrían surgir si el riesgo identificado llegara a materializarse.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Riesgo | Descripción | Categoría | Impacto | ER | Magnitud | Consecuencias asociadas |
|  |  |  |  |  |  |  |

**Plan de Control de Riesgos y Estrategias de Mitigación**

En esta sección se detalla el plan de control de riesgos del proyecto, especificando los factores analizados y las estrategias para su tratamiento. Las estrategias de control de riesgos se presentan de acuerdo con la guía PMBOK e incluyen:

* **Evitar:** Se busca eliminar la amenaza o proteger al proyecto de su impacto, reduciendo la probabilidad de ocurrencia.
* **Transferir:** Se traslada el impacto de una amenaza a un tercero, delegando a otra parte la responsabilidad de su gestión.
* **Mitigar:** Se actúa para reducir la probabilidad de ocurrencia o el impacto de un riesgo, implementando acciones preventivas de manera temprana.
* **Aceptar:** Se reconoce el riesgo y no se toman medidas preventivas a menos que el riesgo se materialice.

# Anexos

Formulario de Pedidos y Detección de Cambios

|  |  |
| --- | --- |
| Formulario de Pedidos y Detección de Cambios | |
| Fecha de Petición: |  |
| Nombre y Versión del Elemento |  |
| Nombre del Solicitante: |  |
| Necesidad de Cambio: |  |
| Descripción del cambio pedido: |  |
| Prioridad: |  |
| Estado: |  |
| Fecha del cambio: |  |
| Identificador de la nueva versión: |  |
| Que fue afectado por este cambio |  |