Mantenimiento de software

Unidad de aprendizaje 2

Actividad 4 - Utilizando sistemas de control de versiones

Hernán Ricardo Loaiza Doncel
David Felipe Trejos Mirando

Facultad de ingeniería

Docente tutor. Rogelio Vasquez
Ingeniería de software

Corporación Universitaria Iberoamericana
Bogotá – Colombia
Mayo 25 de 2025

Tabla de Contenido

Introducción

El presente documento describe el proceso de implementación de un proyecto utilizando Git y GitHub como herramientas de control de versiones y trabajo colaborativo. La actividad tiene como objetivo desarrollar competencias en el manejo básico de Git, desde la creación de un repositorio hasta el trabajo en equipo mediante ramas e implementación de historias de usuario.

Historia, Promoción y Objetivo de GitHub

Historia y Evolución

1. Fundación y Crecimiento Inicial (2008-2009)

GitHub fue fundada en 2008 por Chris Wanstrath, PJ Hyett, Tom Preston-Werner y Scott Chacon. Desde sus inicios, destacó por su enfoque en la colaboración y facilidad de uso. En su primer año ya contaba con más de 46,000 repositorios y 100,000 usuarios.

2. Expansión y Consolidación (2010-2013)

La plataforma creció rápidamente, alcanzando el millón de repositorios en 2010 y superando a competidores como SourceForge y Google Code. En 2013 ya tenía más de 3 millones de usuarios y 5 millones de repositorios.

3. Adquisición por Microsoft (2018)

Microsoft adquirió GitHub por 7.500 millones de dólares. A pesar de las dudas iniciales, GitHub mantuvo su enfoque en el código abierto y se integró con servicios como Azure DevOps, fortaleciendo su ecosistema.

4. Impacto Actual

Hoy, GitHub es el centro global del desarrollo colaborativo, con más de 100 millones de usuarios y millones de repositorios activos. Es clave en proyectos de software, inteligencia artificial, ciencia de datos, aplicaciones móviles y más.

Promoción de GitHub

GitHub creció sin campañas publicitarias masivas, gracias a estrategias centradas en la comunidad y el desarrollador:

Eventos tecnológicos: Participación en conferencias y encuentros clave del sector.

Código abierto: Su enfoque atrajo a comunidades influyentes, generando un efecto multiplicador.

Interfaz amigable: Su diseño intuitivo facilitó la adopción por parte de desarrolladores de todos los niveles.

Redes sociales y comunidad: El apoyo en plataformas como Twitter, Reddit y blogs técnicos impulsó su difusión.

Estas acciones fomentaron un crecimiento orgánico, posicionando a GitHub como referente mundial en colaboración para el desarrollo de software.

Objetivo de GitHub

El objetivo principal de GitHub es facilitar la colaboración eficiente, segura y estructurada en el desarrollo de software. Para ello, ofrece herramientas como:

Control de versiones: Basado en Git, permite gestionar cambios, versiones y errores.

Colaboración distribuida: Funciones como pull requests, issues y code reviews permiten trabajo simultáneo en equipo.

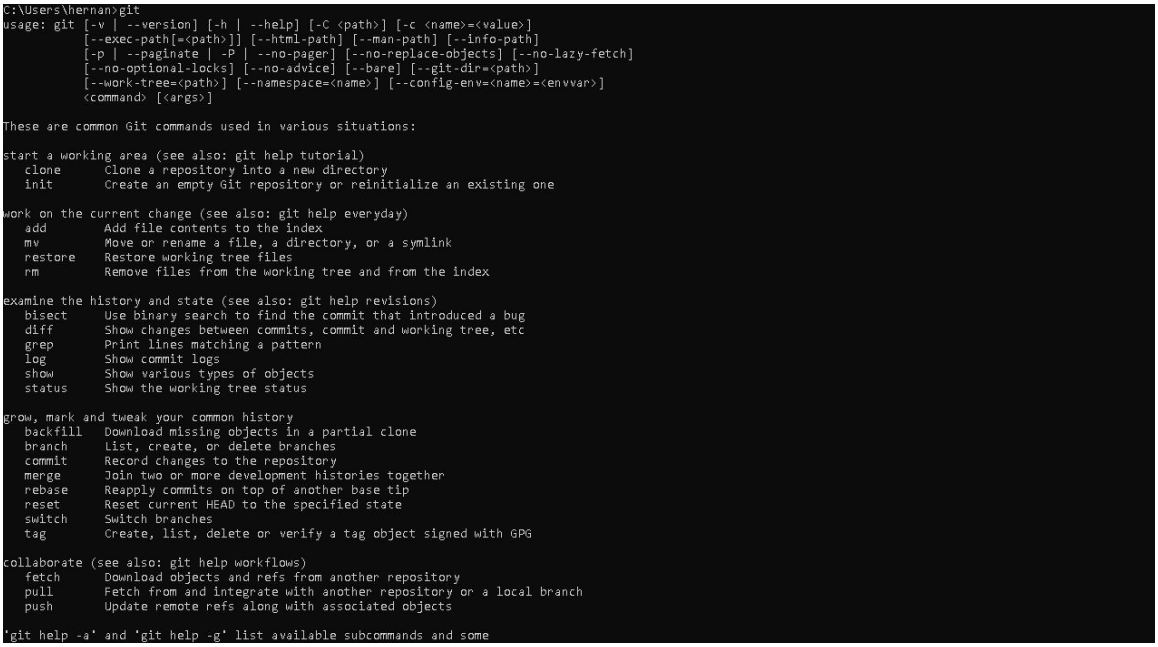
Automatización (CI/CD): Con GitHub Actions, se automatizan pruebas, despliegues y tareas repetitivas.

Seguridad integrada: Detecta vulnerabilidades y sugiere parches automáticos.

Documentación accesible: Archivos README, wikis y páginas estáticas mejoran la comprensión de los proyectos.

Procesos realizados

Comprobamos si temenos git instalado



Miramos que version temenos en la cual temenos la version 2.49.0 en la cual es la ultima version.

Pasos para realizar la instalacion

Se realizó la instalación de Git en los equipos de cada uno de los integrantes del grupo desde el sitio oficial https://git-scm.com/. Versión instalada: git --version Ejemplo de salida:

git version 2.49.0

Creacion de cuentas en GitHub

Cada integrante creó su cuenta en GitHub y compartió su usuario para ser añadido como colaborador.

Creacion del repositorio

Uno de los integrantes Hernan Loaiza creó el repositorio llamado mi- proyecto-equipo Enlace al repositorio: https://github.com/Hernansoft96/mi-proyecto-equipo

Subida del proyecto

Se utilizó un proyecto previamente desarrollado en otro módulo. El proyecto consiste en la cual es una app web “ficticio” es un formulario de contactos en la cual se guarda su registro

El equipo de trabajo
Un formulario funcional “aunque no envía datos, sirve como maqueta”
Galería “aunque los íconos no cargaron eso se puede ajustar a futuro ”
Tabla “en la cual se encuentra mas abajo del proyecto ”

Historia de usuario

Como administrador, quiero ver una lista de usuarios registrados, para poder gestionarlos eficientemente.

Como usuario, quiero editar mi perfil, para mantener mi información actualizada.

Como visitante, quiero registrarme en la plataforma, para acceder a las funcionalidades disponibles.

Como usuario, quiero ver un historial de mis actividades, para llevar un seguimiento de mi uso.

Como administrador, quiero eliminar usuarios inactivos, para mantener la base de datos limpia.

Trabajo colaborativo

Cada integrante tomó una historia de usuario y trabajó en su propia rama usando los siguientes comandos:

git checkout -b historia-usuario-x
git add .git commit -m "Implementación de historia x"git push origin historia-usuario-x

Los cambios fueron revisados y luego fusionados al repositorio principal mediante pull requests.

Comandos que vamos a utilizar.

Comando	Uso
git init	Inicializar repositorio local
git add	Añadir archivos al área de staging
git commit	Crear un snapshot del proyecto
git branch	Crear o listar ramas
git remote	Enlazar repositorio remoto
git push	Subir cambios al repositorio remoto
git status	Ver estado actual del repositorio
git log	Ver historial de commits
git clone	Clonar un repositorio remoto
git ignore	Ignorar archivos específicos con .gitignore

Evidencias de los pasos

inicializaste un repositorio Git local “git init”

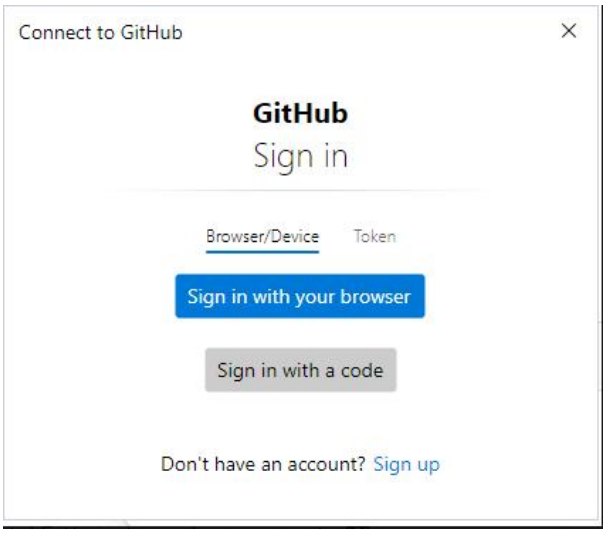
Configuraste tu usuario Git

Agregaste tu repositorio remoto de GitHub

Subiste tu primer commit con todos tus archivos “git push”

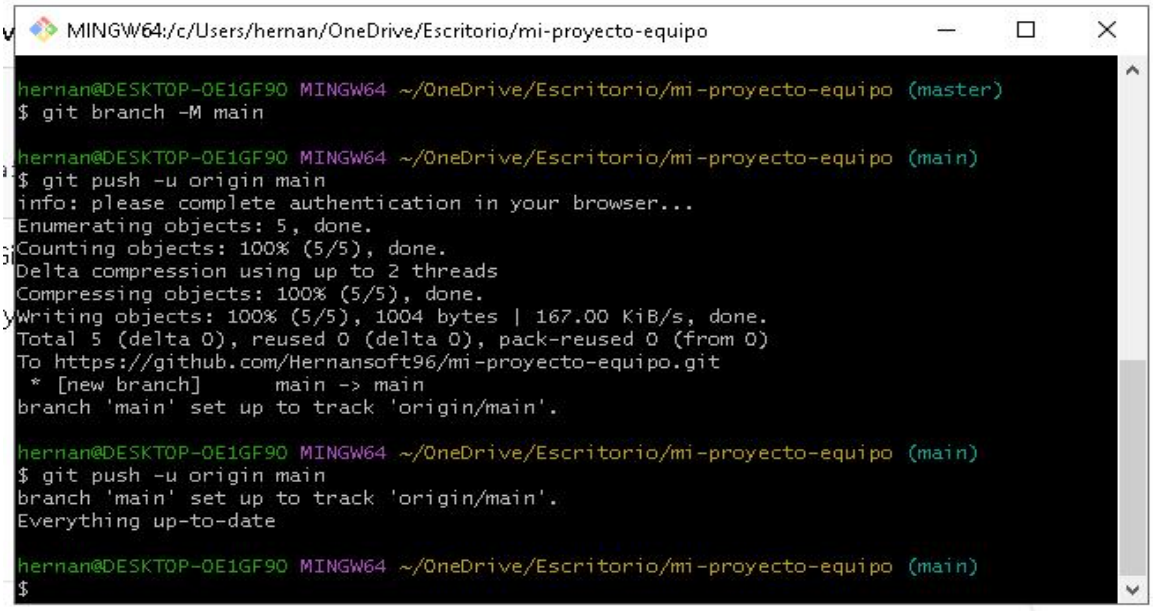
Cambiaste la rama por defecto de master a main “una buena práctica moderna”

Git necesita conectarse a tu cuenta de GitHub para poder **subir (push)** tu proyecto al repositorio remoto.

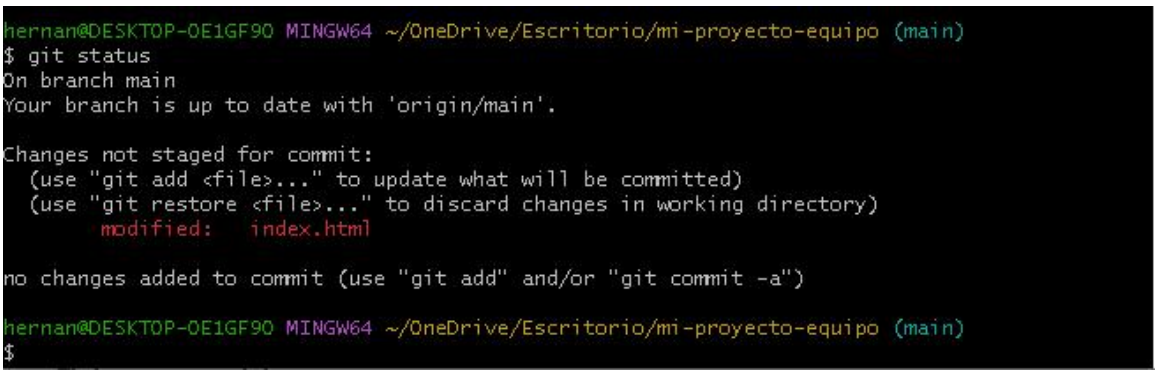


Comandos utilizados:

Git push y Git Branch



Git status

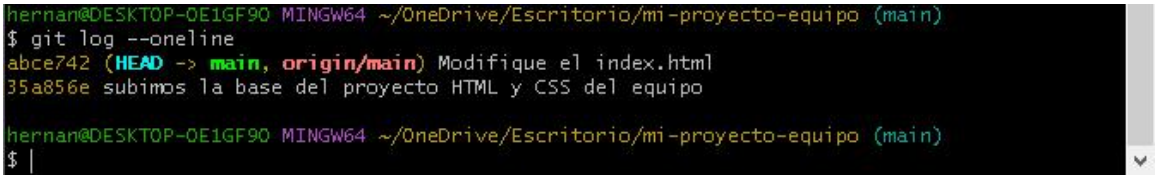


Aqui utilizamos los commandos **git add** index.html lo que nos indica es que esta listo para incluir proximo commit
el siguiente commando utilizado es **git commit -m** “Modifique el index.html”
en la cual nos indica que en ese commit añadieron 85 lineas nuevas y se eliminaron 3

Este commando **git remote** quiere decir que etamos conectados con elrepositorio local con el remote GitHub.

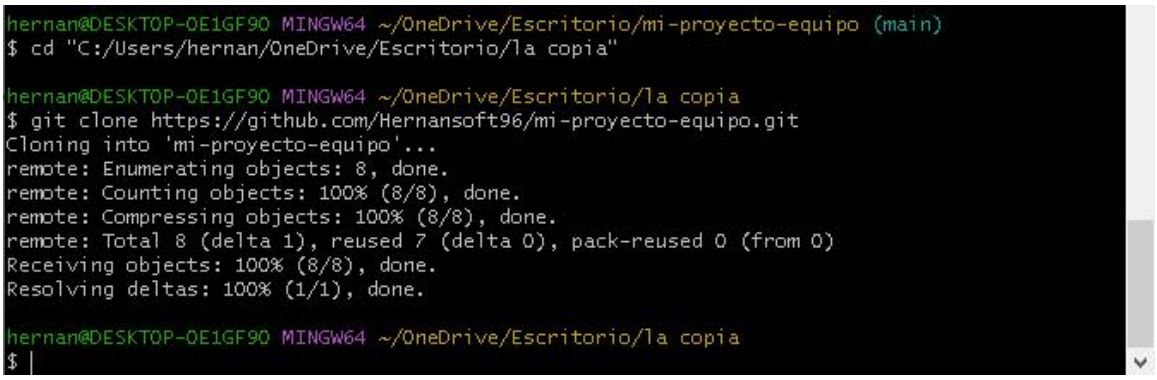
Se utilizó el comando **git push -u origin** main para subir los cambios realizados en el repositorio local al repositorio remoto alojado en GitHub. Esto permitió que el historial de commits y los archivos del proyecto se sincronizaran con la nube.

Este commando **git log –oneline** nos sirve para ver el historial de commits:

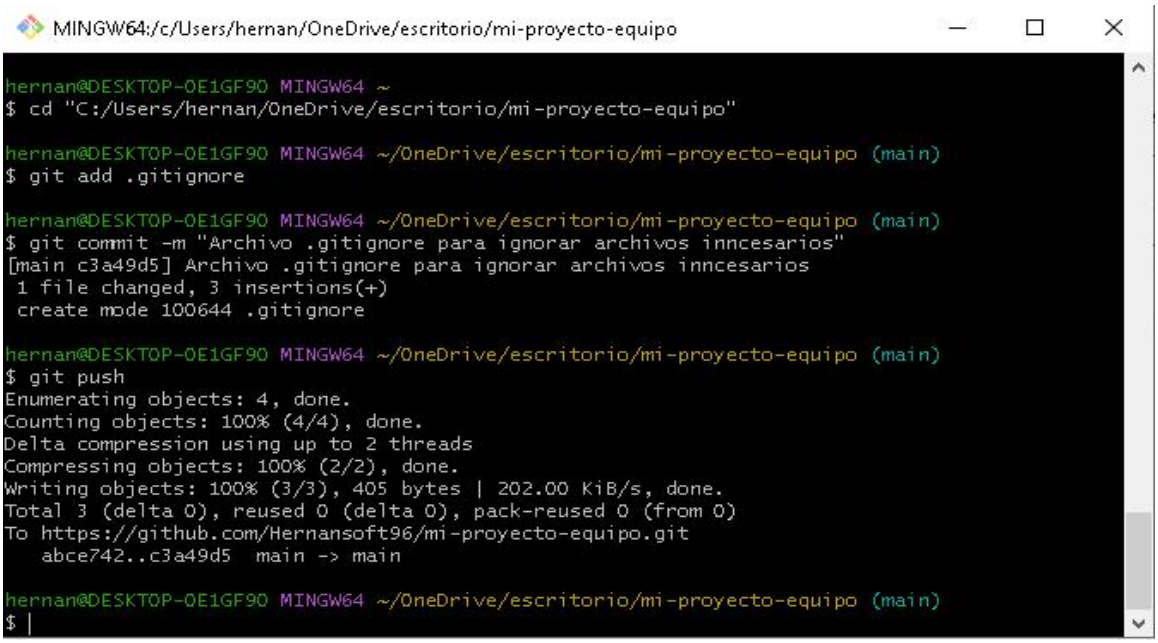


Para organizar el trabajo en el proyecto, se utilizó el comando **git branch** para visualizar y gestionar las ramas. Primero, se confirmó que la rama principal (main) estaba activa. Luego, se creó una nueva rama llamada nueva-rama con el comando **git branch** nueva-rama. Finalmente, se verificó que ambas ramas existían en el repositorio local, lo que permite desarrollar nuevas características sin afectar la rama principal.

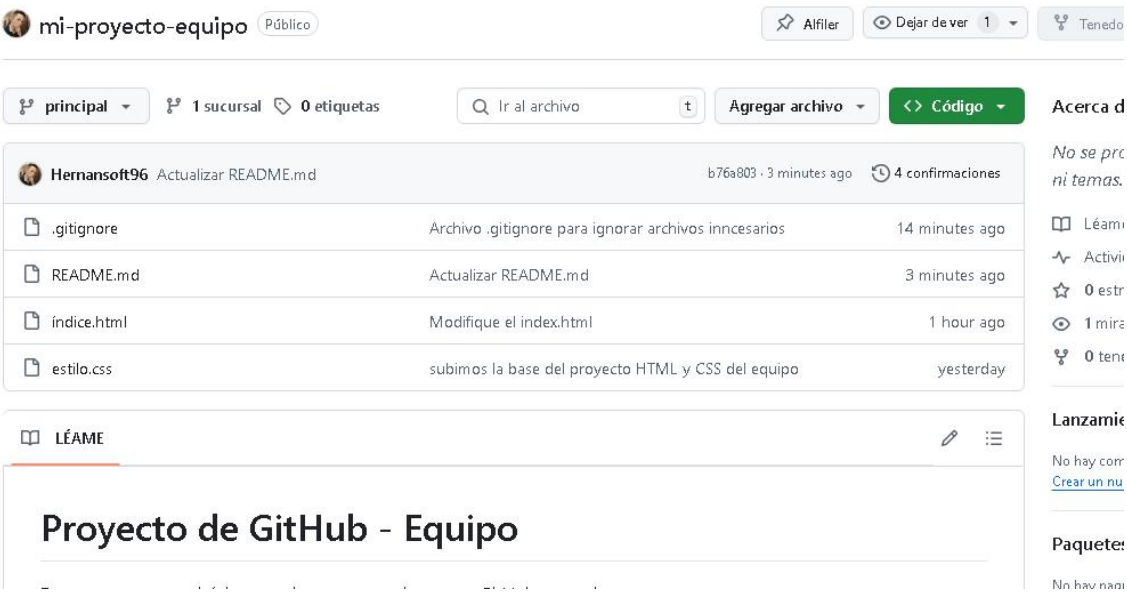
Se utilizó el comando **git clone** para obtener una copia exacta del repositorio remoto desde GitHub hacia el entorno local. Esto permite a cualquier integrante del equipo acceder al proyecto y colaborar.



Se creó un archivo **.gitignore** con el fin de evitar que archivos innecesarios, como archivos de log o carpetas de dependencias, se suban al repositorio. Esto ayuda a mantener limpio el historial del proyecto y evita conflictos.



Todo queda listo y actulizado al entorno remote de GitHub.



Conclusiones

El uso de Git y GitHub facilita el trabajo colaborativo y el control de versiones. Aprendimos a manejar ramas para trabajar en paralelo sin conflictos.- El uso de comandos básicos permitió una correcta gestión del proyecto.Trabajar como equipo nos ayudó a dividir tareas, mejorar la comunicación y aprender herramientas del mundo real del desarrollo de software.

Referencias Bibliograficas

Lardinois, F. (2022, octubre 26). Four years after being acquired by Microsoft, GitHub keeps doing its thing. TechCrunch. https://techcrunch.com/2022/10/26/four-years-after-being-acquired-by-microsoft-github-keeps-doing-its-thing/

Microsoft. (2018, junio 4). Microsoft + GitHub = Empowering Developers. The Official Microsoft Blog. https://blogs.microsoft.com/blog/2018/06/04/microsoft-github-empowering-developers/

Microsoft Learn. (s.f.). Aprendizaje de GitHub. Microsoft Learn. https://learn.microsoft.com/es-es/training/github/

Enlace del repositorio

https://github.com/Hernansoft96/mi-proyecto-equipo