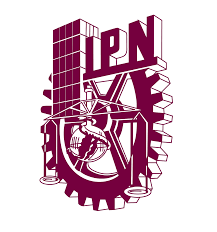
**-INSTITUTO POLITECNICO NACIONAL**

**ESCUELA SUPERIOR DE COMERCIO Y ADMINISTRACIÓN UNIDAD**

**SANTO TOMÁS**

**CARRERA:**

NEGOCIOS DIGITALES

**MATERIA:**

LABORATORIO DE NEGOCIOS DIGITALES

**PRÁCTICA 1:**

PRÁCTICA DE SOFTWARE DE CONTROL DE VERSIONES

**PROFESOR:**

JOVAN DEL PRADO LÓPEZ

**ALUMNA:**

ROMERO RUIZ MELANIE DANIELA

**GRUPO:**

2GM1

**INTRODUCCION DEL TEMA**

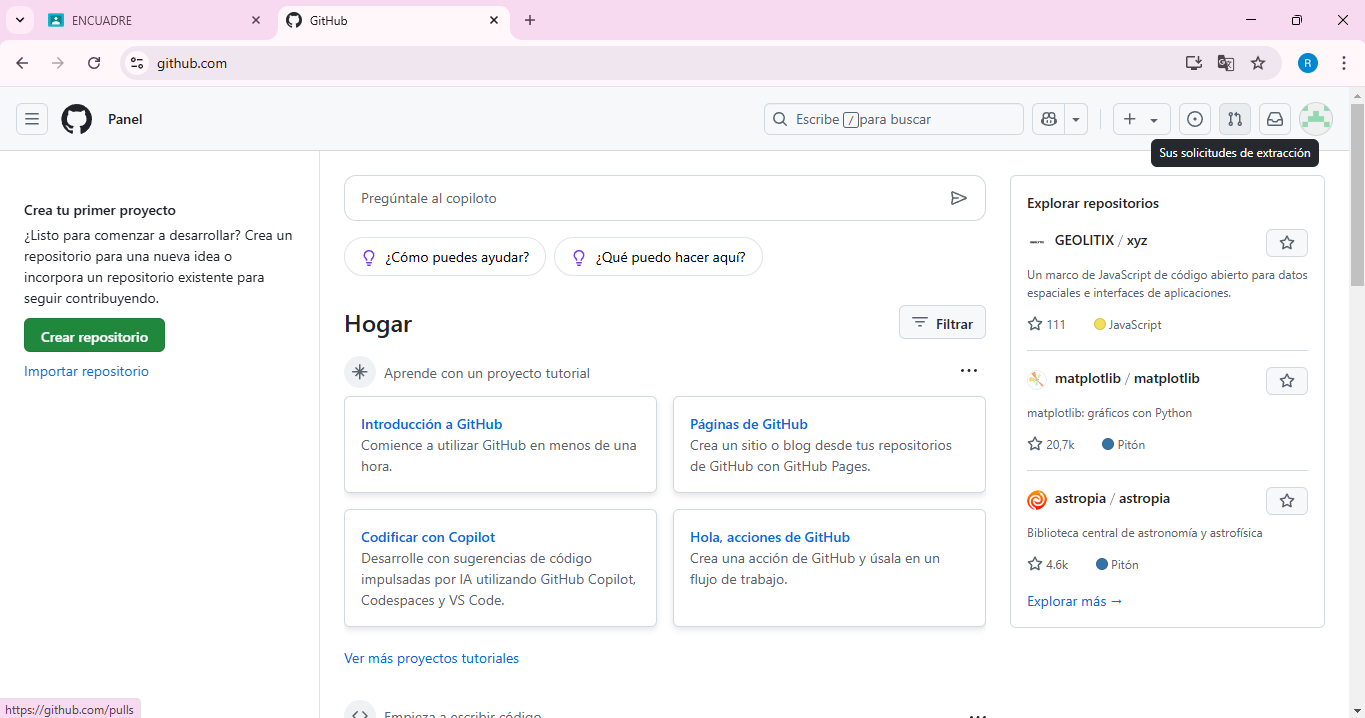
El software de control de versiones es una herramienta muy útil a la hora de editar un archivo entre varias personas y es eficiente cuando es necesario regresar a una versión anterior del documento, añadir algo, eliminar o modificar alguna acción, además el repositorio con el que cuenta nos ayuda a tener un registro de todas las versiones por las que pasa un archivo.

Otra de sus características que lo caracterizan es poder regresar a su versión original y la copia de seguridad externa que respalda el trabajo.

En caso de necesitar este tipo de software hay 5 opciones diferentes que se adaptan a las necesidades de cada usuario y que se describen a continuación.

**DESARRO DE CUENTA DE GITHOB (FOTOS)**

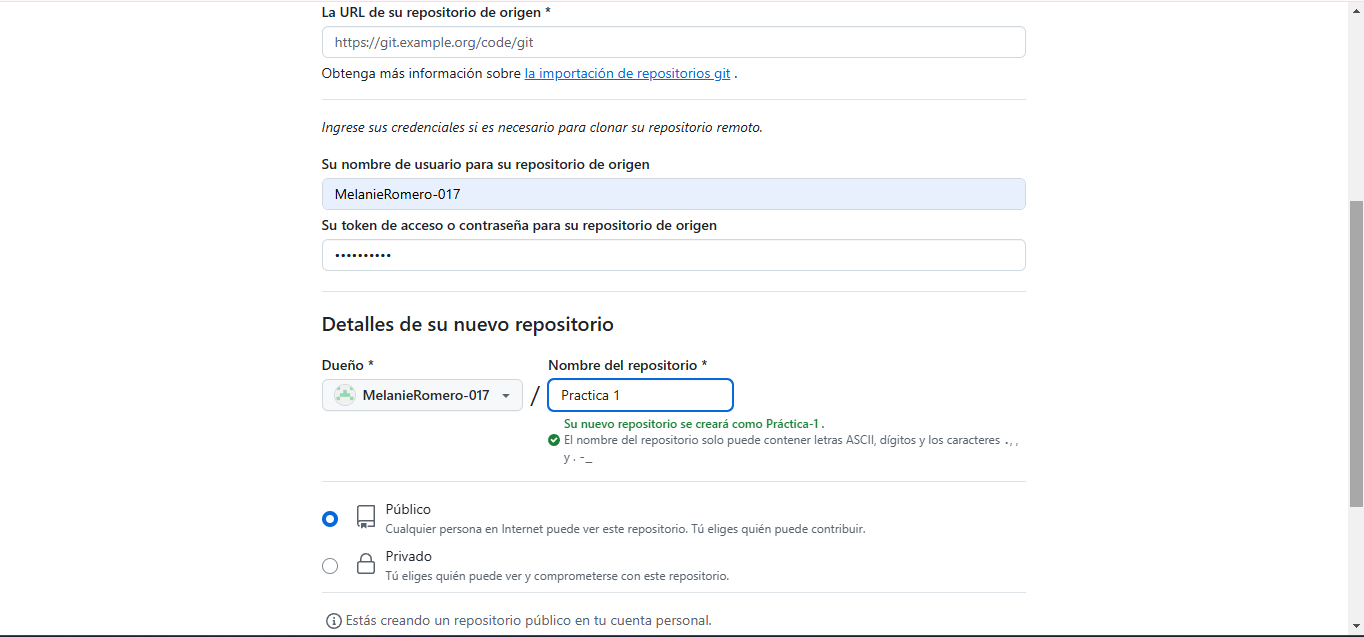
Una vez iniciada sesión esta es la pagina principal de la plataforma



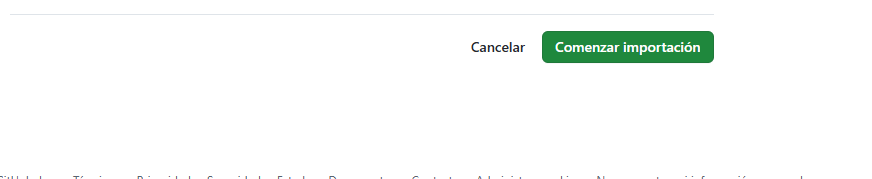
Nos vamos a donde dice importar repositorio y nos aparece esta sección donde tenemos que colocar datos específicos antes de subir el archivo



Después ingresamos los datos



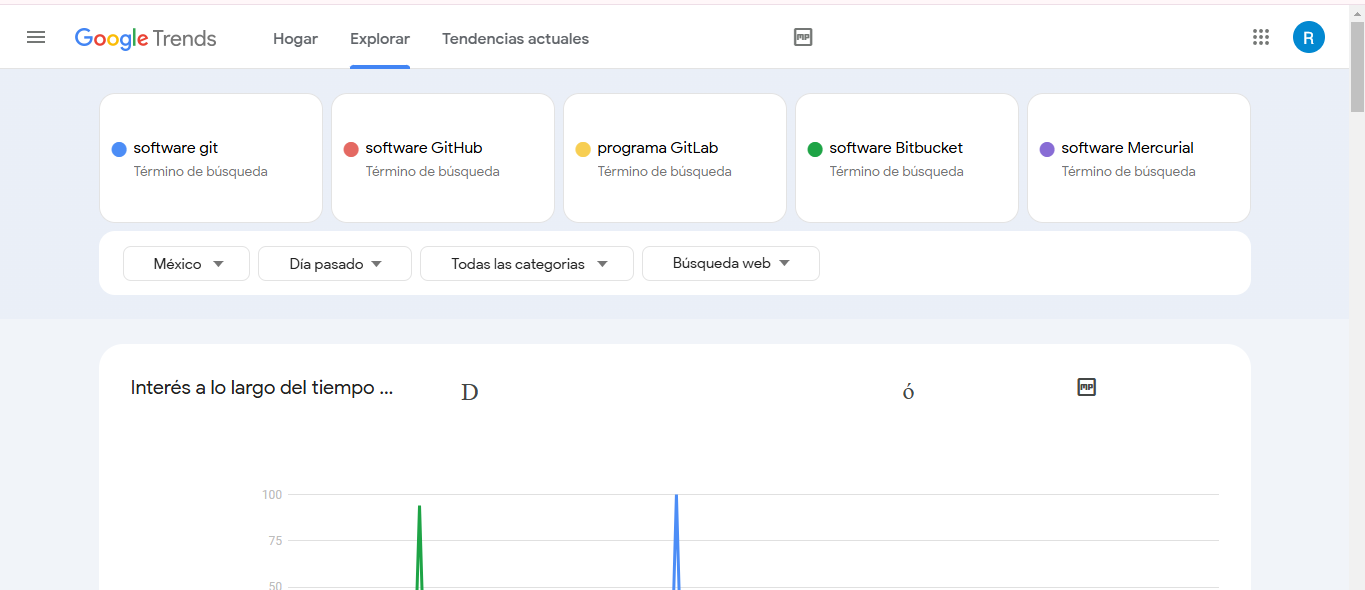
Y finalmente cargamos el archivo



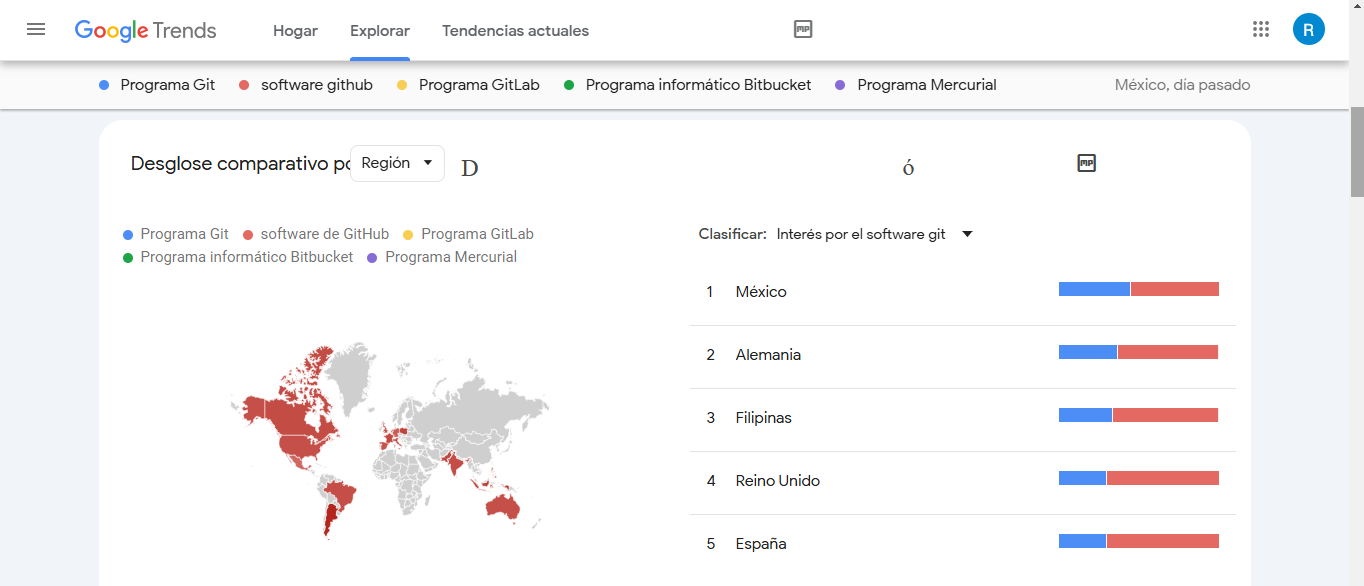
**TABLA**

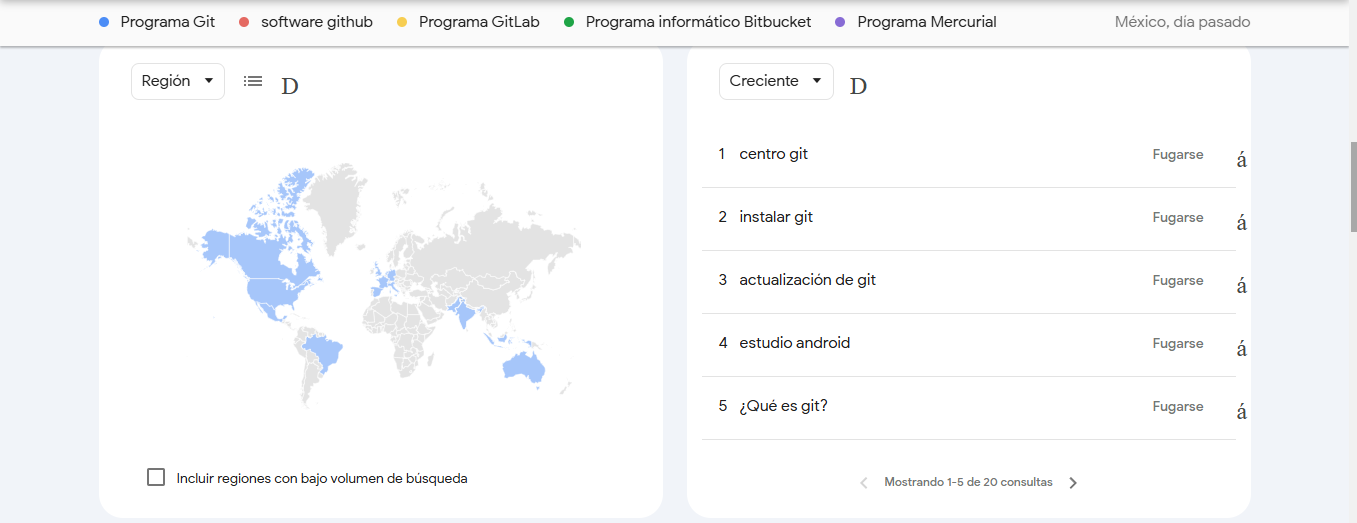
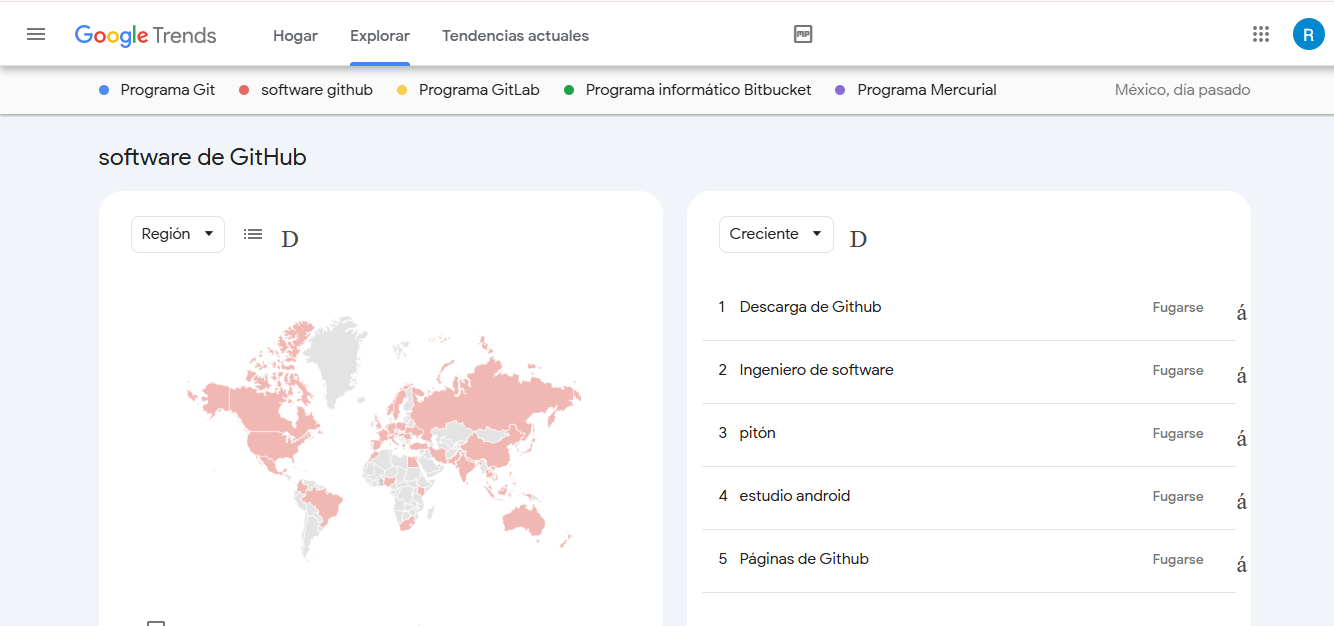
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SOFTWARE | VENTAJAS | DESVENTAJAS | $ | CÓDIGO ABIERTO |
| Git | Distribuido y Descentralizado  A diferencia de los sistemas de control de versiones centralizados, en los que se almacena una única copia del repositorio, Git es distribuido y como desarrolladores tenemos una copia completa del historial de cambios. Esto nos permite trabajar de forma independiente y fusionar nuestros cambios en el repositorio principal de manera eficiente. Rastreo Preciso de Cambios Git realiza un seguimiento preciso de los cambios realizados en cada archivo a lo largo del tiempo. Esto nos permite como desarrollador ver el historial completo de cambios, incluidas las diferencias entre versiones y la posibilidad de regresar a versiones anteriores en caso de problemas. Ramificación Eficiente Git hace que la creación y gestión de ramas branches sea sencillo. Como desarrollador podemos crear ramas para trabajar con nuevas características o arreglos sin afectar la rama principal. Esto fomenta el desarrollo paralelo y facilita la colaboración en equipos grandes. Fusiones Simplificadas La capacidad de fusionar cambios entre diferentes ramas es una característica clave de Git. Como desarrollador podemos combinar cambios de manera eficiente, lo que facilita la integración continua y evita conflictos de código. | * **Es más complejo** que los sistemas centralizados tradicionales porque entran en juego más repositorios, más operaciones y más posibilidades para trabajar en equipo, que hay que decidir. * **La curva de aprendizaje es empinada**. Lo básico lo aprendes enseguida, pero la realidad te demuestra que no es suficiente "tocar de oído" con él. La documentación es tan compleja que muchas veces no resulta de ayuda. * **Los comandos y algunos conceptos** que usa pueden llegar a ser **confusos**, al igual que algunos mensajes que muestra. * Por defecto, **se lleva mal con archivos binarios muy grandes**, como vídeos o documentos gráficos muy pesados. Por suerte existen soluciones para ello (Git LFS). | Se puede utilizar sin la necesidad de tener que pagar una cuota | Palomita | |
| GitHub | * Servicio gratuito, aunque también tiene servicios de pago. * Búsqueda muy rápida en la estructura de los repos. * Amplia comunidad y fácil encontrar ayuda. * Ofrece prácticas herramientas de cooperación y buena integración con Git. * Fácil integrar con otros servicios de terceros. * Trabaja también con TFS, HG y SVN. | * No es absolutamente abierto. * Tiene limitaciones de espacio, ya que no puedes exceder de 100MB en un solo archivo, mientras que los repositorios están limitados a 1GB en la versión gratis. | GitHub ofrece planes gratuitos y de pago para cada desarrollador o equipo. | Palomita |  A pesar de su compromiso con el software de código abierto, GitHub en sí no es de código abierto. |
| GitLab | * Plan gratuito y sin limitaciones, aunque tiene planes de pago. * Es de licencia de código abierto. * Permite el auto hospedaje en cualquier plan. * Está muy bien integrado con Git. | * Su interfaz puede ser algo más lenta con respecto a la competencia. * Existen algunos problemas habituales con los repositorios. | Su función gratuita incluye 400 minutos de cálculo por mes. 5 usuarios por grupo principal.  Por otro lado, cuenta con un plan premium y uno en el que en caso de ser necesario te puedes comunicar con su equipo de ventas para cualquier gestión que la empresa necesite. | Palomita | |
| Bitbucket | * **Facilidad de uso:** Bitbucket es una plataforma relativamente sencilla de utilizar, lo que la hace accesible a desarrolladores de todos los niveles de experiencia. * **Precio:** Bitbucket ofrece una serie de planes de precios flexibles que se adaptan a las necesidades de cualquier equipo. * **Integración con otros servicios:** Bitbucket se integra con una amplia gama de servicios de terceros, lo que facilita a los desarrolladores la gestión de su flujo de trabajo. * **Integración con Jira y Trello** que facilita la gestión de proyectos, la posibilidad de **crear repositorios privados ilimitados** o el contar con **Bitbucket Pipelines**, que simplifica la automatización de flujos de trabajo de desarrollo. | * Interfaz confusa * No es tan popular como otros softwares como GitHub, por mencionar alguno. * Limitaciones del plan gratuito a proyectos grandes. * Las capacidades de CI/CD dependen en gran medida de herramientas externas como Bamboo * Menos énfasis en el escaneo de seguridad integrado en comparación con GitLab * Limitado por las funcionalidades de las herramientas integradas   ​ | Es gratuito para personas y equipos pequeños con hasta 5 usuarios por espacio de trabajo, con repositorios públicos y privados ilimitados.  Por otro lado, hay una versión premium y cuenta con distintos planes que se ajustan a las necesidades del usuario. | Palomita |  Puede funcionar en códigos abiertos y cerrados. |
| Mercurial | * Escalable y adaptable al tamaño y exigencias del proyecto. * Escrito en Python por ende es más fácil su manejo. * Funciona bien sobre páginas y directorios web | * Pocas caracterizas añadidas por default * Comunidad de desarrollo muy pequeña | Sistema de control de versiones distribuidas gratuitas. | Palomita | |

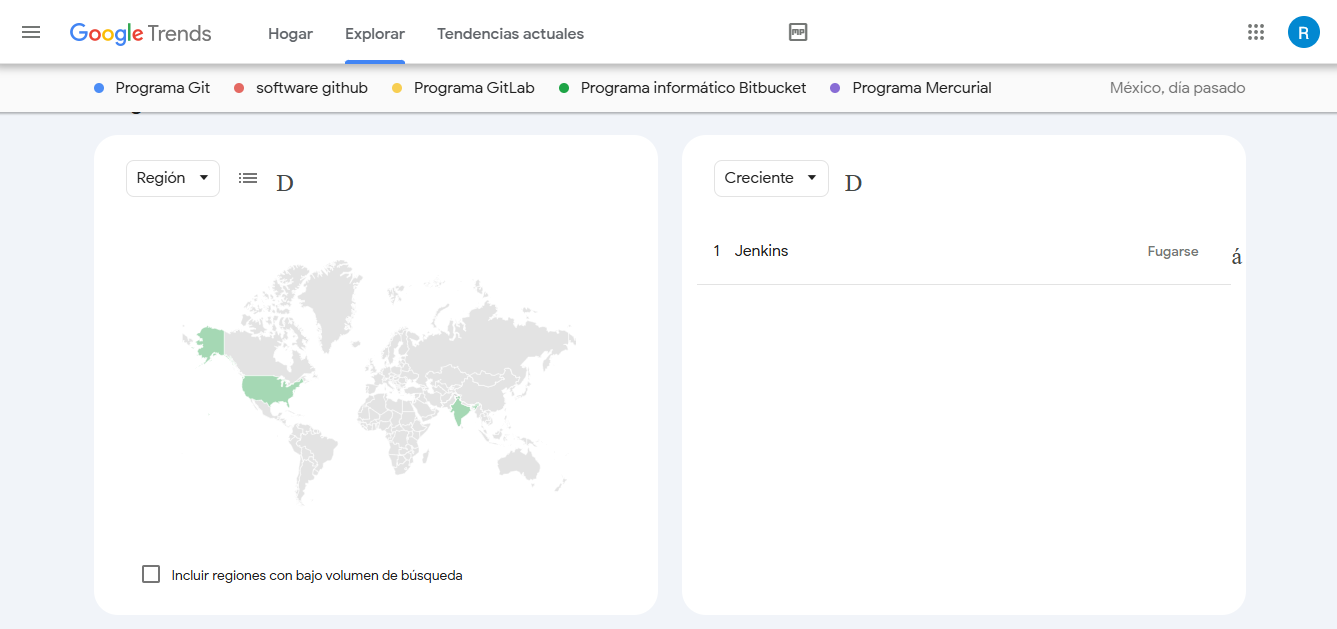
**GOOGLE TRENDS A NIVEL MUNDIAL**

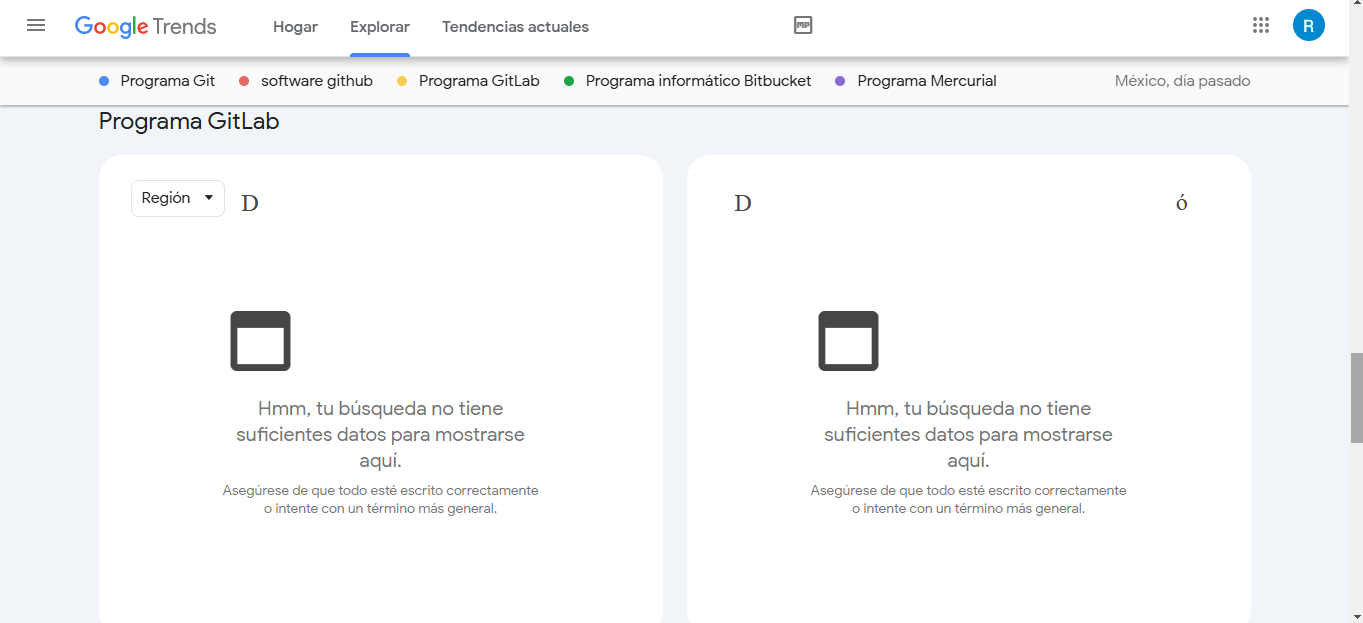
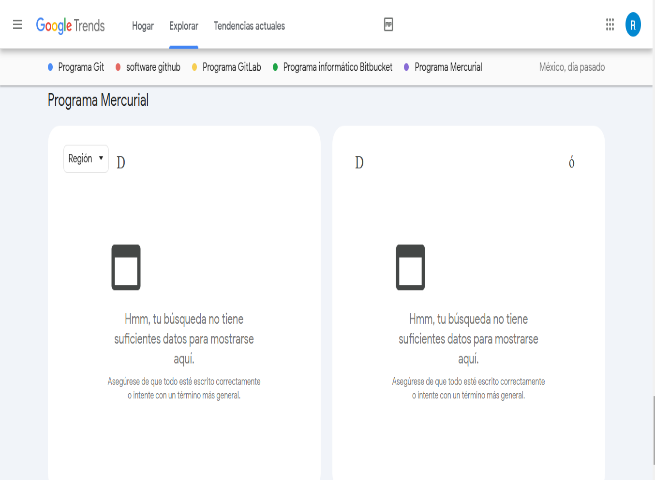










Con ayuda de Google trends se analizaron los 5 distintos softwares de versión a nivel mundial, podemos observar que los que predominan son GitHub y Git en América Latina algunos países de Europa, Asia y Oceanía.

Ahora bien, el software Bitbucket no tiene tanta presencia a nivel mundial, pero si sobresale en países como EE.UU. y algunos otros de América del norte.

Por otra parte, hubo dos tipos de software de los cuales no nos lanzó algún resultado, los cuales fueron GitLab y Mercurial, por lo tanto, no tenemos una comparativa tangible entre los 5 software.

**CONCLUSIONES**

Fue interesante aprender la funcionalidad de los softwares de control de versiones y porque son tan importantes para los usuarios a la hora de editar algún archivo, conocer las distintas opciones disponibles si se necesita, las características de cada uno, ingresar a estos softwares y explorarlos para tener un conocimiento base de como se utilizan me parece muy importante.

**BIBLIOGRAFIA**

*Blog: Descubriendo Git: Características y Ventajas*. (2023). Kranio.io. <https://www.kranio.io/blog/descubriendo-git-caracteristicas-y-ventajas>

Alarcón, J. M. (2020, June 1). *Qué es Git, ventajas e inconvenientes y por qué deberías aprenderlo (bien)*. CampusMVP.es. <https://www.campusmvp.es/recursos/post/que-es-git-ventajas-e-inconvenientes-y-por-que-deberias-aprenderlo-bien.aspx>

Isaac. (2021, March 2). *GitHub vs GitLab: ventajas y desventajas de estas plataformas*. Desde Linux. <https://blog.desdelinux.net/github-vs-gitlab/>

leninmhs. (2023, November 9). *Bitbucket: qué es, ventajas, quiénes lo usan, diferencia con GitHub y Gitlab | Leninmhs*. Leninmhs. <https://leninmhs.com/que-es-bitbucket/>

Paul, F. (2024, June 14). *Bitbucket: La plataforma de control de versiones explicada*. GoDaddy Resources - LATAM. <https://www.godaddy.com/resources/latam/desarrollo/bitbucket-plataforma-de-control-versiones-explicada>

*Bitbucket vs GitLab: ¿Cuál herramienta de gestión de proyectos es la mejor para ti?* (2025). Getguru.com. <https://www.getguru.com/es/reference/bitbucket-vs-gitlab>

*Mercurial (sistema de control de versiones) - EcuRed*. (2015). Ecured.cu. <https://www.ecured.cu/Mercurial_(sistema_de_control_de_versiones)#Ventajas>

Atlassian. (2025). *Precios de Bitbucket: encuentra el plan adecuado para ti | Atlassian*. Atlassian.com. <https://www.atlassian.com/es/software/bitbucket/pricing>

*Acerca de la facturación de planes - Documentación de GitHub*. (2025). GitHub Docs. <https://docs.github.com/es/billing/managing-the-plan-for-your-github-account/about-billing-for-plans>

*GitLab*. (2021). Gitlab.com. <http://about.gitlab.com/pricing/>

‌