Documento de gestión del código

decide-part-chiquito

Grupo 2
Curso escolar: 2023/2024
Evolución y gestión de la configuración

Integrantes

Juan Jesús Campos Garrido	decide-part-chiquito-1
Alejandro Campano Galán	decide-part-chiquito-1
Antonio Carretero Díaz	decide-part-chiquito-1
Pablo Mera Gómez	decide-part-chiquito-1
Carlos Baquero Villena	decide-part-chiquito-1
David Cortabarra Romero	decide-part-chiquito-1
Alejandro Pérez Vázquez	decide-part-chiquito-2
Juan Carlos Ramírez López	decide-part-chiquito-2
Carmen Ruiz Porcel	decide-part-chiquito-2
Alejandro Santiago Félix	decide-part-chiquito-2
Sergio Santiago Sánchez	decide-part-chiquito-2
María Vico Martín	decide-part-chiquito-2

En cuanto a la gestión del código, dispondremos de un repositorio de código en la plataforma de GitHub en el que guardaremos toda la historia del proyecto. Este repositorio será un “fork” del repositorio decide de la asignatura al que le limpiaremos el historial de commits y comenzaremos con toda la producción en la que trataremos de realizar una participación homogénea entre todos los componentes.

Para trabajar, hemos decidido seguir ciertos acuerdos:

Política de commits

El patrón que se utilizará para los mensajes de commits será el siguiente:

```
<Type> [title]

[description]
```

- Donde los **tipos** a utilizar serán:
  - fix
  - feat
  - refactor
  - test
  - docs
  - build
  - release
- El **título** podrá tener un máximo de 50 caracteres, debe ser capitalizado, estar en modo indicativo y no podrá terminar con un punto.
- La **descripción** podrá tener un máximo de 72 caracteres y debe tratar de explicar el qué y por qué se ha hecho.

Política de gestión de ramas

Nuestra estrategia se basará en la metodología de *GitFlow*, con esto tendremos hasta dos tipos de ramas principales en las que no trabajaremos directamente. Estas son las siguientes:

- La rama **master** en las que se encuentran las versiones de lanzamiento con las distintas versiones de la aplicación.
- La rama **develop** a la que iremos introduciendo todos los cambios que desarrollemos.

Para introducir los cambios realizados, lo realizamos a través de un *merge* que añade en estas ramas todo el contenido desarrollado. Para que se produzca este *merge,* se necesita la aprobación de al menos otras dos personas mediante una *pull request* ya que las ramas están protegidas. En estas *pull request* se debe detallar el trabajo hecho para que las demás personas puedan entender y comprobar que todo funciona, sin tener previa idea de la tarea.

Por otro lado, tendremos todas las ramas de trabajo en las que iremos desarrollando todo el proyecto y lo iremos incorporando a las ramas principales. Estas son:

- Las ramas de **hotfix** que son para corregir bugs en las versiones de lanzamiento.

- Las rama **release** en las que prepararemos las distintas versiones de lanzamiento.
- Las ramas de **feat** son las ramas en la que iremos incorporando todas las nuevas características que vamos desarrollando.
- Las ramas **doc** que son para subir los documentos.
- Las ramas **fix** sirven para realizar correcciones de bugs durante el desarrollo, previamente a una versión de lanzamiento.

Siempre que se quiera trabajar en alguna de estas ramas, se debe crear una nueva a partir de la rama principal correspondiente y su nombre debe seguir la siguiente disposición:

```
<Type>/[title]
```

Donde los tipos pueden ser *feature, release, hotfix* o *doc* y el título describe la tarea a realizar descrita en pocas palabras con disposición “snake_case” o la versión a lanzar si se trata de una *release*. Dentro de estas ramas, desarrollaremos individualmente la tarea descrita en el título y, al finalizar, realizaremos el *merge* anteriormente descrito.

Finalmente, hay que destacar que cuando se haya preparado el lanzamiento de una versión a través de una rama *release*, se debe actualizar también la rama *develop* a la misma vez que se lanza la versión a la rama *master* y marcar esta última con una *tag* indicando el versionado del lanzamiento.

Política de versionado

Para cada hito del proyecto se hará un lanzamiento que contendrá una versión, siguiendo el formato X.Y.Z.

Solo tendrán versión los lanzamientos.

- La X identificará al lanzamiento, por ejemplo 1 significa que es el primer lanzamiento.
- La Y identificará si tras hacer el lanzamiento se ha añadido alguna pequeña funcionalidad.
- La Z se aumentará si se hace alguna corrección sobre lo ya lanzado en la versión.