**Brahyan Steven Marquez Acosta**

**Prueba Técnica: Conocimientos en Git**

**Instrucciones:**

Responde cada pregunta de manera clara y concisa. Para las preguntas prácticas, proporciona los comandos exactos que usarías.

**1. Conceptos Básicos**

¿Qué es Git y en qué se diferencia de GitHub?

* Git es el sistema de control de versiones, GitHub es la plataforma que lo aloja.
* Git es un sistema de control de versiones que permite rastrear cambios en el código, mientras que GitHub es una plataforma en línea que ofrece alojamiento y colaboración para proyectos que utilizan Git. Git se enfoca en la gestión del código, mientras que GitHub se enfoca en la colaboración y el almacenamiento en la nube.
* Explica la diferencia entre git pull y git push.
* Git pull descarga cambios del repositorio remoto, mientras que git push sube cambios al repositorio remoto.
* ¿Qué es un repositorio en Git?
* Un repositorio en Git es un almacenamiento centralizado para el código y su historial de cambios.
* ¿Qué es un commit en Git?
* Un commit es una instantánea de los cambios realizados en el código, con un mensaje descriptivo.
* ¿Cómo puedes ver el historial de commits de un repositorio?
* Con el comando "git log".

**2. Clonación y Ramas**

* ¿Cómo clonar un repositorio desde GitHub a tu máquina local?
* Con el comando git clone seguido de la URL del repositorio
* ¿Cómo listar todas las ramas de un repositorio?
* Con el comando git branch
* ¿Cómo cambiar a una rama específica?
* Con el comando git checkout y el nombre de la rama
* ¿Cómo crear una nueva rama y cambiar a ella inmediatamente?
* Con el comando git checkout -b y el nombre de la nueva rama
* ¿Qué comando se usa para eliminar una rama local?
* git branch -d y el nombre de la rama
* ¿Qué sucede si intentas eliminar una rama que no está fusionada?
* Git no permite eliminarla, se debe usar git branch -D para forzar la eliminación.

**3. Trabajando con Cambios y Sincronización**

* Explica la diferencia entre git add y git commit.
* git add selecciona los cambios que quieres preparar para guardar
* git commit guarda esos cambios en el historial del proyecto con un mensaje descriptivo
* ¿Cómo subir una rama nueva al repositorio remoto?
* crea y cambia a la rama con git checkout -b nombre-rama.
* Luego, haz tus cambios, usa git add .
* para preparar todo y git commit -m "mensaje" para guardar.
* Finalmente, sube la rama con git push origin nombre-rama
* ¿Cómo descargar los cambios del repositorio remoto sin fusionarlos automáticamente?
* usando git fetch.
* Esto trae las actualizaciones del remoto a tu repositorio local, pero no las mezcla con tu trabajo actual.
* ¿Cómo traer solo los cambios de un archivo específico desde la rama main sin cambiar el resto de los archivos en tu rama actual?
* git checkout con la ruta del archivo específico desde la rama main
* git checkout main -- ruta
* ¿Qué diferencia hay entre git merge y git rebase?
* Merge es mejor para ramas compartidas (preserva historial).
* Rebase es Ideal para limpiar historial antes de fusionar (evita commits de fusión).

**4. Inspección y Seguimiento de Cambios**

* ¿Cómo puedes ver en qué rama estás actualmente?
* git branch
* ¿Cómo puedes ver el historial de commits con detalles como el autor y la fecha?
* git log
* ¿Cómo ver los cambios realizados en el último commit sin abrir un editor?
* git show
* ¿Cómo deshacer un commit sin perder los cambios?
* git reset --soft HEAD~1
* ¿Cómo revertir un commit que ya ha sido enviado al repositorio remoto?
* git revert <commit-hash>

**5. .gitignore y Configuración**

* ¿Qué es el archivo .gitignore y para qué sirve?
* .gitignore es un archivo de texto que especifica qué archivos o directorios Git debe ignorar y no rastrear.
* ¿Cómo configurar Git para que ignore todos los archivos .log en un proyecto?
* En el archivo .gitignore en la raíz de tu proyecto y agrega la siguiente línea \*.log
* ¿Si un archivo ya fue versionado en Git y luego lo agregas a .gitignore, se ignorará automáticamente? ¿Por qué?
* No, no se ignorará automáticamente. Git ya está rastreando el archivo, por lo que debes eliminarlo manualmente del índice de Git. Usa el siguiente comando: git rm --cached <nombre-del-archivo>

**6. Resolución de Problemas**

* ¿Qué significa cuando al hacer git push obtienes el error "rejected non-fast-forward"?
* ocurre cuando el repositorio remoto tiene cambios que no están en tu rama local, y Git no puede fusionarlos automáticamente.
* ¿Cómo lo solucionas?
* git pull origin <nombre-rama>
* git add .
* git commit -m "Resuelve conflictos"
* git push origin <nombre-rama>
* git push --force origin <nombre-rama>
* ¿Cómo solucionar conflictos de merge en Git?
* Identifica los archivos en conflicto
* git status
* Abre los archivos conflictivos y decidir qué cambios conservar.
* Agrega los archivos resueltos
* git add <archivo-resuelto>
* Finaliza el merge
* git commit
* Si realizaste cambios en varios archivos, pero solo quieres deshacer los cambios de uno sin afectar los demás, ¿qué comando usarías?
* git checkout -- <ruta-del-archivo>
* ¿Qué diferencia hay entre git reset --hard y git reset --soft?
* git reset --soft: deshace commits, pero mantiene los cambios en el área de staging.
* Útil si quieres rehacer el commit con cambios adicionales.
* git reset --hard: deshace commits y elimina todos los cambios en el working directory y el staging area.
* ¿Cómo podrías recuperar un commit eliminado con git reset --hard?
* git checkout <commit-hash>

7. Trabajo en Equipo y Buenas Prácticas

* ¿Por qué es importante hacer commits pequeños y con mensajes descriptivos?
* Facilita la revisión y detección de errores.
* Mejora la colaboración y otros entienden los cambios.
* Mantiene un historial limpio y útil.
* ¿Qué es un fork en GitHub y en qué casos se usa?
* Un fork es una copia de un repositorio en tu cuenta.
* Contribuir a proyectos ajenos.
* Experimentar sin afectar el original.
* Crear proyectos derivados.
* ¿Cómo hacer un pull request en GitHub?
* realizar un fork o clona el repositorio.
* Crea una rama nueva y realizar cambios.
* subir los cambios al repositorio con push.
* En GitHub, ir al repositorio, selecciona una rama y hacer clic en "New Pull Request".
* Describir los cambios y envíalo para revisión.
* git clone <url>
* git checkout -b <nueva-rama>   
  git add .
* git commit -m "Mensaje"
* git push origin <nueva-rama>
* ¿Por qué se recomienda usar git pull --rebase en lugar de git pull en algunos flujos de trabajo?
* Se recomienda git pull --rebase en lugar de git pull en algunos flujos de trabajo porque mantiene un historial de commits más limpio y lineal. Con git pull se crea un commit de fusión adicional, lo que puede complicar el historial, especialmente en proyectos colaborativos.
* ¿Qué es un submódulo en Git y para qué sirve?
* Un submódulo en git es un repositorio anidado dentro de otro repositorio.
* Sirve para gestionar dependencias o proyectos relacionados, permitiendo mantener su código separado pero vinculado al proyecto principal.
* ¿Qué comando usarías para ver quién modificó una línea específica en un archivo?
* git blame
* ejemplo: git blame -L 10,10 archivo.txt muestra el autor de la línea 10 en "archivo.txt".