ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA DE SISTEMAS

CONSTRUCCIÓN Y EVOLUCIÓN DE SOFTWARE

Nombre: Belén Cholango.
Curso: GR2SW.

Proyecto 001: Plan Maestro de SCM

Nombre del proyecto: KateikApp.

Descripción: KateikApp es una aplicación web orientada a la organización y asignación de tareas del hogar entre los miembros de una familia o personas que comparten una vivienda. El sistema permite crear hogares, asignar tareas, establecer fechas límite y llevar un seguimiento del cumplimiento de las responsabilidades domésticas.

El objetivo del proyecto es mejorar la coordinación, la responsabilidad compartida y la comunicación dentro del hogar mediante una herramienta sencilla, accesible desde el navegador web y fácil de mantener a largo plazo.

Tecnologías Propuestas

- Frontend: HTML5 + CSS3 + JavaScript.
- Backend: PHP o Python (Flask). Alternativa: Java (servlets).
- Base de datos: MySQL o SQLite.
- SCM: GitHub, Git.

FASE 1: ANTES de Escribir la Primera Línea de Código.

Configuración del Entorno y Repositorio

- **Herramientas SCM:**
  - Git para control de versiones.
  - GitHub como repositorio remoto.
- **Repositorio:**
  - Un solo repositorio llamado KateikApp.
  - Estructura monorepo (frontend + backend en el mismo repositorio).
- **Rama principal:**
  - Nombre: main.
  - **Rama protegida:**
    - No se permiten pushes directos.
    - Todos los cambios deben entrar mediante Pull Request.
- **Política de Ramas (Branching Strategy)**

Se utilizará un flujo simple y fácil de mantener:

- main: versión estable del sistema.
- feature/nombre-funcionalidad: nuevas funcionalidades.
    - Ejemplo: feature/crear-tareas.
- bugfix/descripcion-bug: corrección de errores.
- hotfix/bug-critico: errores críticos en producción.

Definición del Flujo de Trabajo (Workflow)

- Todo desarrollo comienza desde main.
- Cada cambio se realiza en una rama independiente.
- Los cambios se integran mediante Pull Request (PR).
- El PR debe ser revisado antes de ser fusionado.

Gestión de Artefactos (No-Código)

El control de versiones no se limita solo al código.

- Requisitos del sistema:
    - Archivos Markdown dentro de /docs/requisitos.md
- Diagramas y diseño:
    - Diagramas UML y flujos guardados en /docs/diagramas/
- Documentación SCM:
    - Plan Maestro de SCM almacenado en /Proyecto-001/

FASE 2: DURANTE el Desarrollo

Gestión de Cambios – Pull Requests

Todo cambio en el proyecto KateikaApp se integrará exclusivamente mediante Pull Requests hacia la rama main. No se permiten integraciones directas.

Un Pull Request solo podrá fusionarse si cumple todas las siguientes condiciones:

1. El PR ha sido revisado y aprobado por al menos un integrante del equipo.
2. El código compila o se ejecuta sin errores evidentes.
3. Se respetan las convenciones de nombres de archivos y funciones.
4. El PR pasa correctamente los flujos de Integración Continua (CI).
5. El mensaje del commit referencia explícitamente el Issue asociado.

Formato obligatorio de commit: Agrega funcionalidad de creación de tareas (closes #12). Corrige error al asignar tareas (fixes #18)

Trazabilidad

- Cada funcionalidad, mejora o corrección se registra como un Issue en GitHub.
- Los Issues se clasifican con etiquetas:
    - feature

    - bug
    - enhancement
    - documentation
- Ningún PR puede existir sin un Issue asociado.

Esto permite:

- Auditar cambios
- Analizar impacto
- Mantener historial del proyecto

Integración Continua (CI)

Se configurará GitHub Actions para ejecutar automáticamente los siguientes trabajos:

Job 1: Validación de estructura

- Verifica que la estructura del proyecto no sea alterada incorrectamente.
- Comprueba la existencia de carpetas obligatorias (docs/, frontend/, backend/).

Job 2: Validación de código

Dependiendo del lenguaje utilizado:

- PHP: verificación de sintaxis
- Python: ejecución de scripts básicos
- JavaScript: verificación de errores comunes

Estos trabajos se ejecutan:

- En cada push a una rama feature/
- En cada Pull Request hacia main

Un PR no puede fusionarse si algún job falla.

Gestión de Línea Base (Baselining)

Los hitos importantes se marcarán mediante tags de Git:

- v0.1-alpha → estructura inicial.
- v0.5-beta → gestión básica de tareas.
- v1.0.0 → versión estable inicial.

Esto permitirá regresar a estados estables del proyecto si ocurre algún problema.

FASE 3: DESPUÉS del Lanzamiento

Gestión de Releases y Versionamiento

Se usará Versionamiento Semántico:

MAJOR.MINOR.PATCH

- PATCH: corrección de errores.
- MINOR: nuevas funcionalidades.
- MAJOR: cambios incompatibles.

Ejemplo:

- v1.0.1 → bug corregido.
- v1.1.0 → sistema de puntos agregado.
- v2.0.0 → cambio importante de arquitectura.

Flujo de Release:

1. Se completa un conjunto de Issues planificados.
2. Se crea un Pull Request final hacia main.
3. Tras la aprobación, se crea un tag de release en main.
4. El release se documenta en GitHub Releases.

Plan de Mantenimiento Proactivo

Mantenimiento Correctivo

- Los errores detectados en producción se registran como Issues tipo bug.
- Para errores críticos:
    1. Se crea una rama hotfix/descripcion-bug desde main.
    2. Se aplica la corrección mínima necesaria.
    3. Se fusiona nuevamente a main.
    4. Se libera una nueva versión PATCH.

Ejemplo: hotfix/error-fecha-limite → v1.0.1

Mantenimiento Adaptativo

Para asegurar la compatibilidad del sistema con el entorno tecnológico:

- Cada 6 meses se realizará:
    - Revisión de versiones del lenguaje utilizado (PHP / Python / Java)
    - Verificación de compatibilidad con navegadores web modernos
- Se documentarán los cambios necesarios como Issues enhancement.

Mantenimiento Perfectivo

Se asignará tiempo específico para mejoras internas del sistema:

- Refactorización de código

- Optimización de consultas a base de datos
- Simplificación de funciones complejas
- Mejora de legibilidad y documentación

Frecuencia:

- Una revisión mensual del código
- Issues etiquetados como technical-debt

Mantenimiento Preventivo

Para reducir fallos futuros:

- Uso de herramientas automáticas (Dependabot) para detectar dependencias vulnerables.
- Revisión periódica de permisos y roles de usuario.