Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Estructuras de Datos
Proyecto #2
Ingenieros

- Ing. Luis Espino
- Ing. Jesús Guzmán
- Ing. Álvaro Hernández

Auxiliares

- Dennis Masaya
- Ricardo Cutz
- Antonio Hernández

EDD DRIVE

Objetivos

1. Familiarizarse con los lenguajes de programación Java.
2. Comprender y desarrollar distintas estructuras de datos no lineales como lo son árboles, grafos.
3. Familiarizarse con tecnologías de Cifrado y Compresión.
4. Definir algoritmos de búsqueda y recorrido.
5. Familiarizarse con la herramienta Graphviz para la generación de reportes de estructuras gráficos.

Definición del problema

La facultad de ingeniería desea tener una forma de almacenar archivos importantes, pero actualmente no cuentan con un sistema que se ajuste a sus necesidades por lo que se plantea la solución de crear un sistema propio. Este sistema debe de ser capaz de ser utilizado en cualquier sistema operativo por lo que se plantea la posibilidad de que el sistema sea en ambiente de escritorio. El sistema debe tener un funcionamiento similar a Google Drive con la característica que la Universidad de San Carlos sea propietario del mismo.

Descripción

Según los requerimientos antes mencionados se desea que usted como estudiante de ingeniería en sistemas desarrolle la aplicación para el manejo de archivos para la Universidad de San Carlos de Guatemala de la facultad de Ingeniería. El sistema denominado como EDD DRIVE debe de llevar el control de usuarios, donde cada uno de los cursos de la carrera de ingeniería en sistemas debe de contar con un espacio de almacenamiento donde se puede subir, crear y eliminar carpetas, así como también archivos. Los usuarios también tendrán la opción de modificar los nombres de carpetas y archivos ya creados. La aplicación debe de ser responsiva y amigable al usuario. A continuación, se da una explicación más detallada de lo solicitado.

Descripción del Sistema

Usuarios

Registro de Usuarios e Inicio de Sesión

El sistema debe de contar con la funcionalidad de poder crear usuarios utilizando un formulario en el sistema donde se pueda ingresar el nombre de usuario **ÚNICO** y una contraseña de al menos **8 CARACTERES**. Los usuarios ya registrados al sistema deben de poder ingresar al sistema mediante una pantalla de inicio de sesión, si el usuario no existe debe de poder notificar en pantalla que las credenciales no son correctas.

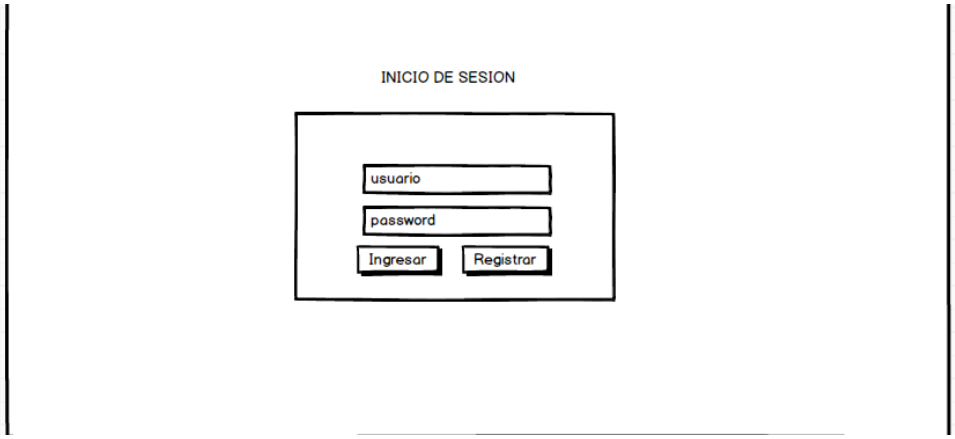


Figura 1. Sugerencia de Inicio de Sesión

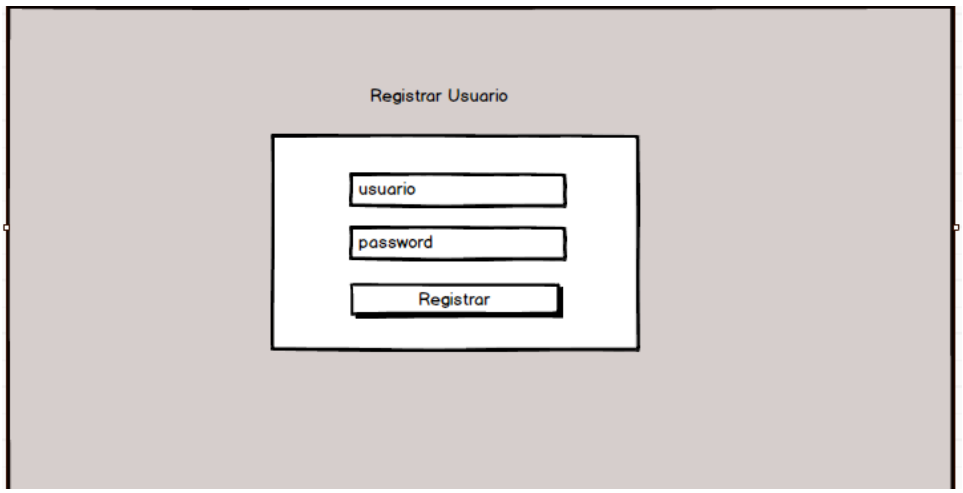


Figura 2. Sugerencia de Pantalla de Registro

Para el manejo de los usuarios la estructura de datos que se manejará sera una tabla hash (Descrita su funcionalidad más adelante)

Carga Masiva de Usuarios

En el sistema por defecto debe de existir un usuario: Admin, con la contraseña Admin. Este usuario al loguearse al sistema va a poder cargar un archivo csv (carga masiva) para poder agregar usuarios al sistema. **Es indispensable contar con esta funcionalidad para validar el comportamiento de la tabla hash. Debe de validar las condiciones de que el Usuario sea único y que la contraseña tenga al menos 8 caracteres**. En la pantalla de carga masiva, debe de mostrar el número de usuarios que se insertaron con éxito. Para los que no cumplieron con las caracteristicas descritas, debe de mostrar una tabla con el usuario y razón del porque no se logró insertar. El formato del archivo CSV será el siguiente, tendrá los encabezados “Usuario” y “Password”:

	A	B
1	Usuario	Password
2	Juan123	MiContrasena123
3	Ricardo123	OtraContrasena
4	Pedro456	Yhsnkdhfmndf
5	Dennis	EddSeccionC
6	Antinio24	EddSeccionB

Figura 3. Carga de usuarios al sistema

Encriptación de contraseña

Todos los usuarios que se almacenen en el sistema de archivos de deben de tener encriptación en su contraseña por lo cual deben de hacer uso de **sha256** nativo de java

Manejo de Carpetas

Los usuarios del sistema podrán crear cuantas carpetas deseen en todos los cursos, con el nombre que quieran y con cuantas carpetas y archivos quieran dentro de ellas. Cuando un usuario se crea, de una vez se le crea una carpeta que será la raíz de todo su directorio. Para manejar la ruta de las carpetas se hará como el sistema operativo Linux las maneja, a continuación, un ejemplo:

```
/
  Mis_Documentos/
    Primer_Semestre
        Mate1/
        Mate2/
        MateInter1/
    Segundo_Semestre/
        EDD/
        Compiladores/
    Tercer_Semestre/
        Orga/

  Mis_Fotos/
    Vacaciones_2018/

    Vacaciones_2019/

  Mis_Videos/
    Videos_Musicales/
        Rock/

  Mis_Descarga
    Programas/
        *Chrome.exe
```

Figura 4. Arbol de Archivos

- Ruta para carpeta **EDD:** /Mis_Documentos/Segundo_Semestre/EDD
- Ruta para carpeta **ROCK:** /Mis_Videos/Videos_Musicales/Rock
- Ruta para la **RAIZ:** /

Las carpetas son administradas a través de un grafo dirigido.

Manejo de Archivos

Cada usuario puede crear sus archivos en cualquier directorio que desee. Cada uno de los archivos debe de contener un nombre, extensión (cualquier extensión), contenido (una cadena solamente) y timestamp de creación del archivo.

Los archivos en cada carpeta deben de ser manejados mediante un árbol AVL, con el criterio de orden en el árbol, utilizar el nombre del archivo.

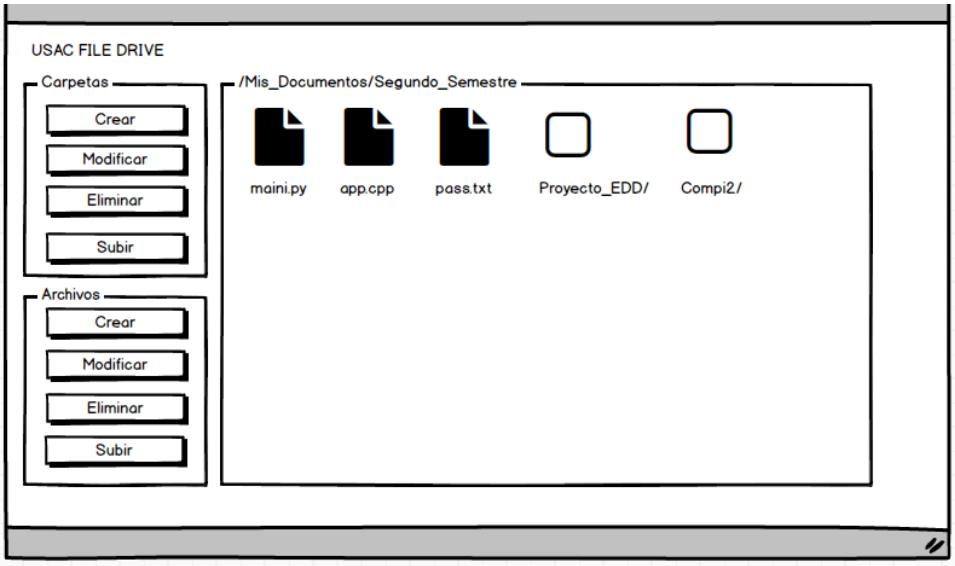


Figura 5. Sugerencia de UI

Eliminación

Los usuarios pueden eliminar los archivos que deseen. También pueden eliminar carpetas enteras (menos la raíz de cada curso), y cuando se eliminan carpetas, se eliminan todos los archivos y carpetas que estaban dentro de la carpeta eliminada.

Modificación

Los usuarios pueden cambiarle el nombre a los archivos y carpetas que quieran. Las estructuras que contengan al objeto modificado tienen que adecuarse al cambio. El cambio no se acepta si ya existe un archivo o carpeta con el mismo nombre. La modificación incluye el poder cambiar la cadena que representa el contenido del archivo (aplica solamente a archivos).

Carga de Archivos

Para simular la carga de archivos, ser hará el uso de un csv que trae una lista de archivos a crear que deben de ser insertados dentro del árbol de archivo AVL del nodo de directorio en donde se desean crear. El usuario debe de estar posicionado en la carpeta donde desea crear los archivos. La aplicación debe de permitir visualizar el contenido del archivo, **EL CONTENIDO DEL ARCHIVO SIEMPRE SERA UNA CADENA**.

Archivo	Contenido
Main.py	"Python Script"
Header.h	"este es un archivo cabecera"
Hola.txt	"Hola mundo!!!!"
chrome.exe	"Este es un archivo ejecutable"

Figura 6. CSV de archivos a crear

Sobreescritura

Cuando se cargan archivos y ya existe un archivo con el mismo nombre y extensión, el sistema deberá de sobrescribir el archivo viejo, pero se deberá de mostrar una notificación de confirmación antes de hacer el cambio.

Descarga de Archivos

Para simular la descarga de archivos se hará la escritura de un archivo en disco (con el nombre del archivo en el sistema) que contenga el contenido del archivo. La aplicación debe de abrir el archivo de texto para visualizar el contenido.

Compartir Archivos

La aplicación debe ser capaz de poder compartir archivos. Para agregar esta funcionalidad el usuario que desea compartir el archivo debe de ingresar el nombre de usuario destino a quien desea compartir, junto con la ruta y nombre del archivo que se desea, si el usuario a quien se le desea compartir el archivo existe, entonces se creará un nuevo archivo con el mismo contenido, nombre y extensión en la carpeta raíz de este, en caso de no existir el usuario debe de mostrar una notificación en pantalla de que no se pudo realizar la operación. A continuación, un ejemplo: el usuario Ricardo desea compartir un archivo.txt a Antonio:

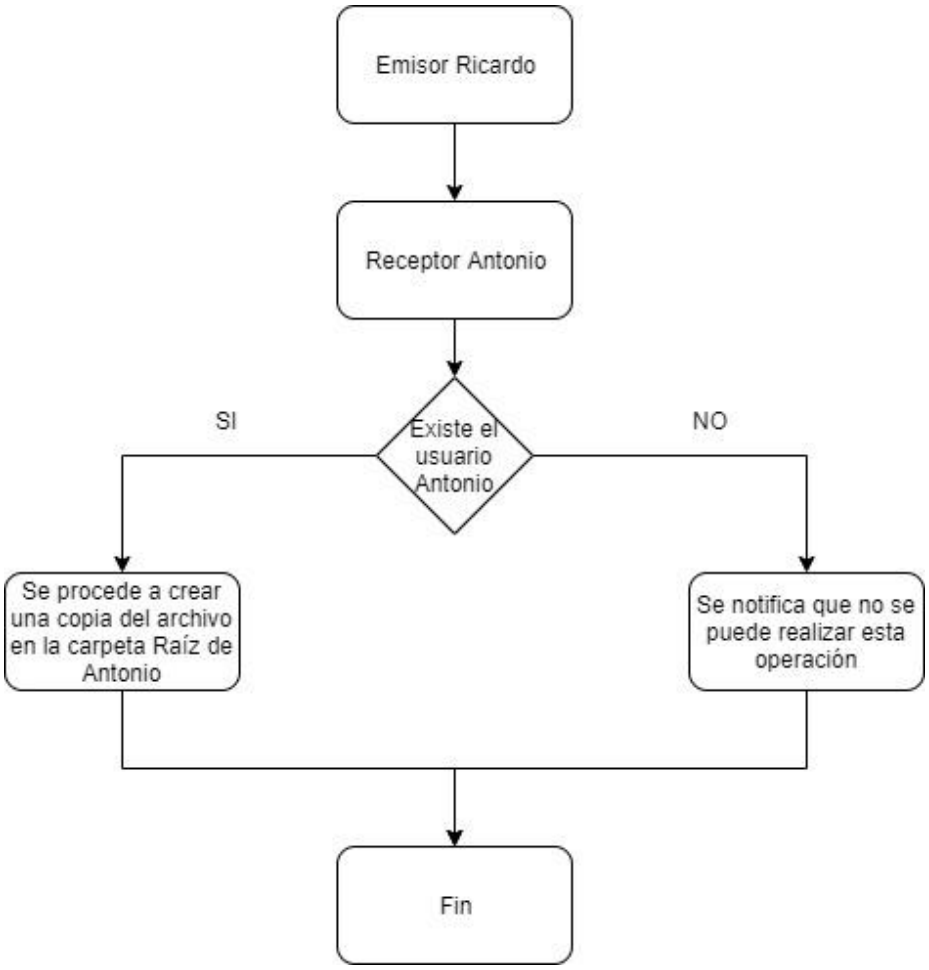


Figura 7. Flujo de compartir archivos

Historial de Cambios

Todo el sistema debe de contar con una bitácora de cambios. Esta almacenará un registro cada vez que se crea o elimina o modifica una carpeta, cuando se sube un archivo, cuando se modifica un nombre de archivo, y así para todas las operaciones del sistema. La bitácora debe de mostrar fecha, hora, operación realizada (compartir, eliminar, modificar, etc.…) y usuario que ejecutó la operación.

La bitácora se manejará utilizando un PILA, así se podrá visualizar desde el cambio más recienta hasta el cambio más antiguo.

Resumen de Estructuras

- Para el manejo de los usuarios se hará uso de una tabla HASH en donde se deben de almacenar en cada nodo el nombre del usuario, su contraseña y un apuntador a su sistema de archivos (Matriz de adyacencia). Se debe comenzar con un tamaño de 7 espacios libres en la tabla la cual incrementara su tamaño hasta el siguiente número primo cuando el porcentaje de utilización de la misma llegue a un 75%. El valor de referencia para insertar en la tabla hash será el nombre del usuario el cual se deberá codificar para obtener un valor entero el cual servirá como parámetro para una **función hash por división** y así encontrar el índice en el cual se insertará el nodo.
- Para las colisiones dentro de la tabla se utilizará una resolución **direccionamiento abierto** por salto al cuadrado el cual tomará el valor hash calculado y lo elevará al cuadrado e intentará insertar el nodo en la nueva posición en caso el nuevo hash sobrepase el arreglo se debe de empezar a recorrer el arreglo desde el inicio con los saltos restantes hasta que se encuentre uno vacío.
- Para el manejo del grafo dirigido se tendrá una lista enlazada simple con todas las carpetas y su información, es aquí donde por cada nodo de esta lista se manejará la referencia hacia su árbol AVL de archivos, es decir un apuntador a donde comienza esta estructura.
- Para el manejo del grafo dirigido de carpetas(directorios) se hará uso de una matriz de adyacencia en donde los nodos de la matriz contendrán la información relevante de la relación entre ambos directorios (nombre, fecha de creación, etc.) por directorio, además de una referencia a su apuntador de su árbol AVL de archivos del directorio hijo (Columnas) y las cabeceras tanto de filas como columnas contendrán el nombre del directorio el cual servirá también para que la matriz esta ordenada, estas cabeceras serán la copia tal cual de la lista donde están almacenados las carpetas (lista enlazada).
- El AVL contiene todos los archivos del directorio actual. Cada nodo guardará tanto el nombre, extensión, contenido y timestamp de creación.
- Para la bitácora de todo el sistema se debe de usar una PILA. La estructura debe de mostrar todos los datos: fecha, hora, operación y usuario que ejecutó la operación.

Reportes

En la aplicación debe de haber un apartado para visualizar los reportes en el navegador. **No se permite buscar el reporte en carpetas al momento de la calificación ni abrirlas con el visor de imágenes**. Los reportes requeridos son los siguientes: NOTA: **TODOS LOS REPORTES DEBEN SER REALIZADOS CON GRAPHVIZ**

Tabla Hash

La tabla hash se debe visualizar:

- Nombre de usuario

- Contraseña
- Índice que ocupa en la tabla hash
- Timestamp de creación de usuario

Ejemplo

Tabla Usuarios

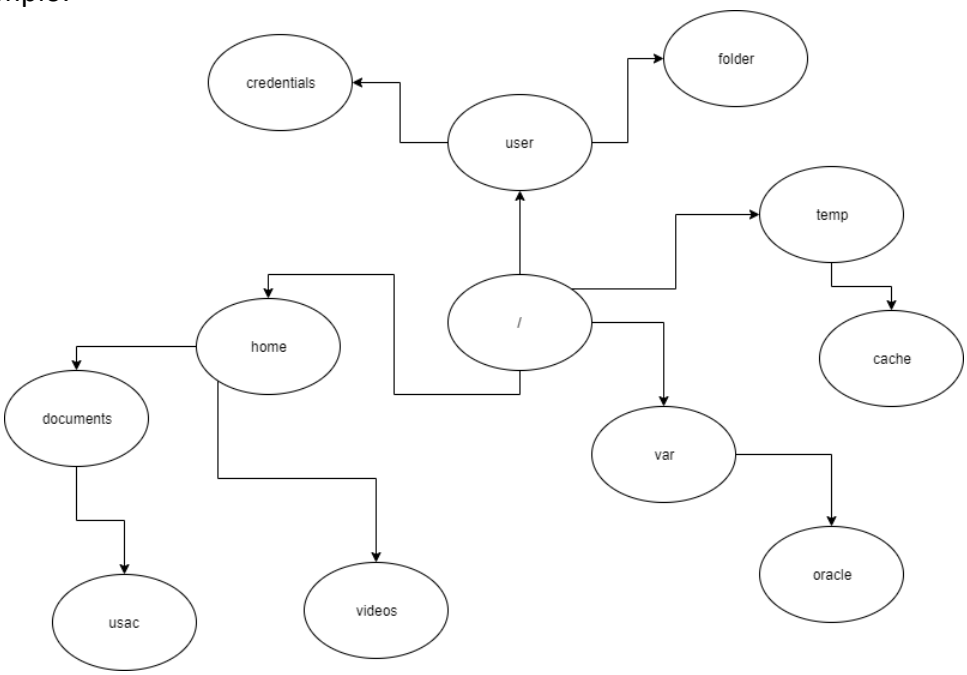
0)	
1)	Nombre: Brandon Alvarez Contraseña: 34547653
2)	Nombre: Antonio Gramajo Contraseña: 24796754
3)	Nombre: Abel Gutierrez Contraseña: 124652211
4)	Nombre: Alumno c Contraseña: BEDDS219
5)	
6)	Nombre: Jimmy Garcia Contraseña: hola1235
7)	Nombre: Jose Lopez Contraseña: COntrsena
8)	Nombre: Maite Gomez Contraseña: 45578787
9)	Nombre: Nery Galvez Contraseña: 12129329
10)	
11)	Nombre: Alumno b Contraseña: Gato5775
12)	Nombre: Fernando Hernandez Contraseña: 32321299
13)	Nombre: Joel Perez Contraseña: 23168643
14)	Nombre: Juan Perez Contraseña: 89764432
15)	
16)	
17)	Nombre: Alan Hurtarte Contraseña: 23546576
18)	Nombre: Alumno a Contraseña: Perro1211

Grafo

Grafo representativo de las relaciones entre los directorios.
Se debe de mostrar por nodo lo siguiente:

- Nombre del directorio
- Cantidad de carpetas del directorio

Ejemplo.



Matriz de adyacencia

La cual será la representación gráfica de cómo están almacenados los cursos en el grafo.

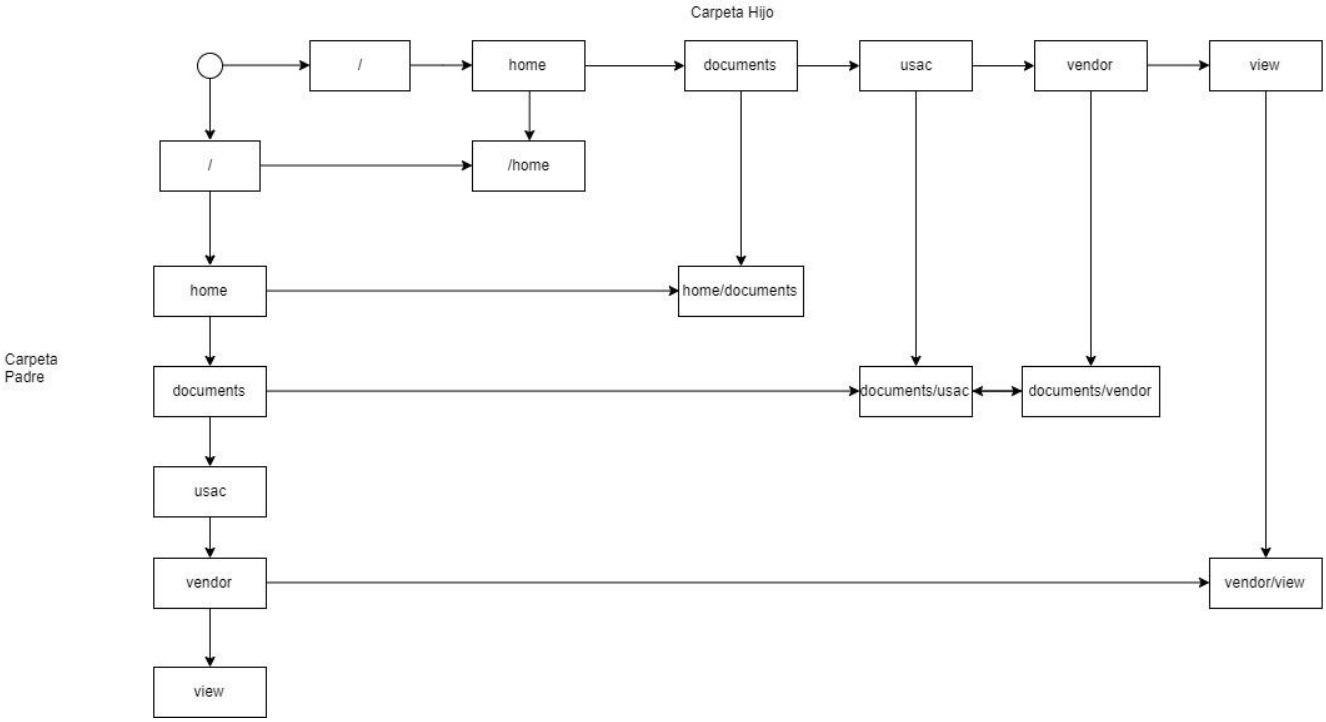
Se debe de mostrar por cabeceras Filas y Columnas la siguiente información:

- Nombre de la carpeta

Se debe de mostrar por nodo de la matriz de adyacencia.

- Toda la información de ambos directorios relacionados.

Ejemplo.

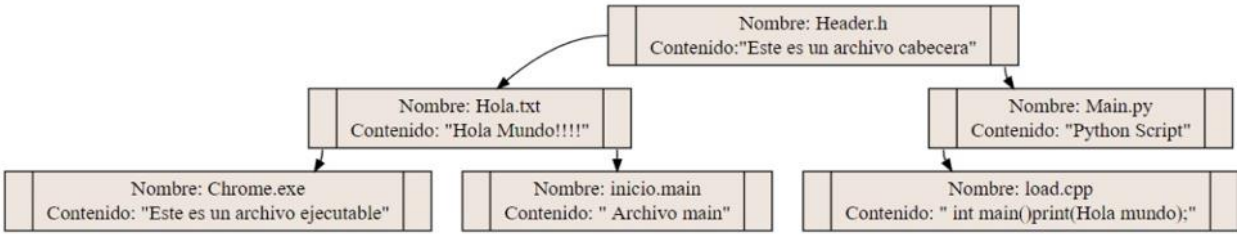


Árbol AVL

Se debe de seleccionar una carpeta en específico y se generara el árbol con los archivos en ese directorio.

Se debe de mostrar por nodo

- Nombre archivo
- Contenido
- Factor de equilibrio
- Altura
- Timestamp de creación archivo
- Propietario del archivo



Bitácora

Pila con todos los cambios hechos en el sistema el cual por nodo debe mostrar

- Descripción del cambio realizado
- TimeStamp
- Usuario quien realizo el cambio

Restricciones

Las estructuras deben de ser desarrolladas por los estudiantes sin el uso de ninguna librería o estructura predefinida en el lenguaje a utilizar Los reportes son esenciales para verificar si se trabajaron correctamente las estructuras solicitadas, Si no se tiene el reporte de alguna estructura se anularán los puntos que tengan relación tanto al reporte como a la estructura en cuestión.

Penalizaciones

1. Falta de seguimiento de desarrollo continuo por medio Github tendrá una penalización del 15% (mínimo 2 commits por semana).
2. Falta de seguimiento de instrucciones conforme al método de entrega (nombre del repositorio) tendrá una penalización del 5%.
3. Falta de puntualidad conforme a la entrega tendrá una penalización de la siguiente manera:
    a. 0-6 horas – 20%.
    b. 6-12 horas – 40%
    c. 12-18 horas 60%
    d. 18-24 horas – 80%.
    e. Pasados 24 horas tendrá una nota de 0 y no se calificará.

Observaciones

1. Lenguaje a utilizar: **Java**
2. Herramienta para desarrollo de reportes gráficos: **Graphviz.**
3. La entrega se realizará por medio de: **Github**, cada estudiante deberá crear un repositorio con el nombre: **EDD_2S2019_PY2_#carnet**, y agregar a su auxiliar correspondiente como colaborador del mismo, para poder analizar su progreso y finalmente a partir del mismo repositorio realizar la calificación correspondiente.
   Dennis Masaya: https://github.com/Dennis201503413
   Ricardo Cutz: https://github.com/ricardcutzh
   Antonio Hernández: https://github.com/fernando29hernandez
4. Además de tener a su auxiliar como colaborador del repositorio para tener un control y orden de las personas que entreguen deberán de colocar el Link de su repositorio en la Tarea que cada auxiliar asignara en su classroom correspondiente.
5. Fecha y hora de entrega: **viernes 15 de noviembre, a las 23:59 horas**.
6. **Copias serán penalizadas con una nota de 0 y castigadas según lo indique el reglamento.**