Preguntas Teóricas (16 pts, 2pts c/u)

1) ¿Diferencie la herramienta Git de Github?

En términos simples la diferencia principal entre el Git y el GitHub es que el **Git** se define como un sistema de control de versiones de archivos de código abierto que permite llevar un registro del historial y los cambios que se le hacen al código fuente de un proyecto, la base del código y sus versiones se encuentran disponibles en cualquier computador; su principal característica es que utiliza el “Branching model”. Mientras que el **GitHub** es un servicio de alojamiento basado en la nube (similar a Google Drive) que permite que los desarrolladores de software almacenen y administren su código. Además de llevar un registro de los cambios de código

2) ¿Qué es un branch?

En Git, una Branch es una forma de organización de la plataforma GitHub. Puede ser definido como un conjunto único de cambios o modificaciones que se hacen al proyecto sin alterar el código principal (master), y es por esta misma razón que el empleo de “branches” es una herramienta muy poderosa y eficiente para los trabajos colaborativos.

3) ¿Cómo se crea un nuevo Branch?

Para crear un Branch se debe ingresar al repositorio respectivo, una vez en la rama master se debe realizar el comando “git branch xxxx” donde las letras “x” representan el nombre que se le dará a esta nueva rama en la cual se puede trabajar sin alterar el funcionamiento del repositorio master.

4) ¿Qué es un commit?

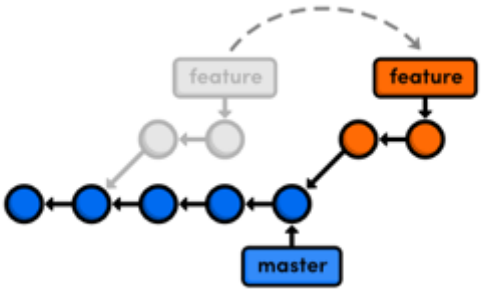
Se puede definir como un comando que ejecuta una determinada actualización de los archivos que están siendo creados en git, logrando así guardar estos cambios realizados. Cuando hacemos un commit, estamos congelando el estado de todos los ficheros y subdirectorios, y guardandolo en un histórico. Al comenzar a usar Git, un error muy común es tenerlo como un simple sistema de backups: una práctica habitual al principio es pensar en los commits como un guardado masivo sobre todos nuestros archivos.

5) ¿Qué es la operación “git stash”?

La operación “git stash” funciona para guardar o mejor dicho “reservar” los cambios realizados en un código sin la necesidad de confirmarlos, esto funciona para cuando es necesario pausar el trabajo en un código para dedicarse a otra cosa y que los cambios queden guardados pero no se apliquen a la rama hasta que estén confirmados.

6) Explique de forma gráfica: ¿Qué sucede en mi Branch local cuando hago “git rebase master”?

Realizar un git rebase master es una forma de mover en su totalidad un branch hasta otro punto del árbol. Para realizar el método es necesario que todos los comits se encuentren en un master branch. Se consulta la rama que se desea cambiar la base y se usa el comando git rebase master, donde master es la rama en la que desea cambiar la base.



Como se observa en la imagen anterior, el branch “feature” fue movilizado para que su base sea el “master” con el “git rebase master”

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

Como su nombre lo dice el “Unittest” o “Prueba Unitaria” consiste en realizar la prueba de una pequeña parte o unidad del código, esto funciona para comprobar que cada bloque del código cumple con sus funciones a la perfección. Un ejemplo de esto sería tener el código para una calculadora, en donde para probar su funcionamiento se realiza una prueba al bloque que realiza las sumas, otra al bloque que realiza las restas y así sucesivamente con las demás operaciones desde la más simple hasta la más compleja. Esto a su vez es muy útil al momento de encontrar errores de funcionamiento en el proyecto.

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

Un “assert” es un comando en el lenguaje de Python que permite hacer comprobaciones en el código, es decir que hace un momento se quiera comprobar el funcionamiento de un bloque o una función que se programó, se utiliza el comando “assert” junto con la comparación de datos que se quiera comprobar, si esta comparación no es correcta, es decir el valor esperado es diferente al valor obtenido, el comando arrojará un “False” como resultado.

Ejemplo:

```
assert(1==2)
# AssertionError
```

Debido a que 1 no es igual a 2, el comando da como resultado el mensaje de “AssertionError”. Este comando puede ser utilizado y muy útil para realizar Unittest.

9) ¿Qué es Flake 8?

Flake8 es una librería de Python que contiene Pyflakes, pycodestyle y Ned Batchelder´s McCabe script. Es una herramienta conocida como “Linter” y lo que hace es correr el código y revisarlo contra el estilo de programación PEP8, errores de programación y revisar la complejidad ciclomática del proyecto. Básicamente es una herramienta que analiza el código y encuentra varios tipos de errores de formato para ser corregidos por el usuario y obtener un código más limpio y eficiente.

10) Explique la diferencia entre un “log de error” y un “valor de error de retorno”

Un valor de error de retorno corresponde a un valor específico que indica por qué el proceso no pudo completarse, dependiendo de este valor de error es posible identificar la causa del error. Por otra parte el log de error corresponde a un registro de los errores críticos que pueden producirse en el sistema, este registro resulta muy útil al momento de realizar el troubleshooting correspondiente.

Bibliografía:

Albert, M. (2017). ¿Qué es y para qué sirve Github? | Deusto Formación. Retrieved from https://www.deustoformacion.com/blog/programacion-diseno-web/que-es-para-que-sirve-github

Empezando - Fundamentos de Git. (2018). Retrieved from https://git-scm.com/book/es/v1/Empezando-Fundamentos-de-Git

Fundamentos de Git - Guardando cambios en el repositorio. (2018). Retrieved from https://git-scm.com/book/es/v1/Fundamentos-de-Git-Guardando-cambios-en-el-repositorio

Git Cherry Pick. (2019). Retrieved from https://www.atlassian.com/es/git/tutorials/cherry-pick

Pytest: helps you write better programs. (2017). Retrieved from https://docs.pytest.org/en/latest/

¿Qué es GitHub? Una Guía para Principiantes sobre GitHub. (2018). Retrieved from https://kinsta.com/es/base-de-conocimiento/que-es-github/

Reescribiendo la Historia con Git Rebase. (2015). Retrieved from https://code.tutsplus.com/es/tutorials/rewriting-history-with-git-rebase--cms-23191

The Ultimate Guide to Git Merge and Git Rebase. (2019). Retrieved from https://www.freecodecamp.org/news/the-ultimate-guide-to-git-merge-and-git-rebase/#:~:text=To%20rebase%2C%20make%20sure%20you,you%20want%20to%20rebase%20on).

The writing and reporting of assertions in tests. (2017). Retrieved from https://docs.pytest.org/en/latest/assert.html#assert

Trabajando con GIT, introducción al uso de los branch y git-completion.bash - Adictos al trabajo. (2011). Retrieved from https://www.adictosaltrabajo.com/2011/06/27/git-branch-bash/