Políticas para el desarrollo

Grupo 7

ISPP-MapYourWorld

Devising a project

Alfonso Alonso, Alejandro Aragón, José María Baquero, Pablo Caballero, Ricardo Carreño, Franco Dell ‘Aguila, Alberto Escobar, Jaime Gómez, Claudio González, Ángel Neria, Pablo Olivencia, Antonio Porcar, Alba Ramos, Pedro Pablo Santos, Manuel Vélez, Gonzalo García.

19/02/2025

CONTROL DE VERSIONES			
VERSIÓN	FECHA	COMENTARIOS	AUTOR
v1.0	19/02/2025	Creación del documento	Pedro Pablo Santos
v1.1	19/02/2025	Revisión del formato	Ricardo Carreño

Índice

1. Convenciones de Desarrollo

Este documento define las buenas prácticas para la gestión del código, commits, issues, pull requests y estrategia de ramas en nuestro equipo de desarrollo.

1.1. Política de Commits

Para mantener claridad y uniformidad en el historial del repositorio, seguiremos la siguiente convención en los mensajes de commit:

- **feat:** Cuando se implemente una nueva funcionalidad. El modelo de mensaje será "*feat: nombre de la funcionalidad, explicación breve*".
- **fix:** Cuando se realicen correcciones o ajustes en una funcionalidad existente. El modelo será "*fix: nombre de la funcionalidad, explicación breve*".
- **test:** Cuando se agreguen o modifiquen pruebas. El modelo será "*test: nombre de la funcionalidad, que test se han realizado*"

Cada commit debe ser claro y conciso, evitando mensajes genéricos como "arreglos" o "cambios".

1.2. Gestión de Issues y Pull Requests

Para el esquema de la issues, lo haremos estableciendo un título descriptivo, realizando seguidamente una breve restricción y poniendo la prioridad de esta.

Trabajaremos con un **Project** en el que las tareas estarán organizadas en diferentes columnas:

- **To Do:** Contiene las tareas pendientes de desarrollo.
- **In Progress:** Cuando un desarrollador comience a trabajar en una tarea, deberá moverla a esta columna y asignarse como responsable.
- **Review:** Una vez completado el desarrollo, se creará una pull request y se asignará un revisor, quien revisará la implementación antes de aprobar la integración. Esta aprobación deberá hacerse escribiendo un mensaje positivo, en caso de no estar correcto el código se deberá intentar en medida de lo posible poner en que falla.
- **Done:** Cuando la pull request sea aprobada y fusionada, la tarea se moverá a esta columna y se considerará finalizada.

Para las pull request, habrá que ser minucioso por lo que se asignaran dos revisores, que se encargaran de que no haya archivos que no deban estar cambiados, y de que el código sea correcto. Esto se hará para evitar problemas

a la hora de implementar las funcionalidades en la rama main, y controlar posibles bugs o errores.

1.2.1. Responsabilidades

- **Desarrollador:** Asignarse la tarea, moverla a "In Progress", crear la pull request y asignar un revisor.
- **Revisor:** Revisar el código asegurando que cumpla con los estándares antes de aprobar la pull request.

1.3. Estrategia de Ramas

Para mantener un flujo de trabajo ordenado, seguiremos la siguiente estructura de ramas:

- **feat/***: Para nuevas funcionalidades. Ejemplo: *feat-loginPage*.
- **hotfix/***: Para correcciones urgentes en producción. Ejemplo: *hotfix-paymentBug*.
- **main:** Rama estable con el código listo para producción.

Por último, las pruebas se realizarán en la rama a la que correspondan dichas pruebas.

1.4. Flujo de Trabajo

1. Crear una nueva rama a partir de main según el tipo de tarea.
2. Desarrollar la funcionalidad o corrección.
3. Abrir una pull request contra main.
4. Una vez revisada y aprobada, integrar la funcionalidad en main.