*Pull requests* en GitHub (actividad grupal)

Objetivos

En esta actividad los estudiantes aprenderán la funcionalidad de *pull requests* de GitHub y el manejo de repositorios remotos. El GitHub *workflow* está cada vez más extendido en entornos empresariales, por lo que estar familiarizado con él es muy ventajoso a la hora de enfrentarse a un nuevo proyecto.

Pautas de elaboración

FinTech Solutions S. A. se centró en el desarrollo de *software* personalizado para instituciones financieras y empresas de tamaño pequeño y mediano. Su enfoque en la calidad, la seguridad y la innovación les permitió ganar reconocimiento en el mercado y establecer relaciones sólidas con sus clientes.

El objetivo de esta actividad es aplicar el flujo de GitHub sobre el código del caso práctico. Para ello, uno de los miembros del equipo debe actuar como administrador del repositorio principal, mientras que los demás integrantes del grupo deberán actuar como desarrolladores individuales, los cuales implementarán una funcionalidad diferente.

Para realizar esta actividad, todos los estudiantes deben tener cuentas personales en [GitHub](https://github.com/). Por lo tanto, si aún no se han registrado, pueden hacerlo a través de su página web. Asimismo, deben tener acceso a una máquina con Git. Para instalarlo, pueden seguir las instrucciones del caso práctico del tema tres, «Flujos de trabajo con Git». Por último, deben decidir quién va a actuar como administrador del repositorio. El responsable de este puesto es quien debe inicializar un repositorio que se desarrolle como principal.

El administrador, para inicializar el repositorio, debe llevar a cabo las siguientes tareas:

* Crear un repositorio nuevo en GitHub. **Este debe estar** **marcado como público.** Puede llevar como nombre, por ejemplo, practica-unir-fintech.
* Clonar el repositorio en local con git clone <url>.
* Descargar el código desde la página de [Github](https://github.com/srayuso/unir-git) como un ZIP y copiar los archivos del ZIP a la carpeta del repositorio.
* Ejecutar git add . (cuidado con el punto), git commit -m "primer commit" y git push origin master.
* También puede clonar el repositorio unir-git directamente y añadir su repositorio como remoto. El objetivo es que el repositorio no sea un *fork* del usado en el caso práctico para no despistar durante la creación de *pull requests.*

Una vez inicializado el repositorio, el administrador compartirá la URL con el resto del grupo, pero no debe dar acceso de escritura a nadie. Los demás integrantes deberán seguir los pasos del caso práctico:

* Crear un *fork* a partir del repositorio del administrador.
* Una vez clonado en local, crear una rama nueva e implementar la funcionalidad, *commit.* Al clonar en local, es necesario comprobar que la carpeta contenga los archivos correctos (el README.md, el fichero Python, etcétera).
* Por último, hay que subir la rama a su *fork* y crear la *pull request.*

El grupo puede implementar las funcionalidades que desee. A continuación, mencionamos algunos ejemplos:

* Actualizar el README.md con una ejecución de prueba para demostrar cómo se usa la aplicación.
* Traducir los mensajes que se imprimen por pantalla al inglés.
* Aceptar un nuevo parámetro de línea de comandos que indique si el orden debe ser ascendente o descendente.
* Aceptar un nuevo parámetro de línea de comandos que indique si es necesario eliminar palabras duplicadas de la lista antes de imprimirlas (en este caso habrá que modificar la función sort\_list).
* Añadir una directiva en el Makefile que ejecute el comando en local, sin arrancar un contenedor de Docker.
* Añadir un fichero de ejemplo, palabras.txt, que incluya varias palabras desordenadas, una por línea, y modificar el Makefile para que use el nombre del fichero como parámetro de línea de comandos.

El objetivo de esta actividad no es aprender a escribir ni Python ni Makefiles, por lo que no deben preocuparse si el código no funciona perfectamente. La idea es que los integrantes del grupo trabajen en paralelo sobre el mismo repositorio. Algunos van a editar los mismos ficheros, pero en repositorios diferentes (tanto en local como con cada *fork*, que son independientes del repositorio principal).

El administrador debe implementar también una funcionalidad y abrir una *pull request* contra su propio repositorio. **Es importante recordar que deben crear una nueva rama antes de añadir los cambios.** De lo contrario, no podrán abrir una *pull request.* Si los cambios se añaden, por ejemplo, a master:, cuando subamos master al repositorio, con git push origin master, ya estaremos integrando los cambios en esa rama. No tiene sentido abrir una *pull request* de master contra master.

Asimismo, todos los integrantes del grupo deben iniciar *fork* a la vez para partir del mismo código y facilitar la revisión de *pull requests.* Cuando las *pull requests* estén abiertas, deben acceder a ellas entre todos. Posteriormente, es necesario comprobar el código de sus compañeros.

Una vez que han revisado el código de todas las *pull requests,* el administrador puede empezar a fusionarlas. Si hay algunas que hayan modificado las mismas líneas de un fichero, es posible que solo se pueda fusionar una de ellas. Al intentar fusionar la siguiente, GitHub indicará que hay conflictos y no ofrecerá la opción de fusionar. En ese caso, lo recomendado es fusionar las que puedan, ya que la resolución de conflictos no entra en el alcance de esta práctica.

Aunque solo uno actúe como administrador, los demás integrantes del grupo deben ayudarlo en los primeros pasos para inicializar el repositorio. Durante la implementación de funcionalidades nuevas, deben ayudarse si hay dudas en el código Python, si no identifican cómo ampliar el Makefile, etc. Aunque no sea el objetivo de la práctica, trabajar sobre código que funciona es siempre más gratificante.

Una vez cerrado el equipo de trabajo, el grupo se puede poner en contacto a través de sus cuentas @comunidadunir.net para comenzar a trabajar. Asimismo, pueden ampliar la información sobre el trabajo en equipo consultando los [Tutoriales de trabajo en grupo.](http://tv.unir.net/secciones/3967/4883/0/0/0/0/)

Aquellos estudiantes que no comiencen su trabajo dentro de los siete primeros días, contados a partir del día de inicio de la actividad, quedarán excluidos de la actividad, sin poder tomar parte en ella. Se trata de una actividad colaborativa, por lo que unos estudiantes no pueden beneficiarse del trabajo que hayan realizado sus compañeros.

Al finalizar el trabajo, **todos los miembros del equipo entregarán la misma actividad** a través del apartado «Envío de actividades» del aula virtual. El documento debe ir nombrado de la siguiente manera: **APELLIDO1\_APELLIDO2\_NOMBRE\_Titulo\_actividad** (sin tildes ni apóstrofes ni ningún otro carácter que pudiera resultar conflictivo).

Asimismo, la actividad debe indicar el nombre de todos los miembros del equipo, los cuales deben cumplimentar la siguiente tabla de valoración individual:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Sí** | **No** | **A veces** |
| Todos los miembros se han integrado al trabajo del grupo |  |  |  |
| Todos los miembros participan activamente |  |  |  |
| Todos los miembros respetan otras ideas aportadas |  |  |  |
| Todos los miembros participan en la elaboración del informe |  |  |  |
| Me he preocupado por realizar un trabajo cooperativo con mis compañeros |  |  |  |
| Señala si consideras que algún aspecto del trabajo en grupo no ha sido adecuado |  |  |  |

Extensión y formato

No es necesario entregar el código, pero sí incluir capturas de pantalla de:

* El repositorio principal en GitHub.
* El *fork* de cada desarrollador en GitHub.
* El comando git push origin <rama> de cada desarrollador (la captura debe ser de la ventana de la terminal).
* Cada *pull request,* una vez abierta. Debe aparecer el nombre de la rama y el *fork* desde el que se ha abierto. La captura tiene que incluir la pestaña de *Files changed.*
* Lista de *pull requests* en el repositorio principal.
* El árbol de *commits* del repositorio principal tras fusionar todas las *pull requests posibles.* Para ello, acceded a la ruta /commits/master del repositorio (por ejemplo, https://github.com/<id-usuario-admin>/practica-unir-eiec/commits/master). Deben aparecer los *commits* de las *pull requests* que se han fusionado y los autores.

Rúbrica

|  |  |  |  |
| --- | --- | --- | --- |
| *Pull requests* en GitHub | Descripción | Puntuación máxima  (puntos) | Peso  % |
| Criterio 1 | El administrador ha conseguido crear un repositorio principal. | 3 | 30 % |
| Criterio 2 | Cada desarrollador ha creado su *fork* y lo ha clonado en su máquina. | 3 | 30 % |
| Criterio 3 | Cada desarrollador ha subido su rama y ha abierto una *pull request* contra el repositorio principal. | 4 | 40 % |
|  |  | **10** | **100 %** |