Actividades

1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas) :

• ¿Qué es GitHub?
Github es una plataforma que almacena repositorios de forma remota

• ¿Cómo crear un repositorio en GitHub?
Para crear un repositorio en github, primero debemos tener una cuenta y loguearnos. Una vez logueados dirigirse /new (desde barra lateral o superior en el ícono con el “+”)

• ¿Cómo crear una rama en Git?
Para crear una rama se usa el comando git branch <nombre de la rama>

• ¿Cómo cambiar a una rama en Git?
Para cambiar de rama se usa el comando git checkout <rama>

• ¿Cómo fusionar ramas en Git?
Para fusionar ramas se usa el comando git merge <rama>

• ¿Cómo crear un commit en Git?
Para crear un commit se usa el comando git commit -m “mensaje”

• ¿Cómo enviar un commit a GitHub?
Para enviar un commit se usa el comando git push <rama remota>/origin <rama remota>

• ¿Qué es un repositorio remoto?
Un repositorio remoto es una forma de almacenar un repositorio en una plataforma online, sincronizado con un repositorio local, de forma que pueda ser accedido, forkeado y clonado por otros usuarios, y que así puedan colaborar en nuestro proyecto

• ¿Cómo agregar un repositorio remoto a Git?
Para agregar un repositorio en Git, se usa el comando git add remote origin <repositorio remoto>; o bien se crea un for/clona el repositorio remoto con git clone <url>

• ¿Cómo empujar cambios a un repositorio remoto?
Para empujar cambios a un repositorio remoto se usan los comandos:

```
git add .
git commit -m “mensaje”
git push origin <rama>
```

• ¿Cómo tirar de cambios de un repositorio remoto?
Para traer cambios de un repositorio remoto se usa el comando git pull/origin <rama>

• ¿Qué es un fork de repositorio?
Un fork es una copia de un repositorio que se almacena en nuestra cuenta, para que podamos hacer cambios sin afectar al repositorio original

• ¿Cómo crear un fork de un repositorio?
Para crear un fork, nos dirigimos al repositorio en GitHub y hacemos click en botón fork. Esto nos lleva a una ventana donde podemos elegir cambiar nombre y descripción y qué rama queremos

• ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
Para enviar un pull requestprimero creamos una rama para realizar nuestros cambio.
Lluego de commitear los cambios se pushea la nueva rama al repositorio remoto. Luego vamos a nuestro repositorio con la rama que pusheamos y clickeamos “pull request” y rellenamos los datos que nos pide

• ¿Cómo aceptar una solicitud de extracción?
Para aceptar un pull equest vamos a nuestro repositorio en github y luego a la pestaña “pull requests”. Luego vamos a la solicitud que queremos revisar y vemos los cambios que se hicieron en “files changed”. Se puede hacer algún comentario de feedback o petición de cambios desde “review changes” y “request changes” y si está todo correcto clickeamos “approve”

• ¿Qué es una etiqueta en Git?
Una etiqueta o tag en git es una referencia a un punto específico en la historia del repositorio, en general usado para marcar reléase de versiones (por ejemplo, se crea la versión 1.0 de una app y se asigna el tag 1.0)

• ¿Cómo crear una etiqueta en Git?
En git GUI se puede crear co Create tag y desde consola con git tag <nombre>

• ¿Cómo enviar una etiqueta a GitHub?
Para enviar un tag a un repositorio remoto se debe hacer de forma explícita al hacer push:
git push <remoto> tags

• ¿Qué es un historial de Git?
El historial de git muestra todos los movimientos que se hicieron en una rama particular del repositorio desde que se inicializó: muestra commits, tags, merges y merge/conflicts, y también el usuario que realizó los cambios y la hora en que se hizo.

• ¿Cómo ver el historial de Git?
El historial se puede ver con:

```
git log
git log –graph
git log –graph -oneline o
git log --pretty=oneline
```

• ¿Cómo buscar en el historial de Git?

Se puede buscar en el historial de Git con git log por el hash del commit

• ¿Cómo borrar el historial de Git?

El historial de Git puede borrarse si borramos la carpeta oculta .git

• ¿Qué es un repositorio privado en GitHub?
Un repositorio privado es uno al que sólo el usuario que lo creó tiene acceso.

• ¿Cómo crear un repositorio privado en GitHub?
Para hacer privado un repositorio debemos cambiar su visibilidad desde repositorio > configuración > privacidad > visibilidad: privado

• ¿Cómo invitar a alguien a un repositorio privado en GitHub?
Para invitar a alguien a colaborar debe dirigirse a configuración > colaboradores > agregar colaborador

• ¿Qué es un repositorio público en GitHub?
Un repositorio público es aquel que es visible para cualquiera en internet.

• ¿Cómo crear un repositorio público en GitHub?
Para crear un repositorio público simplemente creamos un nuevo repositorio, y dejamos tildada la opción de repositorio publico

• ¿Cómo compartir un repositorio público en GitHub?

Para compartir un repositorio público simplemente copiamos la url del mismo y lo compartimos con quien queramos.

2) Realizar la siguiente actividad:

• Crear un repositorio.

Dale un nombre al repositorio. ✓

Elije el repositorio sea público. ✓

Inicializa el repositorio con un archivo. ✓

• Agregando un Archivo

- Crea un archivo simple, por ejemplo, "mi-archivo.txt". ✓
- Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos. ✓
- Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente). ✓

• Creando Branchs o Crear una Branch

- Realizar cambios o agregar un archivo ✓
- Subir la Branch ✓
- Realizar la siguiente actividad:

Paso 1: Crear un repositorio en GitHub

• Ve a GitHub e inicia sesión en tu cuenta.

• Haz clic en el botón "New" o "Create repository" para crear un nuevo repositorio.

• Asigna un nombre al repositorio, por ejemplo, conflict-exercise.

• Opcionalmente, añade una descripción.

• Marca la opción "Initialize this repository with a README".

• Haz clic en "Create repository".

Paso 2: Clonar el repositorio a tu máquina local

• Copia la URL del repositorio (usualmente algo como https://github.com/tuusuario/conflict-exercise.git).

• Abre la terminal o línea de comandos en tu máquina.

• Clona el repositorio usando el comando:

git clone https://github.com/tuusuario/conflict-exercise.git

• Entra en el directorio del repositorio:

cd conflict-exercise

Paso 3: Crear una nueva rama y editar un archivo

• Crea una nueva rama llamada feature-branch: ✓

git checkout -b feature-branch ✓

• Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo:

Este es un cambio en la feature branch.

• Guarda los cambios y haz un commit:

git add README.md ✓

git commit -m "Added a line in feature-branch" ✓

Paso 4: Volver a la rama principal y editar el mismo archivo

• Cambia de vuelta a la rama principal (main):

git checkout main ✓

• Edita el archivo README.md de nuevo, añadiendo una línea diferente:

Este es un cambio en la main branch. ✓

• Guarda los cambios y haz un commit:

git add README.md ✓

git commit -m "Added a line in main branch" ✓

Paso 5: Hacer un merge y generar un conflicto

• Intenta hacer un merge de la feature-branch en la rama main: Git merge feature-branch ✓

• Se generará un conflicto porque ambos cambios afectan la misma línea del archivo README.md.

Paso 6: Resolver el conflicto

• Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:

<<<<<<< HEAD

Este es un cambio en la main branch.

=======

Este es un cambio en la feature branch.

>>>>>>> feature-branch

• Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera. ✓

• Edita el archivo para resolver el conflicto y guarda los cambios(Se debe borrar lo marcado en verde en el archivo donde estes solucionando el conflicto. Y se debe borrar la parte del texto que no se quiera dejar).

• Añade el archivo resuelto y completa el merge:

git add README.md ✓

git commit -m "Resolved merge conflict" ✓

Paso 7: Subir los cambios a GitHub

• Sube los cambios de la rama main al repositorio remoto en GitHub:

git push origin main✓

• También sube la feature-branch si deseas:

git push origin feature-branch ✓

Paso 8: Verificar en GitHub

• Ve a tu repositorio en GitHub y revisa el archivo README.md para confirmar que los cambios se han subido correctamente. ✓

• Puedes revisar el historial de commits para ver el conflicto y su resolución. ✓

```
Usuario@DESKTOP-QVNJCFS MINGW64 /d/VSCODE/TUP/semana_2/nata_c15 (master)
$ git log --graph --oneline
*   ca88ada (HEAD -> master, origin/master) resolve merge conflict
|\
| * df56897 (origin/feature-branch, feature-branch) cambios en readme y archivo txt
* | 3f96899 cambio en la rama master
|/
* e8604b9 Agregando mi-archivo.txt
```

Repositorio en github: https://github.com/nosoynati/nata_c15.git