SERVICIO NACIONAL DE APRENDIZAJE – SENA

CENTRO DE COMERCIO REGIONAL ANTIOQUIA

TECNOLOGÍA EN ANÁLISIS Y DESARROLLO DE SOFTWARE - 2675805

Evidencia de conocimiento: GA7-220501096-AA1-EV01 informe técnico de plan de trabajo para construcción de software´

DANIEL FELIPE ARIAS CORREDOR

2023

Introducción

Este documento tiene como objetivo presentar una noción general acerca de los repositorios y sistemas de control de versiones para un desarrollo de software. Adicional se explicará uno de estos sistemas de control de versiones y sus comandos principales.

¿Qué es un repositorio?

Un repositorio es un sistema de almacenamiento digital utilizado para guardar código fuente de una aplicación, así como ir guardando versiones de un software en construcción. Permite hacer cambios en la versión del código sin comprometer el código inicial, ofreciendo la oportunidad de retroceder a versiones anteriores y desarrollar código de manera colaborativa y simultánea.

¿Qué es un sistema de control de versiones?

Los repositorios permiten el control de versiones, que viene a ser ala practica de gestionar los cambios en el código fuente de un proyecto, así como rastrear estos cambios y deshacer avances si los resultados no son los deseados. Los sistemas de control de versiones permiten a los equipos de desarrollo implementar actualizaciones exitosas y un mayor control del trabajo colaborativo, al ofrecer un historial competo de los cambios de todos los módulos, la creación de ramificaciones y su posterior fusión y por ultimo la trazabilidad, permitiendo conectar el sistema de control de versiones a software de seguimiento como Jira.

¿Qué tipos de sistema de control de versiones se utilizan?

Los sistemas de control de versiones se dividen en tres tipos: locales, centralizados y distribuidos.

Los sistemas de control de versiones locales almacenan las versiones en una base de datos, por lo que para acceder a un aversión anterior había que buscar en esa única base de datos (en disco duro), por lo que el control de versiones solo puede ser realizado por un usuario.

Los sistemas de control de versiones centralizados permiten el almacenamiento de versiones en un servidor externo, permitiendo a dos o mas desarrolladores editar un mismo archivo, al momento de terminar debía haber un consenso para aprobar los cambios efectuados. Ejemplos de sistemas centralizados son SVN y CVS.

Los sistemas de control de versiones distribuidos, por su parte permiten ofrecerle a cada desarrollador su propia copia de todo el proyecto, sin existir un repositorio central, pro lo que cada parte puede trabajar a su propio ritmo y entregar resultados funcionales para su posterior fusión. Ejemplos de sistemas distribuidos son Git y Mercurial.

¿Qué es Git?

Git es un software de control de versiones. Ha sido escogido para el desarrollo del proyecto de aplicación web poque facilita el desarrollo y mantenimiento de aplicaciones de manera individual o en equipo. Guarda un registro con todos los archivos y las versiones que se han venido creando. Permite la creación de ramificaciones o copias de trabajo independientes y sus posterior fusión. Lo anterior junto con su uso extendido permitirán su aprendizaje y aportes en el desarrollo del proyecto.

Git es un software descargable, permite introducir comandos en su propia consola o por medio de una extensión en un entorno de desarrollo como Visual Studio Code o Eclipse.

Github por su parte es un servicio de alojamiento de repositorios en la nube. los usuarios de GitHub pueden rastrear y gestionar los cambios que se realizan en el código fuente en tiempo real, a la vez que tienen acceso a todas las demás funciones de Git disponibles.

Comandos básicos de Git

Para configurar un usuario y correo:

>git config –global user.name *“nombreusuario*”

>git config –global user.email “*usuario@correo.com*”

Para creación de un repositorio

>git init = Comando que se ejecuta cuando se quiere hacer seguimiento a proyecto

>git add = asigna los archivos a un staging area o área de puesta en espera.

>git commit = traslada los archivos del staging área al repositorio.

>git reset –hard *códigoversion* = Comando que se usa cuando se quiere reiniciar el avance de un proyecto hasta una versión anterior determinada.

Para subir de almacenamiento local a github:

>git commit –am = Permite hacer simultaneamente al add y el commit.

>pull origin main= permite almacenar en local la información proveniente de Github (en este caso la info viene de la rama principal o main).

Para crear o gestionar ramas:

>git branch *nombrerama* = Crea una rama con un nombre.

>git checkout *nombrerama* = Permite moverse de una rama a otra para trabajar

>git merge *nombrerama a fusionar* = Permite fusionar dos ramas (se debe estar ubicado en la rama de destino)

>git branch -d *nombrerama* = Permite eliminar una branch

Referencias:

¿Qué es Repo? Recuperado de: https://aws.amazon.com/es/what-is/repo/#:~:text=Un%20repositorio%2C%20o%20repo%2C%20es,de%20documentos%20al%20desarrollar%20software.

¿Qué es el control de versiones? Recuperado de: https://www.atlassian.com/es/git/tutorials/what-is-version-control

Sistemas de Control de Versiones, qué son y por qué amarlos. Recuperado de: https://medium.com/@jointdeveloper/sistemas-de-control-de-versiones-qué-son-y-por-qué-amarlos-24b6957e716e#:~:text=Los%20sistemas%20de%20control%20de%20versiones%20han%20ido%20evolucionando%20a,Versiones%20Locales%2C%20Centralizados%20y%20Distribuidos.

Qué es GitHub y cómo empezar a usarlo Recuperado de: https://www.hostinger.es/tutoriales/que-es-github

Git & GitHub. Recuperado de: https://www.pildorasinformaticas.es/course/curso-de-git-github/