**INFORME SOBRE GIT**

**ALVARO ANDRES AGÜERO BARRERA**

**SV70454892**

**IDAT - 2025**

**Resumen Ejecutivo**  
  
Este informe proporciona una visión detallada sobre Git, un sistema de control de versiones distribuido ampliamente utilizado en entornos de desarrollo de software. Se exploran los conceptos fundamentales, las ventajas del uso de Git, los principales comandos, y los flujos de trabajo comunes como Git Flow y GitHub Flow. Este documento está orientado a estudiantes, profesionales de TI e ingenieros que desean consolidar sus conocimientos sobre esta herramienta esencial para el trabajo colaborativo y el manejo eficiente del código fuente.

**Índice**  
  
1. Introducción 1  
2. ¿Qué es Git? 1  
3. Historia de Git 1  
4. Ventajas de usar Git 2  
5. Instalación de Git 2  
6. Configuración inicial 2  
7. Comandos básicos de Git 3  
8. Flujos de trabajo en Git 3  
 8.1 Git Flow 3  
 8.2 GitHub Flow 3  
9. Buenas prácticas 4  
10. Herramientas de apoyo (Git GUI, GitHub, GitLab) 4  
11. Conclusiones 5  
12. Referencias 5

**1. Introducción**  
  
En el mundo del desarrollo de software moderno, el control de versiones es una herramienta crucial. Git ha ganado una posición destacada debido a su potencia, velocidad y naturaleza distribuida. Fue desarrollado originalmente por Linus Torvalds en 2005 para apoyar el desarrollo del kernel de Linux.  
  
**2. ¿Qué es Git?**  
  
Git es un sistema de control de versiones distribuido. Permite que múltiples desarrolladores trabajen simultáneamente sobre el mismo proyecto sin interferir en el trabajo de los demás. Git guarda un historial completo de los cambios realizados, lo que permite regresar fácilmente a versiones anteriores.  
  
**3. Historia de Git**  
  
Git fue creado como una respuesta a las limitaciones encontradas en otros sistemas de control de versiones, como CVS y Subversion. Fue diseñado con tres objetivos principales: velocidad, integridad de los datos y soporte para flujos de trabajo distribuidos.

**4. Ventajas de usar Git**  
  
- Control total del historial del proyecto.  
- Trabajos paralelos mediante ramas.  
- Rapidez en operaciones locales.  
- Integridad de datos garantizada con SHA-1.  
- Accesibilidad desde múltiples plataformas.  
  
**5. Instalación de Git**  
  
Git puede instalarse en los principales sistemas operativos desde el sitio oficial: [https://git-scm.com](https://git-scm.com). En Linux puede usarse el comando `sudo apt install git`, en Windows puede descargarse el instalador, y en macOS puede instalarse con `brew install git`.  
  
**6. Configuración inicial**  
git config --global user.name "Tu Nombre"  
git config --global user.email "tu.email@ejemplo.com"

**7. Comandos básicos de Git**  
  
- `git init`: Inicializa un repositorio Git.  
- `git clone <url>`: Clona un repositorio remoto.  
- `git add <archivo>`: Añade archivos al área de staging.  
- `git commit -m "mensaje"`: Guarda los cambios en el historial.  
- `git status`: Muestra el estado del repositorio.  
- `git log`: Muestra el historial de commits.  
  
**8. Flujos de trabajo en Git**  
  
**8.1 Git Flow**  
  
Este flujo utiliza ramas separadas para features, releases y hotfixes, ideal para proyectos complejos y colaborativos.  
  
**8.2 GitHub Flow**  
  
Es más ágil y centrado en la integración continua. Cada nueva funcionalidad se desarrolla en una rama aparte y se integra mediante un pull request.  
  
**9. Buenas prácticas**  
  
- Commits claros y frecuentes.  
- Uso adecuado de ramas.  
- Revisiones de código vía pull requests.  
- Documentación del código y del proceso de desarrollo.  
  
**10. Herramientas de apoyo**  
  
- Git GUI: Interfaces gráficas como Sourcetree o GitKraken.  
- GitHub: Plataforma de alojamiento basada en Git.  
- GitLab: Alternativa con opciones CI/CD integradas.

**11. Conclusiones**  
  
Git es una herramienta fundamental para cualquier desarrollador moderno. Su adopción facilita el trabajo en equipo, mejora la trazabilidad de los cambios y optimiza el desarrollo de software.  
  
**12. Referencias**  
  
Chacon, S., & Straub, B. (2014). Pro Git (2nd ed.). Apress. Recuperado de https://git-scm.com/book/en/v2

Atlassian. (n.d.). What is version control? Recuperado 19 de abril de 2025, de https://www.atlassian.com/git/tutorials/what-is-version-control

Atlassian. (2025). Gitflow workflow. Recuperado 19 de abril de 2025, de https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

GitHub Docs. (n.d.). GitHub flow. Recuperado 19 de abril de 2025, de https://docs.github.com/en/get-started/using-github/github-flow

DigitalOcean. (2021). How to use Git: A reference guide. Recuperado de https://www.digitalocean.com/community/cheatsheets/how-to-use-git-a-reference-guide