

Universidad San Carlos de Guatemala
Facultad de ingeniería.
Ingeniería en ciencias y sistemas



Proyecto 2: **[GoDisk 2.0]**

PONDERACIÓN: 30

Horas Aproximadas: 168

Índice

1. Resumen Ejecutivo.....	3
2. Competencia que desarrollaremos.....	3
3. Objetivos del Aprendizaje.....	3
3.1 Objetivo General.....	3
3.2 Objetivos Específicos.....	4
4. Enunciado del Proyecto.....	5
4.1 Descripción del problema a resolver.....	5
Arquitectura.....	5
Frontend.....	5
Backend.....	5
Primer Parte (Frontend).....	6
Iniciar Sesión:.....	7
Visualizador del Sistema de Archivos.....	7
Se agrega el manejo de Sistema de Archivos Ext3.....	12
EXT3.....	12
Nueva estructura.....	13
Journal.....	13
Information.....	14
Nuevos comandos que se deben agregar al Sistema de Archivos.....	14
1. FDISK (ADD y DELETE).....	14
2. UNMOUNT.....	17
3. MKFS (FS).....	18
4. Remove.....	19
5. Edit.....	20
6. Rename.....	20
7. Copy.....	21
8. Move.....	23
9. Find.....	24
10. Chown.....	25
11. Chmod.....	25
Pérdida y Recuperación del Sistema de Archivos EXT3.....	27
Recovery File System.....	27
Simulate System Loss.....	27
Journaling.....	28
Documentación.....	29
Instrucciones de Entrega.....	29
Requisitos Mínimos.....	30
Consideraciones.....	30
4.2 Alcance del proyecto.....	31

4.4 Entregables.....	32
5. Metodología.....	33
6. Desarrollo de Habilidades Blandas.....	34
6.1 Proyectos Individuales.....	34
7. Cronograma.....	35
8. Rúbrica de Calificación.....	36
8.1 Requisitos para optar a la calificación.....	36
8.2 Resumen de Puntuaciones.....	37
8.3 Detalle de la Calificación.....	37
8.4 Valores.....	40
8.5 Comentarios Generales.....	41

1. Resumen Ejecutivo

Este proyecto consiste en la evolución del sistema de archivos simulado del Proyecto 1, con el objetivo de mejorar su accesibilidad y usabilidad mediante una interfaz gráfica de usuario (GUI) basada en la web. El problema que se busca resolver es la dificultad de interactuar y visualizar la estructura del sistema de archivos únicamente a través de comandos de consola. Para solucionarlo, se desarrollará una aplicación web que permite explorar visualmente discos, particiones, carpetas y archivos de forma intuitiva.

Además de la visualización, se incorporarán nuevas funcionalidades como manejo del sistema de archivos EXT3, comandos avanzados (como `fdisk`, `remove`, `edit`, `copy`, `chown`, entre otros), y un sistema de journaling para registrar las operaciones. La solución estará desplegada en la nube utilizando servicios de AWS, con un frontend alojado en un bucket S3 y un backend en una instancia EC2 con sistema operativo Linux, asegurando así escalabilidad, accesibilidad y eficiencia.

2. Competencia que desarrollaremos

- Implementa un sistema de archivos con estructuras ext2/ext3 expuesto mediante una API e interfaz gráfica mediante la integración de funciones del backend y visualización de resultados en el frontend para gestionar operaciones relacionadas a la construcción y manejo de un sistema de archivos .
- Integra soluciones de almacenamiento local y en la nube mediante la selección y combinación de tecnologías de almacenamiento físico y virtualizado para diseñar infraestructuras eficientes, seguras y escalables
- Analiza arquitecturas de virtualización de almacenamiento mediante la comparación de modelos y evaluación de retos operativos para optimizar la administración de recursos en entornos virtualizados

3. Objetivos del Aprendizaje

3.1 Objetivo General

El estudiante será capaz de diseñar y desarrollar una aplicación web que permita la visualización y gestión de un sistema de archivos simulado, integrando tecnologías como Go para el backend y frameworks modernos para el frontend. La solución incluirá soporte para sistemas de archivos EXT2 y EXT3, nuevas funcionalidades de administración, y será desplegada en la nube utilizando servicios de AWS como EC2 y S3, aplicando conocimientos de programación, diseño de sistemas, y despliegue en entornos distribuidos.

3.2 Objetivos Específicos

Al finalizar el proyecto, los estudiantes deberán ser capaces de:

1. Desarrollar un sistema funcional basado en la nube

Aplicar los conocimientos adquiridos en el curso para diseñar, implementar y probar una aplicación web que permita visualizar y gestionar un sistema de archivos simulado en formato EXT2 y EXT3. El sistema será desplegado en AWS, utilizando EC2 para el backend en Go y S3 para el frontend.

Ejemplo: Los estudiantes podrán simular operaciones del sistema de archivos como creación de discos, montaje, navegación y recuperación desde una interfaz web accesible desde cualquier dispositivo.

2. Diseñar interfaces gráficas funcionales e intuitivas

Implementar una interfaz gráfica de usuario (GUI) basada en tecnologías web modernas que permita una navegación sencilla del sistema de archivos, facilitando la interacción visual con discos, particiones, carpetas y archivos.

Ejemplo: Los estudiantes podrán desarrollar una interfaz similar a un explorador de archivos que permita al usuario explorar visualmente la jerarquía del sistema y ejecutar comandos mediante una terminal embebida.

3. Documentar adecuadamente el sistema desarrollado

Elaborar un manual técnico que incluya la arquitectura del sistema, los comandos implementados, la explicación de estructuras internas como journaling e inodos, y el proceso de despliegue en AWS.

Ejemplo: Los estudiantes serán capaces de entregar un documento que describa detalladamente el funcionamiento interno del sistema de archivos y cómo fue implementado y desplegado en la nube.

4. Enunciado del Proyecto

En el mundo de la informática, las tecnologías y las necesidades de los usuarios están en constante evolución. Esto nos impulsa a mejorar y optimizar continuamente los sistemas existentes. En este proyecto, estamos evolucionando el Proyecto 1, haciéndolo más accesible y visualmente atractivo. Nuestro objetivo es desarrollar una interfaz gráfica de usuario (GUI) basada en web que permita visualizar fácilmente todo el sistema de archivos creado a través de comandos. Esto facilitará la navegación entre discos, particiones, carpetas y archivos. Además, se incorporarán nuevas funcionalidades que se detallarán más adelante y se ampliará el soporte para incluir sistemas de archivos EXT3.

Para asegurarnos de que todos puedan acceder al sistema y de que este pueda crecer fácilmente, hemos decidido usar la nube. Específicamente, estamos utilizando los servicios de Amazon Web Services (AWS). Aprovechando así sus capacidades de almacenamiento, procesamiento y despliegue de aplicaciones de manera eficiente y segura.

4.1 Descripción del problema a resolver

Arquitectura

El proyecto tendrá la siguiente arquitectura:

Frontend

La interfaz gráfica de usuario deberá ser desarrollada en una página web mediante un framework como React, Angular, Vue, u otro, dejando a discreción del estudiante el framework a utilizar.

Dicha página web deberá ser desplegada mediante el servicio de bucket S3 de AWS.

Backend

Para esta implementación se requerirá la creación de un bucket, que será una API Rest desarrollada en lenguaje Go. Este backend deberá integrarse con el proyecto 1.

Dicho backend deberá ser desplegado en una instancia EC2 de AWS. Esta instancia deberá tener como sistema operativo alguna distribución Linux, se recomienda Ubuntu.

La infraestructura final será:

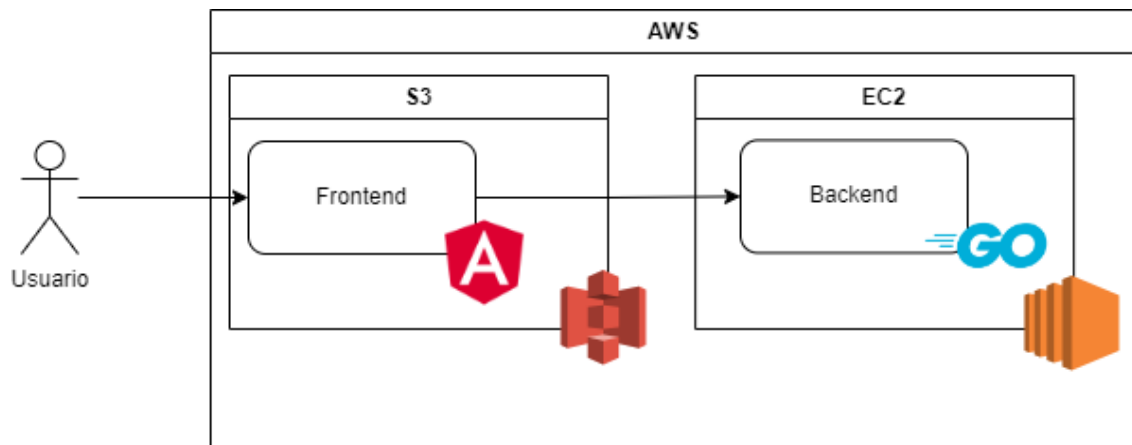


Figura 1 - Infraestructura en nube

Primer Parte (Frontend)

Primero se mostrará la página principal del programa, la cual contará con una terminal para poder ejecutar comandos. Esta servirá para poder ejecutar todos los comandos iniciales (para la creación de discos, montaje y formateo de las particiones). Esta contará con un botón para poder iniciar sesión con alguno de los usuarios de una partición.

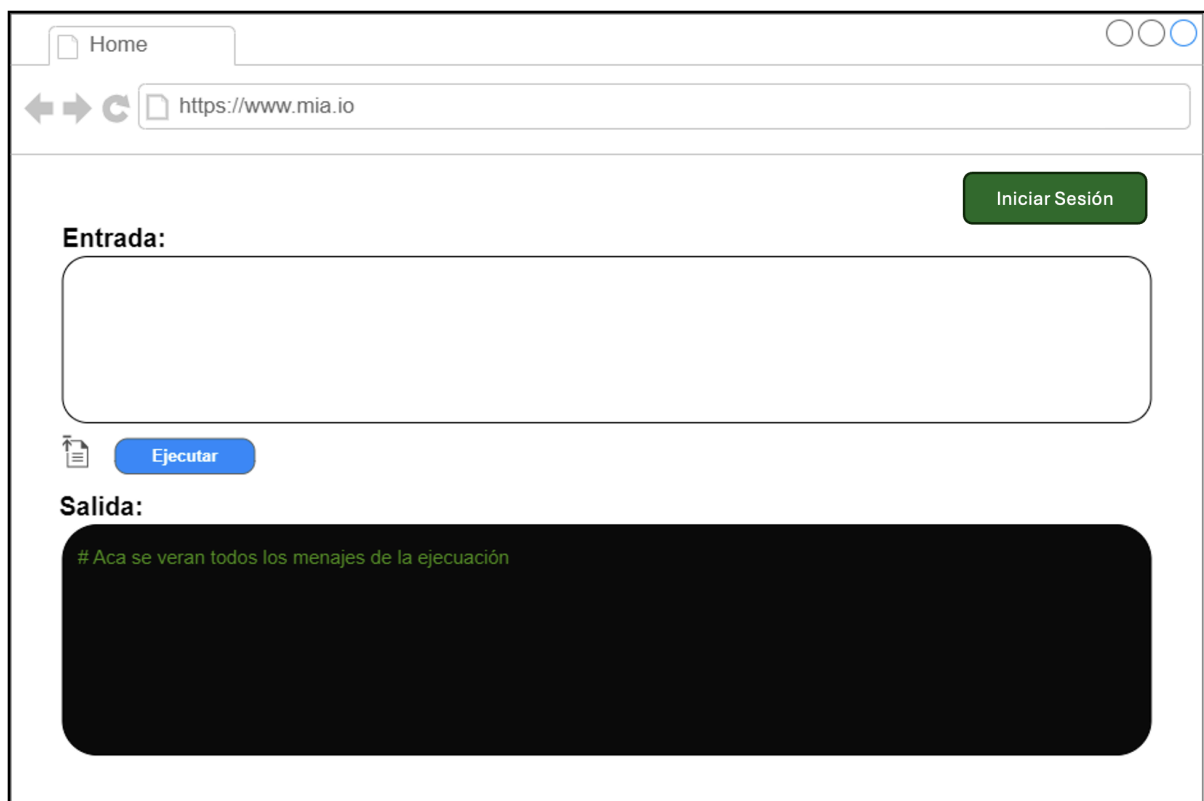
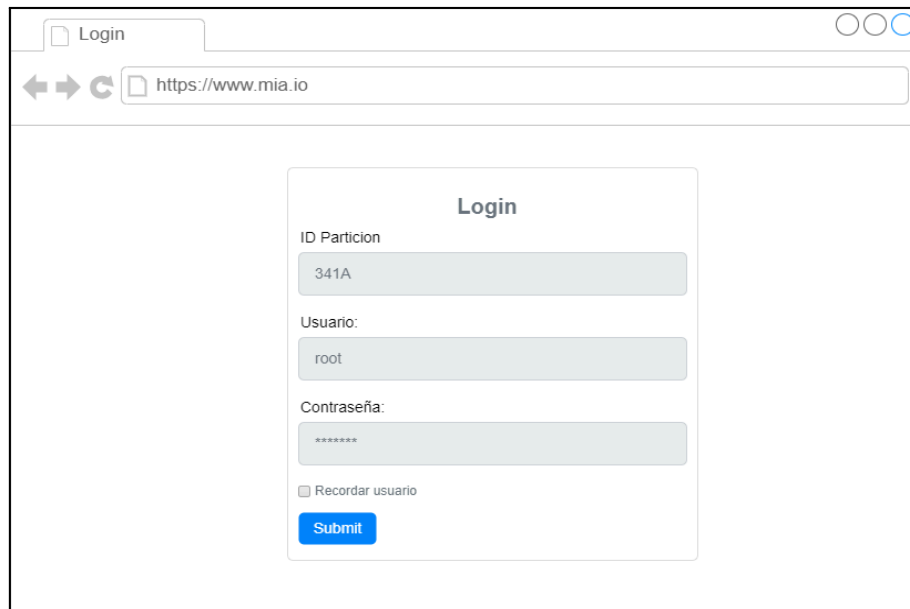


Figura 2 - Página Principal

Iniciar Sesión:

Se requiere una página para iniciar sesión ahora se realizará el login por medio de esta interfaz gráfica y no por comando, los comandos que requieren iniciar sesión previo a utilizarse son MKGRP, RMGRP, MKUSR, RMUSR y otros que usted considere necesarios.



The image shows a web browser window with a single tab titled 'Login'. The address bar displays 'https://www.mia.io'. The main content area features a centered login form. The form has a title 'Login' and three input fields: 'ID Particion' with the value '341A', 'Usuario' with the value 'root', and 'Contraseña' with masked characters '*****'. Below the password field is a checkbox labeled 'Recordar usuario'. At the bottom of the form is a blue button labeled 'Submit'.

Figura 3 - Página para iniciar sesión

Visualizador del Sistema de Archivos

Una vez logueado se ofrecerá una interfaz de usuario que permitirá explorar el sistema de archivos de manera visual, similar a un explorador de archivos, pero en modo solo lectura. Los usuarios podrán examinar la estructura de directorios y archivos sin la capacidad de modificar, agregar o eliminar elementos. La interfaz está diseñada para facilitar una navegación intuitiva y clara, mostrando la jerarquía de discos, particiones y carpetas de forma organizada. La visualización detallada permitirá a los usuarios acceder a la información necesaria con facilidad, proporcionando una agradable experiencia de usuario.

El proceso consta de los siguientes pasos:

1. **Selección de disco**

Se deberá poder seleccionar alguno de los discos creados mediante la siguiente vista:

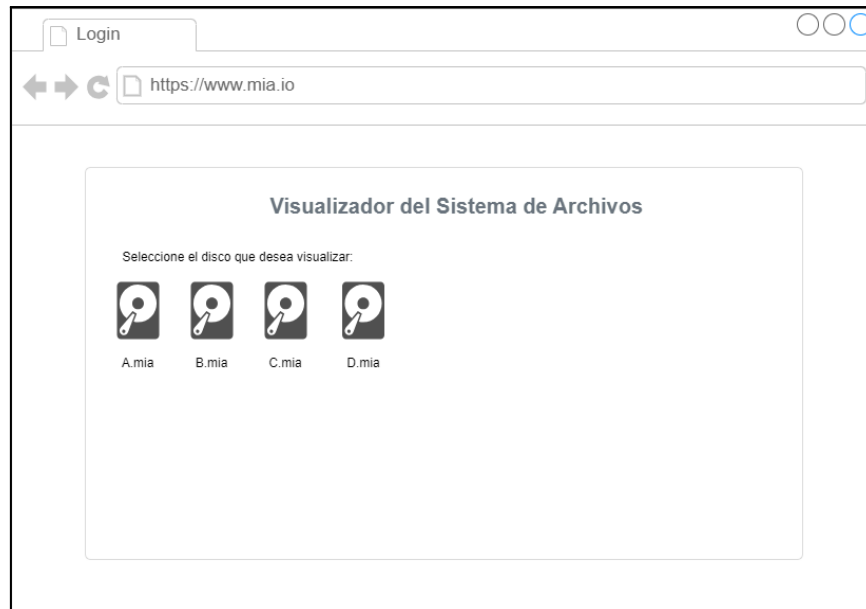


Figura 4 - Página inicio del visualizador y selección del disco

Cada disco también deberá mostrar su información básica, ya sea la capacidad del disco, el fit y las particiones montadas que cuenta.

2. Selección de partición

Se deberá poder seleccionar alguna de las particiones del disco seleccionado:

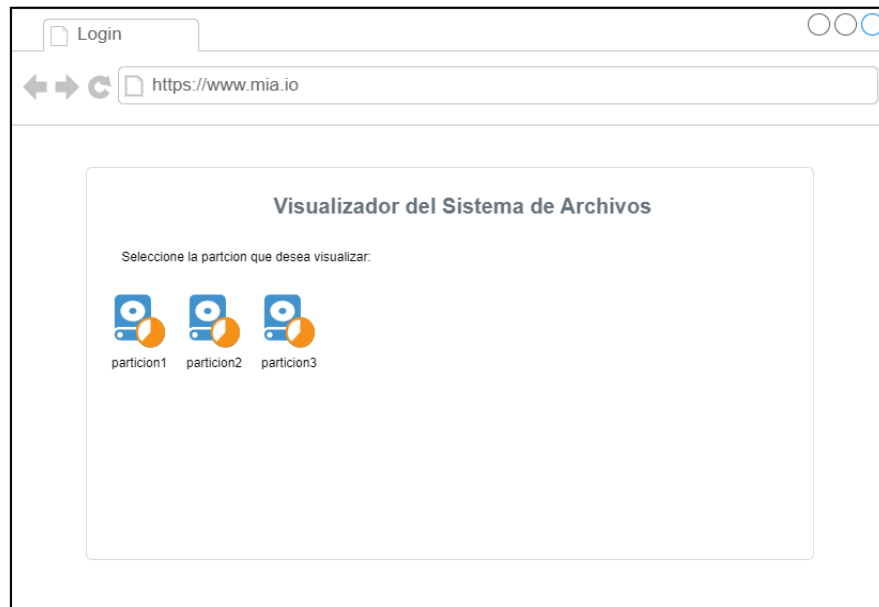


Figura 5 - Página selección de partición

Igualmente, cada una de las particiones deberá de mostrar su información básica, siendo **imprescindible que se muestre su tamaño**, fit y el estado de la misma.

3. Navegación con el sistema de archivos

Permite comenzar desde la carpeta raíz "/" cuando se utiliza el visualizador:

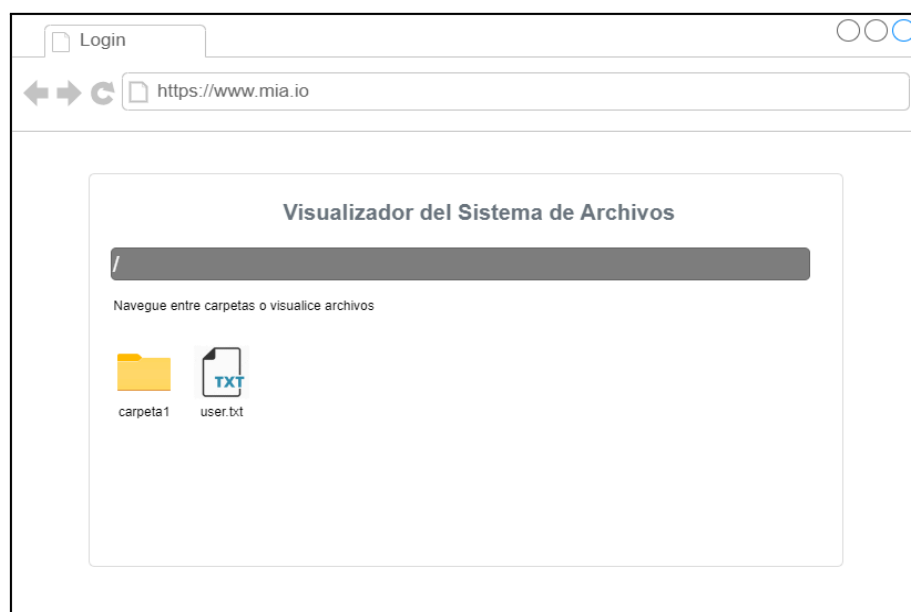


Figura 6 - carpeta root

Se deberá poder navegar por el sistema de archivos, donde la creación de archivos o carpetas será por medio de comandos en la página home y se podrán ver reflejados los cambios en esta pantalla. Tanto los archivos como las carpetas deben mostrar su información básica, incluyendo los permisos.

En esta pantalla, se podrá acceder a otras carpetas para navegar dentro de ellas.

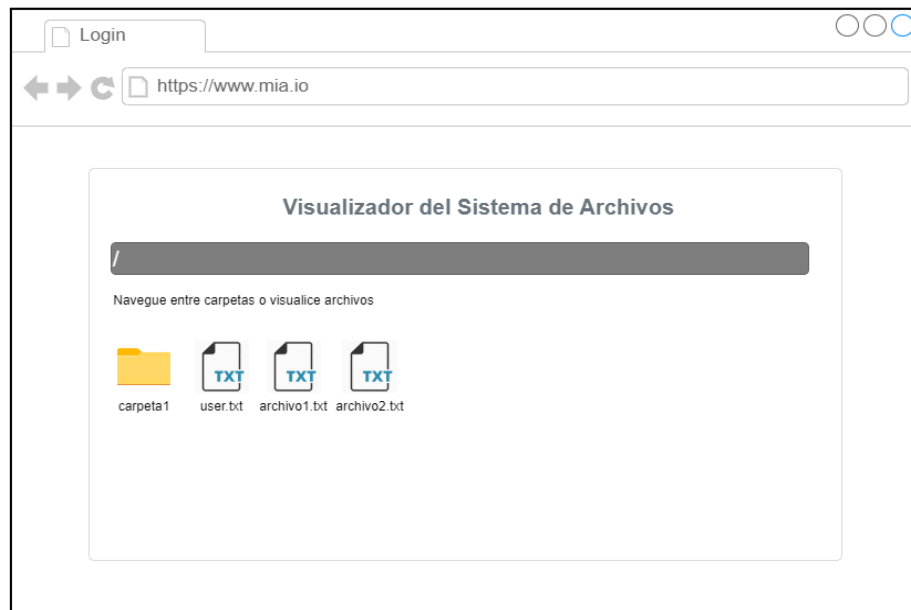


Figura 7 - Visualizador del sistema de archivos actualizado

Dentro de esta interfaz, también se debe poder mostrar el contenido de los archivos.



Figura 8 - Visualizador de archivo de texto

Al estar logueado, en la pantalla principal se deberá tener un botón para cerrar sesión. La sesión activa servirá para ejecutar comandos en la terminal que requieran de una misma (por ejemplo cat, mkdir, mkfile, entre otros que verán más adelante).

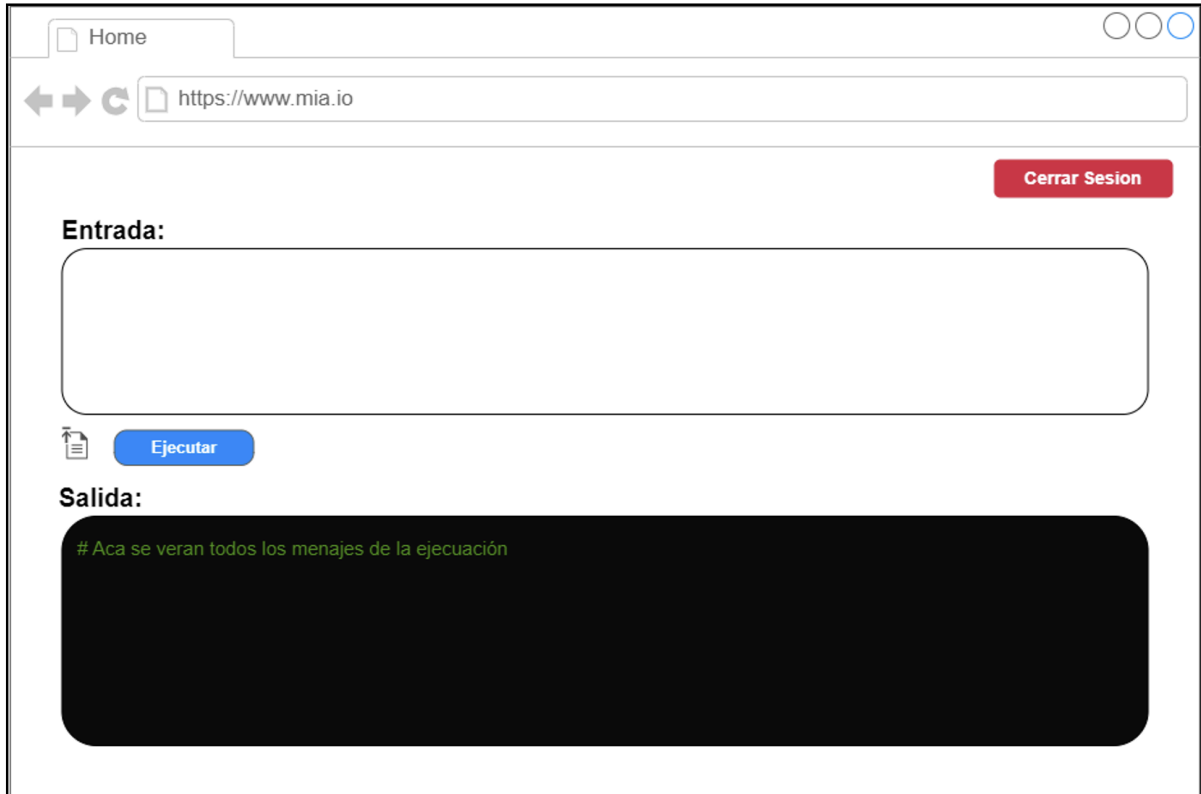
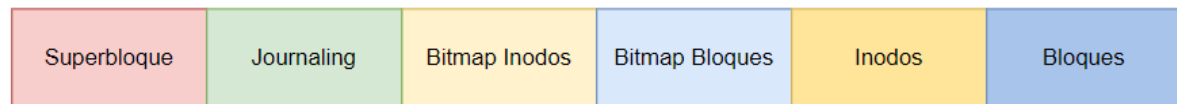


Figura 9 - Cerrar Sesión activa

Se agrega el manejo de Sistema de Archivos Ext3

EXT3

Para el caso del sistema de archivos EXT3, se deberán implementar las estructuras como se especifican a continuación. La estructura en bloques es la siguiente:



El número de bloques será el triple que el número de inodos. El número de inodos y bloques a crear se puede calcular despejando n de la primera ecuación y aplicando la función floor al resultado:

- $\text{tamaño_particion} = \text{sizeof}(\text{superblock}) + n * \text{sizeof}(\text{Journaling}) + n + 3 * n + n * \text{sizeof}(\text{inodos}) + 3 * n * \text{sizeof}(\text{block})$
- $\text{numero_estructuras} = \text{floor}(n)$

Observaciones:

1. sizeof es el tamaño de los Structs.
2. En el Bitmap de Bloques y Bloques se multiplica por tres debido a que existen tres tipos de bloque que son: bloques carpetas, bloques archivos y bloques de contenido.
3. El valor del Journaling se va a manejar con una constante de 50.

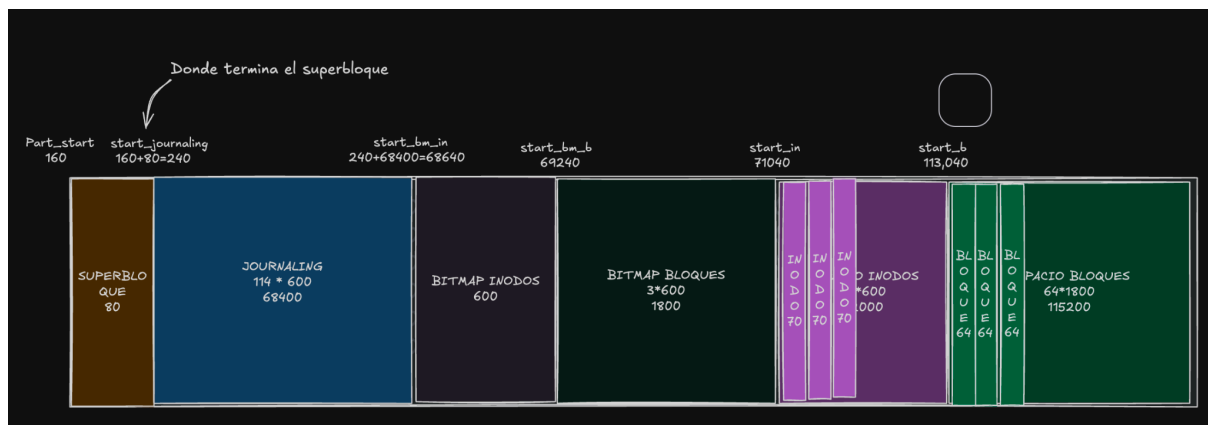


Figura 9 - Espacio de la partición 1

En este caso se tomó la estructura de bloques del sistema EXT2 que en este ejemplo se toma como si se hubiera asignado a la partición 1 con dicho sistema, **se aclara** que los bloques de Inodos y Bloques solo son reserva de espacio, ya que

dentro de ese espacio se escribirán múltiples estructuras contiguas según corresponda como por ejemplo la siguiente figura.

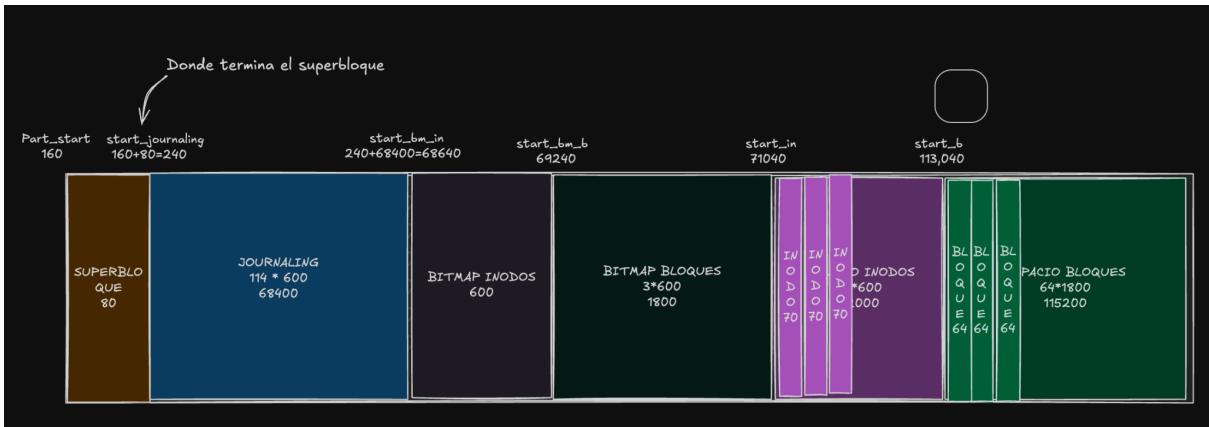


Figura 10 - Espacio de la partición 1

En la figura se muestra como dentro del espacio reservado de Inodos se escribieron múltiples estructuras de inodos los cuales se entrar a detalle a continuación de dichas estructuras.

Nueva estructura

Journal

Esta viene siendo la nueva estructura a agregar en este proyecto, siendo una bitácora de todas las acciones que se realizan en el sistema de archivos logueado, este cuenta con la siguiente estructura:

NOMBRE	TIPO	DESCRIPCION
j_count	int	Lleva el conteo del journal que es.
j_content	information	Contiene toda la información de la acción que se hizo

Information

Este es el contenido que va a llevar el journal, donde se cuenta tanto con la operación realizada como la fecha en la que se realizó; está compuesto de la siguiente manera:

NOMBRE	TIPO	DESCRIPCION
<code>i_operation</code>	char[10]	Contiene la operación que se realizó.
<code>i_path</code>	char[32]	Contiene el path donde se realizó la operación.
<code>i_content</code>	char[64]	Contiene todo el contenido (si es un archivo.)
<code>i_date</code>	float	Contiene la fecha en la que se hizo la operación.

Nuevos comandos que se deben agregar al Sistema de Archivos

En el Proyecto 1, actualmente se cuenta con una serie de comandos iniciales que permiten la ejecución de tareas básicas. Sin embargo, para mejorar la funcionalidad y hacer el proyecto más completo, se requiere la implementación de nuevas funciones. Para ello, es necesario desarrollar y agregar nuevos comandos que amplíen las capacidades del sistema.

Se solicita que se implementen las siguientes funciones, apoyándose en la creación de nuevos comandos o agregar ciertos parámetros en comandos existentes:

1. FDISK (ADD y DELETE)

En este comando se agregan los parámetros delete y add para tener las funcionalidades de agregar, quitar espacio o eliminar particiones.

Parámetro	Categoría	Descripción
<code>-size</code>	Obligatorio al crear	Este parámetro recibirá un número que indicará el tamaño de la partición a crear. Debe ser positivo y mayor a cero, de lo contrario se mostrará un mensaje de error.

-unit	Opcional	<p>Este parámetro recibirá una letra que indicará las unidades que utilizará el parámetro size. Podrá tener los siguientes valores:</p> <p>B: indicará que se utilizarán bytes.</p> <p>K: indicará que se utilizarán Kilobytes(1024 bytes)</p> <p>M: indicará que se utilizarán Megabytes(1024 * 1024 bytes).</p> <p>Este parámetro es opcional, si no se encuentra se creará una partición en Kilobytes. Si se utiliza un valor diferente mostrará un mensaje de error.</p>
-path	Obligatorio	<p>Este parámetro será la ruta en la que se encuentra el disco en el que se creará la partición. Este archivo ya debe existir, si no se mostrará un error.</p>
-type	Opcional	<p>Indicará que tipo de partición se creará. Ya que es opcional, se tomará como primaria en caso de que no se indique.</p> <p>Podrá tener los siguientes valores:</p> <p>P: en este caso se creará una partición primaria.</p> <p>E: en este caso se creará una partición extendida. L: Con este valor se creará una partición lógica.</p> <p>Las particiones lógicas sólo pueden estar dentro de la extendida sin sobrepasar su tamaño. Deberá tener en cuenta las restricciones de teoría de particiones: La suma de primarias y extendidas debe ser como máximo 4. Solo puede haber una partición extendida por disco.</p> <p>No se puede crear una partición lógica si no hay una extendida.</p> <p>Si se utiliza otro valor diferente a los anteriores deberá mostrar un mensaje de error.</p>
-fit	Opcional	<p>Indicará el ajuste que utilizará la partición para asignar espacio. Podrá tener los siguientes valores:</p> <p>BF: Indicará el mejor ajuste (Best Fit) FF: Utilizará el primer ajuste (First Fit) WF: Utilizará el peor ajuste (Worst Fit)</p> <p>Ya que es opcional, se tomará el peor ajuste si no está especificado en el comando. Si se utiliza otro valor que no sea alguno de los</p>

		anteriores mostrará un mensaje de error.
-delete	Opcional	<p>Este parámetro indica que se eliminará una partición. Este parámetro se utiliza junto con -name y -path. Se deberá mostrar un mensaje que permita confirmar la eliminación de dicha partición.</p> <p>Si la partición no existe deberá mostrar error. Si se elimina la partición extendida, deben eliminarse las particiones lógicas que tenga adentro.</p> <p>Recibirá los siguientes valores:</p> <p>Fast: Esta opción marca como vacío el espacio en la tabla de particiones.</p> <p>Full: Esta opción además marcar como vacío el espacio en la tabla de particiones, rellena el espacio con el carácter \0. Si se utiliza otro valor diferente, mostrará un mensaje de error.</p>
-name	Obligatorio	<p>Indicará el nombre de la partición. El nombre no debe repetirse dentro de las particiones de cada disco. Si se va a eliminar, la partición ya debe existir, si no existe debe mostrar un mensaje de error.</p>
-add	Opcional	<p>Este parámetro se utilizará para agregar o quitar espacio de la partición. Puede ser positivo o negativo. Tomará el parámetro units para las unidades a agregar o eliminar.</p> <p>En el caso de agregar espacio, deberá comprobar que exista espacio libre después de la partición. En el caso de quitar espacio se debe comprobar que quede espacio en la partición (no espacio negativo).</p>

Ejemplos:

```
#Elimina de forma rápida una partición llamada
Partición 1
fdisk -delete=fast -name="Particion1"
-path=/home/Disco1.dk
```

```
#Elimina de forma completa una partición llamada
Partición 1
```

```
fdisk -name=Particion1 -delete=full
-path=/home/Disco1.dk
```

```
#Quitan 500 Kb de Partición 4 en Disco4.dk
```

```
#Ignora los demás parámetros (size)
```

```
#Se toma como válido el primero que aparezca, en
este caso add
```

```
fdisk -add=-500 -size=10 -unit=K
-path="/home/misdiscos/Disco4.dk"
-name=Particion4
```

```
#Agrega 1 Mb a la partición Partición 4 del
Disco4.dk
```

```
#Se debe validar que haya espacio libre después de
la partición
```

```
fdisk -add=1 -unit=M -path="/home/mis
discos/Disco4.dk" -name="Particion 4"
```

2. UNMOUNT

Desmonta una partición del sistema. Se utilizará el id que se le asignó a la partición al momento de cargarla. Recibirá los siguientes parámetros:

Parámetro	Categoría	Descripción
-id	Obligatorio	Especifica el id de la partición que se desmontará. Si no existe el id deberá mostrar un error.

Ejemplos:

```
#Desmonta la partición con id 341A (En Disco1.dk)
```

```
umount -id=341
```

```
#Si no existe, se debe mostrar error umount -id=341A
```

Observaciones:

- En este paso debe cambiar el valor del correlativo de la partición a su estado inicial que es 0.

3. MKFS (FS)

Este comando realiza un formateo completo de la partición ya sea en EXT2 y EXT3, se formatea como ext2 por defecto si en caso el parámetro fs no está definido.

Tendrá los siguientes parámetros:

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
<code>-id</code>	Obligatorio	Indicará el id que se generó con el comando mount. Si no existe mostrará error. Se utilizará para saber la partición y el disco que se utilizará para hacer el sistema de archivos.
<code>-type</code>	Opcional	Indicará que tipo de formateo se realizará. Podrá tener los siguientes valores: Full : en este caso se realizará un formateo completo. Ya que es opcional, se tomará como un formateo completo si no se especifica esta opción.
<code>-fs</code>	Opcional	Indica el sistema de archivos a formatear. Podrá tener los siguientes valores: 2fs : Para el sistema EXT2 3fs : Para el sistema EXT3 Por defecto será ext2 .

Ejemplos:

```
#Realiza un formateo completo de la partición en el id 061A en
ext2
```

```
mkfs -type=full -id=061A
```

```
#Realiza un formateo completo de la partición que ocupa el id
062A en EXT3
```

```
mkfs -id=062A -fs=3fs
```

4. Remove

Este comando permitirá eliminar un archivo o carpeta y todo su contenido, si el usuario que actualmente está logueado tiene acceso al permiso de escritura sobre el archivo y en el caso de carpetas, eliminará todos los archivos o subcarpetas en los que el usuario tenga permiso de escritura. Si no pudo eliminar un archivo o subcarpeta dentro de la carpeta por permisos, no deberá eliminar nada dentro de esa carpeta ni la carpeta como tal.

Tendrá los siguientes parámetros:

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
<code>-path</code>	Obligatorio	Este parámetro será la ruta del archivo o carpeta que se eliminará. Si lleva espacios en blanco deberá encerrarse entre comillas. Si no existe el archivo o no tiene permisos de escritura en la carpeta o en el archivo, debe mostrarse un mensaje de error. Si no pudo eliminar algún archivo o carpeta no deberá eliminar los padres.

Ejemplos:

```
#Elimina el archivo a.txt, b.txt muestra error si no tiene
permiso
```

```
remove -path=/home/user/docs/a.txt
```

```
remove-path=/home/user/docs/b.txt
```

```
#Error por permisos
```

```
#Elimina la carpeta user y todo su contenido (docs, a.txt)
```

```
#Si el usuario no tuviera permiso de escritura sobre b.txt
```

```
#No debería eliminar las carpetas padres docs ni user, solo
```

a.txt

remove -Path=/home/user

remove -Path=/home/user

5. Edit

Este comando permitirá editar el contenido de un archivo para asignarle otro contenido. Funcionará si el usuario que actualmente está logueado tiene acceso al permiso de lectura y escritura sobre el archivo, si no debe mostrar error.

Tendrá los siguientes parámetros

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
-path	Obligatorio	Este parámetro será la ruta del archivo o carpeta que se editará. Si lleva espacios en blanco deberá encerrarse entre comillas.
-contenido	Obligatorio	Contiene la ruta a un archivo en el sistema operativo que contendrá el contenido que será Agregado a la edición.

Ejemplos:

#Modifica el archivo a.txt

edit -path=/home/user/docs/a.txt

-contenido=/root/user/files/a.txt

6. Rename

Este comando permitirá cambiar el nombre de un archivo o carpeta, si el usuario actualmente logueado tiene permiso de escritura sobre el archivo o carpeta.

Tendrá los siguientes parámetros:

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
<code>-path</code>	Obligatorio	Este parámetro será la ruta del archivo o carpeta al que se le cambiará el nombre. Si lleva espacios en blanco deberá encerrarse entre comillas. Si no existe el archivo o carpeta o no tiene permisos de escritura deberá mostrar un mensaje de error.
<code>-name</code>	Obligatorio	Especificará el nuevo nombre del archivo, debe verificar que no exista un archivo con el mismo nombre, de ser así debe mostrar un mensaje de error.

Ejemplos:

```
#Cambia el nombre del archivo a.txt a b1.txt
```

```
rename -path=/home/user/docs/a.txt -name=b1.txt
```

```
#Deberá mostrar error ya que el archivo b1.txt ya existe
```

```
rename -path=/home/user/docs/c.txt -name=b1.txt
```

7. Copy

Este comando permitirá realizar una copia del archivo o carpeta y todo su contenido hacia otro destino. Tendrá los siguientes parámetros:

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
-path	Obligatorio	Este parámetro será la ruta del archivo o carpeta que se desea copiar. Si lleva espacios en blanco deberá encerrarse entre comillas. Debe copiar todos los archivos y carpetas con todo su contenido, a los cuales tenga permiso de lectura. Si no tiene permiso de lectura, no realiza la copia únicamente de ese archivo o carpeta. Muestra un error si no existe la ruta
-destino	Obligatorio	Este parámetro será la ruta a donde se va a copiar el contenido. Debe tener permisos de escritura sobre esta carpeta, si no deberá mostrar un mensaje de error. De no existir la carpeta deberá mostrar un mensaje de error.

Ejemplos:

```
#/
# home #664
#user #664
#documents #664
#a.txt #664
#b.txt #224
#images #664
#Copia documents a images
copy -path="/home/user/documents" -destino="/home/images"
# b.txt no se copia debido a falta de permisos #/
# home #664
#user #664
#documents #664
#a.txt #664
#b.txt #224
#images #664
#documents #664
#a.txt #664
```

8. Move

Este comando moverá un archivo o carpeta y todo su contenido hacia otro destino. Si el origen y destino están dentro de la misma partición, solo cambiará las referencias, para que ya no tenga el padre origen sino, el padre destino, y que los padres de la carpeta o archivo ya no tengan como hijo a la carpeta o archivo que se movió. Solo se deberán verificar los permisos de escritura sobre la carpeta o archivo origen. Tendrá los siguientes parámetros:

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
-path	Obligatorio	Este parámetro será la ruta del archivo o carpeta que se desea mover. Si lleva espacios en blanco deberá encerrarse entre comillas. Debe mover todos los archivos y carpetas con todo su contenido, a los cuales tenga permiso de escritura. Si no tiene permiso de escritura, no realiza el movimiento Muestra un error si no existe la ruta.
- destino	Obligatorio	Este parámetro será la ruta de la carpeta a la que se moverá el archivo o carpeta. Debe tener permiso de escritura sobre la carpeta. Si lleva espacios en blanco deberá encerrarse entre comillas. Debe mostrar un mensaje de error si no tiene permisos para escribir o si la carpeta no existe.

Ejemplos:

```
#/
# home #664
# user #664
# documents #664
# a.txt #664
# b.txt #224
# images #664
#Copia documents a images
```



```

move -path="/home/user/documents" -destino="/home/images"
# mueve b.txt ya que solo se comprueban los permisos
#/
# home #664
# user #664
# images #664
# documents #664
# a.txt #664
# b.txt #224

```

9. Find

Este comando permitirá realizar una búsqueda por el nombre del archivo o carpeta. Permitirá los siguientes caracteres especiales:

CARÁCTER	DESCRIPCIÓN
?	Un solo carácter
*	Uno o más caracteres

Recibe los siguientes parámetros:

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
-path	Obligatorio	Este parámetro será la ruta de la carpeta en el que se inicia la búsqueda, deberá buscar en todo su contenido. Si lleva espacios en blanco deberá encerrarse entre comillas.
-name	Obligatorio	Debe tener permisos de lectura en los archivos que buscará. Indica el nombre de la carpeta o archivo que se desea buscar.

Ejemplos:

```
find -path = "/" -name=*
```

```
# Arbol Actual
```

```
# /
```

```
# |_ home #664
```

```
# |_ user #664
```

```
# |_ a.txt #664
```

```
# |_ b.txt #664
```

```
# |_ images #664
```

```
# |_ a.jpg #664
```

```
# |_ abcd.jpg #664
```

```
find -path = "/" -name = "?.*"
```

```
# El resultado del comando sería
```

```
# /
```

```
# |_ home
```

```
# |_ user
```

```
# |_ a.txt
```

```
# |_ b.txt
```

```
# |_ images
```

```
# |_ a.jpg
```

10. Chown

Cambiará el propietario de uno o varios archivos o carpetas. Lo podrá utilizar el usuario root en todos los archivos o carpetas y también lo podrán utilizar otros usuarios, pero solo sobre sus propios archivos. Recibirá los siguientes parámetros:

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
-path	Obligatorio	Este parámetro será la ruta en la que se encuentra el archivo o carpeta a la que se le cambiará el propietario. Si no existe la ruta deberá mostrar mensaje de error
-r	opcional	Debe tener permisos de lectura en los archivos que buscará. Indica el nombre de la carpeta o archivo que se desea buscar.

-usuario	Obligatorio	Nombre del nuevo propietario, si no existe debe mostrar error
-----------------	-------------	---

Ejemplos:

#Cambia el propietario de la carpeta

home recursivamente

```
chown -path=/home -r -usuario=user2
```

#Cambia los

permisos de la

carpeta home

```
Chown -path=/home
```

```
-usuario=user1
```

11. Chmod

Este comando elimina un usuario en la partición. Solo lo puede ejecutar el usuario **root**, si lo utiliza otro usuario deberá mostrar un error. Recibirá los siguientes parámetros

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
-path	Obligatorio	Este parámetro será la ruta en la que se encuentra el archivo o carpeta a la que se le cambiarán los permisos.
-ugo	Obligatorio	<p>Indica los permisos que tendrán los usuarios. Serán tres números:</p> <p>U G O [0-7][0-7][0-7]</p> <p>Usuario Grupo: usuarios dentro del grupo Otros: usuarios fuera del grupo. Cada número tendrá los valores desde el 0 al 7.</p> <p>Si el número está fuera de este rango se mostrará un error. A nivel de bits significan permisos para lectura, escritura y ejecución. Por ejemplo: El número 5 (101) indica permisos para leer y ejecutar.</p>

		<p>El número 7 indica permisos para las tres operaciones anteriores.</p> <p>El número 0 indica que no tendrá permisos para utilizar el archivo.</p>
-r	Opcional	<p>Indica que el cambio será recursivo en el caso de carpetas. El cambio afectará a todos los archivos y carpetas en la que la ruta contenga la carpeta especificada por el parámetro path y que sean propiedad del usuario actual</p>

Ejemplos:

```
#Cambia los permisos de la carpeta home recursivamente #Todos
los archivos o carpetas que tengan /home cambiarán #Por
ejemplo si existiera /home/user/docs/a.txt
#Cambiaría los permisos de las tres carpetas y del archivo
chmod -path=/home -r -ugo=764
```

```
#Cambia los permisos de la carpeta home
#Se debe comprobar que la carpeta home pertenezca al usuario
#actual, si no deberá mostrar un mensaje de error.
chmod -path=/home -ugo=777
```

Pérdida y Recuperación del Sistema de Archivos EXT3

Recovery File System

La recuperación del sistema se hará por medio del journaling y el superbloque. Se recuperará el sistema a un estado consistente antes del último formateo.

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
-id	Obligatorio	Especifica el id de la partición a la que se le simulara la recuperación del sistema.

```
#Recuperando el sistema de archivos EXT3 en una partición
recovery -id=061Disco1
```

Simulate System Loss

Este formatea los siguientes bloques de datos para simular un fallo en el disco (una partición en específica), una inconsistencia o pérdida de información. Se deberán limpiar los siguientes bloques con el carácter /0.

- Bloque de bitmap de Inodos
- Bloque de bitmap de Bloques
- Área de Inodos
- Área de Bloques.

Ejemplo:

```
#Simulando la pérdida del sistema de archivos EXT3 en una partición
```

```
loss -id=061Disco1
```

Journaling

Este mostrará la información de todas las transacciones realizadas mostrando la operación, la ruta, contenido, fecha y hora. Debe existir un apartado dentro de la interfaz gráfica para visualizarlo.

PARÁMETRO	CATEGORÍA	DESCRIPCIÓN
-id	Obligatorio	Indica el id de la partición que se utilizará.

```
# Journaling en una partición
```

```
journaling -id=061Disco1
```

Operacion	Path	Contenido	Fecha
mkdir	/	-	07/04/2025 19:07
mkfile	/ejemplo.txt	12345678901234567890	07/04/2025 19:07

Esta es solo la estructura que deberá llevar este reporte, pero queda a discreción del estudiante como mostrarlo por medio de la interfaz gráfica (se debe mostrar en la interfaz, no generar un reporte como el proyecto anterior).

Documentación

La documentación del proyecto consistirá en un **Manual Técnico** que brindará una visión completa del funcionamiento interno y uso del sistema de archivos EXT3 simulado en esta aplicación web. También deberá incluir una explicación de todo el despliegue que se realizó en AWS. Este manual deberá incluir los siguientes componentes:

- **Descripción de la Arquitectura del Sistema:** Una explicación detallada de la estructura y conexión entre los módulos frontend y backend, apoyada por diagramas que muestran cómo se integran y comunican estos componentes. En esta explicación también deberá entrar el despliegue a AWS junto con su arquitectura.
- **Explicación de las Estructuras de Datos:** Descripción de las estructuras de datos fundamentales utilizadas en el sistema, tales como el MBR, EBR, inodos y bloques. Este apartado debe aclarar la función de cada estructura dentro del sistema de archivos y cómo se organizan y gestionan dentro del archivo binario .mia.
- **Descripción de los Comandos Implementados:** Un listado de todos los comandos disponibles en la aplicación (por ejemplo, MOVE, REMOVE, UNMOUNT), junto con una descripción detallada de cada uno. Esto incluye ejemplos de uso, parámetros requeridos y una explicación de sus efectos específicos sobre las estructuras internas del sistema de archivos.

Este **Manual Técnico** brindará una guía clara para el diseño y operación del sistema de archivos EXT3 en la aplicación web, además de ofrecer una referencia detallada para el uso de los comandos y la visualización de sus efectos en el sistema.

Instrucciones de Entrega

El proyecto se entregará el **26/10/2025 hasta las 23:59 horas**. Se utilizará un repositorio de github para que suban su proyecto y se habilitará una opción en UEDI para que puedan subir el link de su repositorio, los auxiliares de cada curso deberán tener acceso a los repositorios respectivos en cualquier momento de la duración del laboratorio, si no se cuenta con acceso se anulara el proyecto, se recomienda que sea un repositorio privado para evitar copias. La impuntualidad será penalizada según la siguiente tabla. Se calificará el último commit que suban a la hora estipulada.

Minutos tarde en su entregable o Minutos transcurridos luego de la entrega en la que se notifica al auxiliar	Penalización
1-5 minutos	-5 Puntos
5-10 minutos	-10 Puntos
10-15 minutos	-15 Puntos
15-20 minutos	-20 Puntos

25-60 minutos	-25 Puntos
1 hora a 10 horas	-40 Puntos
10 horas en adelante	-100 Puntos (Anulado)

Nombre del repositorio: **MIA_2S2025_P2_carnet**

Usuarios de github de los auxiliares de cada sección:

1. **Sección A:** joshi20022021
2. **Sección B:** melladodaniel
3. **Sección C:** SaulCerezo
4. **Sección D:** kmsu

Requisitos Mínimos

Para tener derecho a calificación se deberá contar con requisitos mínimos los cuales son:

- Aplicación Web **COMPLETA** en la nube
- Ejecución Completa del Script

Consideraciones

Los mockups son sugerencias para la interfaz, la cual puede ser implementada por el estudiante según considere conveniente, siempre y cuando se mantenga el flujo.

El proyecto debe realizarse de forma individual, **Se utilizará software para la detección de copias, las copias tendrán una nota de 0 y serán reportadas a la escuela.**

- El lenguaje del backend a utilizar es Go. No se permite otro lenguaje.
- El framework para la página web queda a discreción del estudiante.
- El proveedor de nube es AWS. No se permite el uso de otro proveedor.
- El frontend debe estar desplegado desde un bucket S3
- El backend debe estar desplegado en una instancia EC2 con una distribución Linux.
- Se calificará basado en **el visualizador del sistema de archivos** en su mayor parte

4.2 Alcance del proyecto

Alcance obligatorio: Los estudiantes deberán desarrollar las siguientes funcionalidades mínimas para que el proyecto sea considerado completo:

- La aplicación web debe permitir la ejecución de comandos definidos para la administración de discos, particiones, archivos, carpetas y usuarios.
- El backend debe estar completamente implementado en Go y simular un sistema de archivos EXT2/EXT3 funcional utilizando archivos binarios `.mia`.
- El frontend debe contar con un área de entrada y salida de comandos, y permitir la carga de scripts (`.smia`).
- Se deben implementar correctamente los comandos: `mkdisk`, `fdisk`, `mount`, `mkfs`, `login`, `logout`, `mkfile`, `mkdir` e interfaz gráfica del sistema de archivos.
- Se debe entregar documentación técnica detallada y funcional (manual técnico, arquitectura, estructuras y comandos).

Alcance opcional: Estas funcionalidades no son obligatorias, pero pueden mejorar significativamente la calidad del proyecto y la nota final:

- Permitir exportar los reportes generados como archivos descargables desde la interfaz.

4.3 Requerimientos técnicos

Los estudiantes deberán utilizar el lenguaje de programación Go (Golang) para implementar toda la lógica del backend, incluyendo la simulación del sistema de archivos EXT2 y la ejecución de comandos. Para el desarrollo del frontend, se recomienda el uso de frameworks modernos como Angular, React o Vue.js, que permitan construir una interfaz web dinámica e intuitiva. Además, se utilizará Graphviz como herramienta principal para la generación de reportes visuales. El sistema deberá ejecutarse en la plataforma de AWS utilizando un bucket S3 para página estática para el frontend y una ec2 para el backend.

4.4 Entregables

Describe los productos concretos que se espera que los estudiantes entreguen al finalizar el proyecto. Pueden ser prototipos, informes técnicos, documentación, etc.

Ejemplo:

Tipo	Descripción
Sitio Web Funcional	Aplicación web ejecutable localmente que permita interactuar con el sistema de archivos simulado, ejecutar comandos y visualizar resultados.
Comandos Implementados	Implementación de la mayoría de los comandos definidos en el enunciado (especialmente aquellos relacionados a discos, usuarios y archivos).
Funcionalidades de Reportes	Generación de reportes visuales (MBR, inodos, bloques, árbol de directorios, etc.) usando Graphviz, accesibles desde el sitio web.
Comentarios y Scripts	Soporte para ejecutar scripts <code>.smia</code> con múltiples comandos y mostrar comentarios dentro del área de salida de comandos.
Documentación Técnica	Manual que incluya la arquitectura del sistema, explicación de las estructuras implementadas y ejemplos de uso de los comandos.
Repositorio GitHub	Proyecto completo subido a un repositorio privado de GitHub, con acceso habilitado para los auxiliares, siguiendo el formato requerido.
Manual de Usuario	Documento que explica cómo usar el sistema desarrollado, incluyendo capturas de pantalla, pasos detallados y resolución de problemas comunes.

5. Metodología

Desarrollo en Fases (SCRUM)

Se recomienda dividir el desarrollo en sprints (iteraciones de 1 semana), priorizando las funcionalidades esenciales:

Sprint 1: Revisión y complementación del proyecto 1

- Creación de discos virtuales (comando MKDISK).
- Manejo de particiones (comando FDISK).
- Montaje de particiones (comando MOUNT).
- Formateo (MKFS)

Sprint 2: Sistema de Archivos EXT3

- Formateo de particiones (comando MKFS).
- Implementación de inodos, bloques y bitmaps.

Sprint 3: Frontend y Comunicación con el Backend

- Desarrollo de la interfaz web (área de comandos, área de salida, carga de scripts).
- Conexión con el backend mediante APIs RESTful.
- Visualización de resultados de comandos.

Sprint 4: Revisión y prubeas

- Subirlo a la nube (AWS)
- Pruebas integrales del sistema.
- Depuración de errores.
- Redacción del Manual Técnico (arquitectura, estructuras, comandos).

6. Desarrollo de Habilidades Blandas

Además del enfoque técnico, este proyecto busca fortalecer competencias interpersonales que son esenciales en el ámbito profesional. A lo largo del desarrollo, los estudiantes deberán poner en práctica y mejorar las siguientes habilidades blandas:

- **Comunicación efectiva:**
Los estudiantes deberán expresar ideas técnicas de forma clara tanto en la documentación como al presentar el proyecto, fomentando la comprensión entre compañeros, tutores y evaluadores.
- **Colaboración y trabajo en equipo:**
Aunque el proyecto es individual, se promoverá la colaboración a través de foros, sesiones de consulta y espacios de retroalimentación entre estudiantes y tutores, simulando entornos reales de desarrollo.
- **Gestión del tiempo y responsabilidad:**
La entrega puntual del proyecto, la planificación de tareas y el cumplimiento de objetivos definidos impulsan una mayor disciplina y organización personal.
- **Pensamiento crítico y resolución de problemas:**
El estudiante deberá enfrentar desafíos técnicos de forma autónoma, buscando soluciones viables mediante el análisis, prueba y validación de su implementación.

6.1 Proyectos Individuales

Los proyectos individuales permiten a los estudiantes desarrollar autonomía y responsabilidad sobre su propio trabajo. En este tipo de proyectos, cada estudiante debe gestionar su tiempo, investigar, planificar y ejecutar todas las fases del proyecto de manera independiente.

6.1.1 Autogestión del Tiempo

Los estudiantes deben crear un cronograma personal para cumplir con los plazos establecidos. Esto les ayuda a mejorar su disciplina y capacidad de priorización, habilidades esenciales en cualquier entorno profesional.

6.1.2 Responsabilidad y Compromiso

En un proyecto individual, el estudiante asume la totalidad de las responsabilidades, desde la investigación hasta la entrega final. Esto fomenta el sentido de compromiso y permite una mayor personalización en la solución del problema planteado.

6.1.3 Resolución de Problemas

Trabajar de manera independiente impulsa a los estudiantes a buscar soluciones de forma

creativa y a enfrentarse a los desafíos sin depender del apoyo constante de otros. Esto fortalece su capacidad para resolver problemas de manera autónoma.

6.1.4 Reflexión Personal

Al concluir el proyecto, el estudiante realiza una autoevaluación, reflexionando sobre sus decisiones, lo aprendido y las áreas en las que podría mejorar. Esta práctica promueve el desarrollo continuo y el autoaprendizaje.

7. Cronograma

El cronograma describe las etapas clave del proyecto, los plazos estimados para cada una, y el proceso de asignación, elaboración y calificación de las tareas. Los estudiantes deberán seguir este plan para asegurar que el proyecto avance de manera organizada y cumpla con los plazos establecidos. Cada fase incluye la asignación de tareas, el tiempo estimado para su elaboración, y el momento de su calificación.

Tipo	Fecha Inicio	Fecha Fin
Asignación de Proyecto		
Backend completo		
Entrega No. 2 – Frontend funcional		
Entrega No. 3 – Reportes y ejecución de scripts		
Calificación		

8. Rúbrica de Calificación

8.1 Requisitos para optar a la calificación

Antes de la evaluación del proyecto, los estudiantes deben cumplir con los requisitos que se indiquen en esta sección.

Tema	Descripción	Cumple (Si/No)
Cumplimiento de la tecnología establecida	El backend debe estar desarrollado exclusivamente en Go (Golang) .	
Uso de herramientas o frameworks requeridos	El frontend debe estar implementado con un framework moderno como React , Angular o Vue.js . Debe usarse Graphviz para la generación de reportes.	
Gestión y entregas del proyecto	Todas las entregas parciales deben haberse completado según el cronograma. El proyecto debe estar en un repositorio privado de GitHub , con acceso habilitado.	
Documentación obligatoria	Se debe entregar un manual técnico con la arquitectura del sistema, estructuras utilizadas, comandos implementados y un manual de usuario con capturas.	
Pruebas y funcionalidad mínima	El sistema debe simular correctamente el sistema de archivos EXT2 y permitir la ejecución de comandos, generación de reportes y carga de scripts .	

8.2 Resumen de Puntuaciones

Área	Puntos Totales	Puntos Obtenidos
1. Conocimiento		
Funcionalidad del Proyecto	80	
Procedimiento y Desarrollo	10	
Sub-Total	90	
2. Habilidades		
Preguntas relacionadas al proyecto	5	
documentación	5	
Sub-Total	10	
TOTAL	100	
1. Conocimiento		

8.3 Detalle de la Calificación

No.	Requerimientos Mínimos	Cumple	No Cumple
1.	Aplicación Web en la nube		
2.	Ejecución Completa del Script		

Conocimiento

Parte 1: Comandos Previos

Descripción de Ponderación	Valor	Observación	Punteo
MKFS	1		
LOGIN	1		
LOGOUT	1		
MKDIR	1		

MKFILE	1		
TOTAL	5		

Parte 2: Aplicacion web en la nube

Descripción de Ponderación	Valor	Observación	Punteo
Instancias EC2	10		
Instancia con distribución linux	4		
Ejecucion del backend en GO	3		
Security Groups	3		
Almacenamiento S3	10		
Bucket Sitio Web Estático	4		
Permisos de S3	3		
Link del Sitio Web	3		
Aplicación Web	20		
Iniciar Sesion	2		
Seleccion del disco	3		
Seleccion de partición	3		
Seleccion de Carpetas	4		
Seleccion de Archivos y visualizacion de su contenido	4		
Manejo de Sesion del Usuario	2		
Cerrar Sesion	2		
TOTAL	40		

Parte 3: Nuevos Comandos

Descripción de Ponderación	Valor	Observación	Punteo
FDISK	5		
ADD	2.5		
DELETE	2.5		
UNMOUNT	2.5		

MKFS	2.5		
FS (EXT2 y EXT3)	2.5		
REMOVE	2.5		
EDIT	2.5		
RENAME	2.5		
COPY	2.5		
MOVE	2.5		
FIND	2.5		
CHOWN	2.5		
CHMOD	2.5		
TOTAL	30		

Parte 4: Journaling

Descripción de Ponderación	Valor	Observación	Punteo
RECOVERY	5		
LOSS	5		
JOURNALING	5		
TOTAL	15		

Habilidades

Parte 5: Documentación y conocimientos

Descripción de Ponderación	Valor	Observación	Punteo
Documentación	5		
Pregunta 1	2.5		
Pregunta 2	2.5		
Subtotal de la Parte 6	10		

Sumatoria total

Subtotal Parte 1	5		
Subtotal Parte 2	40		
Subtotal Parte 3	30		
Subtotal Parte 4	15		

Subtotal Parte 5	10		
Penalización por entrega tardía	%		
Penalización por veracidad del código	40%		
TOTAL	100		

8.4 Valores

En el desarrollo del proyecto, se espera que cada estudiante demuestre honestidad académica y profesionalismo. Por lo tanto, se establecen los siguientes principios:

1. **Originalidad del Trabajo**
 - Cada estudiante o equipo debe desarrollar su propio código y/o documentación, aplicando los conocimientos adquiridos en el curso.
2. **Prohibición de Copias y Plagio**
 - Si se detecta la copia total o parcial del código, documentación o cualquier otro entregable, la calificación será de **0 puntos**.
 - Esto incluye la reproducción de código entre compañeros, la reutilización de proyectos de semestres anteriores o el uso de código externo sin la debida referencia.
3. **Uso Responsable de Recursos Externos**
 - El uso de bibliotecas, frameworks y ejemplos de código externos está permitido, siempre y cuando se referencian correctamente y se comprendan plenamente. (Consultar con el catedrático su política)
4. **Revisión y Detección de Plagio**
 - Se podrán utilizar herramientas automatizadas y revisiones manuales para identificar similitudes en los proyectos.
 - En caso de sospecha, el estudiante deberá justificar su código y demostrar su desarrollo individual o en equipo. Si este extremo no es comprobable la calificación será de **0 puntos**.

Al detectarse estos aspectos se informará al catedrático del curso quien realizará las acciones que considere oportunas.

8.5 Comentarios Generales
