TEC | Tecnológico de Costa Rica

Tarea 1

Luis Diego Brenes Vargas (2022145221), Juan Alberto Solano Méndez (2022058409)

Escuela de Ingeniería Mecatrónica, Instituto Tecnológico de Costa Rica

MT7003: Microprocesadores y Microcontroladores

Ing. Rodolfo Piedra Camacho

16 de febrero del 2024

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

Git permite que varios colaboradores sean capaces de trabajar en paralelo en un mismo proyecto sin que cada cambio que realicen se vea reflejado sobre las copias locales con que cuenta cada programador. Es decir, se trata de un sistema de control de versiones distribuido donde se sincroniza el código generado de forma individual al servidor que contiene la versión oficial del código del proyecto.

Esto permite que se puedan hacer modificaciones y comprobaciones de código independientes del proyecto principal, lo cual puede evitar errores o cambios que perjudiquen su funcionamiento o estructura (Aziz, 2023).

2) ¿Qué es un branch?

Una vez que se decide trabajar sobre el proyecto independiente se crea un *branch* o copia local del código que se encuentra en el servidor principal. Esto permite realizar toda clase de cambios en el código sin alterar el original o *master copy* (Github, s.f.).

3) En el contexto de github. ¿Qué es un Pull Request?

Un pull request actua como una solicitud efectuada por un colaborador de un proyecto para que se revise o modifique una sección del código del proyecto. Esta solicitud permite contrastar la copia modificada con la original para que el resto de programadores puedan analizar y revisar dicha solicitud, y con base en esto tomar una decisión sobre si se acepta la modificación o no (Github, s.f.).

4) ¿Qué es un commit?

En el contexto de Git, es posible generar un *branch* o copia local del código principal del proyecto y realizar todos los cambios o pruebas necesarias para alterarlo; sin embargo, si se desea sincronizar el código local con el *master* se debe realizar un *commit* que permita sincronizar los cambios realizados en el servidor que aloja el código principal, lo cual genera una nueva versión del mismo (Github, s.f.).

5) Describa lo que sucede al ejectuar la siguiente operación: “git cherry-pick <SHA>”.

Este comando es sumamente útil para traer a nuestro trabajo cambios específicos de un *commit* sin necesidad de modificar la *branch* de desarrollo. Según Burgos (2020), lo que hace esta operación es identificar el *commit* específico que necesitamos, este se identifica con el <SHA>. Luego aplica los

cambios que se quieren sobre nuestro *branch*. Si todo se aplica de forma exitosa se crea un nuevo *commit* en la *branch* actual.

6) Explique que es un “merge conflict” y como lo resolvería.

Los conflictos de fusión ocurren cuando se hacen cambios contrapuestos en la misma línea de un archivo o cuando una persona edita un archivo y otra persona borra el mismo archivo. Para resolverlo primero debería de identificar el archivo de conflicto, luego abrir este para así poder resolverlo. Esto se hace decidiendo que cambios deseo conservar en el archivo. Una vez que decidí que cambios deseo mantener los guardo y por último es de buena práctica agregar y confirmar los cambios con un comentario (Github, s.f.).

7) ¿Qué es una Prueba Unitaria o Unittest en el contexto de desarrollo de software?

Las pruebas unitarias son pequeños bloques de código que me permiten realizar una verificación de una parte específica del código. Esto es para evitar probar programas enteros cuando solo necesitamos realizarle una prueba a una función o método en específico (Hillard, 2020).

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

El assert nos permite realizar comprobaciones de código, ya que lo que realiza esta función es comprobar si una condición es verdadera. Con la función se pueden realizar pruebas unitarias y así asegurarse de que las funciones del programa se encuentren libres de errores. Asimismo, nos es útil para la protección de errores inesperados, ya que con la función podemos probar condiciones críticas fuera del tiempo de ejecución (Hillard, 2020).

9) ¿Mencione y explique 3 errores de formato detectables con Flake8?

Flake 8 es capaz de identificar errores de formato en el código a partir de las guías de convención PEP8 (Github,s.f.). Esta herramienta posee una lista de códigos asociados a un error en específico, a continuación, se muestran algunos ejemplos:

- E501: la línea del código es demasiado larga (se establece un máximo de 79 caracteres).
- E999: Error de identación: hay una identación no esperada. Puede ser una instrucción dentro de un If que no está espaciado por tab.
- E302: Esperaba 2 líneas en blanco, encontró 0. Esto suele ocurrir cuando se definen funciones y clases, se espera que hayan dos líneas en blanco entre estas.

10) Explique la funcionalidad de parametrización de pytest.

Esta función nos permite probar diferentes entradas con el mismo código de prueba. Es de suma utilidad cuando se desean probar algunas funciones con conjuntos de datos. Así se podría verificar su comportamiento en los distintos casos (Pytest, s.f.).

Referencias

Github(s.f.). *Flake8 Rules*. https://www.flake8rules.com

Github(s.f.). *Resolución de los conflictos de combinación en Git.* https://docs.github.com/es/pull-requests/collaborating-with-pull-requests/addressing-merge-conflicts/resolving-a-merge-conflict-using-the-command-line

Hillard, D. (2020). *Effective Python Testing With Pytest*. Real Python. https://realpython.com/pytest-python-testing/

Panda, A. (2023). *Understanding GIT: Meaning, Definition and Purpose*. myHQ. https://digest.myhq.in/git-full-form/#:~:text=In%20conclusion%2C%20the%20GIT%20full,not%20interfere%20with%20one%20another

Pytest. (s.f.). H*ow to parametrize fixtures and test functions*. Pytest. https://docs.pytest.org/en/stable/how-to/parametrize.html

Richardson, D. (2023). *Git vs Github: What's the difference?* Hubspot. https://blog.hubspot.com/website/git-vs-github#:~:text=Git%20is%20a%20version%20control,cannot%20use%20GitHub%20without%20Git