Preguntas Teóricas (20 pts, 2pts c/u)

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

Git es un sistema de control de versiones. Permite crear una copia local del proyecto de forma que se pueda trabajar remotamente y sin conexión. Al trabajar de forma local, posteriormente se sincronizan los cambios con el repositorio en el servidor. En un proyecto con varios desarrolladores se guardan los cambios de forma local y por medio de "ramas" permite aislar los cambios para que al unirse a la rama principal se combinen las modificaciones. Con esto los participantes de un proyecto tienen la posibilidad de trabajar de forma simultánea.

2) ¿Qué es un branch?

Un "branch" (rama) en git representa una línea independiente de desarrollo de un proyecto y es un puntero a una "foto" de un "commit" especifico. Los proyectos inician en la rama principal "master" o "main", el cual contiene los archivos en el estado deseado. Cuando se decide desarrollar una función, probar algo nuevo o corregir se crea una nueva rama, la cual no afecta directamente la rama principal. Al finalizar el trabajo en la nueva rama se pueden fusionar los cambios a la rama principal mediante un "merge".

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

Un "pull request" o una solicitud de incorporación de cambios es una propuesta para combinar un conjunto de cambios de una rama con otra. Con un "pull request" se muestran las diferencias en el contenido de las ramas, con lo cual los participantes del proyecto pueden revisar los cambios que se van a integrar al código principal.

4) ¿Qué es un commit?

Un "commit" en git representa un conjunto de "fotos" de los cambios que se desean realizar en una rama, los cuales están "staged" después de haber sido agregados, medianre un "add" a una lista de espera. Cada uno de los "commit" tienen un identificador ("hash") propio de 40 caracteres, pero se puede usar una versión corta que puede ser de hasta 5 caracteres mientras estos sean únicos en el respectivo repositorio. Cada "commit" tiene un mensaje con una descripción que explica lo realizado en dicho "commit".

5) Describa lo que sucede al ejecutar la siguiente operación: "git cherry-pick ".

Git Cherry Pick es un commando que permite seleccionar "commits" de una rama y adjuntarlos a otra rama. El comando "git cherry-pick <SHA>" selecciona el "commit" con el identificador SHA y los añade a la rama donde se ejecute.

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

Un "merge conflict" ocurre cuando se realiza un "merge" de dos ramas y se tienen diferentes cambios en las mismas líneas de un mismo archivo. Para resolver el conflicto se procede ha abrir el archivo, en el cual se marcan los cambios realizados por el "usuario 1" y el "usuario 2". Al comparar los cambios que realizaron los usuarios es necesario decidir que se va a realizar, ya sea eliminar lo que realizó uno de los usuarios o unir ambos cambios. Al finalizar los cambios deseados

es necesario agregar el archivo con “add” y luego un “commit” para que finalice la unión. Al realizar el “commit” se agrega automáticamente un comentario sobre el conflicto y se puede agregar un comentario detallando la solución elegida.

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

Una prueba unitaria se refiere a un proceso en el que se prueba únicamente una sección pequeña del código, como puede ser un método. De esta forma se puede determinar con mayor precisión la ubicación de un error en un código.

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

Un “assert” en pytest permite verificar si una condición es verdadera. Esto se realiza al comparar el “return” de la función deseada en Python con el valor esperado que se ingresa. Si la condición es falsa el programa falla y genera un mensaje mostrando el valor que retornó la función o un mensaje personalizado.

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

Flake8 es una herramienta de Python que permite verificar el formato y la calidad de un código. Dentro de los errores de formato que puede detectar son:

Errores de indentación: Detecta la mezcla de espacios y tabulador, así como la incorrecta indentación en el código.

Errores de importaciones: Al importar una librería después de ejecutar la línea de comando con una función dentro de esta.

Errores de cantidad de caracteres: Cuando una línea de código sobrepasa 79 caracteres reporta como error sobrepasar el límite. Se puede ignorar el error añadiendo “# noqa: E501”.

10) Explique la funcionalidad de parametrización de pytest.

La funcionalidad de una parametrización en pytest es ejecutar una misma prueba con diferentes entradas.

Referencias

Atlassian. (s. f.). Git Branch | Atlassian Git Tutorial. https://www.atlassian.com/git/tutorials/using-branches

Atlassian. (s. f.-b). Git Commit | Atlassian Git Tutorial. https://www.atlassian.com/git/tutorials/saving-changes/git-commit

Atlassian. (s. f.-c). Git Merge Conflict tutorials. https://www.atlassian.com/git/tutorials/using-branches/merge-conflicts

How to write and report assertions in tests — pytest documentation. (s. f.). https://docs.pytest.org/en/7.1.x/how-to/assert.html

Pytest - Parameterizing tests. (s. f.). https://www.tutorialspoint.com/pytest/pytest_parameterizing_tests.htm