
**Instituto Tecnológico de Las Américas**

**Facultad de Desarrollo de Software**

# Trabajo de Investigación:

Tarea 3

**Estudiante:**

Heger Arias Santos

**Matricula:**

2021-0758

**Asignatura:**

Programación 3

**Docente:**

Kelyn Tejada

**Práctica individual Valor 10 Puntos**

**Trabajar de forma individual**

# 1- Desarrolla el siguiente Cuestionario

## 1- ¿Qué es Git?

Git es un proyecto de código abierto maduro y con un mantenimiento activo que desarrolló originalmente Linus Torvalds, el famoso creador del kernel del sistema operativo Linux, en 2005. Un asombroso número de proyectos de software dependen de Git para el control de versiones, incluidos proyectos comerciales y de código abierto. Los desarrolladores que han trabajado con Git cuentan con una buena representación en la base de talentos disponibles para el desarrollo de software, y este sistema funciona a la perfección en una amplia variedad de sistemas operativos e IDE (entornos de desarrollo integrados).

Git, que presenta una arquitectura distribuida, es un ejemplo de DVCS (sistema de control de versiones distribuido, por sus siglas en inglés). En lugar de tener un único espacio para todo el historial de versiones del software, como sucede de manera habitual en los sistemas de control de versiones antaño populares, como CVS o Subversión (también conocido como SVN), en Git, la copia de trabajo del código de cada desarrollador es también un repositorio que puede albergar el historial completo de todos los cambios.

Además de contar con una arquitectura distribuida, Git se ha diseñado teniendo en cuenta el rendimiento, la seguridad y la flexibilidad.

## 2- ¿Para que funciona el comando Git init?

Este comando inicializa un proyecto de git, creando a nivel de esa carpeta un archivo. git el cual tiene todas las configuraciones necesarias para poder comenzar a utilizar git en dicha carpeta.

## 3-Como saber es que rama estoy?

En git existe un comando llamado git Branch, este comando enlistara las ramas que hay disponibles en nuestro proyecto e indicara con un * en cual rama nos encontramos actualmente, adicional a esto la mayoría de herramientas que utilizamos a diario como por ejemplo git bash el cual es la terminar predeterminada de git delante de la ubicación de la carpeta actual también dice en que rama nos ubicamos.

## 4-Que es una rama?

Las ramas son una división del estado del código, esto permite crear nuevos caminos a favor de la evolución del código. Normalmente la creación de ramas en otros sistemas de control de versiones puede tomar mucho tiempo y ocupar demasiado espacio en el almacenamiento. Por el contrario, en Git, las ramas son parte diaria del desarrollo, son una guía instantánea para los cambios realizados.

Por ejemplo, imagina que quieres añadir una nueva función o tal vez arreglar un error, sin importar su tamaño, generas una nueva rama en la cual se alojan estos cambios que realizaste, al realizar esta acción va resultar más complicado que algún error o fallo del código inestable se incorpore al código base principal, dando la oportunidad de limpiar tu historial antes de fusionarlo todo con la rama principal, mejorando tu eficiencia de trabajo.

Para poder utilizar las ramas en Git es necesario el comando git branch y la implementación que estas realizan en Git es mucho más sencilla que la de otros modelos de sistemas de control de versiones. En vez de copiar entre directorios, Git almacena las ramas como referencia a una confirmación. Por lo tanto, una rama representa el extremo de una serie de confirmaciones, es decir, no es un contenedor de confirmaciones. El historial de una rama se extrapola de las relaciones de confirmación.

## 5-Quien creo git?

El famosísimo programador, padre del open source Linus Torvalds.

## 6-Cuales son los comandos más esenciales de Git?

- Git clone
- Git branch
- Git checkout
- Git status
- Git add
- Git commit
- Git push
- Git pull

## 7-Cuales son los repositorios de git más conocido?

La mayoría de los respositorios del open source de herramientas como: React, vue e incluso lenguajes de programación como javascript están en repositorios hechos en git y subidos a la plataforma github, aunque aquí les traigo uno nuevo para innovar, el cual es de un lenguaje creado por Google con el motivo de hacer de sustituto a C++
https://github.com/carbon-language/carbon-lang

## 2- Desarrolle un ejercicio práctico en Azure Devops o GitHub (ZenHub) con las siguientes características

Link del proyecto: https://github.com/HegerAriasSantos/tarea-individual-prog-3

### Proyecto Anotalo

### Historias de usuario

| | | |
|---|---|---|
| **HU01- Acceder** | Yo como vendedor quiero acceder al sistema de forma rápida y sencilla para trabajar con más fluidez en mi negocio. | |
| **CRITERIOS DE ACEPTACIÓN** | | |
| 1. | **Interfaz** | En caso de que quiera acceder al sistema este permitirá acceder de forma sencilla. |
| 2. | **Campos obligatorios** | En caso de que el vendedor deje un campo vacío, cuando proceda a acceder el sistema mostrara una advertencia. |
| 3. | **Espacio crear cuenta** | En caso de que el administrador quiera crear una cuenta de usuario nueva esta opción debe estar presente. |

| | | |
|---|---|---|
| **HU02- Crear cuentas de usuario** | Yo como administrador quiero crear nuevas cuentas de usuarios para que los empleados nuevos hagan uso del sistema. | |
| **CRITERIOS DE ACEPTACIÓN** | | |

| | | |
|---|---|---|
| 1. | **Crear Cuenta** | En caso de que el administrador quiera crear una nueva cuenta de usuario el sistema mostrara un espacio para acceder a realizar este proceso. |

| | | |
|---|---|---|
| 2. | **Nombres Repetidos** | En caso de que el administrador introduzca un nombre de usuario que ya está en uso el sistema enviara un mensaje de alerta e impedirá su creación. |
| 3. | **Contraseña débil** | En caso de que el administrador introduzca una contraseña débil el sistema proporcionara un mensaje diciendo que debe usar una contraseña más fuerte con símbolos, mayúsculas y caracteres. |

| | | |
|---|---|---|
| **HU03- Almacenar información de cliente** | Yo como vendedor quiero tener un espacio en el sistema para guardar la información de los clientes. | |
| **CRITERIOS DE ACEPTACIÓN** | | |
| 1. | **Guardar cliente** | En caso de que el vendedor quiera guardar un cliente nuevo el sistema tendrá una opción que se lo permita. |
| 2. | **Interfaz** | En caso de que el vendedor solicite guardar un nuevo cliente el sistema presentara una interfaz clara y sencilla de entender. |
| 3. | **Campos** | En caso de que el vendedor se encuentre en el área de guardar clientes del sistema este deberá introducir el nombre, apellidos, cedula, dirección y teléfono del cliente para registrar el cliente. |
| 4. | **Identificadores** | En caso de que el vendedor guarde un cliente nuevo el sistema de forma automática le asignara un identificador único. |

| | | |
|---|---|---|
| **HU04- Deudas y abonos** | Yo como vendedor quiero agregar deudas y abonos para el cliente que seleccione. | |
| **CRITERIOS DE ACEPTACIÓN** | | |
| 1. | **Agregar deudas** | En caso de que el vendedor quiera agregar una deuda a un cliente el sistema mostrara una opción que le permita realizar esta acción. |
| 2. | **Agregar abonos** | En caso de que el vendedor quiera agregar un abono a un cliente el sistema mostrara una opción que le permita realizar esta acción. |
| 3. | **Sumar deudas** | En caso de que el vendedor agregue deudas a un cliente estas se irán sumando a medida que se agreguen. |
| 4. | **Visualizar abonos** | En caso de que el vendedor agregue abonos a un cliente estos se les irán restando al total de la deuda. |
| 5. | **Campos de información de agregar deuda** | En caso de que el vendedor quiera agregar una deuda esta se guardara en un campo. |
| 6. | **Campos de información agregar abono** | En caso de que el vendedor quiera agregar un abono este se guardara en un campo. |

| | | |
|---|---|---|
| **HU05- Vista abonos y deudas** | Como vendedor quiero visualizar el estado de la deuda y abono de cada uno de los clientes para saber su estado de cuenta. | |
| **CRITERIOS DE ACEPTACIÓN** | | |
| 1. | **Visualizar** | En caso de que quiera visualizar la deuda y abonos de un cliente el sistema mostrara un espacio con dicha información asociada al cliente que corresponde. |
| 2. | **Búsqueda** | En caso de que el vendedor quiera buscar un cliente en específico para conocer el su deuda y abonos hasta el momento el sistema cuenta con un espacio que le permite realizar dicha acción. |

| | | |
|---|---|---|
| **HU06- Saldar deudas** | Como vendedor quiero saldar las deudas de los clientes para archivar que ese cliente ya pago su deuda. | |
| **CRITERIOS DE ACEPTACIÓN** | | |
| 1. | **Vista** | En caso de que el vendedor seleccione un cliente el sistema debe proporcionar un espacio que le permita saldar la deuda del cliente. |
| 2. | **Estado de cuenta** | En caso de que el vendedor salde la deuda de un cliente su cuenta se establecerá como saldada sin deudas. |
| 3. | **Eliminar cuenta** | En caso de que el vendedor salde la deuda de un cliente el sistema le dará la opción de si eliminar o conservar los datos del cliente. |

| | | |
|---|---|---|
| 4. | **Saldar por abono** | En caso de que el vendedor agregue un abono que cubra el total de la deuda del cliente o supere su totalidad la cuenta del cliente seleccionado se saldara. |

| | | |
|---|---|---|
| **HU07- Vista principal** | Como vendedor quiero visualizar todos los clientes que me deben para poder acceder a su información fácilmente. | |
| **CRITERIOS DE ACEPTACIÓN** | | |
| 1. | **Organizar de mayor a menor** | En caso de que el vendedor entre a la vista, deberá ver de primero los clientes con más deuda acumulada. |
| 2. | **Campos** | En caso de que el vendedor entre a la vista, deberá ver el cliente desglosado en diversos campos: deuda, nombre, apellido, fecha de ultimo pago, ultimo pago |
| 3. | **Total, de deudas acumuladas** | En caso de que el vendedor entre a la vista, deberá ver en un apartado de con una forma muy simple y atractiva a la vista, la sumatoria de las deudas de los clientes, para conocer el dinero que deberá cobrar. |

**Estimación de historias** **Puntos de historia**

| | |
|---|---|
| **HU01- Acceder** | **Yo como vendedor quiero acceder al sistema de forma rápida y sencilla para trabajar con más fluidez en mi negocio.** |

10

| | |
|---|---|
| **HU02- Crear cuentas de usuario** | Yo como administrador quiero crear nuevas cuentas de usuarios para que los empleados nuevos hagan uso del sistema. |

14

| | |
|---|---|
| **HU03- Almacenar información de cliente** | Yo como vendedor quiero tener un espacio en el sistema para guardar la información de los clientes. |

15

| | |
|---|---|
| **HU04- Deudas y abonos** | Yo como vendedor quiero agregar deudas y abonos para el cliente que seleccione. |

12

| | |
|---|---|
| **HU05- Vista abonos y deudas** | Como vendedor quiero visualizar el estado de la deuda y abono de cada uno de los clientes para saber su estado de cuenta. |

15

| | |
|---|---|
| **HU06- Saldar deudas** | Como vendedor quiero saldar las deudas de los clientes para |

10

| | |
|---|---|
| | archivar que ese cliente ya pago su deuda. |

| | |
|---|---|
| **HU07- Vista principal** | Como vendedor quiero visualizar todos los clientes que me deben para poder acceder a su información fácilmente. |

18

## Sprint Backlog

| Épicas | Historias de Usuario |
|---|---|
| **Sistema de inicio de Sesión** | – **HU01- Acceder**<br>– **HU02- Crear cuentas de usuario** |
| **Almacenar Información de Clientes** | – **HU03- Almacenar información de cliente** |
| **Vista de los Datos Generales** | – **HU07- Vista principal**<br>– **HU05- Vista abonos y deudas** |
| **Gestión de Deudas de Clientes** | – **HU06- Saldar deudas**<br>– **HU04- Deudas y abonos** |