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

   Git es una herramienta de control de versiones distribuido que se utiliza principalmente en el desarrollo de software para gestionar y mantener un historial de cambios en el código fuente. Sus principales utilidades incluyen: Control de versiones, colaboraciones, branches, descentralización, integración y despliegue continuo, documentación y seguimiento, entre otras. Por lo tanto, Git es una herramienta esencial para gestionar el desarrollo de software de manera eficiente y colaborativa, proporcionando un entorno estructurado y seguro para la evolución continua de proyectos de código fuente.

2. Explique la diferencia entre git y github

   Su principal diferencia es que Git es la herramienta que se usa para gestionar el historial de versiones del código, mientras que GitHub es una plataforma que proporciona una interfaz y un conjunto de herramientas para usar Git de manera más efectiva y colaborativa.

3. ¿Qué es un branch?

   Un Branch en Git es una bifurcación del código fuente que permite desarrollar funcionalidades, corregir errores o experimentar con cambios sin afectar el código principal del proyecto. Cada rama es una línea independiente de desarrollo, lo que facilita el trabajo en paralelo y la colaboración entre desarrolladores.

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

   En el contexto de GitHub, un Pull Request es una solicitud para que los cambios realizados en una rama específica de un repositorio sean revisados y potencialmente fusionados en otra rama del mismo repositorio, generalmente la rama principal. Los Pull Requests son fundamentales para la colaboración y revisión de código en equipos de desarrollo de software.

5. ¿Qué es un commit?

   Un commit es una acción que guarda un conjunto de cambios en el historial del repositorio. Es una de las operaciones fundamentales en Git y representa una instantánea del estado del proyecto en un momento dado.

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

   El Git rebase main es una herramienta poderosa para actualizar la rama actual con los últimos cambios de main, manteniendo un historial de commits más limpio y lineal. Es fundamental manejar los conflictos con cuidado y entender cómo el rebase afecta al historial del proyecto. Lo que sucede es:
   - Identificación de la rama actual y la rama base
   - Desplazamiento de commits
   - Aplicación de cambios
   - Resolución de conflictos

- Finalización del rebase

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

Un "merge conflict" ocurre cuando Git no puede automáticamente combinar los cambios de dos ramas diferentes debido a modificaciones incompatibles en el mismo fragmento de código. Esto es común cuando se intenta fusionar ramas que han modificado las mismas líneas de un archivo. Para resolverlo es necesario identificar los conflictos, revisarlos, realizar los cambios pertinentes y proceder con la fusión de ramas de código.

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

Una Prueba Unitaria es una técnica de prueba en el desarrollo de software que se enfoca en verificar que las unidades individuales de código (generalmente funciones, métodos o clases) funcionen correctamente de manera aislada. La idea es asegurarse de que cada componente del software realiza su función correctamente antes de integrarlo con otros componentes.

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

En el contexto de pytest, un assert es una declaración que se utiliza para verificar que una expresión es verdadera. Si la expresión es falsa, pytest genera un error, lo que indica que la prueba ha fallado. Los assert son fundamentales para escribir pruebas unitarias, ya que permiten comparar valores esperados con los resultados obtenidos y confirmar que el código se comporta como se espera.

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

- Líneas excesivamente largas: este error ocurre cuando una línea de código excede los 79 caracteres recomendados.
- Indentación incorrecta: este error se produce cuando la indentación no sigue las normas de PEP 8, generalmente se recomienda usar 4 espacios por nivel de indentación.
- Espacios innecesarios: se refieren a la presencia de espacios innecesarios, se puede producir cuando hay espacios al final de la línea, o cuando hay espacios innecesarios antes de los comentarios.