Trabajo Práctico 2 - Programación 1

Estudiante: Valentín Mattoni

Actividad 1: Preguntas teóricas

1) **¿Qué es GitHub?**

GitHub es una plataforma basada en la web que permite alojar proyectos que utilizan el sistema de control de versiones Git. Facilita la colaboración entre desarrolladores mediante funcionalidades como repositorios remotos, ramas, pull requests, y gestión de versiones.

2) **¿Cómo crear un repositorio en GitHub?**

Inicia sesión en GitHub, clic en "new", escribí el nombre del repositorio, seleccionás la visibilidad (público o privado), inicializá con README, clic en "Create repository".

3) **¿Cómo crear una rama en Git?**

Vas a la terminal y escribís git branch nombre-rama.

4) **¿Cómo cambiar a una rama en Git?**

Vas a la terminal y escribís git checkout nombre-rama.

5) **¿Cómo fusionar ramas en Git?**

Vas a la terminal y escribís “git checkout main” y luego git merge feature-1.

6) **¿Cómo crear un commit en Git?**

Vas a la terminal y escribís git add . para agregar todos los cambios hechos o git add- nombre del archivo para agregarlos individualmente y luegp git commit -m “Mensaje descriptivo”

7) **¿Cómo enviar un commit a GitHub?**

Después de hacer el commit localmente, vas a la terminal y escribís git push origin nombre-rama para subir esos cambios a la rama correspondiente del repositorio en GitHub.

8) **¿Qué es un repositorio remoto?**

Un repositorio remoto es una copia del proyecto alojada en un servidor externo, como GitHub. Permite trabajar en colaboración con otras personas y mantener una copia segura del código en la nube. Los cambios realizados localmente pueden ser sincronizados con el repositorio remoto mediante comandos como git push y git pull.

9) **¿Cómo agregar un repositorio remoto a Git?**

Vas a la terminal y escribís git remote add origin URL-del-repositorio. Esto le indica a Git que ese repositorio local está vinculado con uno remoto en GitHub. A partir de ahí, podés usar comandos como git push o git pull para enviar o recibir cambios.

10) ¿Cómo empujar cambios a un repositorio remoto?

Después de hacer commits, escribís git push origin nombre-rama en la terminal. Esto sube todos los cambios que hiciste localmente a la rama correspondiente del repositorio remoto, permitiendo que otros puedan verlos y usarlos.

11) ¿Cómo tirar de cambios de un repositorio remoto?

Para bajar los últimos cambios hechos por otros (o por vos mismo en otra computadora), escribís git pull origin nombre-rama. Este comando actualiza tu rama local con lo que haya en la rama remota, ayudando a mantener el proyecto sincronizado.

12) ¿Qué es un fork de repositorio?

Un fork es una copia de un repositorio que se crea en tu cuenta de GitHub. Sirve para modificar un proyecto que no te pertenece sin tocar el original. Es muy común en proyectos de código abierto donde la comunidad colabora mejorando el código desde sus propias copias.

13) ¿Cómo crear un fork de un repositorio?

Entrás al repositorio original en GitHub y hacés clic en el botón “Fork”. Automáticamente, GitHub va a crear una copia en tu cuenta para que puedas trabajar desde ahí sin afectar el repositorio original.

14) ¿Cómo enviar una solicitud de extracción (pull request)?

Luego de hacer el fork y subir los cambios:

- Ir a GitHub

- Clic en “Compare & pull request”

- Describir los cambios

- Clic en “Create pull request”

15) ¿Cómo aceptar una solicitud de extracción?

Vas a la pestaña “Pull requests” del repositorio, elegís la solicitud que querés revisar, y si todo está bien, hacés clic en “Merge pull request”. Esto integra los cambios en el código principal y los deja listos para usarse.

16) ¿Qué es una etiqueta en Git?

Una etiqueta (o *tag*) es una marca especial que se usa para señalar un punto importante en el historial del proyecto, como una versión estable o un lanzamiento. A diferencia de las ramas, las etiquetas no cambian con el tiempo.

17) ¿Cómo crear una etiqueta en Git?

Vas a la terminal y escribís git tag nombre-etiqueta para marcar el último commit, o git tag nombre-etiqueta id-commit si querés etiquetar un commit específico.

18) ¿Cómo enviar una etiqueta a GitHub?

Después de crear la etiqueta, podés subirla con git push origin nombre-etiqueta. Si querés subir todas las etiquetas juntas, usás git push origin --tags.

19) ¿Qué es un historial de Git?

Es el registro cronológico de todos los commits que se hicieron en el repositorio. Muestra quién hizo cada cambio, cuándo y con qué mensaje, lo cual permite entender cómo fue evolucionando el proyecto.

20) ¿Cómo ver el historial de Git?

Vas a la terminal y escribís git log

21) ¿Cómo buscar en el historial de Git?

Hay dos opciones:

Por mensaje: git log --grep="texto"

Por autor: git log --author="nombre"

22) ¿Cómo borrar el historial de Git?

Si querés editar o eliminar commits anteriores, usás git rebase -i HEAD~N, donde “N” es la cantidad de commits a revisar. Desde ahí podés elegir qué commit borrar, editar o reordenar. Ojo: este comando modifica el historial, así que hay que usarlo con cuidado.

23) ¿Qué es un repositorio privado en GitHub?

Es un repositorio visible solo para el propietario y sus colaboradores.

24) ¿Cómo crear un repositorio privado en GitHub?

Al crear el repositorio, seleccionás la opción Private.

25) ¿Cómo invitar a alguien a un repositorio privado en GitHub?

Vas a “Settings”, luego a “Collaborators”, ingresás su nombre de usuario o email y hacés clic en Add.

26) ¿Qué es un repositorio público en GitHub?

Es un repositorio visible para cualquier usuario de GitHub o incluso sin cuenta. Aunque todos lo pueden ver, solo los colaboradores con permiso pueden hacer cambios en el código.

27) ¿Cómo crear un repositorio público en GitHub?

Desde tu cuenta de GitHub, hacés clic en “New Repository”, ponés el nombre, y en la sección de visibilidad seleccionás “Public”. Después hacés clic en “Create repository”.

28) ¿Cómo compartir un repositorio público en GitHub?

Copiás la URL del repositorio (desde la barra del navegador o desde el botón verde “Code”) y la compartís por mensaje, mail o donde quieras. Cualquiera con ese enlace puede ver tu proyecto.

ACTIVIDAD 2) https://github.com/vmattoni/Tp2_Programacion1_Act2

ACTIVIDAD 3) https://github.com/vmattoni/conflict-exercise