Nombre: Julián Aleksei Narváez Vinueza
Curso: GR2SW
Fecha: 10 de diciembre de 2025

Proyecto

Nombre del proyecto: SuperManga

Descripción:

SuperManga es una aplicación web diseñada para que cada usuario gestione su propia biblioteca personal de mangas. Dentro de la plataforma, el usuario puede buscar títulos ya existentes y marcar los mangas que ha leído, manteniendo un registro ordenado y visual de su progreso.

Además, cada lector puede dejar reseñas y comentarios sobre los mangas disponibles, permitiendo expresar su opinión, compartir experiencias y revisar lo que otros usuarios piensan. La aplicación funciona como un espacio centralizado donde cada persona puede llevar el control de sus lecturas y consultar información y opiniones de la comunidad, todo en un entorno sencillo, organizado y orientado a la lectura de manga.

Plan Maestro SCM

Fase 1: ANTES de Escribir la Primera Línea de Código (Planificación y Diseño)

Configuración del Entorno y Repositorio

Se utilizará Git y GitHub como herramientas principales. El repositorio se organizará como un monorepo llamado SuperManga-App, con carpetas separadas para frontend, backend y documentación. La rama principal será main y estará protegida para evitar modificaciones directas. Se requerirá al menos una aprobación de Pull Request antes de fusionar cambios, y no se permitirá el force push en main.

Definición del Flujo de Trabajo (Workflow)

Nomenclatura de ramas de trabajo:

- Nuevas funcionalidades: feature/nombre-feature (Ej. feature/sistema-busqueda-manga)
- Corrección de errores: bugfix/descripcion-bug (Ej. bugfix/corregir-login-usuario)
- Arreglos urgentes: hotfix/descripcion-hotfix (Ej. hotfix/vulnerabilidad-auth)

Gestión de Artefactos (No-Código)

Los requisitos y las historias de usuario se documentarán en Markdown dentro de /docs/requisitos. Los diseños de interfaz se crearán en Figma, cuyo enlace estará en el README principal, y las capturas aprobadas se guardarán en /docs/diseños.

Fase 2: Durante el desarrollo (Codificación y Pruebas)

Gestión de Cambios en Acción

El Proceso del Pull Request (PR)

Un Pull Request podrá ser aprobado únicamente cuando cumpla con la siguiente Definición de Hecho (Definition of Done):

1. Feature completa: La feature implementada funciona correctamente y cubre los criterios de aceptación.
2. Peer Review: Al menos un revisor debe aprobar el PR.
3. Pruebas automáticas pasadas: Todas las pruebas unitarias y de integración deben ejecutarse exitosamente.
4. Sin regresiones: Las pruebas existentes deben continuar pasando sin fallos.
5. Documentación actualizada: Si la funcionalidad lo requiere, se debe actualizar el archivo correspondiente dentro de /docs.
6. Sin conflictos: El PR debe poder fusionarse a main sin conflictos.
7. Código limpio: El linter no debe reportar errores ni advertencias.

Trazabilidad:

1. Commits: Todos los commits deben incluir referencia al ID del Issue.
2. Pull Requests: Los PR deberán estar asociados obligatoriamente a un Issue mediante sintaxis GitHub.
3. Etiquetas: Los Issues se etiquetarán según su tipo: feature, bug, enhancement, documentation, tech-debt.

Integración Continua (CI)

Se implementarán workflows de GitHub Actions que se ejecutarán en cada push a ramas feature, bugfix y en todos los PR hacia main. Habrá un workflow de pruebas automáticas para backend, frontend y base de datos. Otro workflow verificará estilo y formato con ESLint y Prettier. Un workflow adicional realizará análisis de seguridad buscando vulnerabilidades o secretos expuestos. Otro generará una build de producción y, en PRs, un preview temporal del despliegue.

Gestión de Línea Base (Baselining)

Para el control de líneas base se utilizarán tags de Git, siguiendo versionamiento semántico. Estos tags representarán hitos importantes del proyecto como autenticación, búsqueda, biblioteca del usuario, reseñas, perfiles y la versión final, ejemplo:

Tags del desarrollo:

Tag	Descripción
v0.1-alpha	Módulo de búsqueda de mangas.
v0.2-alpha	Biblioteca personal del usuario.
v0.3-beta	Sistema de reseñas y comentarios.

Fase 3: Después del Lanzamiento (Despliegue y Mantenimiento)

Gestión de Releases y Despliegue (CD):

El despliegue funcionará mediante un pipeline de CD. Cada merge aprobado en main pasará automáticamente al entorno de staging. Para producción se realizará el despliegue generando un tag desde main, permitiendo un control preciso de la versión publicada.

El versionamiento seguirá SemVer: los cambios mayores incrementan la versión MAJOR, nuevas funcionalidades incrementan MINOR y los arreglos pequeños incrementan PATCH.

Plan de Mantenimiento Proactivo

Mantenimiento Correctivo:

- Flujo estándar: Issue → rama → PR → patch release.
- Hotfix para problemas críticos.

Mantenimiento Adaptativo:

- Cada 4 meses: revisar compatibilidades, APIs, frameworks.
- Cada 6 meses: auditoría completa.

Mantenimiento Perfectivo:

- 15% del tiempo de desarrollo semanal para deuda técnica.

Mantenimiento Preventivo:

- Dependabot para vulnerabilidades.
- Backups automáticos diarios, semanales y mensuales.
- Revisiones trimestrales de seguridad.