**III Proyecto Programado – Sistemas Operativos**

**Tercer proyecto programado: Crear un repositorio con control de versiones**

El proyecto consiste en diseñar un programa en Phyton que sirva como una interfaz (línea de comandos o gráfico), el cual acepta comandos de usuario (crear archivo, eliminar archivo, abrir archivo, guardar archivo). Este proyecto debe ser desarrollado en un ambiente Linux o Windows.

La interfaz de debe proporciona la posibilidad de incluir usuarios, definir los permisos de cada usuario.

**Carpeta repositorio**

El primer paso es crear un usuario en el sistema operativo para la aplicación, crear una carpeta (repositorio (raiz)), a este repositorio solo el usuario creado va a tener premiso de ingresar y realizar modificaciones.

**Usuarios**

Los usuarios que se van a crear serán usuarios de la aplicación y no del sistema operativo, los mismos crearán su propia carpeta dentro de la carpeta raíz, cada una de estas carpetas se le podrán asignar permisos de más usuarios. (podría ser que otro usuario tenga permiso de solo lectura, o bien que les permita modificar también la información).

**Permisos**

Los permisos se definirán por medio de cada usuario. Por ejemplo, el UsuarioA podría ver, modificar, eliminar archivos del Usuario B si le fueron asignados los premisos correspondientes.

**Estructura de la aplicación**

* Cada usuario tendrá dos carpetas asignadas como mínimo, una temporal y otra permanente.
  + En la carpeta permanente se incluirán solo los archivos que se le haya hecho permanentes (commit)
  + En la temporal tendrán todos los archivos.
* Cada carpeta asignada al usuario por medio de permisos creara una carpeta temporal para traer los archivos (update).

**Funciones de la aplicación**

* Commit: El commit se realizará por carpeta y la función que realiza es trasladas lo que tiene la carpeta temporal a la carpeta permanente.
* Update: El update se realizará por carpeta, es la función encargada de trasladar una copia de la carpeta permanente a la temporal. Si se tienen mas archivos en la carpeta temporal que en el permanente se perderán al realizar este proceso.
* Control de versiones: Cada vez que se realiza un commit se deben guardar los archivos que están en ese momento en permanente, para luego poder recuperarlos, si así se desea.
* Recuperar un archivo o carpeta: la opción de recuperar tiene dos modos, recuperar un único archivo, recuperar toda una carpeta. Se le debe presentar al usuario una lista de versiones y que el elija cual desea recuperar.
* Como solo se podrá ingresar a esa carpeta por medio de un usuario ya definido, las acciones básicas de cada archivo se realizarán mediante la aplicación, guardar, modificar, eliminar, abrir, etc.

**Notas**:

 La tarea se puede realizar en grupos de máximo 3 personas

 Fecha de entrega: Según cronograma

 Entregar los archivos de código fuente, junto con los pasos para compilar el código

**Tabla de evaluación**

|  |  |  |
| --- | --- | --- |
| Rubro | Valor |  |
| Crear el repositorio | 5% |  |
| Usuarios | 5% |  |
| Permisos | 8% |  |
| Correcta función en las carpetas de los usuarios | 5% |  |
| Carpetas temporales por permisos | 5% |  |
| Commit | 15% |  |
| Update | 15% |  |
| Control de versiones | 20% |  |
| Recuperación | 20% |  |
| Acciones básicas (abrir, etc) | 2% |  |
| Extra |  |  |

|  |  |
| --- | --- |
| CANTIDAD | DESGLOSE |
| NC= No Cumple, D=Deficiente, R=Regular, B=Bueno, MB= Muy Bueno, E=Excelente | |
| 30 | 0=NC, 0-7=D, 8-12=R, 13-18=B, 19-23= MB, 24-30=E |
| 25 | 0=NC, 0-5=D, 6-10=R, 11-15=B, 16-20= MB, 21-25=E |
| 20 | 0=NC, 0-4=D, 5-8=R, 9-12=B, 13-16= MB, 17-20=E |
| 15 | 0=NC, 0-3=D, 4-6=R, 7-9=B, 10-12= MB, 13-15=E |
| 10 | 0=NC, 1-2=D, 3-4=R, 5-6=B, 7-8= MB, 9-10=E |
| 8 | 0=NC, 1-2=D, 3-4=R, 5-6=B, 7= MB, 8=E |
| 5 | 0=NC, 1=D, 2=R, 3=B, 4= MB, 5=E |
| 2 | 0=NC, 1=R, 2=E |