Introducción

GIT es un sistema de control de versiones distribuido, de código abierto, diseñado para manejar proyectos de software de manera eficiente y confiable. GIT permite a cada desarrollador tener una copia completa del historial del proyecto, lo que facilita la colaboración y la administración de cambios en entornos distribuidos.

Una de las características más destacadas de GIT es su capacidad para realizar un seguimiento detallado de los cambios en el código fuente. Cada modificación realizada en el proyecto se registra como un “commit”, que incluye un mensaje descriptivo y una referencia única. Esta estructura permite a los desarrolladores revertir cambios, comparar versiones anteriores y fusionar diferentes ramas del desarrollo de manera segura y eficiente. Además, GIT utiliza un sistema de ramas ligero que facilita la creación y gestión de ramas para desarrollar nuevas características, corregir errores o experimentar sin afectar la estabilidad del código principal.

GIT también es conocido por su alto rendimiento y escalabilidad. Su arquitectura distribuida reduce la carga en el servidor central y permite a los desarrolladores trabajar de manera independiente, incluso sin conexión a internet. Las operaciones comunes, como los commits y las fusiones, son extremadamente rápidas, lo que mejora la productividad y eficiencia del equipo de desarrollo.

Además de sus capacidades técnicas, GIT se integra fácilmente con una amplia variedad de herramientas y servicios de alojamiento de código, como GitHub, GitLab y Bitbucket. Estas plataformas proporcionan funcionalidades adicionales, como la gestión de repositorios remotos, revisión de código, integración continua y despliegue automatizado, lo que convierte a GIT en una herramienta esencial para el desarrollo de software moderno.

En resumen, GIT es una herramienta fundamental para cualquier desarrollador de software. Su capacidad para gestionar el control de versiones de manera distribuida, su eficiencia y su integración con otras herramientas de desarrollo, hacen de GIT una solución robusta y flexible para proyectos de cualquier tamaño y complejidad. Este curso está diseñado para proporcionar a los participantes una comprensión profunda de Git, desde los conceptos básicos hasta las técnicas avanzadas, permitiéndoles gestionar sus proyectos de manera más eficaz y colaborativa.

Objetivos del Taller

Este Taller ofrece una comprensión integral de GIT, enseñando desde los fundamentos hasta técnicas avanzadas. Los participantes aprenderán a gestionar versiones, colaborar eficazmente y optimizar su flujo de trabajo. Se abordarán conceptos como repositorios, commits, ramas, fusiones, resolución de conflictos y mejores prácticas. Además, se explorarán estrategias avanzadas para proyectos complejos, incluyendo reescritura de historial, submódulos y hooks personalizados. Al finalizar, los participantes podrán utilizar GIT con confianza para mejorar la productividad y la calidad del desarrollo de software.

Contenidos

Fundamentos de GIT

- Introducción a los fundamentos de Git
- Primeros comandos de git
- Nuestro primer repositorio
- Cambiar nombre de la rama Master a Main
- Demostración de la creación, puesta en escena y commits
- VSCode add y commits
- Diferentes formas de agregar archivos al escenario
- Diferentes formas de añadir al Stage

Elementos adicionales a los fundamentos de GIT

- Cambios en los archivos
- Actualizar mensaje del commit y revertir commits
- Posible error/warning que tienen algunos
- Preparando un repositorio para viajes en el tiempo
- Cambiar el nombre y eliminar archivos mediante git
- Cambiar el nombre y eliminar archivos fuera de git
- Ignorando archivos que no deseamos

Ramas, uniones, conflictos y tags

- Introducción a la sección de ramas
- Introducción Ramas, uniones y conflictos

Merge y Tags

- Merge: Fast-Forward
- Merge: Union automática
- Merge: Uniones con conflictos
- Tags - Etiquetas
- Creando etiquetas - Tags

GIT Stash y GIT Rebase

- Introducción a la sección - Stash
- Introducción al stash
- Git Stash
- Conflictos con el stash
- Stash avanzado
- Introducción al git rebase
- Rebase - Actualizando una rama
- Rebase - Squash
- Rebase - Reword
- Rebase - edit

Inteligencia Artificial

- Acompañamiento con IA para la resolución específica de temas del Taller

Metodología

- Taller práctico (80% Práctico 20% Teórico).
- Este Taller permitirá a los participantes aplicar GIT en escenarios reales, reforzando la teoría con ejercicios prácticos.

Requisitos

No hay requisitos para este Taller

Dirigido a:

Desarrolladores de Aplicaciones

Generalidades

- Se aplicará un cuestionario previo al inicio del taller a cada participante para evaluar su nivel de conocimiento. Esto permitirá identificar la necesidad de nivelación y/o realizar ajustes en el temario si es necesario.
- Duración 20 horas cronológicas.
- Taller cerrado.