Tarea 1 - GitHub, Pytest y Flake 8

Gabriel O. González Rodríguez
Luis Gabriel Rosales

Preguntas Teóricas

1. **¿Explique la principal utilidad de git como herramienta de desarrollo de código?**

La principal utilidad de git en el desarrollo de software es el control de versiones. La estructura de git permite tener un repositorio central con las versiones funcionales del código, lo que facilita recuperar alguna versión pasada conforme se van realizando cambios y pruebas. También permite trabajar en ramas paralelas al repositorio central las cuales se comparan con este a la hora de unir los cambios. Esto provee bastante seguridad y confianza a la hora de trabajar en un proyecto y compone una forma colaborativa de trabajar y manejar versiones del código.

2. **Explique la diferencia entre git y github**

Git es un sistema de control de versiones distribuido, de código abierto. Github es una plataforma de Microsoft que permite el uso de Git de forma gráfica y más amigable con el usuario.

3. **¿Qué es un branch?**

Es una copia a nivel local del repositorio principal. Esta se utiliza para trabajar sin afectar el repositorio principal, y una vez finalizado el trabajo se puede combinar de nuevo con él.

4. **En el contexto de github. ¿Qué es un Pull Request?**

Cuando un desarrollador está listo para combinar el código en su repositorio local, o “branch”, con el del repositorio principal, o “main”, debe de realizar un “Pull Request” que es un paso extra en donde los demás desarrolladores puedan revisar los cambios realizados y aprobarlos. Esta es la etapa del Pull Request. Una vez aceptados los cambios se procede con el “merge”, o combinación de la rama local y el repositorio principal.

5. **¿Qué es un commit?**

Un “commit” es el proceso de actualización de los archivos modificados por el desarrollador en git. Cuando un desarrollador realiza cambios en su “branch” estos están locales en su computadora, estos deben de ser “staged” para actualizarlos en git y que formen parte de los archivos que se quieren modificar en el repositorio principal. Una vez que todos los archivos deseados están “staged” se realiza un “commit” que recopila todos ellos y los actualiza en git (en la nube).

6. **Describa lo que sucede al ejecutar la siguiente operación: “git rebase main”.**

La operación “rebase” es una alternativa al merge que permite añadir los cambios hechos en una “branch” paralela a la “master”, sobreescribiendo las diferencias de la “branch” paralela en la “master”, dejando de esta forma una sola línea de tiempo en el historial. En caso de ejecutarse la operación, se añaden los cambios al main y de hacerse un “fast-forward” a la “master branch”, se sobreescriben los cambios y se elimina la “branch” experimental.

7. **Explique que es un “merge conflict” y cómo lo resolvería.**

Un “merge conflict” es un error que se produce cuando dos o más colaboradores intentan hacer cambios en “branches” distintas sobre la misma línea de un archivo. Github permite resolver estos conflictos mediante los siguientes pasos:

1. En la ventana de “pull requests” se busca el solucionador de “merge conflicts” y se utiliza el botón de resolver conflictos siempre y cuando estos no sean muy complejos
2. Posteriormente, se eligen los cambios de las ramas que se desean mantener o si se desea hacer un cambio nuevo.
3. Dado que todos los conflictos fueron resueltos, se confirma la combinación

8. **¿Qué es una Prueba Unitaria o Unittest en el contexto de desarrollo de software?**

El unittesting es una infraestructura de código que permite automatizar el proceso de pruebas de un proyecto por medio del uso de funciones encargadas de probar varios posibles resultados deseados.

9. **Bajo el contexto de pytest. ¿Cuál es la utilidad de un “assert”?**

El “assert” permite comparar que el resultado retornado por la función de prueba sea equivalente al valor del deseado, de ser así, se puede asegurar que la prueba sea exitosa o fallida.

10. **Mencione y explique 3 errores de formato detectables con Flake8**

Entre algunos errores que se pueden encontrar en Flake8 son:

1. **E202:** Este error de formato indica las líneas de código en donde existe un espacio previo a un cierre de paréntesis “)”.
2. **W292:** Este error alerta al usuario que el código al final no presenta una nueva línea. Esto es una mala costumbre ya que no se logra distinguir la diferencia entre un archivo que no termina en una nueva línea y otro que sí.
3. **E721:** En este error se recomienda no comparar tipos, sino usar el comando isinstance para determinar el tipo de variable que se encuentra, esto porque isinstance puede identificar subclases también.

Referencias

Git - rebasing. (s. f.). https://git-scm.com/book/en/v2/Git-Branching-Rebasing

Introduction — pycodestyle 2.11.1 documentation. (s. f.). https://pycodestyle.pycqa.org/en/latest/intro.html#error-codes

Kinsta. (2023, 21 agosto). *Git vs GitHub: ¿Cuál es la Diferencia y cómo Empezar?* Kinsta®. https://kinsta.com/es/base-de-conocimiento/git-vs-github/

Mijacobs. (2023, 5 octubre). *¿Qué es Git? - Azure DevOps*. Microsoft Learn. https://learn.microsoft.com/es-es/devops/develop/git/what-is-git

Resolver un conflicto de fusión en GitHub - Documentación de GitHub. (s. f.). GitHub Docs. https://docs.github.com/es/pull-requests/collaborating-with-pull-requests/addressing-merge-conflicts/resolving-a-merge-conflict-on-github

unittest — Infraestructura de tests unitarios — documentación de Python - 3.9.19. (s. f.). https://docs.python.org/es/3.9/library/unittest.html