**UNIVERSIDAD DON BOSCO**

**FACULTAD DE INGENIERÍA**

**ESCUELA DE COMPUTACIÓN**

Diagrama

Descripción generada automáticamente con confianza baja

**Diseño y Programación de Software Multiplataforma**

**Ing. Karens Medrano**

**Grupo 01T**

**Tema:**

Taller Clase 1

|  |  |
| --- | --- |
| **Alumno** | **Código** |
| Christian Giovanni Tobar Cerón | TC192221 |

27 de julio de 2021

Realizar un documento en Word que contenga un consolidad con la siguiente información

1. **Tipos de cuenta de Github**

* Free
* Team
* Empresa

1. **Diferencias entre cuentas**

Cuentas de usuarios personales

Cada persona que usa GitHub tiene su propia cuenta de usuario

Cuentas de organización

Las organizaciones son cuentas compartidas donde grupos de personas pueden colaborar en muchos proyectos a la vez. Los propietarios y administradores pueden gestionar el acceso de los miembros a los datos y proyectos de la organización con sofisticadas funciones administrativas y de seguridad.

Cuentas empresariales

Con las cuentas empresariales, puede administrar de forma centralizada las políticas y la facturación de varias organizaciones de GitHub.com. Las cuentas empresariales están disponibles con GitHub Enterprise Cloud y GitHub Enterprise Server.

1. **Finalidad de cada una de las cuentas**

GitHub Free

* Soporte de la Comunidad de GitHub
* Alertas del Dependabot de GitHub
* Aplicación de la autenticación de dos factores
* 2,000 minutos de GitHub Actions
* 500MB de almacenamiento de Paquetes de GitHub

GitHub Team

Adicionalmente a las características disponibles con GitHub Free para organizaciones, GitHub Team incluye:

* Soporte de GitHub por correo electrónico
* 3,000 minutos de GitHub Actions
* 2GB de almacenamiento de Paquetes de GitHub
* Herramientas y perspectivas avanzadas en repositorios privados:
* Revisores requeridos para solicitudes de extracción
* Revisores múltiples para solicitudes de extracción
* Páginas de GitHub
* Wikis
* Ramas protegidas
* Propietarios del código
* Gráficos de información del repositorio: pulso, contribuyentes, tráfico, confirmaciones, frecuencia de código, red y bifurcaciones
* Solicitudes de extracción en borrador
* Revisores de equipo para solicitudes de extracción
* Recordatorios programados

GitHub Enterprise

Adicionalmente a las características disponibles con GitHub Team, GitHub Enterprise incluye:

* Soporte para GitHub Enterprise
* Controles de seguridad, cumplimiento e implementación adicionales
* Autenticación con inicio de sesión único SAML
* Provisión de acceso con SAML o SCIM
* GitHub Connect

Una captura de pantalla de una computadora

Descripción generada automáticamente

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

**Conceptos sobre comandos GIT**

* ***Configuración***
* **git:** Es el comando a través del cual se lanzan todas las operaciones de Git.
* **git config:** Lo usas para establecer una configuración específica de usuario, podría ser el email, usuario o tipo de formato.
* **git help:** Muestra una lista con los comandos más utilizados en GIT.
* ***Obtener y crear proyecto***
* **git init:** Creará un nuevo repositorio local GIT. Usando git init [nombre del proyecto] También puedes crear un repositorio dentro de un directorio especificando el nombre del proyecto.
* **git clone:** Lo usas para clonar un repositorio.
* ***Seguimiento básico***
* **git status:** Lo usas para que muestre la lista de archivos que has cambiado, junto con archivos que serán preparados y confirmados.
* **git add:** Lo usas para agregar archivos al área de de preparación.
* **git rm:** se puede usar para eliminar archivos sueltos o un conjunto de ellos.
* **git commit:** Lo usas para crear un cambio que se guardará en el directorio git.
* ***Crear y fusionar ramas***
* **git branch:** Nos muestra una lista de los branches que existen en nuestro repositorio.
* **git checkout:** Nos permite crear ramas y navegar entre ellas.
* **git merge:** Hace un merge entre dos ramas, en este caso la dirección del merge sería entre el branch que indiquemos en el comando, y el branch donde estémos ubicados.
* **git stash:** Almacena temporalmente (o guarda en un stash) los cambios que hayas efectuado en el código en el que estás trabajando para que puedas trabajar en otra cosa y, más tarde, regresar y volver a aplicar los cambios más tarde.
* **git tag:** Este comando enumera las etiquetas en orden alfabético; el orden en el que se muestran no tiene importancia real.
* ***Compartir y actualizar proyectos***
* **git fetch:** Es el comando que le dice a tu git local que recupere la última información de los metadatos del original (aunque no hace ninguna transferencia de archivos. Es más bien como comprobar si hay algún cambio disponible).
* **git pull:** Lo usas para fusionar todos los cambios que hiciste en el repositorio local con el directorio de trabajo local.
* **git push:** Lo usas para enviar confirmaciones a la rama maestra/principal del repositorio remoto.
* ***Inspección y comparación de repositorios***
* **git show:** Este comando se utiliza para mostrar el detalle de algunos objetos, como etiquetas (tags) o commits.
* **git log:** El comando que podemos usar para ver el histórico de commits, estando situados en la carpeta de nuestro proyecto
* **Git diff:** Es un comando que se usa para generar los cambios entre dos fuentes dentro del repositorio de git.

Crea una cuenta de GitHub y coloca una captura

Captura de pantalla de computadora

Descripción generada automáticamente

**Fuentes de consulta**

<https://github.com/pricing>

<https://docs.github.com/es/github/getting-started-with-github/learning-about-github/githubs-products>

<https://docs.github.com/en/get-started/learning-about-github/types-of-github-accounts#personal-user-accounts>