**DIFERENCIA ENTRE GIF LOCAL Y GIF REMOTO**

**APRENDIZ:**

**SNEYDER CAMILO ORDOÑEZ USCATEGUI**

**FICHA: 2824182**

**INSTRUCTOR:**

**FABIAN DAVID BARRETO SANCHEZ**

**CENTRO REGIONAL DE CASANARE SENA**

**YOPAL – CASANARE**

**16/10/2024**

**INTRODUCCION**

Git es un sistema de control de versiones distribuido que permite rastrear los cambios en archivos y facilitar la colaboración entre varios desarrolladores en proyectos de software. Funciona como una "máquina del tiempo" para tu código, permitiéndote regresar a versiones anteriores, comparar modificaciones y trabajar de forma simultánea con otros en un entorno eficiente. Gracias a su estructura distribuida, cada colaborador tiene una copia completa del repositorio, lo que mejora la resiliencia y la flexibilidad en el desarrollo.

**Importancia de Git en el Desarrollo de Software**

**Control de versiones:** Git permite rastrear cada modificación en el código, facilitando la identificación de errores, la reversión de cambios indeseados y la comprensión del progreso del proyecto a lo largo del tiempo.

**Colaboración:** Facilita el trabajo en equipo, permitiendo que varios desarrolladores trabajen simultáneamente en el mismo proyecto sin conflictos. Esto mejora la productividad y la sincronización entre los miembros del equipo.

**Respaldo:** Actúa como un sistema de copias de seguridad, protegiendo el código ante pérdidas debidas a fallos o errores, lo que brinda seguridad a los desarrolladores.

**Historial:** Proporciona un registro detallado de todos los cambios realizados, invaluable para auditorías y análisis de código, permitiendo comprender las decisiones pasadas y su impacto en el proyecto.

**Ramificación:** Permite crear ramas independientes para experimentar con nuevas características o realizar pruebas sin afectar la rama principal del proyecto, fomentando así la innovación y el desarrollo seguro.

**Git Local vs. Git Remoto**

**Git Local:** Es una copia completa del repositorio almacenada en tu computadora. Este espacio de trabajo personal te permite realizar cambios, crear nuevas ramas y experimentar sin afectar a otros desarrolladores. La ventaja de Git local es que puedes trabajar de manera independiente y sin conexión, lo que te da flexibilidad en tu flujo de trabajo.

**Git Remoto:** Es la versión del repositorio que se encuentra en un servidor remoto, como GitHub, GitLab o Bitbucket. Este es el punto central donde se comparte el código con el equipo, facilitando la colaboración y la sincronización entre los miembros del proyecto. Git remoto asegura que todos los desarrolladores tengan acceso a la versión más actualizada del código y permite la integración de sus contribuciones de manera efectiva.

**OBJETIVO**

**Objetivo General:**

Analizar las diferencias entre Git local y Git remoto para entender su aplicación adecuada en el desarrollo de proyectos de software.

**Objetivos Específicos:**

* **Describir las características clave de Git local y Git remoto.**

Explicar la estructura y funcionamiento de cada tipo de Git, destacando sus ventajas y desventajas.

* **Comparar los comandos básicos de ambos tipos de Git.**

Identificar y contrastar los comandos más utilizados en Git local y Git remoto, facilitando su comprensión y uso práctico.

* **Identificar los escenarios de uso más comunes para cada tipo de Git.**

Proporcionar ejemplos prácticos de situaciones en las que es preferible utilizar Git local o Git remoto, optimizando el flujo de trabajo en proyectos de desarrollo.

**PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL**

**FORMATO GUÍA DE APRENDIZAJE**

**1. IDENTIFICACIÓN DE LA GUÍA DE APRENDIZAJE**

* **Denominación del programa de formación:** Análisis y desarrollo de software.
* **Código del programa de formación:** 228118
* **Nombre del proyecto:** Construcción de software integrador de tecnologías orientadas a servicios.
* **Fase del proyecto:** Ejecución.
* **Actividad de proyecto:** Codificar los módulos del software.
* **Competencias:**

**Técnicas:**

**220501096 -** Desarrollar la solución de software de acuerdo con el diseño y metodologías de desarrollo.

* **Resultados de aprendizaje a alcanzar:**

**Técnicos:**

**220501096-01 -** Planear actividades de construcción del software de acuerdo con el diseño establecido.

**220501096-04 -** Codificar el software de acuerdo con el diseño establecido.

* **Duración de la guía:** 48 horas Técnico: 48 horas.

1. **PRESENTACIÓN**

Estimado aprendiz, el SENA extiende una cordial bienvenida a la guía de aprendizaje que comprende la competencia técnica de: desarrollar la solución de software de acuerdo con el diseño y metodologías de desarrollo. De acuerdo con Maida y Pacienzia (2015): la metodología en el desarrollo de software es un modo sistemático de realizar, gestionar y administrar un proyecto que comprende los procesos a seguir para idear, implementar y mantener un producto software desde que surge la necesidad del producto hasta que cumplimos el objetivo por el cual fue creado.

Para el desarrollo de las actividades planteadas en esta guía, contará con el acompañamiento de los instructores asignados al programa, los cuales de forma continua y permanente lo orientarán con las pautas necesarias para el logro de las actividades de aprendizaje, brindando herramientas básicas de tipo conceptual y metodológico. Los instructores programarán encuentros de asesoría virtual, para brindar orientaciones específicas relacionadas con las temáticas a desarrollar en las actividades. Es importante que organice su tiempo, dada la exigencia que demanda la realización de esta guía de aprendizaje.

Por consiguiente, se presentan cada una de las acciones de aprendizaje que le permitirán desarrollar lo anteriormente mencionado.

1. **FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE**

En este apartado se describirán las actividades de aprendizaje para cada una de las competencias que plantea la fase de ejecución del proyecto formativo: construcción de software integrador de tecnologías orientadas a servicios.

* 1. **Actividades de aprendizaje de la competencia 220501096 - Desarrollar la solución de software de acuerdo con el diseño y metodologías de desarrollo**

Con base en las metodologías de desarrollo utilizadas en esta competencia se inicia con el desarrollo de las aplicaciones web y móviles utilizando Framework para el desarrollo ágil entre los que se encuentran SprintBoot, React, Android, Swift del lado del servidor Node, go, que en conjunto con las herramientas de versionamiento permiten realizar grandes proyectos en tiempos cortos, después de estudiadas cada una de estas tecnologías podrá escoger la que más se adapte a las características del proyecto a realizar.

**3.1.1 Actividad de aprendizaje GA7-220501096-AA1 configurar herramientas de versionamiento para control de código**

**Duración:** 12 horas.

**Materiales de formación:** para el desarrollo de esta actividad es importante la lectura y análisis de los componentes formativos: “Aplicación del paradigma orientado a objetos” e “Integración continua”.

**Evidencias:**

A continuación, se describen las acciones y las correspondientes evidencias que conforma la actividad de aprendizaje:

* **Evidencia de conocimiento: GA7-220501096-AA1-EV01 identifica herramientas de versionamiento**

Tomando como referencia el componente formativo “Integración continua”, realice una tabla con las diferencias entre el sistema de control de versionamiento git local y git remoto.

**Elementos para tener en cuenta en el documento:**

* Se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción, objetivo, tablas con diferencias y características y comandos de git local y git remoto.
* Realice una tabla con la descripción de los comandos básicos de git remoto y git local

**Lineamientos generales para la entrega de la evidencia:**

* **Productos para entregar:** documento informe técnico
* **Extensión:** Libre.
* **Formato:** PDF.
* Para hacer el envío de la evidencia remítase al área de la actividad correspondiente y acceda al espacio asignado.

**Diferencias entre el sistema de control de versionamiento git local y git remoto:**

|  |  |  |
| --- | --- | --- |
| **Características** | **Git Local** | **Git Remoto** |
| Ubicación | Se ejecuta en el sistema local del usuario. | Se ejecuta en un servidor remoto. |
| Almacenamiento | Almacena el historial y versiones en el disco local. | Almacena el historial y versiones en un servidor (p.ej. GitHub, GitLab). |
| Acceso | Acceso directo y rápido a los archivos y commits. | Requiere conexión a Internet para acceder. |
| Colaboración | Individual, solo el usuario tiene acceso. | Permite colaboración entre múltiples usuarios. |
| Sincronización | No necesita sincronización, trabaja con datos locales. | Necesita sincronización con el repositorio local. |
| Comandos | Utiliza comandos para gestionar cambios locales. | Utiliza comandos para gestionar el estado del repositorio remoto. |
| Backup | No es un respaldo a menos que se guarde en otro medio. | Sirve como respaldo centralizado de versiones. |

**Comandos de git local y git remoto:**

|  |  |  |
| --- | --- | --- |
| **Comando** | **Descripción** | **Tipo** |
| git init | Crea una copia local de un repositorio remoto. | Local |
| git clone <url> | Añade un archivo al área de preparación. | Remoto |
| git add <archivo> | Guarda los cambios en el repositorio local. | Local |
| git commit -m "<mensaje>" | Muestra el estado del repositorio local. | Local |
| git status | Envía los cambios locales al repositorio remoto. | Local |
| git push | Actualiza la copia local con los cambios del remoto. | Remoto |
| git pull | Descarga los cambios del remoto sin aplicarlos. | Remoto |
| git fetch | Muestra las ramas locales. | Remoto |
| git branch | Cambia a una rama específica. | Local |
| git checkout <rama> | Crea una copia local de un repositorio remoto. | Local |

* **Evidencia de desempeño: GA7-220501096-AA1-EV02 instalación y configuración de herramienta de versionamiento (Local / Web)**

Tomando como referencia el componente formativo “Integración continua”, realice la instalación y configuración de las herramientas de control de versionamiento tanto local como remoto.

**Elementos para tener en cuenta en el documento:**

* Se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción, objetivo, paso a paso con pantallazos de la instalación de las herramientas de control de versionamiento tanto local como remota.

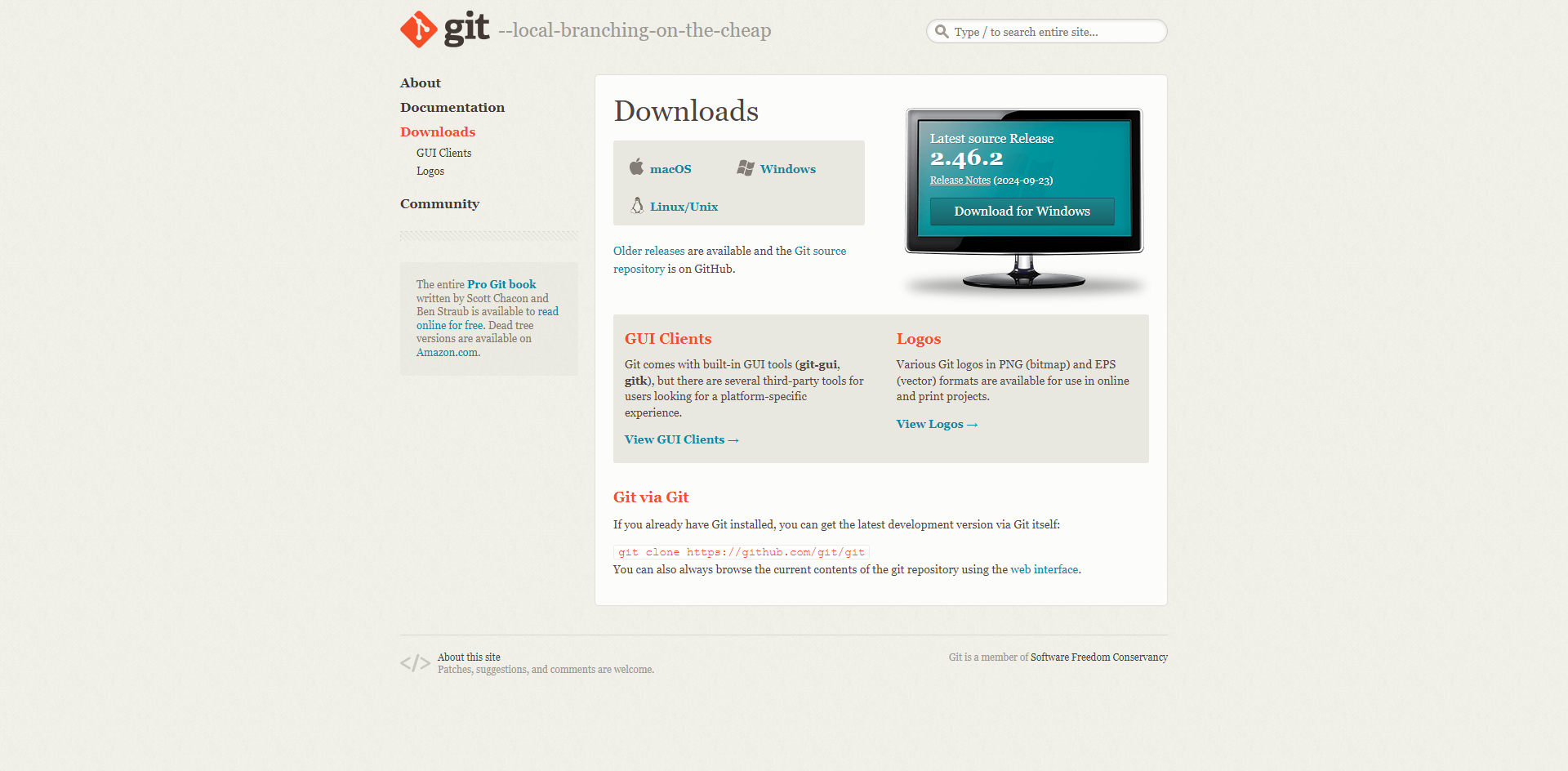
**Lineamientos generales para la entrega de la evidencia:**

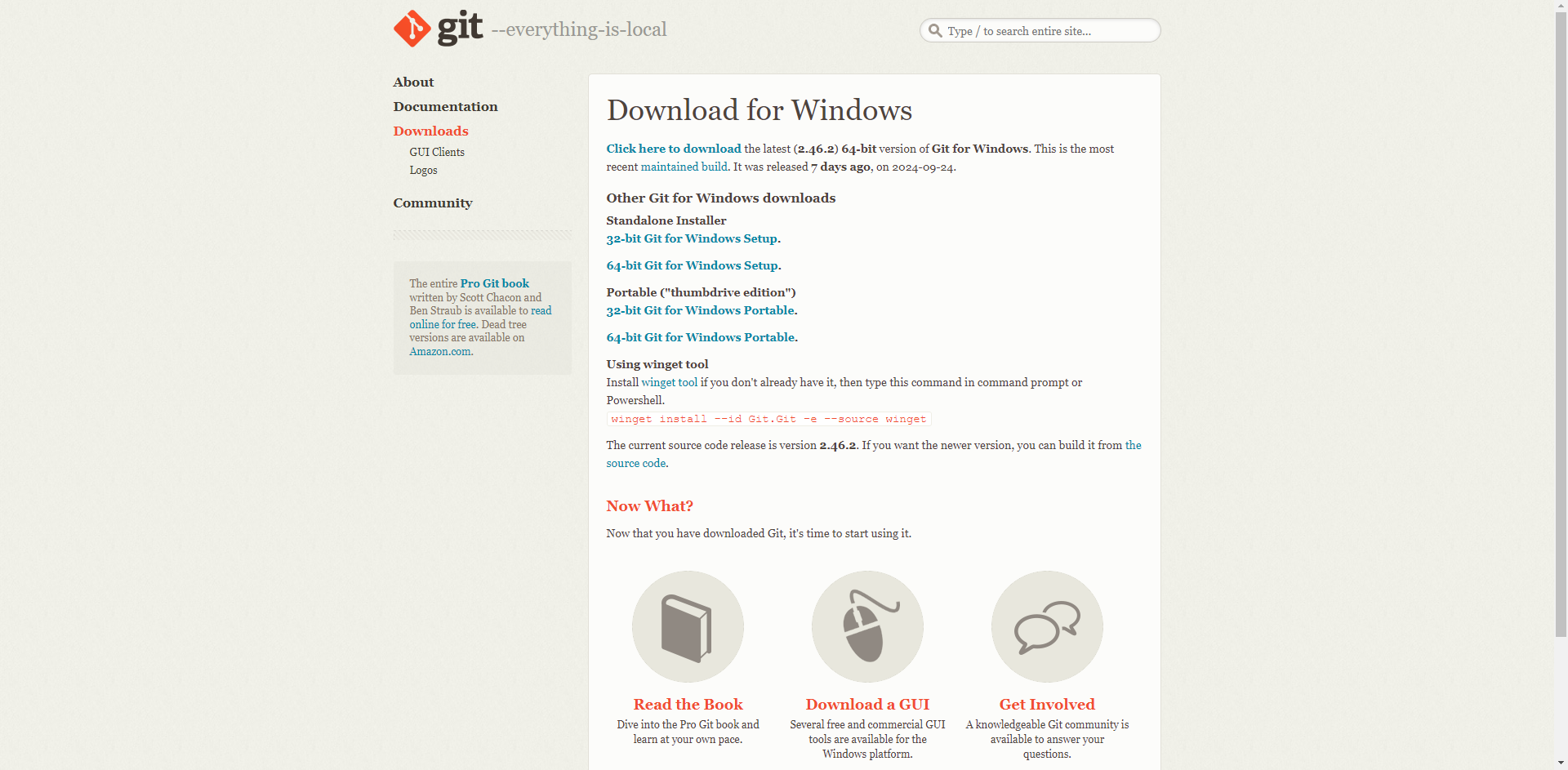
* **Productos para entregar:** documento instalación.
* **Extensión:** Libre.
* **Formato:** PDF.
* Para hacer el envío de la evidencia remítase al área de la actividad correspondiente y acceda al espacio correspondiente.

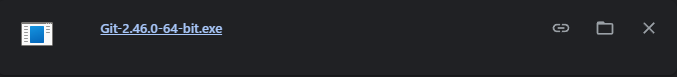
**Instalación de las herramientas de control de versionamiento tanto local como remota:**

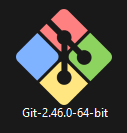
Git (Local)

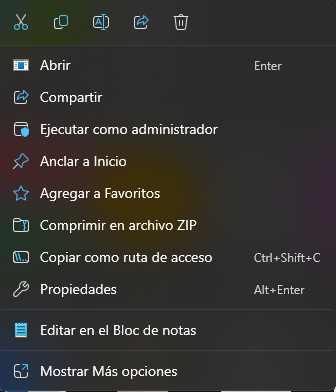
<https://git-scm.com/downloads>

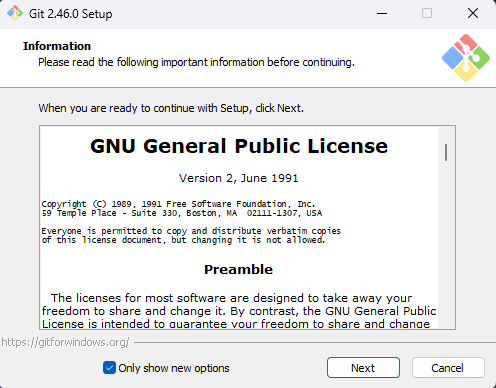


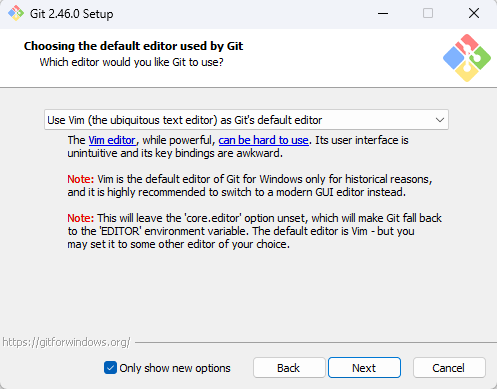


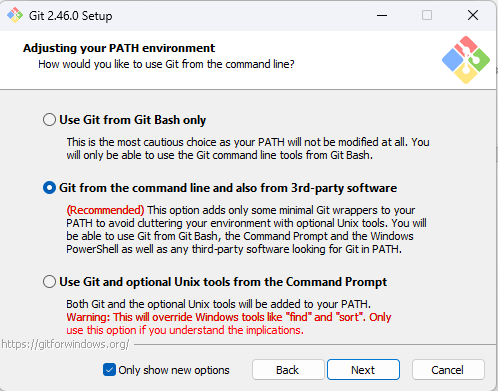


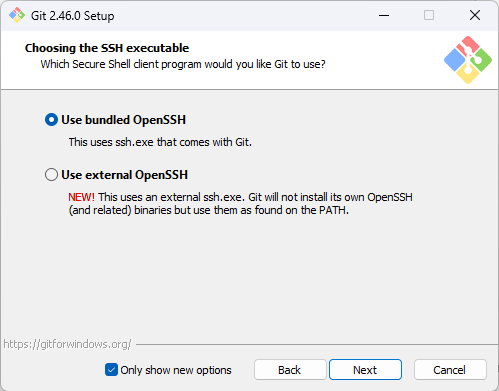


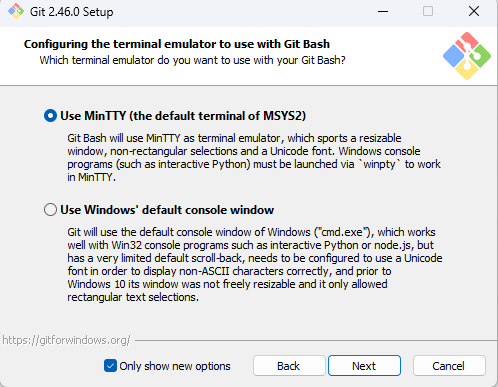


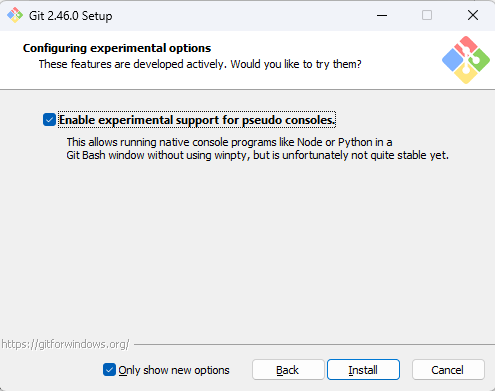


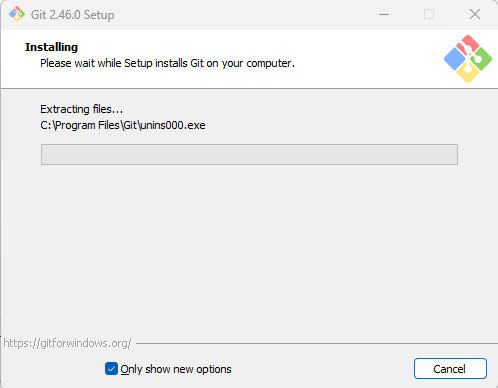


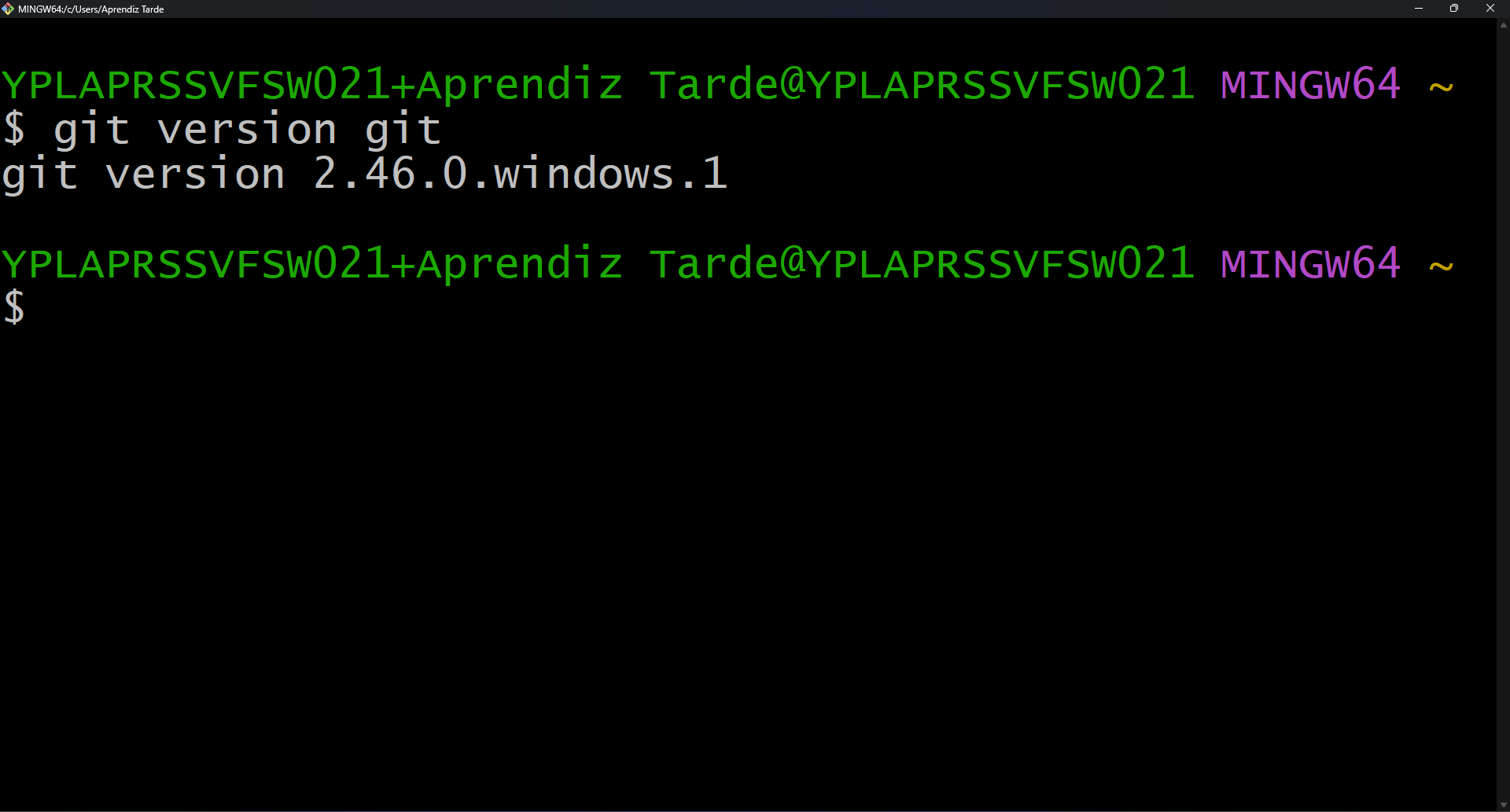




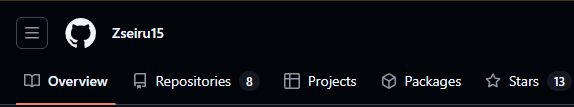




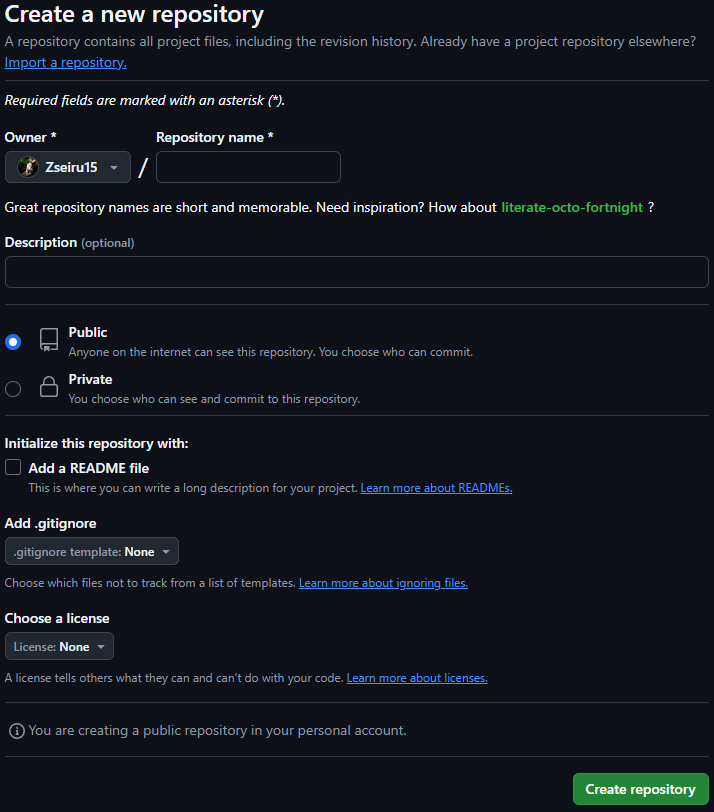




Git (Remoto)







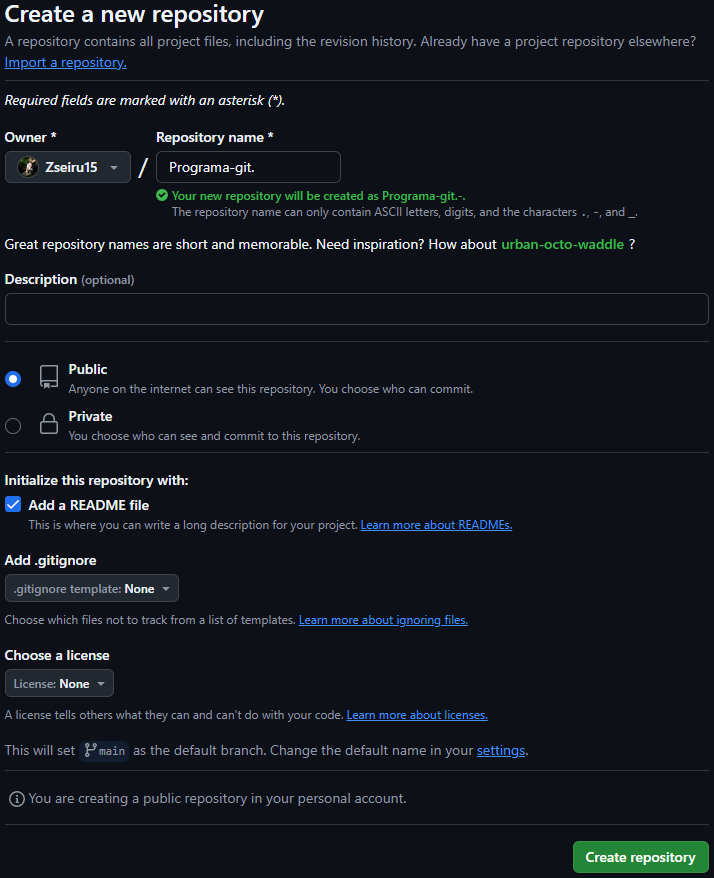
* **Evidencia de producto: GA7-220501096-AA1-EV03 herramientas de versionamiento (GIT) instalada y configurada.**

Tomando como referencia el componente formativo “Integración continua”, conectar el equipo local con el repositorio remoto por medio de Git usando los comandos vistos anteriormente.

Realice la ejecución de cada uno de los comandos básicos de git remoto y local teniendo en cuenta el ejercicio propuesto a continuación:

# Paso 1

1. Crear un nuevo repositorio público en GitHub, gitLab o herramienta de su selección con el nombre Programa-git.
2. Añadirlo al repositorio local del Programa.
3. Mostrar todos los repositorios remotos configurados.

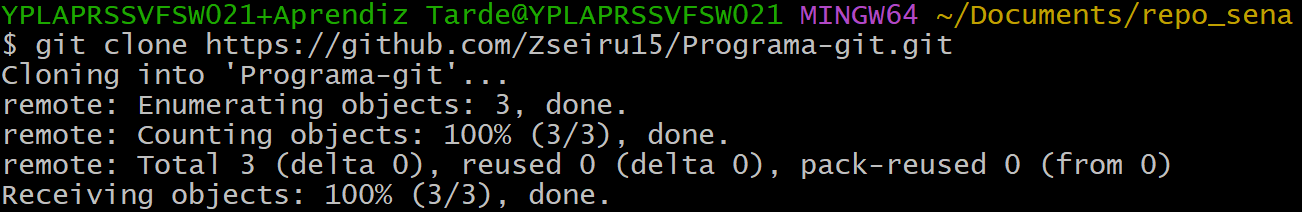


# Paso 1

*# Crear el repositorio en GitHub y copiar su url con el protocolo https.*

> git remote add github url

> git remote -v



# Paso 2

1. Agregar los cambios del repositorio local al repositorio remoto de GitHub o GitLab.
2. Acceder a GitHub o GitLab y comprobar que se han subido los cambios mostrando el historial de versiones.

>git push github master

**Paso 3**

* Agregar el archivo kotlin.txt que contenga el nombre del usuario y su correo electrónico.

* Agregar los cambios.
* Realizar un commit con el mensaje “Añadido datos.”
* Cargar cambios al repositorio remoto.

> cat > kotlin.txt

git add .

> git commit -m

"Añadido datos."

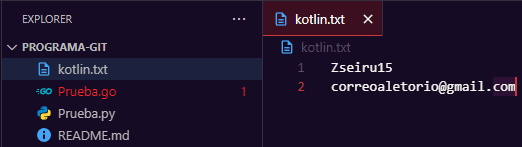
> git push origin master.

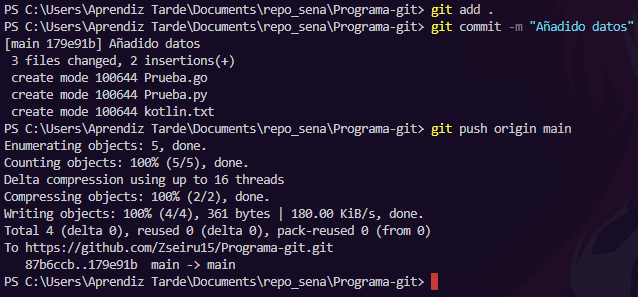
**Elementos para tener en cuenta:**

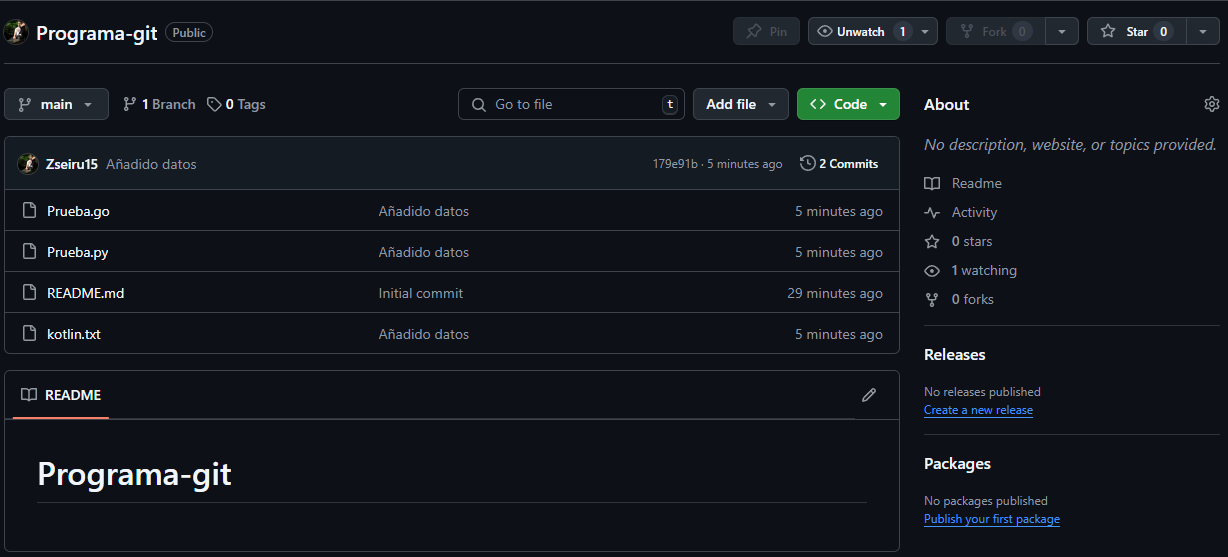
* Se debe realizar el ejercicio en las herramientas de versionamiento instaladas y tomar pantallazos ejecutando cada uno de los puntos solicitados en el ejercicio propuesto.
* Debe ir explicando cada uno de los requerimientos.

**Lineamientos generales para la entrega de la evidencia:**

* **Productos para entregar:** Documento
* **Extensión:** pdf.
* Para hacer el envío de la evidencia remítase al área de la actividad correspondiente y acceda al espacio asignado











**3.1.2 Actividad de aprendizaje GA7-220501096-AA2 - Aplicar estándares de codificación**

Esta actividad se centra en la descripción del plan de trabajo que se debe definir según las características del software a desarrollar, detallando estándares de codificación y lenguaje a utilizar.

**Duración:** 24 horas

**Materiales de formación a consultar:** para el desarrollo de esta actividad es importante la lectura y análisis del componente formativo: “Desarrollo con lenguaje goland o python”.

**Evidencias:** a continuación, se describen las acciones y las correspondientes evidencias que conforma la actividad de aprendizaje**:**

○ **Evidencia de desempeño: GA7-220501096-AA2-EV01 codificación de módulos del software según requerimientos del proyecto**

**Ejercicio: Generar una Factura Simple en PDF**

**Descripción:** Crea un programa en Go o python que solicite al usuario los datos necesarios para generar una factura simple en formato PDF. El programa deberá pedir al usuario que ingrese información como el nombre del cliente, la fecha, una lista de productos con sus precios y cantidades, y finalmente calcular el total.

El PDF debe incluir la siguiente información:

1. Nombre del cliente.
2. Fecha de la factura.
3. Lista de productos, precios unitarios, cantidades y el costo total por producto.
4. Total, general de la factura.

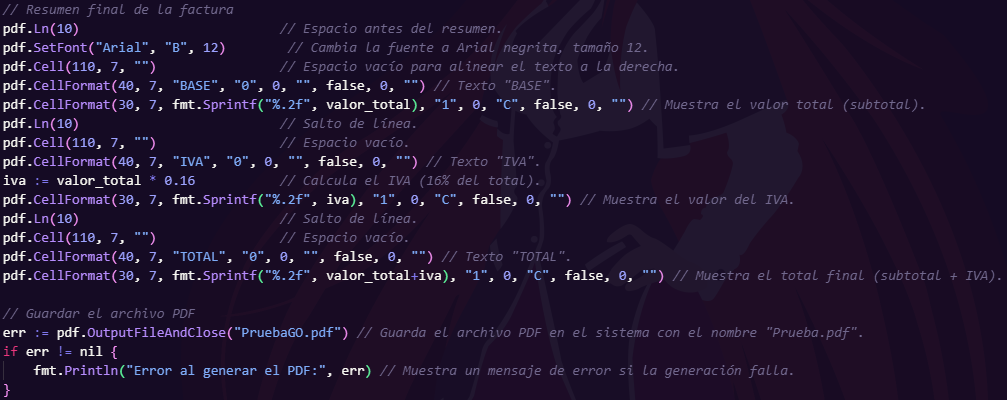
**Requisitos:**

1. **Sentencias de control**: Usa bucles y condiciones para gestionar la entrada de varios productos.

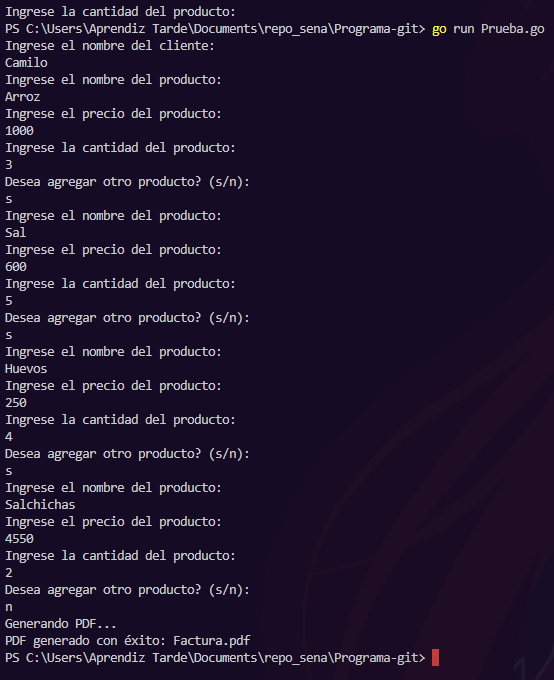
If, For



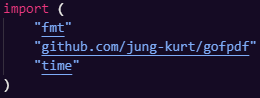
1. **Conversión de variables**: Convierte las cantidades y precios de string a valores numéricos.

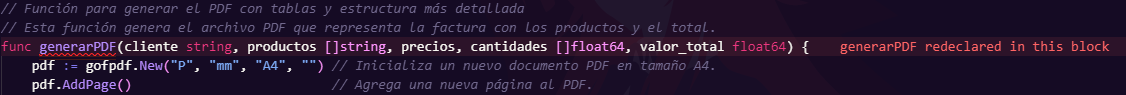


1. **Impresión de datos**: Muestra por consola el resumen de la factura antes de generarla.
2. **Entrada por consola**: Permite al usuario introducir los datos necesarios.

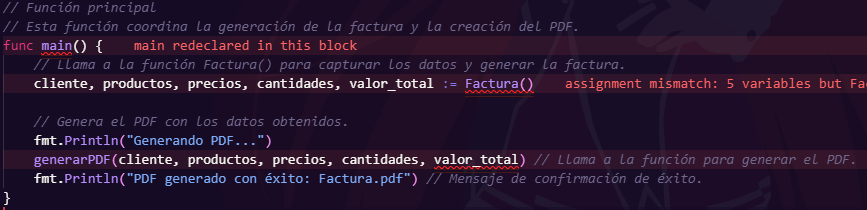


1. **Generar un PDF**: Utiliza una librería como gofpdf para generar el documento.



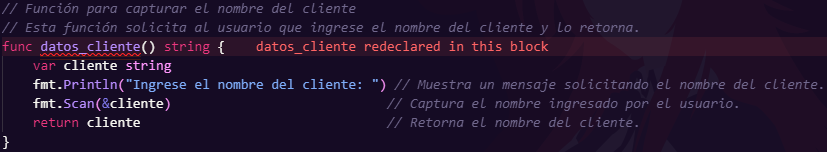




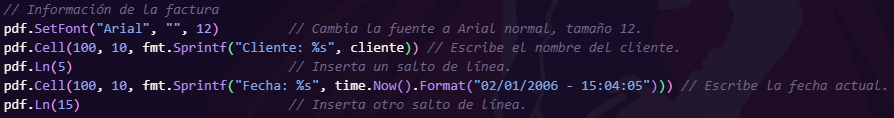


**Instrucciones:**

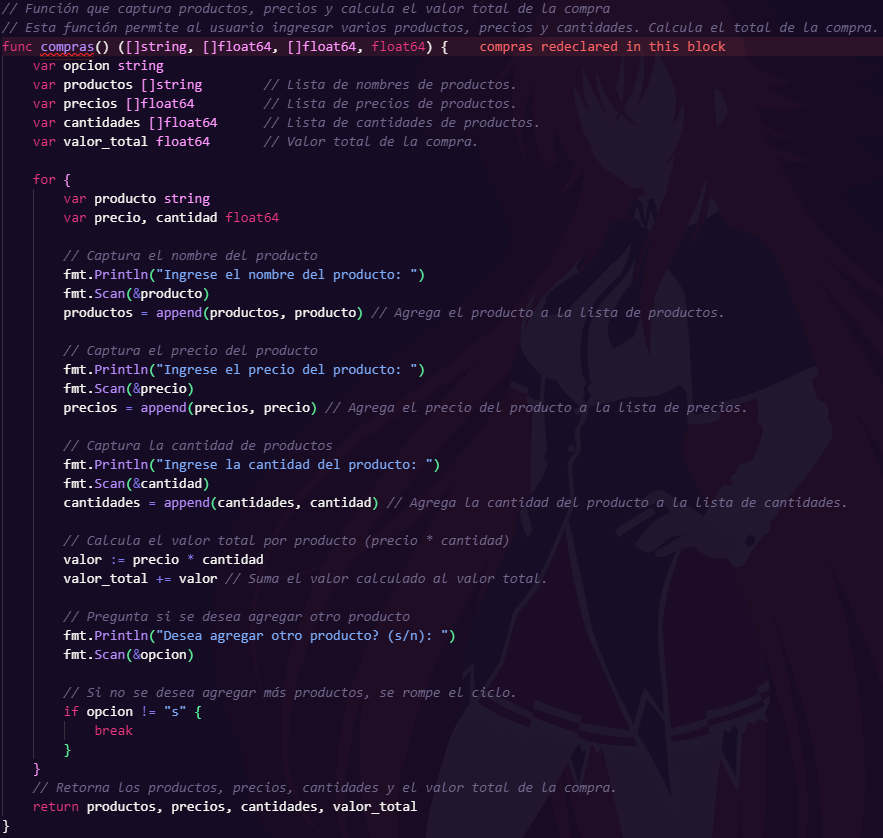
1. **Pedir datos al usuario**:
   * Nombre del cliente.



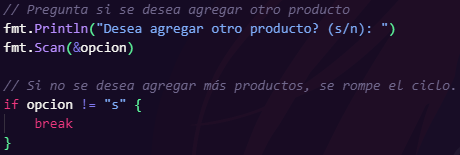
* + Fecha de la factura.



* + Solicitar productos (nombre, cantidad, precio unitario).



* + Permitir al usuario agregar múltiples productos hasta que indique que ha terminado.

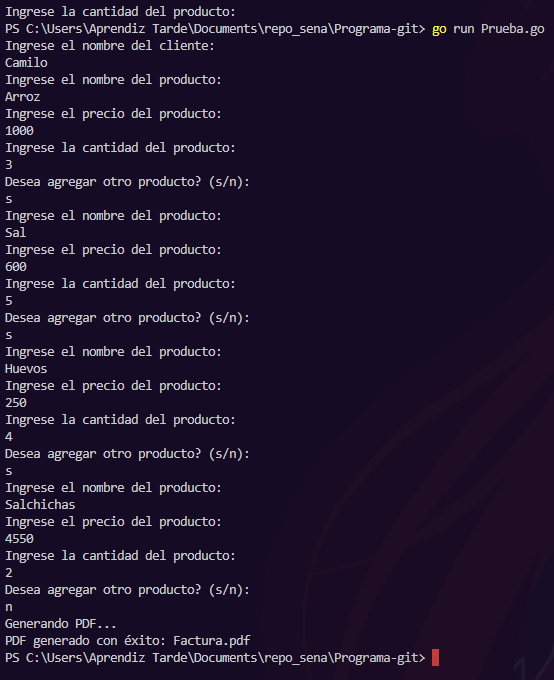


1. **Calcular totales**:

* + Por cada producto, calcular el total (cantidad \* precio unitario).
  + Calcular el total general de todos los productos.



1. **Imprimir por consola** el resumen de la factura antes de generarla.
2. **Generar el PDF** con los datos ingresados.



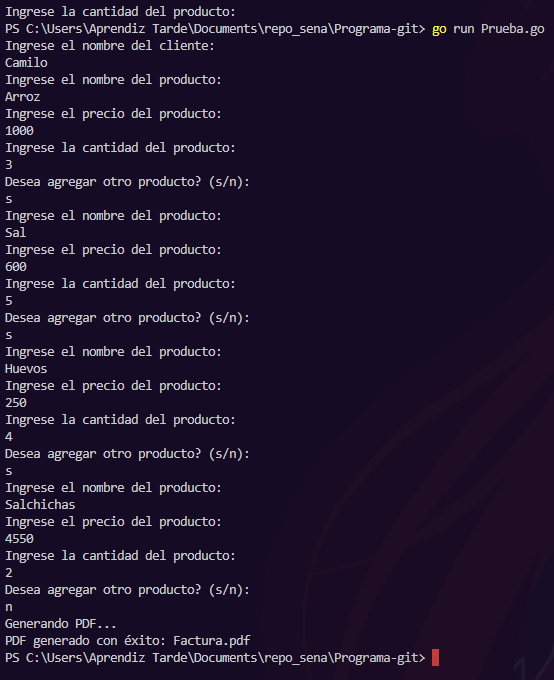
**Lineamientos generales para la entrega de la evidencia:**

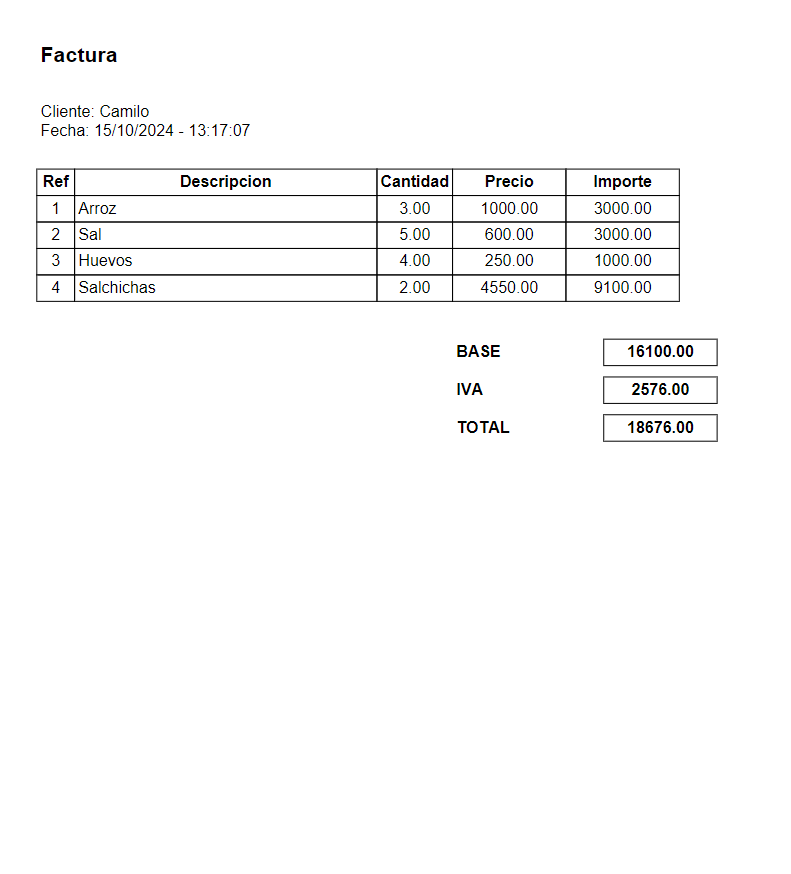
○ **Productos para entregar:** Repositorio que debe tener los siguientes archivos: ● archivos del proyecto,

* archivo con enlace del repositorio, la carpeta comprimida debe tener el nombre del aprendiz y número de la evidencia así: NOMBREAPELLIDO\_AA2\_EV01
* **Extensión:** PDF
* Para hacer el envío de la evidencia remítase al área de la actividad correspondiente y acceda al espacio destinado.

* **Evidencia de producto: GA7-220501096-AA2-EV02 módulos de software codificados y probados**

Teniendo en cuenta la codificación del módulo del proyecto realizada en la evidencia AA2-EV01 realizar las pruebas según requerimientos de las historias de usuario o casos de uso**.**





**Link del repositorio:** <https://github.com/Zseiru15/Programa-git.git>

**Elementos para tener en cuenta:**

* Se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción, objetivo, se requiere tomar un pantallazo de la aplicación por cada uno de los requisitos levantados en las historias de usuario o casos de uso, ósea que debe describirse la historia de usuario y/o casos de uso y debajo debe ir el pantallazo de la interfaz de la aplicación, se deben documentar también las pruebas de validaciones de la aplicación (fechas, números, textos, caracteres especiales, longitudes etc)
* Debe presentar un documento mostrando toda la funcionalidad del módulo codificado con sus respectivas validaciones.
* Debe trabajar con herramientas de versionamiento.

**Lineamientos generales para la entrega de la evidencia:**

* **Productos para entregar:** carpeta comprimida que debe tener los siguientes archivos: archivos del proyecto, documento en Word
* Archivo con enlace del repositorio, la carpeta comprimida debe tener el nombre del aprendiz y número de la evidencia así: NOMBRE\_APELLIDO\_AA2\_EV02
* **Extensión:** ZIP, RAR.
* Para hacer el envío de la evidencia remítase al área de la actividad correspondiente y acceda al espacio destinado.

1. **ACTIVIDADES DE EVALUACIÓN**

|  |  |  |
| --- | --- | --- |
| **Evidencias de aprendizaje** | **Criterios de evaluación** | **Técnicas e instrumentos de evaluación** |
| Evidencias de conocimiento: Informe técnico de plan de trabajo para construcción de software.  **GA7-220501096-AA1-EV01**    Evidencias de desempeño:  Definir estándares de codificación de acuerdo a plataforma de desarrollo elegida.  **GA7-220501096-AA1-EV02** | Interpreta el informe de diseño para definir el plan de trabajo en la construcción del software.    Selecciona y configura herramientas de desarrollo de acuerdo con las condiciones del software a construir.    Define estándares de codificación de acuerdo con las reglas de la plataforma de desarrollo seleccionada. | **IE-GA7-220501096-AA1-EV01**  Lista de chequeo        **IE-GA7-220501096-AA1-EV02**  Lista de chequeo |

|  |  |  |
| --- | --- | --- |
| Evidencia de conocimiento: Identifica herramientas de versionamiento.  **GA7-220501096-AA1-EV03** |  | **IE-GA7-220501096-AA1-EV03**  Lista de chequeo |
| Evidencias de desempeño:  Instalación y configuración de herramienta de versionamiento (Local / Web).  **GA7-220501096-AA1-EV04**  Evidencia de producto: Herramientas de versionamiento (GIT) instalada y configurada.  **GA7-220501096-AA1-EV05** | Selecciona y configura herramientas de versionamiento para el control de cambios en el código. |  |
| Evidencias de desempeño: Codificación de módulos del software según requerimientos del proyecto.  **GA7-220501096-AA2-EV01** | Codifica los módulos del software  Stand alone, web y móvil, de acuerdo | **IE-GA7-220501096-AA2-EV01**  Lista de chequeo |
| Evidencia de producto:  Módulos de software codificados y probados.  **GA7-220501096-AA2-EV02** | con las especificaciones del diseño y el estándar de codificación. |  |
| Evidencias de desempeño: Codificación de módulos del software Stand alone. **GA7220501096-AA3-EV01**  Evidencia de producto: Módulos de software codificados y probados.  **GA7-220501096-AA3-EV02** |  | **IE-GA7-220501096-AA2-EV02**  Lista de chequeo |

|  |  |  |
| --- | --- | --- |
|  |  |  |

1. **GLOSARIO**

**Software**: soporte lógico, programas, parte no mecánica de un sistema. Serie de instrucciones necesarias para ejecutar diversas aplicaciones y tareas.

**WAR**: archivo de aplicación web utilizado para empaquetar las aplicaciones web hechas con tecnología JAVA. La Sun define la estructura de un WAR.

**Servlet**: es una clase hecha en Java utilizada para extender las capacidades de los servidores que alojan aplicaciones que siguen el modelo petición-respuesta.

**RDBMS**: administrador de Bases de Datos Relacionales, se conocen así a los programas que permiten la gestión completa de bases de datos y su mantenimiento.

1. **REFERENTES BIBLIOGRÁFICOS**

Java en Castellano. (2021, Dic 01). Servlets y JSP.<https://programacion.net/articulo/servlets_y_jsp_82>

Maida, E. G., & Pacienzia, J. (2015). Metodologías de desarrollo de software.

Quintas, A. F. (2000). *Java 2: manual de usuario y tutorial*. Grupo Editorial RA-MA.

Sun. (2021, Dic 01). Java.<http://java.sun.com/>

Vall Mainou, A. (2010). Desarrollo para internet con tecnología JAVA.

1. **CONTROL DEL DOCUMENTO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Nombre** | **Cargo** | **Dependencia** | **Fecha** |
| Autor(es) | Jonathan  Guerrero Astaiza | Experto temático | Centro de Teleinformática y Producción Industrial | Diciembre 2021 |
| Zulema  Leon Escobar | Experta temático | Centro de Teleinformática y Producción Industrial | Diciembre 2021 |
| Deivis Eduard  Ramírez Martínez | Diseñador  Instruccional | Regional Distrito Capital - Centro para la Industria de la Comunicación Gráfica | Diciembre 2021 |
| Silvia Milena  Sequeda Cárdenas | Metodológa | Regional Distrito Capital - Centro de diseño y metrología | Febrero de 2022 |
| Rafael Neftalí Lizcano Reyes | Asesor  Pedagógico | Regional Santander - Centro Industrial del Diseño y la Manufactura | Febrero de 2022 |
| Sandra Patricia Hoyos Sepúlveda | Corrección estilo de | Centro de Diseño y Metrología - Distrito capital | Febrero 2021 |
| Fabian David Barreto Sanchez | Ajuste de actividades | Regional Casanare | Septiembre del 2024 |

**7. CONTROL DE CAMBIOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Nombre** | **Cargo** | **Dependencia** | **Fecha** | **Razón del cambio** |
| **Autor (es)** |  |  |  |  |  |