Instituto Tecnológico de Costa Rica **Prof. Lic. Ing. Rodolfo José Piedra Camacho**

Área Académica de Ingeniería Mecatrónica **II Semestre 2020**

Curso: MT-7003 Microprocesadores y Microcontroladores

Tarea 1: GitHub, Pytest y Flake 8

Estudiantes:

Coronado Zúñiga Carlos Alberto	**2017158641**
Monge Barahona Adrián	**2016062596**

II Semestre 2020

Índice:

1. Preguntas Teóricas

1) ¿Qué es Git?

Es un sistema de control de versiones de código abierto. El cual permite administrar y controlar proyectos de programación de todo tipo, de una manera muy eficiente. Existen otros sistemas que cumplen una función parecida a Git. Sin embargo, según se menciona en [1] a diferencia de los otros sistemas, en Git cada vez que se guarda un proyecto se almacena una especie de captura del archivo que se modificó, en lugar de guardar la información como una serie de cambios. En otras palabras Git no vuelve a guardar un archivo con los cambios, si no toma capturas de como lucen los archivos y almacena una referencia a esas capturas.

2) ¿Qué es Github?

Github es una plataforma de alojamiento de código para el control de versiones y la colaboración. Permite a los usuarios, trabajar juntos en proyectos en línea. Github es un repositorio de alojamiento de Git, que proporciona a los desarrolladores herramientas como capas de colaboración para: enviar mejor código a través de funciones de línea de comandos, discusión de temas, solicitudes de extracción, revisión de código y el uso de colecciones gratuitas y de compra.

3) ¿Qué es un branch?

En [3] se menciona que un Branch (“ramificación”) es una manera de apartarse del código principal para poder comenzar a modificarlo, sin que se vea afectado el código principal. O sea, que se puede comenzar a trabajar en cierta parte del código sin que se afecte el código principal, así de esta manera no se pierde lo que ya se ha trabajado y aprobado anteriormente del código.

4) ¿Qué es un commit?

En github, los cambios guardados se denominan confirmaciones (“commits”). Los mensajes de confirmación permiten a otros colaboradores entender lo que fue hecho por los demás. En el historial de revisión de un proyecto las confirmaciones aparecen como instantáneas en el tiempo, como una relación de lista enlazada y se pueden organizar en varias líneas de desarrollo llamadas ramas (“Branches”).

5) ¿Qué es la operación cherry-pick?

Esta operación según [3] es una manera de mover el trabajo hecho en un branch a otro branch. Sin embargo, esta operación es para un solo commit. La manera en que funciona es que toma la información de un commit y la trata de integrar en el branch en el que se encuentre el desarrollador al momento de aplicar esa operación.

6) ¿Qué hace el comando git checkout?

De forma general un git checkout es un comando que cambia o genera ramas, además de que puede restaurar archivos del árbol del trabajo. Allí intervienen los sistemas de control de versiones distribuidos (DVCS) como en git, que permiten no solo revisar la última instantánea de los archivos, sino que permiten "reflejar" el repositorio, y utilizar cada "clon" como copia de seguridad.

7) ¿Qué hace el comando git stash?

Este comando permite trabajar en parte del proyecto sin realizar un commit. Lo cual es útil si se quiere trabajar en ciertas ideas que aún no se tienen claras, o bien, pruebas que no necesariamente irán al código final.

Según [3] lo que hace este comando es tomar el trabajo parcial hecho y almacenarlo en una pila de cambios sin terminar, que se pueden acceder posteriormente para continuar trabajando en ellos.

8) Compare las operaciones git fetch y git pull

El comando git fetch ("extraer") solo descarga los datos al repositorio local, no los fusiona automáticamente con ninguno de los trabajos, ni modifica en lo que el usuario esté trabajando en el momento. Se debe fusionar con el trabajo cuando se esté listo (utilizando git merge).

Por el contrario, el git pull se puede usar para buscar automáticamente y fusionar esa rama remota en la rama actual. La ejecución de git pull obtiene datos del servidor desde el que clona e intenta combinarlo automáticamente con el código de trabajo actual, si la rama actual está configurada para rastrear una rama remota.

9) ¿Qué hace el comando git reset ~HEAD?

En [3] se menciona que este comando permite hacer un reinicio mixto de commits, de manera que desaparece el commit de los archivos, haciendo que los archivos modificados queden como pendientes y que posteriormente se puedan hacer los commits. Esto es útil si se desea dividir un commit en varias partes para luego trabajar en ellas. El comando restablece todas las partes en que se dividió el commit, para que estas puedan modificarse.

10) ¿Qué es Pytest?

Pytest es un entorno de trabajo ("framework"), que facilita la creación de pruebas simples y escalables. Las pruebas automáticas son una metodología indispensable que permite que el trabajo sea más sencillo y agradable. Las pruebas son "expresivas" y "legibles" y no se requiere un código estándar. Pytest facilita la escritura de pequeñas pruebas a escala para admitir pruebas funcionales complejas para aplicaciones y bibliotecas. Con

Pytest, el uso de recursos complejos puede ser manejado por fixtures (“accesorios”) de inicialización.

11) Bajo el contexto de pytest. ¿Qué es un “assert”?

Según lo visto en [7] un “assert” es una afirmación o condición que se hace en la programación de un método de prueba que, si no se cumple, pytest dejará de ejecutar ese método de prueba y continuará probando con los demás métodos. También se puede ver como un resultado que se espera del método de prueba, y si este no se cumple durante la ejecución de la prueba, significa que el método no es correcto.

12) ¿Qué es Flake 8?

Flake8 es un envoltorio (“wrapper”) de estas herramientas: PyFlakes, pycodestyle y " Ned Batchelder’s McCabe script". Flake8 ejecuta todas las herramientas ejecutando el único comando flake8. Muestra las advertencias en una salida combinada por archivo.

También, agrega algunas características para verificar el estilo y la calidad de algunos códigos de Python, además de verificar su base de código con el estilo de codificación (PEP8), buscar errores de programación y para verificar la complejidad ciclomática.

2. Referencias:

[1] Flake8, *"Project Description"*, [online document], June, 2020. Available: https://pypi.org/project/flake8/ [Accessed: Sept. 12, 2020].

[2] Git, "Getting Started - What is Git?", *Git.* [Online]. Available: https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F. [Access: Sept. 12, 2020]

[3] Github Guides, "*Git Handbook*", [online document], July, 2020. Available: https://guides.github.com/introduction/git-handbook/#github [Accessed: Sept. 12, 2020].

[4] Github Guides, "*Hello World*", [online document], July, 2020. Available: https://guides.github.com/activities/hello-world/ [Accessed: Sept. 12, 2020].

[5] I. Stapleton, *"Frequently Asked Questions"*, Flake8, [online document], 2016. Available: https://flake8.pycqa.org/en/latest/faq.html#when-is-flake8-released. [Accessed: Sept. 12, 2020].

[6] H. Krekel, Pytest Documentation, *"Installation and Getting Started",*[online document], Septiembre, 2020. Available: https://docs.pytest.org/en/stable/getting-started.html [Accessed: Sept. 12, 2020].

[7] H. Krekel, *Pytest Documentation*. Pytest, [online document], 2020. Available: https://docs.pytest.org/en/stable/ [Accessed: Sept. 12, 2020].

[8] S. Chacon and B. Straub, *Pro Git*, 2nd ed. New York City, NY: Apress, 2014. [E-book] Available: https://git-scm.com/book/en/v2.