

Puesta en Producción Segura

T1A3

Scripting Avanzado



Cursos 23/24 – IES La Marisma

Trabajo Práctico 3

Pedro Manuel García Álvarez

Índice

1.- Indica al sistema operativo que el script debe ser ejecutado.	3
2.- Introducir en el log fecha y hora de ejecución del script.	3
3.- Declaración de Variables.	3
4.- Declaraciones de funciones.	4
4.1.- Función para mostrar el mostrar_menu().	4
4.2. Función para realizar una copia() de seguridad.	5
4.3.- Función generauser() generación de nombres de usuario.	6
4.4.- Función para el alta() de usuarios.	7
4.5.- Función para dar de baja() a usuarios.	8
4.6.- Función para mostrar usuarios con opción de orden alfabético.	8
4.7.- Función para mostrar el contenido del fichero log.log.	9
4.8.- Función para controlar la existencia del fichero usuarios.csv..	9
5.- Cuerpo principal	9
5.1.- Control de acceso de usuario y menú de opciones	10
5.2.- Control de Acceso y Menú de Opciones	11
6.- Ejecución de fichero menu.sh.....	11
6.1.- Poner el fichero menu.sh para que se pueda ejecutar	11
6.2.- Ejecutamos el fichero menu.sh -root	11
6.3.- Damos de alta a los usuarios.....	12
6.5.- Visualizar el contenido del fichero usuarios.csv	13
6.6.- Realizamos la copia de seguridad.....	13
6.7.- Visualización de los ficheros copia creados	14
6.8.- Dar de baja a un usuario.....	14
6.9.- Ver el fichero usuario.csv	14
6.10.- Mostar los usuarios ordenados si ordenar	16
6.11.- Mostar log del sistema	16
6.12.- Salir	17

1.- Indica al sistema operativo que el script debe ser ejecutado.

```
#!/bin/bash
```

La línea al inicio de un script de Bash, conocida como shebang, indica al sistema operativo que el script debe ser ejecutado usando el intérprete de comandos Bash. Esta línea es fundamental, ya que le dice al sistema dónde encontrar el intérprete de comandos necesario para ejecutar el script. Específicamente, `#!/bin/bash` apunta al intérprete de comandos Bash ubicado en el directorio `/bin` del sistema.

2.- Introducir en el log fecha y hora de ejecución del script.

```
# Introducimos en el log fecha y hora de ejecución del script
echo "-----$(date +%d%m%Y:%H:%M)-----" >> log.log
```

1.- `$(date +%d%m%Y:%H:%M)`: Esta parte utiliza el comando `date` para obtener la fecha y hora actual en el formato especificado (día, mes, año, hora y minutos). El comando `date` es una utilidad en sistemas Unix que muestra la fecha y la hora actuales o establece la fecha y la hora del sistema.

2.- `echo "-----$(date +%d%m%Y:%H:%M)-----"` >> `log.log`: Aquí, la fecha y hora obtenidas se concatenan con guiones para formar un separador visual, y luego se redirigen al archivo `log.log` utilizando `>>`, lo que significa que la salida se agrega al final del archivo sin sobrescribir su contenido existente.

En resumen, esta línea imprime un separador visual que contiene la fecha y hora actual en el archivo de registro `log.log`.

3.- Declaración de Variables.

```
# Declaración de Variables
intentos=0
usuario_valido=false
```

La variable `"intentos"` podría ser utilizada para controlar el número de intentos de inicio de sesión en un script. Por ejemplo, en un escenario de autenticación, esta variable podría incrementarse cada

vez que un usuario ingresa credenciales incorrectas, y el script podría tomar decisiones basadas en el número de intentos, como limitar el acceso después de un cierto número de intentos fallidos. En resumen, la variable "intentos" podría ser empleada para rastrear y controlar los intentos de inicio de sesión en un script.

La variable "usuario_valido" podría ser utilizada para almacenar el estado de validez de un usuario dentro de un script. Por ejemplo, después de que un usuario ingresa credenciales, el script podría verificar las credenciales y luego establecer la variable "usuario_valido" en true si las credenciales son válidas, o en false si no lo son. Esta variable podría ser fundamental para controlar el acceso a ciertas partes del script o para permitir que el script realice ciertas acciones basadas en la validez del usuario.

4.- Declaraciones de funciones.

4.1.- Función para mostrar el mostrar_menu().

```
# Función para mostrar el menú
mostrar_menu() {
    echo "1.- EJECUTAR COPIA DE SEGURIDAD"
    echo "2.- DAR DE ALTA USUARIO"
    echo "3.- DAR DE BAJA AL USUARIO"
    echo "4.- MOSTRAR USUARIOS"
    echo "5.- MOSTRAR LOG DEL SISTEMA"
    echo "6.- SALIR"
}
```

La función mostrar_menu() se utiliza para presentar un menú al usuario, donde se muestran varias opciones numeradas. Cada opción se muestra con un número y una descripción. Por ejemplo, "1.- EJECUTAR COPIA DE SEGURIDAD", "2.- DAR DE ALTA USUARIO", "3.- DAR DE BAJA AL USUARIO", "4.- MOSTRAR USUARIOS", "5.- MOSTRAR LOG DEL SISTEMA" y "6.- SALIR".

4.2. Función para realizar una copia() de seguridad.

```
function copia {  
    # Obtener la fecha y hora actual  
    fecha_actual=$(date +"%d%m%Y_%H-%M-%S")  
  
    # Crear el nombre del archivo de copia de seguridad  
    nombre_copia="copia_usuarios_$fecha_actual.zip"  
  
    # Realizar la copia de seguridad  
    zip -r $nombre_copia usuarios.csv  
  
    # Mantener solo las 2 copias de seguridad más recientes  
    copias=$(ls -t copia_usuarios_*.zip)  
    contador=0  
    for copia in $copias; do  
        if [ $contador -ge 2 ]; then  
            rm $copia  
        fi  
        contador=$((contador+1))  
    done  
  
    # Agregar la entrada de la copia de seguridad al archivo de registro  
    echo "Copia de seguridad realizada: $nombre_copia el $(date +"%d%m%Y") a las $(date +"%H:%M")" >> log.log  
}
```

La función copia realiza varias tareas relacionadas con la creación y gestión de copias de seguridad. A continuación, se explica paso a paso lo que hace cada parte de la función:

1. `fecha_actual=$(date +"%d%m%Y_%H-%M-%S")`: Esta línea obtiene la fecha y hora actual en el formato especificado y la almacena en la variable `fecha_actual`.
2. `nombre_copia="copia_usuarios_$fecha_actual.zip"`: Aquí se crea el nombre del archivo de copia de seguridad, utilizando la fecha y hora actual obtenida anteriormente.
3. `zip -r $nombre_copia usuarios.csv`: Se realiza la copia de seguridad comprimiendo el archivo `usuarios.csv` en un archivo ZIP con el nombre generado.
4. `copias=$(ls -t copia_usuarios_*.zip)`: Se obtiene una lista de todas las copias de seguridad existentes, ordenadas por fecha de modificación.
5. `contador=0`: Se inicializa un contador para realizar un seguimiento del número de copias de seguridad.
6. El bucle `for` recorre la lista de copias de seguridad. Si hay más de 2 copias, se elimina la copia más antigua.
7. Finalmente, se agrega una entrada al archivo de registro, indicando que se ha realizado la copia de seguridad, junto con la fecha y hora de la operación.

En resumen, la función copia crea una copia de seguridad, gestiona el número de copias existentes y registra la operación en un archivo de registro.

4.3.- Función generauser() generación de nombres de usuario.

```
function generauser() {  
    nombre=$(echo "${1:0:1}" | tr '[:upper:]' '[:lower:]') # Primera letra del nombre  
    apellido1=$(echo "${2:0:3}" | tr '[:upper:]' '[:lower:]') # Tres primeras letras del primer apellido  
    apellido2=$(echo "${3:0:3}" | tr '[:upper:]' '[:lower:]') # Tres primeras letras del segundo apellido  
    dni=$(echo "${4: -3}" | tr '[:upper:]' '[:lower:]') # Tres últimos dígitos del DNI en minúsculas  
    nombre_usuario="${nombre}${apellido1}${apellido2}${dni}" # Nombre de usuario final  
    echo $nombre_usuario  
}
```

La función generauser() se encarga de generar un nombre de usuario a partir de los parámetros proporcionados. A continuación, se explica paso a paso lo que hace cada parte de la función:

1. nombre=\$(echo "\${1:0:1}" | tr '[:upper:]' '[:lower:]'): Toma la primera letra del primer parámetro, la convierte a minúscula y la almacena en la variable nombre.
2. apellido1=\$(echo "\${2:0:3}" | tr '[:upper:]' '[:lower:]'): Toma las tres primeras letras del segundo parámetro, las convierte a minúsculas y las almacena en la variable apellido1.
3. apellido2=\$(echo "\${3:0:3}" | tr '[:upper:]' '[:lower:]'): Toma las tres primeras letras del tercer parámetro, las convierte a minúsculas y las almacena en la variable apellido2.
4. dni=\$(echo "\${4: -3}" | tr '[:upper:]' '[:lower:]'): Toma los tres últimos dígitos del cuarto parámetro, los convierte a minúsculas y los almacena en la variable dni.
5. nombre_usuario="\${nombre}\${apellido1}\${apellido2}\${dni}": Concatena las variables anteriores para formar el nombre de usuario final.
6. echo \$nombre_usuario: Imprime el nombre de usuario resultante.

En resumen, la función generauser() toma ciertos datos de entrada y genera un nombre de usuario a partir de ellos, siguiendo un formato específico. Este tipo de funciones son comunes en scripts de automatización de tareas, especialmente en entornos de gestión de usuarios y sistemas.

4.4.- Función para el alta() de usuarios.

```
function alta {  
    read -p "Ingrese el nombre: " nombre  
    read -p "Ingrese el primer apellido: " apellido1  
    read -p "Ingrese el segundo apellido: " apellido2  
    read -p "Ingrese el DNI: " dni  
  
    nombre_usuario=$(generauser "$nombre" "$apellido1" "$apellido2" "$dni")  
  
    # Verificar si el nombre de usuario ya existe  
    if grep -q "$nombre:$apellido1:$apellido2:$dni:$nombre_usuario" usuarios.csv; then  
        echo "El nombre de usuario ya existe"  
    else  
        # Agregar el nuevo usuario al archivo usuarios.csv  
        echo "$nombre:$apellido1:$apellido2:$dni:$nombre_usuario" >> usuarios.csv  
        echo "Usuario agregado correctamente"  
  
        # Agregar la entrada de "INSERTADO" al archivo de registro  
        echo "INSERTADO $nombre:$apellido1:$apellido2:$dni:$nombre_usuario el $(date +%d%m%Y) a las $(date +%H:%M)" >> log.log  
    fi  
}
```

El código presentado define una función en llamada alta, la cual realiza las siguientes tareas:

1. Solicita al usuario que ingrese el nombre, primer apellido, segundo apellido y DNI.
2. Utiliza la función generauser() para generar un nombre de usuario a partir de la información proporcionada.
3. Verifica si el nombre de usuario ya existe en el archivo usuarios.csv.
4. Si el nombre de usuario no existe, agrega la información del nuevo usuario al archivo usuarios.csv y registra la operación en el archivo de registro log.log.

El código define una función llamada alta, la cual se encarga de solicitar al usuario información como nombre, apellidos y DNI para luego generar un nombre de usuario a partir de esta información. Posteriormente, verifica si el nombre de usuario ya existe en un archivo llamado usuarios.csv. Si el nombre de usuario no existe, agrega la información del nuevo usuario al archivo usuarios.csv y registra la operación en el archivo de registro log.log.

4.5.- Función para dar de baja() a usuarios.

```
function baja {  
  read -p "Ingrese el nombre: " nombre  
  read -p "Ingrese el primer apellido: " apellido1  
  read -p "Ingrese el segundo apellido: " apellido2  
  
  nombre_usuario=$(generauser "$nombre" "$apellido1" "$apellido2")  
  
  # Verificar si el nombre de usuario existe en el archivo  
  if grep -q "$nombre:$apellido1:$apellido2:.*:$nombre_usuario" usuarios.csv; then  
    # Crear un nuevo archivo que excluya la línea con el nombre de usuario  
    grep -v "$nombre:$apellido1:$apellido2:.*:$nombre_usuario" usuarios.csv > usuarios_nuevo.csv  
    mv usuarios_nuevo.csv usuarios.csv  
    echo "Usuario dado de baja correctamente"  
  
    # Agregar la entrada de "BORRADO" al archivo de registro  
    echo "BORRADO $nombre:$apellido1:$apellido2:.*:$nombre_usuario con la fecha de borrado $(date +%d/%m/%Y) a las $(date +%H:%M)" >> log.log  
  else  
    echo "El usuario no existe"  
  fi  
}
```

El código define una función llamada baja, la cual realiza las siguientes tareas:

1. Solicita al usuario que ingrese el nombre, primer apellido y segundo apellido del usuario que se desea dar de baja.
2. Utiliza la función generauser() para generar el nombre de usuario a partir de la información proporcionada.
3. Verifica si el nombre de usuario existe en el archivo usuarios.csv.
4. Si el nombre de usuario existe, crea un nuevo archivo que excluye la línea con la información del usuario a dar de baja, lo renombra como usuarios.csv, y registra la operación en el archivo de registro log.log.
5. Si el nombre de usuario no existe, muestra un mensaje indicando que el usuario no se encuentra en el archivo.

En resumen, la función baja se encarga de gestionar la baja de usuarios, verificando su existencia en un archivo, eliminando la información del usuario si es necesario, y registrando la operación en un archivo de registro.

4.6.- Función para mostrar usuarios con opción de orden alfabético.

```
function mostrar_usuarios {  
  read -p "¿Desea mostrar los usuarios por orden alfabético? (s/n): " opcion  
  if [ "$opcion" = "s" ]; then  
    # Mostrar usuarios por orden alfabético  
    sort -t: -k5 usuarios.csv  
  else  
    # Mostrar todos los usuarios  
    cat usuarios.csv  
  fi  
}
```


La función `mostrar_usuarios` solicita al usuario que indique si desea mostrar los usuarios por orden alfabético. Dependiendo de la respuesta, realiza las siguientes acciones:

1. Si la respuesta es "s", muestra los usuarios por orden alfabético utilizando el comando `sort`.
2. Si la respuesta es distinta de "s", muestra todos los usuarios en el archivo `usuarios.csv` utilizando el comando `cat`.

En resumen, esta función proporciona una forma interactiva de mostrar los usuarios almacenados, permitiendo al usuario elegir si desea ver la lista en orden alfabético o no.

4.7.- Función para mostrar el contenido del fichero `log.log`.

```
function mostrar_log {  
    # Mostrar el contenido del archivo de log  
    cat log.log  
}
```

El código define una función llamada `mostrar_log`, la cual se encarga de mostrar el contenido del archivo de registro `log.log` utilizando el comando `cat`. Esta función está diseñada para permitir a los usuarios visualizar el registro de operaciones almacenado en el archivo de registro.

4.8.- Función para controlar la existencia del fichero `usuarios.csv`.

```
# Función para controlar la existencia del archivo "usuarios.csv"  
controlar_archivo_usuarios() {  
    if [ ! -f "usuarios.csv" ]; then  
        touch usuarios.csv  
    fi  
}
```

La función `controlar_archivo_usuarios()` se encarga de verificar la existencia del archivo `"usuarios.csv"` en el directorio actual. Si el archivo no existe, la función lo crea utilizando el comando `touch`. Esta función es útil para asegurar que el archivo `"usuarios.csv"` esté disponible para su posterior manipulación, como la adición, modificación o eliminación de usuarios en un sistema.

5.- Cuerpo principal

```
# Llamada a la función para controlar la existencia del archivo "usuarios.csv"  
controlar_archivo_usuarios
```

Invoca la función `controlar_archivo_usuarios`, la cual verifica si el archivo `"usuarios.csv"` existe en el directorio actual. Si el archivo no existe, la función lo crea utilizando el comando `touch usuarios.csv`.

En resumen, esta línea de código invoca una función que se encarga de verificar y, en su caso, crear el archivo `"usuarios.csv"` en el directorio actual.

5.1.- Control de acceso de usuario y menú de opciones

```
while [ $intentos -lt 3 ] && [ $usuario_valido = false ]; do
    read -s -p "Ingrese su nombre de usuario: " usuario
    echo
    if [ "$usuario" = "admin" ] && [ "$1" = "-root" ]; then
        echo "Acceso como administrador concedido"
        usuario_valido=true
    elif grep -q "$usuario" usuarios.csv; then
        echo "Acceso concedido"
        usuario_valido=true
    else
        echo "Acceso denegado. Inténtelo de nuevo."
        intentos=$((intentos+1))
    fi
done
```

Este bucle `while` controla el acceso al sistema, solicitando al usuario que ingrese su nombre de usuario. Luego, verifica si el nombre de usuario es `"admin"` y si el primer parámetro pasado al script es `"-root"`. Si es así, se otorga acceso como administrador. Si no, verifica si el nombre de usuario está presente en el archivo `"usuarios.csv"`. Si el nombre de usuario está presente, se otorga acceso. De lo contrario, se deniega el acceso y se incrementa el contador de intentos. Este bucle se ejecutará hasta que el usuario sea válido o se superen los tres intentos.

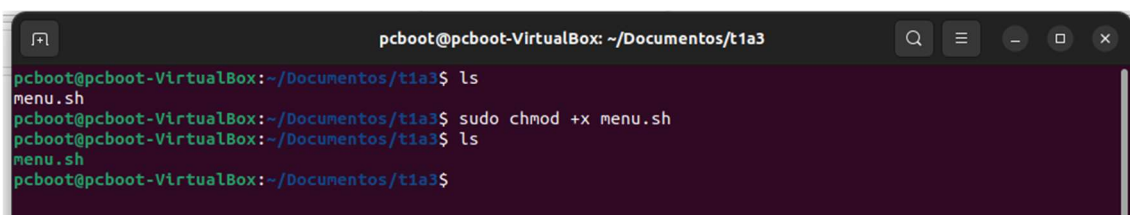
5.2.- Control de Acceso y Menú de Opciones

```
if [ $usuario_valido = true ]; then
    while true; do
        # Mostrar el menú principal
        mostrar_menu
        read -p "Seleccione una opción: " opcion
        case $opcion in
            1) # Llamar a la función para ejecutar copia de seguridad
                copia;;
            2) # Llamar a la función para dar de alta usuario
                alta;;
            3) # Llamar a la función para dar de baja usuario
                baja;;
            4) # Llamar a la función para mostrar usuarios
                mostrar_usuarios;;
            5) # Llamar a la función para mostrar log del sistema
                mostrar_log;;
            6) break
            ;;
            *) echo "Opción no válida. Por favor, seleccione una opción válida."
                ;;
        esac
    done
else
    echo "Máximo de intentos alcanzado. Saliendo del script."
fi
```

Esta estructura condicional verifica si el usuario es válido. Si el usuario es válido, se muestra un menú con varias opciones, como ejecutar una copia de seguridad, dar de alta o baja a un usuario, mostrar usuarios o mostrar el registro del sistema. Si el usuario no es válido, se muestra un mensaje indicando que se ha alcanzado el máximo de intentos y se sale del script.

6.- Ejecución de fichero menu.sh

6.1.- Agregar el permiso de ejecución al fichero menu.sh



```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ ls
menu.sh
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ sudo chmod +x menu.sh
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ ls
menu.sh
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$
```

En esta pantalla, visualizamos los archivos que se encuentran en el directorio actual y agregamos el permiso de ejecución al archivo menu.sh para permitir su ejecución.

6.2.- Ejecutamos el fichero menu.sh -root

```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ ./menu.sh -root
Ingrese su nombre de usuario:
Acceso como administrador concedido
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción:
```

En esta pantalla, se muestra el parámetro `-root` y se solicita al usuario que introduzca el nombre de usuario `admin` para acceder al menú de inicio de sesión por primera vez. El parámetro `-root` indica que se debe acceder al menú de inicio de sesión con permisos de raíz, lo que permite al usuario acceder a todas las funciones del menú.

6.3.- Damos de alta a los usuarios

```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ ./menu.sh -root
Ingrese su nombre de usuario:
Acceso como administrador concedido
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción: 2
Ingrese el nombre: Manuel
Ingrese el primer apellido: Vazquez
Ingrese el segundo apellido: Gento
Ingrese el DNI: 12345678a
Usuario agregado correctamente
```

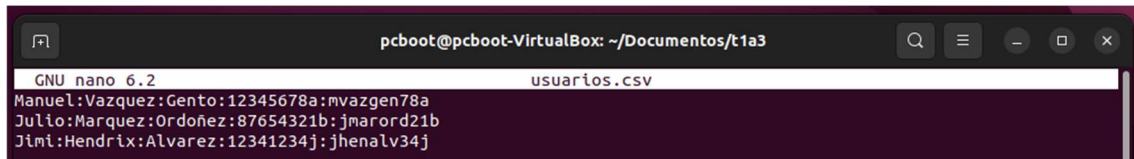
```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción: 2
Ingrese el nombre: Julio
Ingrese el primer apellido: Marquez
Ingrese el segundo apellido: Ordoñez
Ingrese el DNI: 87654321b
Usuario agregado correctamente
```

```
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción: 2
Ingrese el nombre: Jimi
Ingrese el primer apellido: Hendrix
Ingrese el segundo apellido: Alvarez
Ingrese el DNI: 12341234j
Usuario agregado correctamente
```

En esta pantalla, se realiza el alta de usuario de tres nuevos usuarios: Manuel Vázquez Gento con DNI 12345678a, Julio Márquez Ordoñez con DNI 87654321b y Jimi Hendrix Álvarez con DNI

12341234j. Después sale un mensaje usuario agregado correctamente.

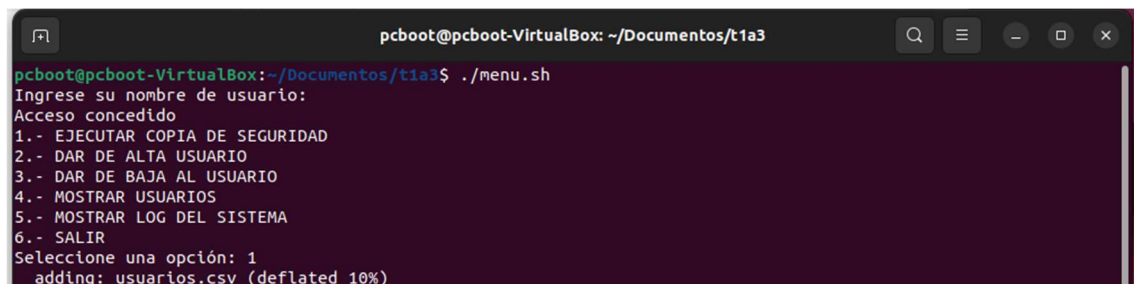
6.5.- Visualizar el contenido del fichero usuarios.csv



```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
GNU nano 6.2 usuarios.csv
Manuel:Vazquez:Gento:12345678a:mvazgen78a
Julio:Marquez:Ordoñez:87654321b:jmarord21b
Jimi:Hendrix:Alvarez:12341234j:jhenalv34j
```

En esta pantalla, se muestra el resultado de la agregación de los usuarios en el archivo usuario.csv. Después de introducir correctamente la información de los usuarios en la pantalla anterior, se ha guardado la información en el archivo usuario.csv y se ha agregado correctamente todos los usuarios. En la pantalla se puede ver el contenido del archivo usuario.csv, donde se pueden ver los nombres, DNI y nombre de usuario asignado de los usuarios agregados. La información se ha agregado correctamente, como se puede ver en la pantalla, donde se muestran los detalles de cada usuario.

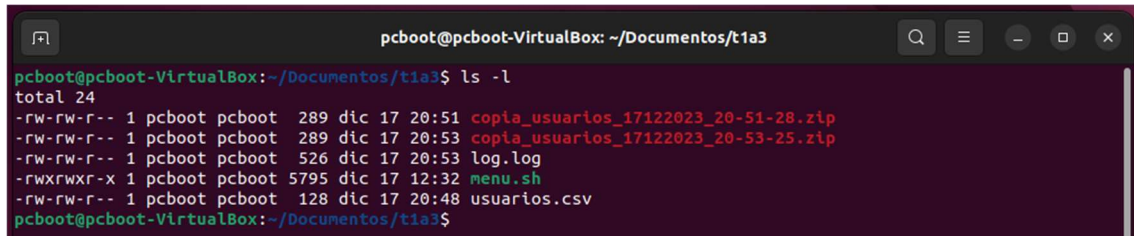
6.6.- Realizamos la copia de seguridad.



```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ ./menu.sh
Ingrese su nombre de usuario:
Acceso concedido
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción: 1
adding: usuarios.csv (deflated 10%)
```

En esta pantalla, se muestra el proceso de copia de seguridad en curso. Después de seleccionar la opción de copia de seguridad en la pantalla anterior, se ha iniciado el proceso de copia de seguridad. En la pantalla se puede ver el progreso de la copia de seguridad, incluyendo el porcentaje.

6.7.- Visualización de los ficheros copia creados



```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ ls -l
total 24
-rw-rw-r-- 1 pcboot pcboot 289 dic 17 20:51 copia_usuarios_17122023_20-51-28.zip
-rw-rw-r-- 1 pcboot pcboot 289 dic 17 20:53 copia_usuarios_17122023_20-53-25.zip
-rw-rw-r-- 1 pcboot pcboot 526 dic 17 20:53 log.log
-rwxrwxr-x 1 pcboot pcboot 5795 dic 17 12:32 menu.sh
-rw-rw-r-- 1 pcboot pcboot 128 dic 17 20:48 usuarios.csv
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$
```

En esta pantalla se ve que se ha realizado las copias de seguridad correctamente.

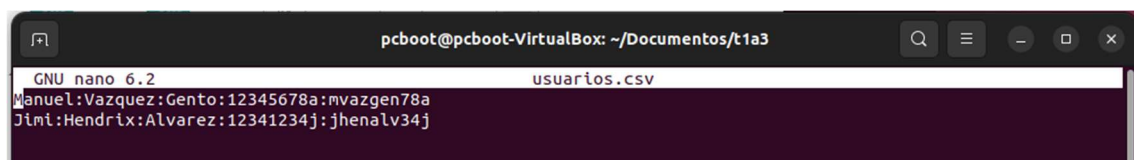
6.8.- Dar de baja a un usuario



```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ ./menu.sh
Ingrese su nombre de usuario:
Acceso concedido
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción: 3
Ingrese el nombre: Julio
Ingrese el primer apellido: Marquez
Ingrese el segundo apellido: Ordoñez
Usuario dado de baja correctamente
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción:
```

En esta pantalla, se muestra el proceso de dar de baja a un usuario en el sistema. Después de seleccionar la opción de dar de baja en la pantalla anterior, se ha iniciado el proceso de eliminación del usuario. En la pantalla se puede ver el nombre del usuario que se va a dar de baja, en este caso Julio Márquez Ordoñez, y un mensaje que indica que el usuario dado de baja correctamente.

6.9.- Ver el fichero usuarios.csv



```
GNU nano 6.2 usuarios.csv
Manuel:Vazquez:Gento:12345678a:mvazgen78a
Jimi:Hendrix:Alvarez:12341234j:jhenalv34j
```

En esta pantalla, se muestra el resultado de dar de baja al usuario Julio Márquez Ordoñez. Después de seleccionar la opción de

dar de baja en la pantalla anterior, se ha eliminado el usuario del sistema y solo se ve los usuarios que esta dado de alta.

6.10.- Mostar los usuarios ordenados o si ordenar

```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ ./menu.sh
Ingrese su nombre de usuario:
Acceso concedido
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción: 4
¿Desea mostrar los usuarios por orden alfabético? (s/n): s
Jimi:Hendrix:Alvarez:12341234j:jhenalv34j
Manuel:Vazquez:Gento:12345678a:mvazgen78a
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción: 4
¿Desea mostrar los usuarios por orden alfabético? (s/n): n
Manuel:Vazquez:Gento:12345678a:mvazgen78a
Jimi:Hendrix:Alvarez:12341234j:jhenalv34j
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción:
```

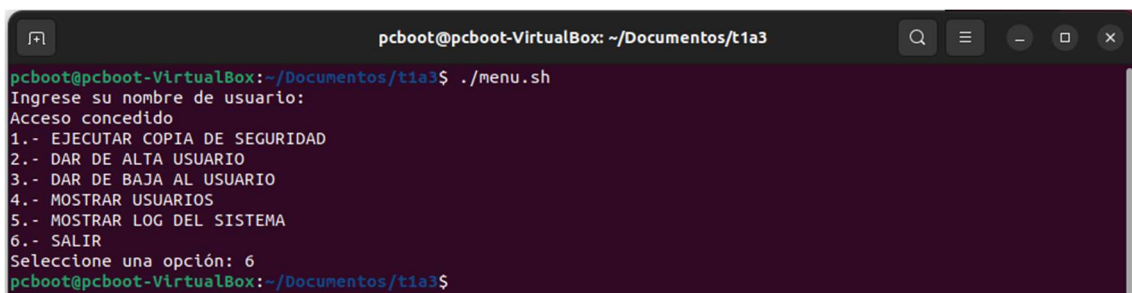
En esta pantalla, se muestra la opción 4, que es "Mostrar usuarios". Esta opción permite al usuario ver la lista de usuarios en el sistema, y elegir si desea verlos en orden alfabético o en orden como se ha escrito en el fichero. Si se selecciona "Ordenado", la lista de usuarios se organizará en orden alfabético, mientras que, si se selecciona "Desordenado", la lista se organizará de manera que está en el fichero.

6.11.- Mostar log del sistema

```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ ./menu.sh
Ingrese su nombre de usuario:
Acceso concedido
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción: 5
-----17122023:20:44-----
INSERTADO Manuel:Vazquez:Gento:12345678a:mvazgen78a el 17122023 a las 20:46
INSERTADO Julio:Marquez:Ordoñez:87654321b:jmarord21b el 17122023 a las 20:47
INSERTADO Jimi:Hendrix:Alvarez:12341234j:jhenalv34j el 17122023 a las 20:48
-----17122023:20:50-----
Copia de seguridad realizada: copia_usuarios_17122023_20-51-28.zip el 17122023 a las 20:51
-----17122023:20:53-----
Copia de seguridad realizada: copia_usuarios_17122023_20-53-25.zip el 17122023 a las 20:53
-----17122023:20:59-----
-----17122023:20:59-----
BORRADO Julio:Marquez:Ordoñez:.*:jmarord con la fecha de borrado 17122023 a las 20:59
-----17122023:21:03-----
-----17122023:21:06-----
```


En esta pantalla, se muestra la opción 5: "Mostrar log del sistema". El log es una registración de todas las acciones y eventos que ocurren en el sistema, incluyendo la acción realizada, la fecha y hora de ejecución, y cualquier otro detalle relevante. Al seleccionar esta opción, se muestra el registro de actividad del sistema, lo que permite ver todo lo que se ha realizado ejecutando el script, como la creación, modificación o eliminación de usuarios, entre otras acciones.

6.12.- Salir



```
pcboot@pcboot-VirtualBox: ~/Documentos/t1a3
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$ ./menu.sh
Ingrese su nombre de usuario:
Acceso concedido
1.- EJECUTAR COPIA DE SEGURIDAD
2.- DAR DE ALTA USUARIO
3.- DAR DE BAJA AL USUARIO
4.- MOSTRAR USUARIOS
5.- MOSTRAR LOG DEL SISTEMA
6.- SALIR
Seleccione una opción: 6
pcboot@pcboot-VirtualBox:~/Documentos/t1a3$
```

En esta pantalla se ve como salir del script con la opción 6.

Índice Alfabético

A

acceso 2, 4, 10
acciones 4, 9, 17
admin 10, 12
alfabético 2, 8, 9, 16
almacena 5, 6
alta 2, 7, 11, 12, 15
alta() 2, 7
antigua 5
año 3
apellido1 6
apellido2 6
archivo 3, 5, 6, 7, 8, 9, 10, 11, 13
autenticación 3
automatización 6

B

baja 2, 8, 11, 14
baja() 2, 8
bash 3
bucle 5, 10

C

cat9
código 7, 8, 9, 10
comando 3, 9, 10
condicional 11
contador 5, 10
controlar 2, 3, 4, 9, 10
controlar_archivo_usuarios() 9
copia() 2, 5
copias 5, 6, 14
credenciales 4

D

date 3, 5
día3
dígitos 6
directorio 3, 9, 10, 11
dni6

E

echo 3, 6
ejecutado 2, 3
entornos 6
entrada 5, 6
escenario 3
estructura 11

F

fallidos 4
fecha 2, 3, 5, 17
fichero 2, 9, 11, 13, 14, 16
for 5
formato 3, 5, 6
función 4, 5, 6, 7, 8, 9, 10
funciones 2, 4, 6, 12

G

genera 6
generauser() 2, 6, 7, 8
gestión 5, 6

H

hora 2, 3, 5, 17

I

imprime 3
información 7, 8, 13
intentos 3, 10, 11
intérprete 3

L

letras 6
lista 5, 9, 16
log 2, 3, 7, 8, 9, 16, 17
log.log 2, 3, 7, 8, 9
lower 6

M

menú 2, 4, 10, 11, 12
menu.sh 2, 11
mes..... 3
minúsculas 6
minutos 3
mostrar 2, 4, 8, 9, 11
mostrar_log..... 9

N

nombre 5, 6, 7, 8, 10, 12, 13, 14
número 3, 4, 5, 6

O

opción 2, 4, 8, 13, 14, 16, 17
operación 5, 6, 7, 8
operativo..... 2, 3

P

pantalla 11, 12, 13, 14, 16, 17
permiso 11

R

rastrear 4
registro..... 3, 5, 6, 7, 8, 9, 11, 17
resumen 3, 4, 6, 8, 9, 10
root 2, 10, 11, 12

S

script 2, 3, 4, 10, 11, 17
seguridad 2, 5, 6, 11, 13, 14
sesión 3, 12
shebang..... 3
sistema 2, 3, 9, 10, 11, 14, 15, 16, 17
solicita 9, 12
sort..... 9

T

tareas 5, 6, 7, 8
touch..... 9, 10

U

upper..... 6
usuario 2, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16
usuarios.csv..... 2, 5, 7, 8, 9, 10, 13

V

validez 4
válido 10, 11
variable 3, 4, 5, 6
verifica 7, 10, 11
visual 3

Z

zip 5