PLAN DE GESTIÓN DE LA CONFIGURACIÓN

GALLERY GUIDE

Antonio Macías Ferrera, Benjamín Ignacio Maureira

Flores

04/02/2025

Índice

Ficha del documento

- **Nombre del Proyecto:** GALLERY GUIDE
- **Autores:** Antonio Macías Ferrera, Benjamín Ignacio Maureira Flores
- **Fecha de Creación:** 04/02/2025
- **Versión:** 1.0

Histórico de Modificaciones

Fecha	Realizada por	Descripción de los cambios
04/02/2025	Antonio Macías Ferrera	Elaboración de la plantilla del documento.
04/02/2025	Benjamín Ignacio Maureira Flores	—-

1. NORMAS Y PROCEDIMIENTOS APLICABLES

Este plan contiene la información sobre cómo el equipo de trabajo realizará el seguimiento y control del cambio durante el desarrollo del proyecto, además de cómo se llevará a cabo el control de versiones durante las fases de desarrollo y cierre. Se excluye de este plan la especificación de cómo se realizará la configuración y el versionado de los cambios en detalle. Ver: ***Plan de Gestión del Cambio***.

2. ELEMENTOS CONFIGURABLES

Los elementos configurables del proyecto incluyen aquellos artefactos y entregables que pueden estar sujetos a modificaciones durante el desarrollo. Cada uno de estos elementos tendrá un identificador único, y se someterá a control de versiones.

1. **Documentación**: todos los documentos que surjan a lo largo de todas las fases del proyecto deberán estar sujetos a un sistema de control de versiones específico y unificado.
2. **Requisitos e HU**: define los requisitos del sistema. De este documento saldrán también los registros de HU y de Casos de Uso del Sistema, que también serán configurables estarán debidamente trazados.
3. **Código Fuente**: código del producto software, sujeto a control de versiones.
4. **Tareas**: las fechas y condiciones para la entrega de los principales entregables de cada Sprint también serán elementos configurables.

3. HERRAMIENTAS A UTILIZAR

Tecnología	Elementos Configurables	Descripción
Pandoc y Eisvogel	Documentación, Registros	Plataforma para la edición de documentos en Markdown.
Clockify	Tareas	Seguimiento del tiempo de trabajo por actividad.

Tecnología	Elementos Configurables	Descripción
Taiga.io	Hitos, requisitos, HU	Herramienta ágil para gestión de proyectos.
Git	Código fuente	Control de versiones del código fuente.
GitHub	Código fuente, Hitos, HU	Gestión del desarrollo colaborativo.

4. CONTROL y POLÍTICA DE VERSIONADO

4.1. Versionado de Documentación y Registros

Se sigue un esquema de versionado semántico:

- **Mayor:** Cambios significativos en el contenido (reestructuración o nueva sección).
- **Menor:** Actualizaciones dentro de secciones existentes o correcciones menores.

Se deberá modificar la versión siempre que se realice alguna modificación en el documento. La versión se indicará tanto en el título del documento como en la tabla de control de versiones que se encontrará tras el índice del documento.

Ejemplo de versionado: `v1.2` (vMayor.Menor)

4.2. HU, tareas y actividades

Cada historia de usuario y caso de uso tiene un identificador único asignado tras su creación.

Cambios importantes en la descripción o criterios de aceptación de una HU se anotan y actualizaremos la versión. Se seguirá un versionado similar al visto en el punto anterior. La versión de cada elemento se añadirá en un campo específico que encontraremos en su tabla.

- **Mayor**: Cambios significativos en el contenido debido a una solicitud de cambio.

- **Menor**: Actualización de información, correcciones ortográficas, formato o pequeños ajustes de expresión.

Ejemplo de versionado: v1 . 2 (vMayor.Menor).

A la hora de crear nuevas tareas en *Taiga.io* en base a las HU, se deberá usar un nombre descriptivo y breve, similar al título de la HU asociada. Como esta tarea estará asociada a una Issue de GitHub irá acompañada de un código de la Issue(#XX). ***Esto ayudará a llevar una trazabilidad entre Issue - Tarea - HU - Clockify***.

La nomenclatura a seguir para los distintos requisitos, CU e HU se con un esquema de numeración para cada tipo de registro:

Categoría	**Código**
Objetivos de alto nivel	OBJ-XXX
Requisitos del proyecto	PRR-XXX
Requisitos de información	IRQ-XXX
Requisitos de reglas de negocio	RRQ-XXX
Requisitos de conducta	CRQ-XXX
Requisitos de fiabilidad	FRQ-XXX
Requisitos de portabilidad	PRQ-XXX
Requisitos de seguridad	SRQ-XXX
Requisitos de organización (incluye entrega, uso de estándares y tecnología)	ORQ-XXX
Requisitos de factores ambientales (incluye requisitos legislativos y de privacidad)	FARQ-XXX
Cambios	CHG-XXX
Historias de usuario	HU-XXX
Casos de Uso	CU-XXX

4.3. Control del tiempo

Clockify no tiene control de versiones explícito, pero el seguimiento se realiza registrando las entradas de tiempo y las tareas completadas para cada miembro del equipo siguiendo la misma nomenclatura que las tareas de *Taiga.io*.

Las tareas se numeran en orden cronológico y deben incluir un identificador único en el formato #XX.

Se excluyen de esta nomenclatura las tareas de las fases de Inicio y Planificación. Todas las tareas deberán estar asignadas a UNA SOLA persona y a un proyecto de Clockify previamente creado. Esto ayudará evaluar mejor el desempeño y el tiempo empleado en las tareas.

Ejemplo de versionado: `Realización de ventana de inicio de sesión` #53.

4.4. Versionado de Código Fuente en Git y GitHub

Estructura de Ramas - **`main`** → Producción
- **`develop`** → Desarrollo
- **`feat/nueva_funcionalidad`** → Funcionalidades nuevas
- **`test/nueva_funcionalidad`** → Ramas de pruebas
- **`hotfix/corrección`** → Correcciones urgentes

Versionado semántico Se emplea versionado semántico en el caso de realizar 'releases' de código: - **Mayor**: Cambio que rompe compatibilidad (nueva arquitectura). - **Menor**: Nuevas funcionalidades añadidas sin afectar lo anterior

Commits (Conventional Commits) La política de nombrado de commits se ajustará a las directrices de 'conventional commits', siendo estos **siempre en inglés**:

```
<tipo>: <descripción breve>
[opcional] Cuerpo detallado del mensaje
[opcional] Pie de mensaje (referencias a tickets, breaking changes, etc.)
    El <tipo> especifica la naturaleza del cambio realizado. Los tipos más comunes incluyen:
    - feat: Una nueva funcionalidad.
    - fix: Corrección de errores.
    - docs: Cambios en la documentación (no relacionados con el código).
    - style: Cambios que no afectan la lógica del código (formato, espacios en blanco, etc.).
    - refactor: Cambios en el código que no corrigen errores ni añaden funcionalidades.
    - test: Adición o modificación de pruebas.
```

La descripción debe ser concisa y clara, expresando en una sola línea el propósito del commit. Se recomienda utilizar el modo imperativo (por ejemplo, .agrega", çorrige") y evitar el uso de mayúsculas al inicio, salvo para nombres propios.

El cuerpo se utiliza para detallar el contexto del cambio, explicar el por qué detrás del commit, y describir cualquier implicación importante.

El pie del mensaje puede incluir: - Referencias a tickets o tareas: Refs #123. - Cambios significativos (breaking changes): BREAKING CHANGE: seguido de una descripción del cambio.

Ejemplo de mensaje de commit

```
feat(auth): add support for token-based authentication

This change introduces a new authentication system based on JWT tokens.
    The user module has been updated, and a new dependency has been
   added.

BREAKING CHANGE: The login API now requires a JWT token instead of a
   cookie-based session.
```

4.5. Solicitudes de cambio (Registro de cambios)

Los registros se mantienen como archivos en formato *Markdown*. Cada cambio, HU o caso de uso tiene un identificador único (ejemplo: REQ-001, CU-003). Cambios aprobados se reflejan en el historial de cambios del documento correspondiente actualizando su versión acorde a lo mencionado anteriormente.

Para las solicitudes de cambio formales, se deberá seguir la plantilla ubicada en el ***Plan De Gestión Del Cambio***.

5. Conclusión

Este documento define las políticas de gestión de configuración del proyecto **GALLERY GUIDE**. Es crucal que estas buenas prácticas sean aplicadas por todos los miembros del equipo a lo largo de todo el proyecto para procurar un orden, trazabilidad y, en resumen, una buena calidad en el desarrollo y resultado del producto.