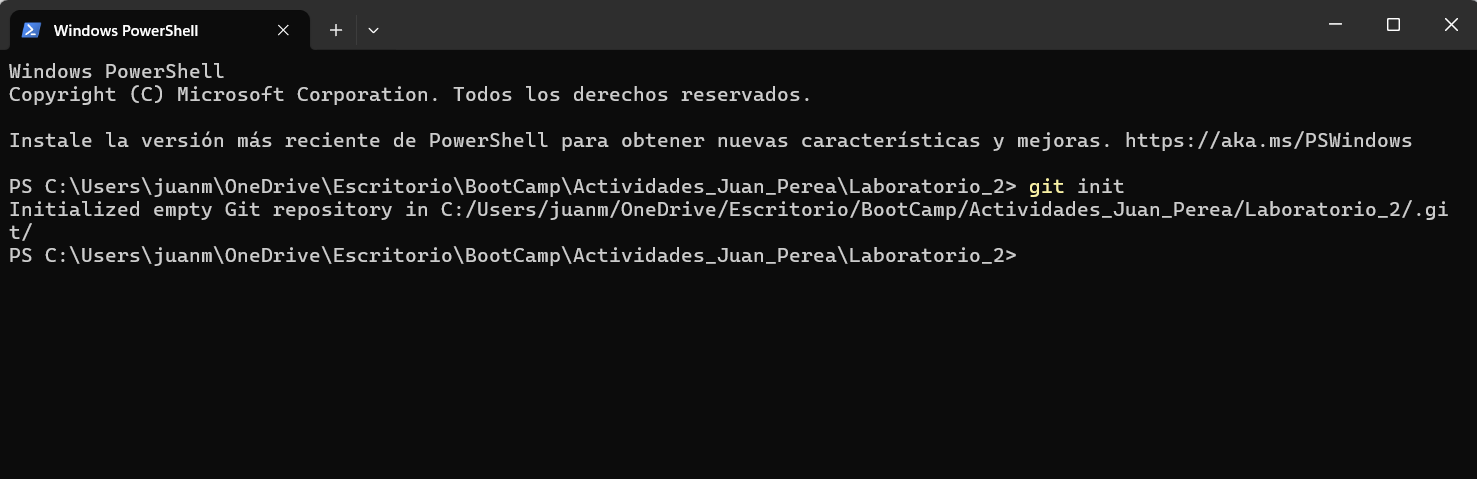
**Nombre:** Juan Manuel Perea Coronado

**DESARROLLO DEL LABORATORIO 2.**

1. **Creación del repositorio local usando GIT:**

Desde la consola de Windows, se debe ingresar a la carpeta donde se hará el repositorio local, como espacio de trabajo. Luego, se usa el comando “git init” para crear el repositorio local ***(Ver Imagen 1)***:



*Imagen 1. Creación del repositorio local.*

Con lo anterior, se creará un archivo (No visible) dentro de la carpeta, indicando la creación exitosa del repositorio.

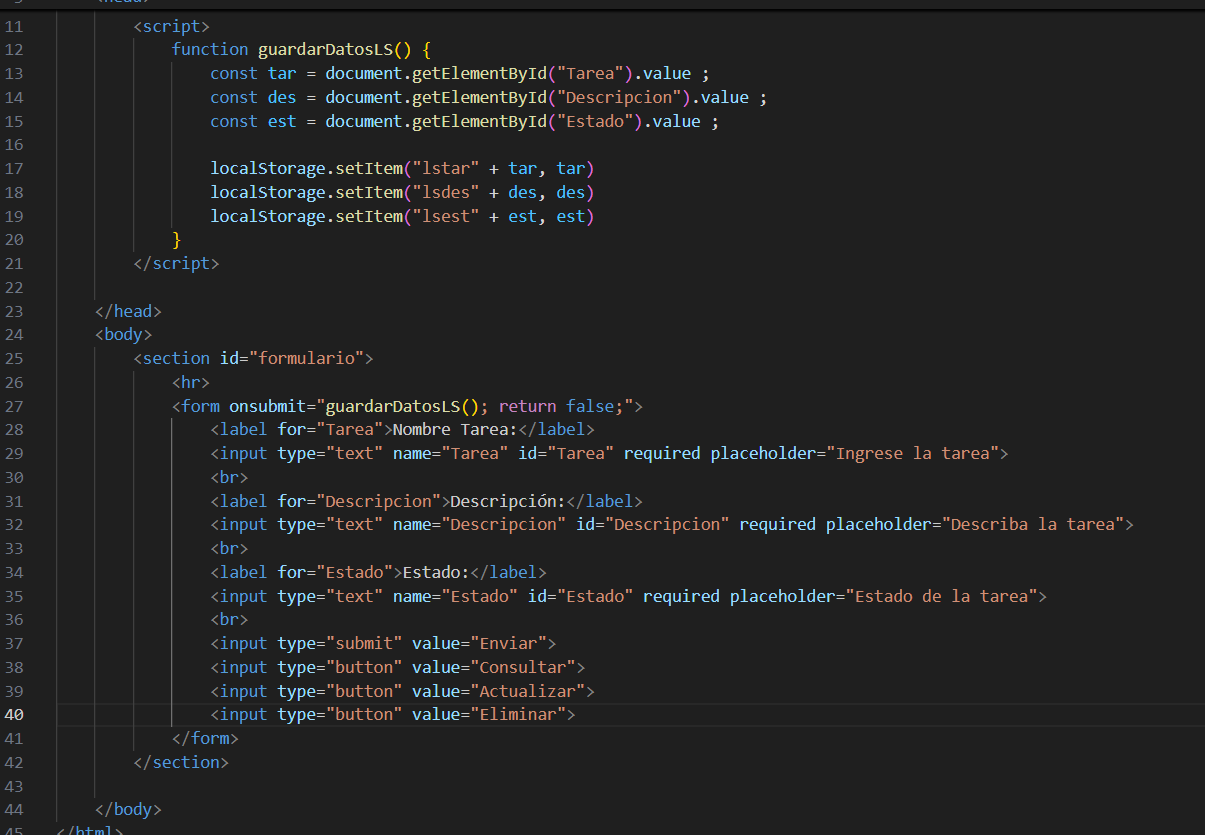
1. **Creación en la rama “master”, donde se agrega la funcionalidad de crear una tarea:**

Primero se crea un archivo .html donde se elabora un formulario junto con la funcionalidad de almacenar la tarea descrita en él dentro del Local Storage ***(Ver Imagen 2)***.

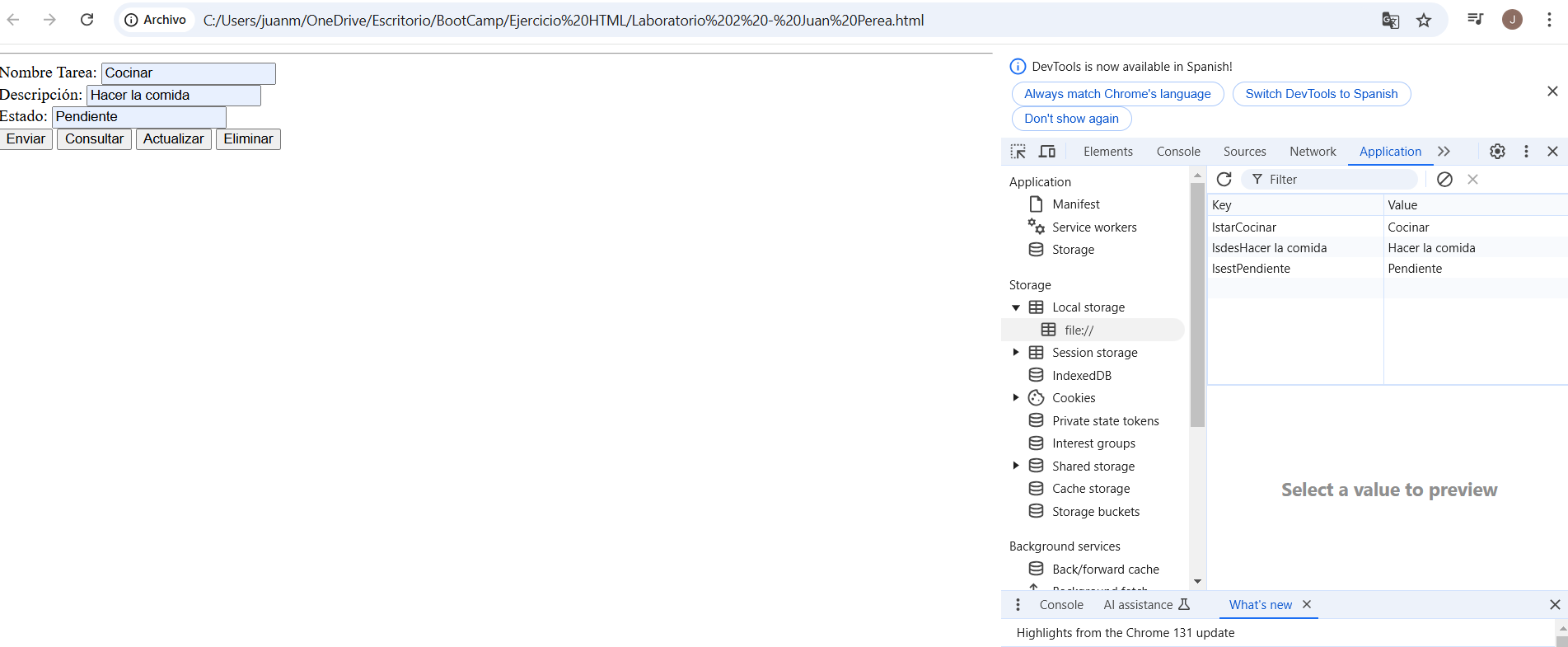
Luego, se guarda y se confirma que esté funcionando adecuadamente en el navegador ***(Ver Imagen 3)***.

Se guarda en la carpeta donde se tiene creado el repositorio. A continuación, se introduce en la consola el comando “git add .” para agregar los cambios a la zona de Stage. Al usar el comando “git status” se puede notar que se está trabajando sobre la rama “master” ***(Ver Imagen 4)***.

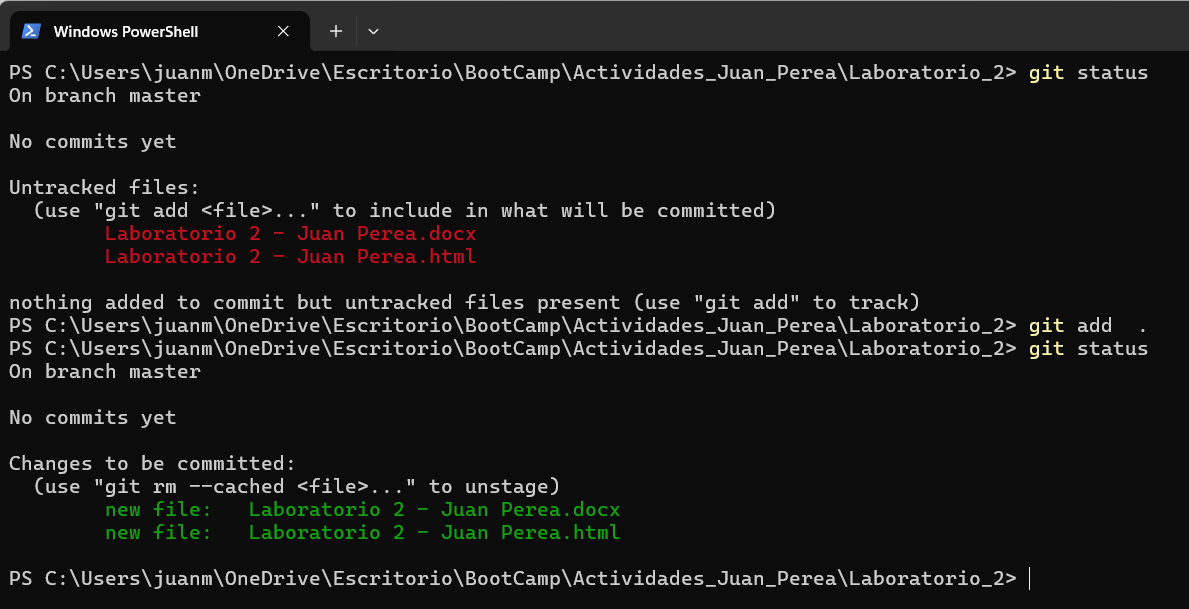
Por último, se hace el Commit a la rama “master”, para pasar los cambios a la zona de Head. Esto se logra usando el comando “git commit -m “Commit 1 - Crear Tarea”” ***(Ver Imagen 5)***.



*Imagen 2. Archivo .html.*



*Imagen 3. Prueba de funcionamiento – Creación de una tarea.*



*Imagen 4. Guardar cambios a Stage.*

Texto

Descripción generada automáticamente

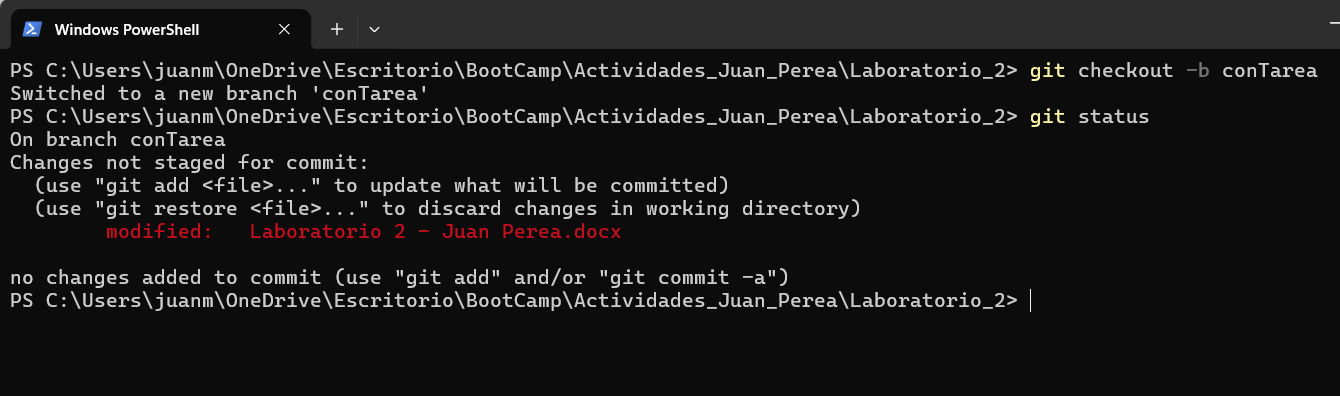
*Imagen 5. Ejecución del Commit.*

1. **Creación de la rama “conTarea”, donde se agrega la funcionalidad de consultar una tarea:**

Se crea la rama “conTarea”, usando el comando “git checkout -b conTarea” y se verifica que se está trabajando sobre ella usando el comando “git status” ***(Ver Imagen 6)***.

Se hacen las modificaciones necesarias en el archivo .html, creando la función para consultar los datos del Local Storage y que se ejecute al darle click en el botón “Consultar” ***(Ver Imagen 7)***. También se verifica la funcionalidad del botón desde el navegador.

Finalmente, se guardan los cambios en la carpeta del repositorio y se procede a guardarlos en la zona Stage de la rama usando el comando “git add .” ***(Ver Imagen 8)***, y se hace el Commit con “git commit -m “Commit 2 – Consultar Tarea”” ***(Ver Imagen 9)***.



*Imagen 6. Creación de la rama “conTarea”.*

Texto

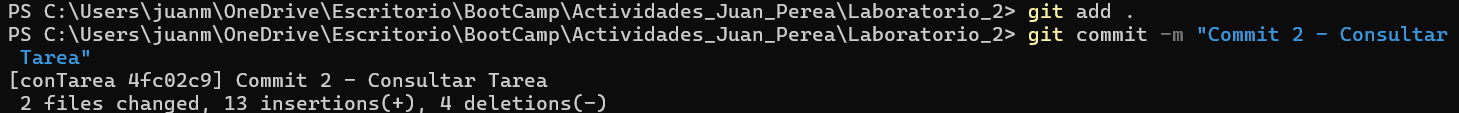
Descripción generada automáticamente

*Imagen 7. Creación de la función para consultar datos en el Local Storage.*

Texto

Descripción generada automáticamente

*Imagen 8. Cambios guardados en la rama “conTarea”.*

**

*Imagen 9. Commit en la rama “conTarea”.*

1. **Fusión de la rama “conTarea” en la rama “master”:**

Se usa el comando “git checkout master” para cambiar a la rama “master. Luego se usa el comando “git merge conTarea” para fusionar el contenido de la rama “conTarea” dentro de la rama “master” ***(Ver imagen 10)***.

Texto

Descripción generada automáticamente

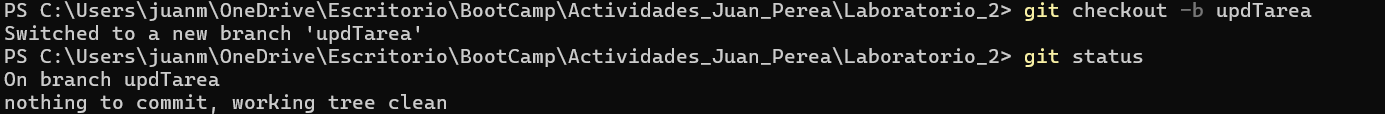
*Imagen 10. Fusión de la rama “conTarea” dentro de la rama “master”.*

1. **Creación de la rama “udpTarea”, donde se agrega la funcionalidad de actualizar una tarea:**

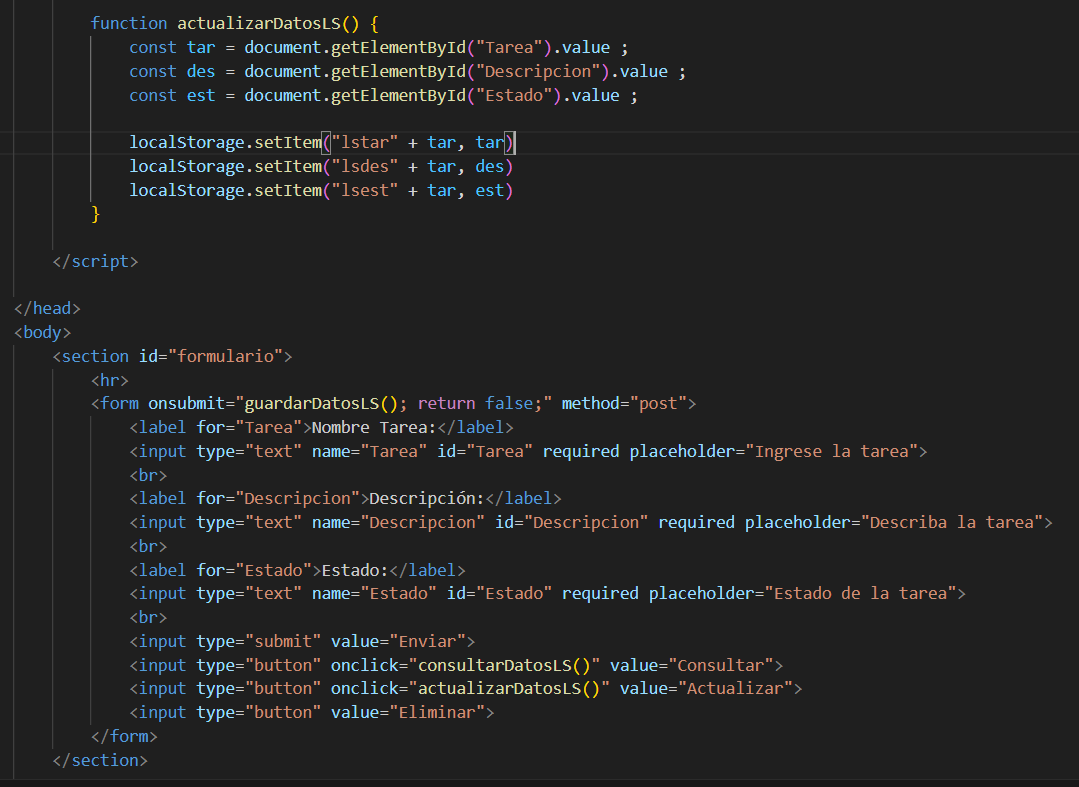
Se crea la rama “updTarea”, usando el comando “git checkout -b updTarea” y se verifica que se está trabajando sobre ella usando el comando “git status” ***(Ver Imagen 11)***.

Se hacen las modificaciones necesarias en el archivo .html, creando la función para actualizar los datos del Local Storage y que se ejecute al darle click en el botón “Actualizar” ***(Ver Imagen 12)***. También se verifica la funcionalidad del botón desde el navegador.

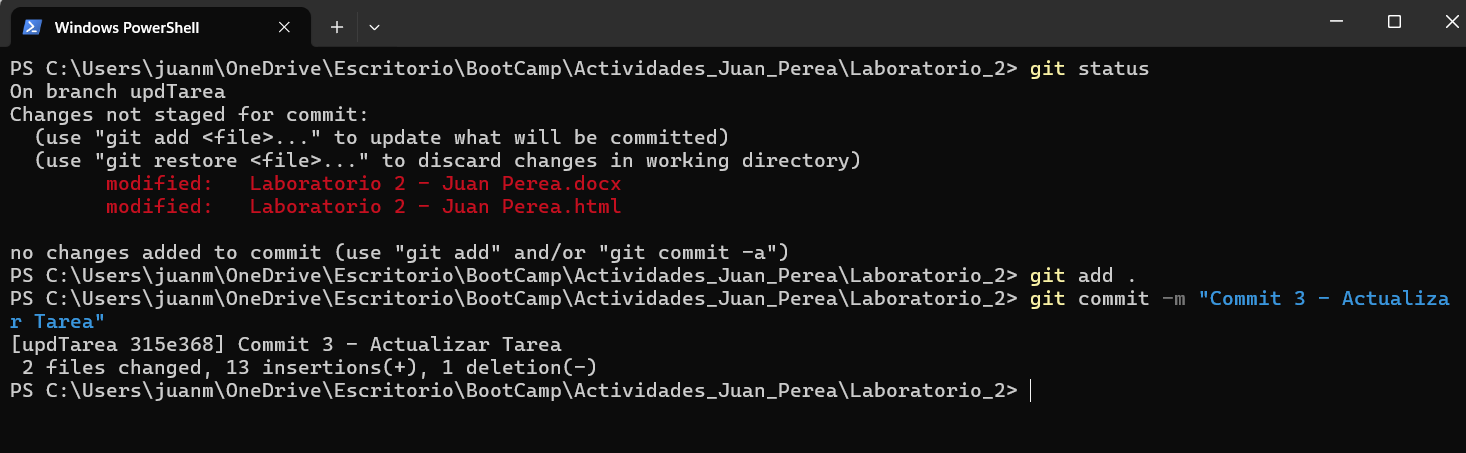
Finalmente, se guardan los cambios en la carpeta del repositorio y se procede a guardarlos en la zona Stage de la rama usando el comando “git add .” y se hace el Commit con “git commit -m “Commit 3 – Actualizar Tarea”” ***(Ver Imagen 13)***.



*Imagen 11. Creación de la rama “updTarea”.*



*Imagen 12. Creación de la función para actualizar datos en el Local Storage.*



*Imagen 13. Cambios guardados en la rama “updTarea”.*

1. **Fusión de la rama “updTarea” en la rama “master”:**

Se usa el comando “git checkout master” para cambiar a la rama “master. Luego se usa el comando “git merge updTarea” para fusionar el contenido de la rama “updTarea” dentro de la rama “master” ***(Ver imagen 14)***.

Texto

Descripción generada automáticamente

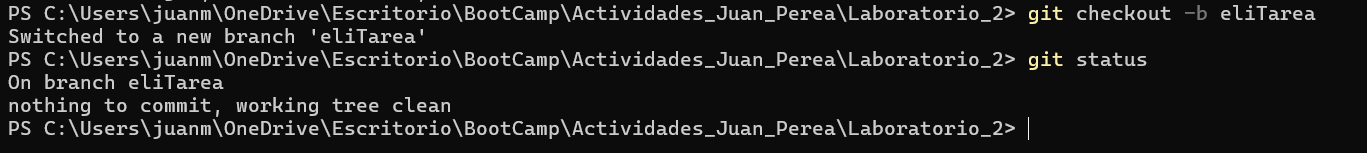
*Imagen 14. Fusión de la rama “updTarea” dentro de la rama “master”.*

1. **Creación de la rama “eliTarea”, donde se agrega la funcionalidad de eliminar una tarea:**

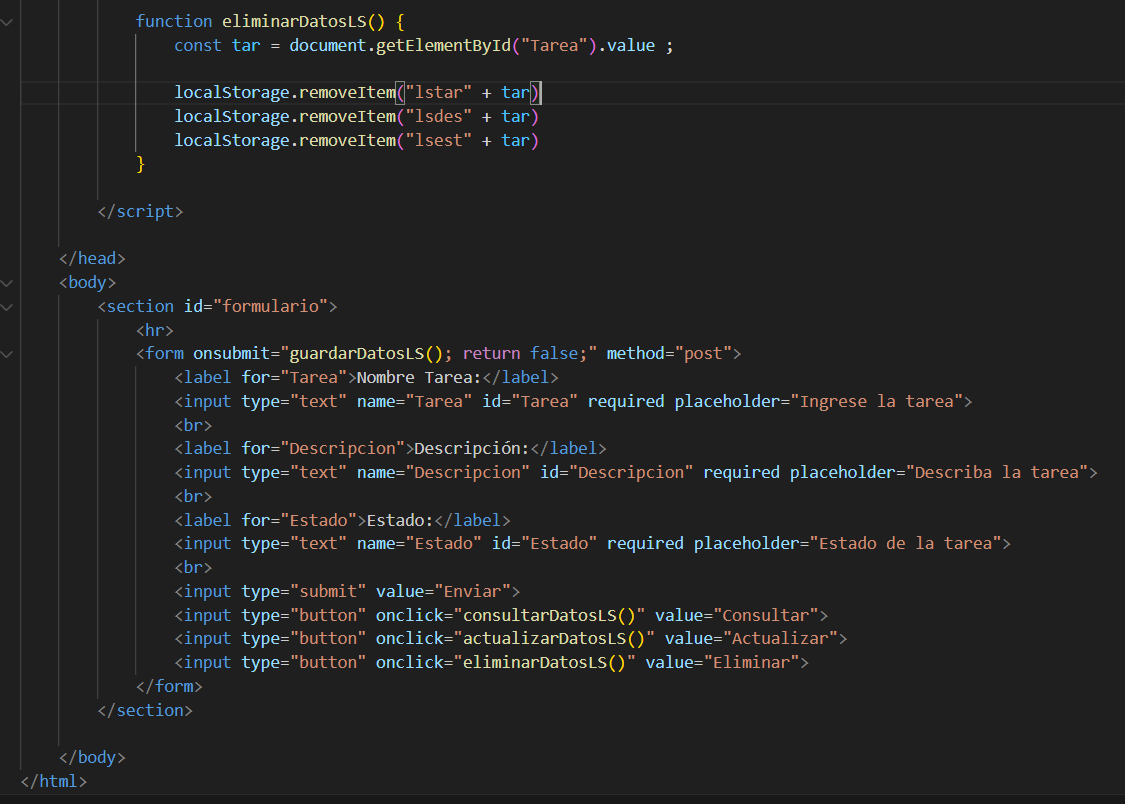
Se crea la rama “eliTarea”, usando el comando “git checkout -b eliTarea” y se verifica que se está trabajando sobre ella usando el comando “git status” ***(Ver Imagen 15)***.

Se hacen las modificaciones necesarias en el archivo .html, creando la función para eliminar los datos del Local Storage y que se ejecute al darle click en el botón “Eliminar” ***(Ver Imagen 16)***. También se verifica la funcionalidad del botón desde el navegador.

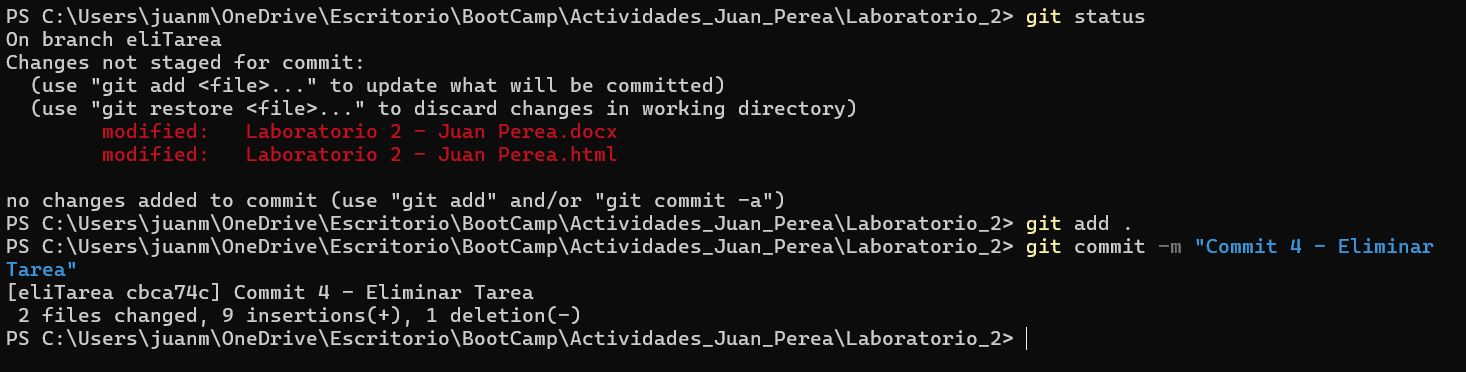
Finalmente, se guardan los cambios en la carpeta del repositorio y se procede a guardarlos en la zona Stage de la rama usando el comando “git add .” y se hace el Commit con “git commit -m “Commit 4 – Eliminar Tarea”” ***(Ver Imagen 17)***.



*Imagen 15. Creación de la rama “eliTarea”.*



*Imagen 16. Creación de la función para eliminar los datos del Local Storage.*



*Imagen 17. Cambios guardados en la rama “eliTarea”.*

1. **Fusión de la rama “eliTarea” en la rama “master”:**

Se usa el comando “git checkout master” para cambiar a la rama “master. Luego se usa el comando “git merge eliTarea” para fusionar el contenido de la rama “eliTarea” dentro de la rama “master” ***(Ver Imagen 18)***.

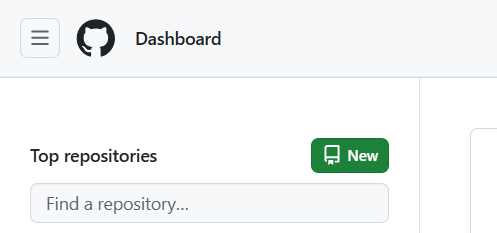
Texto

Descripción generada automáticamente

Imagen 18. Fusión de la rama “eliTarea” dentro de la rama “master”.

1. **Creación de un repositorio público en GitHub:**

Se crea un nuevo repositorio en la página de github.com. Para esto, luego de crear una cuenta, se da click en el botón “New” ***(Ver Imagen 19)***. Luego en la nueva página que aparece, se llenan los datos del repositorio, como el nombre (Laboratorio 2 – Juan Perea), la descripción y la disposición del repositorio (Público) ***(Ver Imagen 20)***.



*Imagen 19. Botón “New” de GitHub para crear un nuevo repositorio.*

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

*Imagen 20. Formulario para crear un nuevo repositorio.*

Luego se da click en el botón “Create Repository” para finalizar su creación. A continuación se presenta la página principal del repositorio creado (Ver Imagen 21).

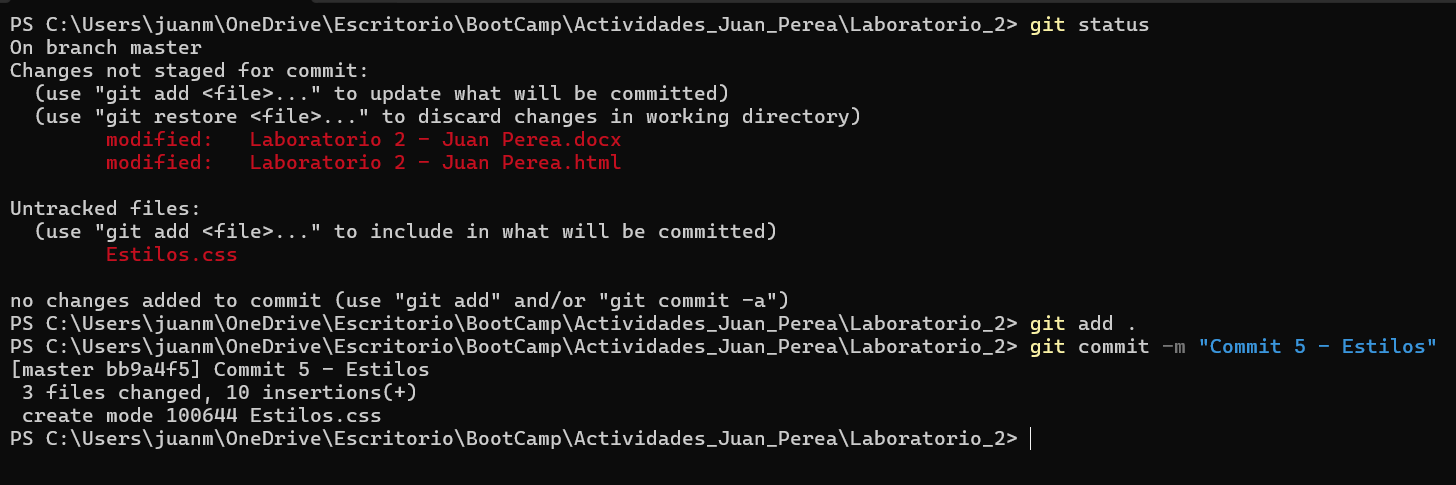
Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

*Imagen 21. Repositorio creado.*

1. Copia del proyecto al repositorio:

Para finalizar con el laboratorio, se crea una hoja Estilos.css para darle un poco de color al formulario creado y se hacen algunas modificaciones al archivo .html. Estos cambios son guardados en la rama master usando “git add .” y luego haciendo el Commit con “git commit -m “Commit 5 – Estilos” ***(Ver Imagen 22)***.



*Imagen 22. Cambio guardado de los estilos en la rama “master.*

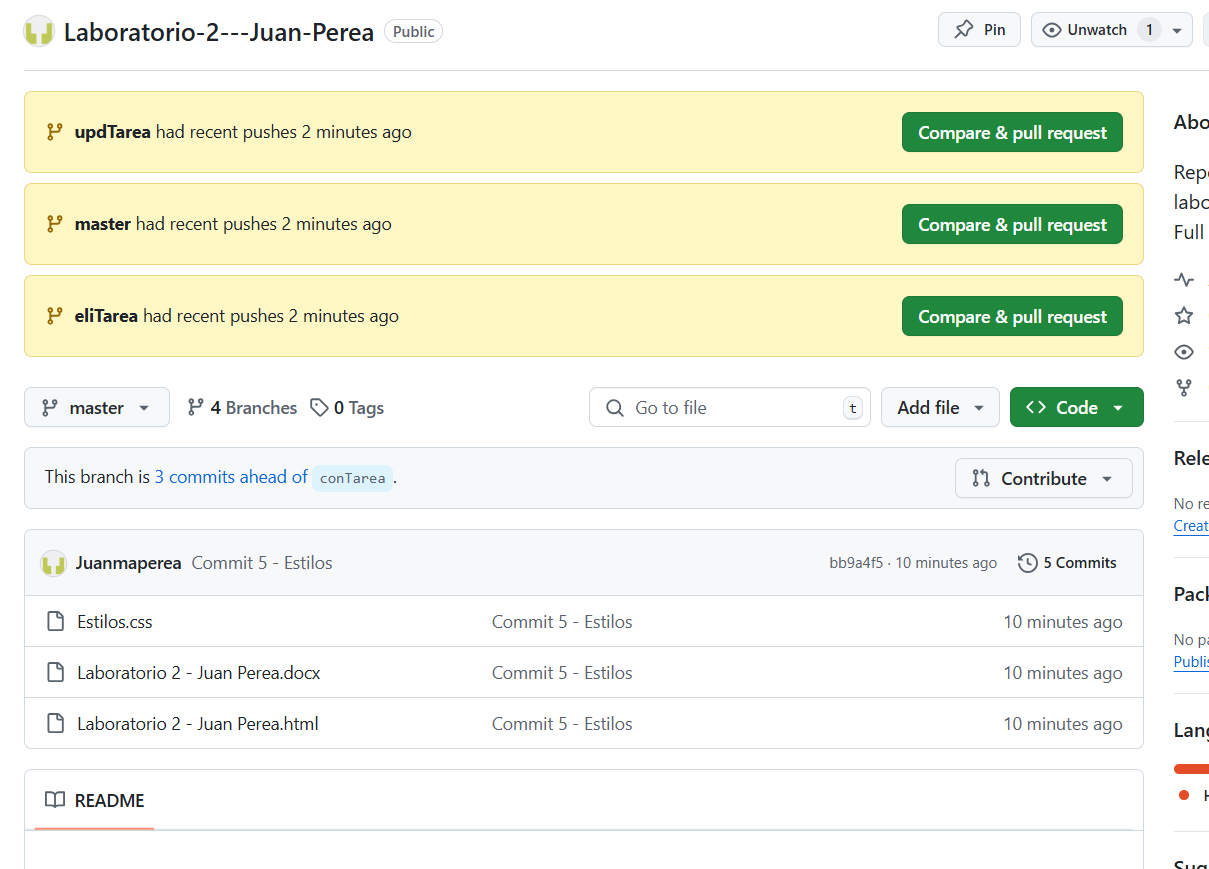
Ahora. Para asociar el repositorio local con el remoto de GitHub, se usa el comando “git remote add origin <https://github.com/Juanmaperea/Laboratorio-2---Juan-Perea.git>”, donde la URL que se agrega es la del repositorio creado anteriormente en GitHub. Finalmente, se llevan los cambios al repositorio remoto usando el comando “git push –all” ***(Ver Imagen 23)***.

Texto

Descripción generada automáticamente

*Imagen 24. Copia del repositorio local en el remoto.*

Lo anterior se puede corroborar entrando en el repositorio remoto desde el navegador, donde se podrán encontrar todos los documentos creados ***(Ver Imagen 25)***.



*Imagen 25. Repositorio remoto en GitHub.*

**URL del repositorio de GitHub donde se encuentra el laboratorio:**

<https://github.com/Juanmaperea/Laboratorio-2---Juan-Perea.git>