



Proyecto Linux

Muñoz Norberto David Julian
Montero Quintero Francisco Joshua

22 de Septiembre 2023

Contents

1	Introducción	2
1.1	Objetivo	2
2	Desarrollo	2
2.1	Sistema de Acceso	2
2.2	Menu	4
2.3	Información del Sistema	6
2.4	Fecha y hora	8
2.5	Buscar	8
2.6	Creditos	9
2.7	Juego	10
2.8	Reproductor MP3	14
3	Conclusiones	17

1 Introducción

Durante el desarrollo de este proyecto se elaboró una terminal usando shell-crypt, esta terminal ejecuta los comandos dentro del Sistema Operativo Linux.

Este proyecto representa nuestro conocimiento en programación y la comprensión profunda de las necesidades de los usuarios en entornos de línea de comandos, por ello creamos unos cuantos comandos que creemos pueden ser de mucha ayuda para el usuario y sobre todo es una oportunidad para demostrar los conocimientos adquiridos.

1.1 Objetivo

Como becarios demostraremos los conocimientos adquiridos durante el curso de Linux, así mismo, pondremos a prueba nuestra capacidad de investigación, creatividad y análisis para poder cumplir con las especificaciones del proyecto.

2 Desarrollo

En esta sección de desarrollo, profundizaremos en la arquitectura y funcionalidades clave que hacen de nuestra terminal una herramienta útil.

2.1 Sistema de Acceso

```
#!/bin/bash
# Funci n para manejar la se al SIGINT (Ctrl+C)
ctrl_c_handler() {
    echo "Ctrl+C desactivado"
}

# Asociar la funci n al manejo de la se al SIGINT
trap ctrl_c_handler SIGINT

#Funcion que bloquea ctrl+z
stty susp ""

while true; do
    echo "Ingresa el nombre del usuario o 'salir' para salir:"
    read user

    if [ "$user" = "salir" ]; then
        echo "Gracias por usar la terminal PREBE:)"
    fi
done
```

```

        break
    fi

    echo "Ingresa la contraseña del usuario:"
    read -s pass

    if su "$user" <<EOF
$pass
EOF
    then
        ./menu.sh
    else
        echo "Credenciales incorrectas."
    fi
done

```

- Primeramente en la primer linea del script indica que el intérprete de comandos a utilizar es Bash. Esto es necesario para que el sistema sepa cómo ejecutar el script.
- Se implementan dos funciones, una que desactiva el comando CTRL+C y la otra que desactiva CTRL+Z durante la ejecución de todo el programa
- Inicia un bucle infinito que solicita al usuario un nombre de usuario y una contraseña.
- Si el usuario ingresa "salir," el programa muestra un mensaje de despedida y finaliza.
- Si el usuario ingresa un nombre de usuario, se solicita una contraseña y se intenta cambiar al usuario utilizando el comando 'su'.
- Si las credenciales son correctas, se ejecuta un script llamado 'menu.sh'.
- Si las credenciales son incorrectas, se muestra un mensaje de error.

```

joshuaqm@miPC:~/Documentos/GitHub/ProyectoLinux$ ./main.sh
Ingresa el nombre del usuario o 'salir' para salir:
joshuaqm
Ingresa la contraseña del usuario:
Contraseña:

Valida
Bienvenid@ joshuaqm

```

2.2 Menu

```
#!/bin/bash
# Función para manejar la señal SIGINT (Ctrl+C)
ctrl_c_handler() {
    echo "Ctrl+C desactivado"
}

# Asociar la función al manejo de la señal SIGINT
trap ctrl_c_handler SIGINT
echo -e "\n \nValida"
echo -e "Bienvenid@ \e[33m$USER"\e[0m"

opcion=""

while [ "$opcion" != "salir" ];do
    echo -e "\n\e[32mBienvenido al menu de opciones, escribe "ayuda" para ver las opciones d
    echo -e -n "\e[33m$USER"\e[0m:\e[36m$PWD"\e[0m> "
    read opcion

    if [ "$opcion" == "ayuda" ];then
        echo ayuda:      Muestra la lista de comandos disponibles en la terminal PREBE
        echo infosys:   Muestra la informacion del sistema
        echo fyh:       Muestra la fecha y hora actual
        echo buscar:    Busca un archivo en la carpeta especificada por el usuario
        echo creditos:  Muestra los creditos de los programadores
        echo juego:     Inicia un juego en la terminal
        echo musica:    Reproduce musica en la terminal
        echo salir:     Redirige al usuario al sistema de login
    fi

    if [ "$opcion" == "infosys" ];then
        echo "La informacion del sistema es:"
        ./infosys.sh
    fi

    if [ "$opcion" == "fyh" ];then
        echo "La fecha y hora actual es:"
        ./fechayhora.sh
    fi

    if [ "$opcion" == "buscar" ];then
        ./buscar.sh
    fi
fi
```

```

if [ "$opcion" == "creditos" ];then
    ./creditos.sh
fi

if [ "$opcion" == "juego" ];then
    ./juego.sh
fi

if [ "$opcion" == "musica" ];then
    ./musica.sh
else
    eval $opcion 2>/dev/null
fi

done

```

Este comando muestra una interfaz con un menú de opciones.

- Al principio, se desactiva el comando CTRL+C y muestra un mensaje de bienvenida al usuario.
- Se ejecuta un bucle, donde se muestra la línea de comandos y que a su vez muestra el usuario activo de color amarillo y la ruta actual en azul, esperando a que sea ingresado algún comando
- Mientras el valor de la variable 'opcion' no sea igual a "salir", permite que el programa se ejecute continuamente hasta que el usuario elija salir.
- El usuario puede ingresar comandos, y el script verificará cuál se ha ingresado utilizando estructuras condicionales.
- Si se ingresa "ayuda", se muestra una lista de comandos disponibles.
- Si se ingresa "infosys", ejecuta el script 'infosys.sh' que muestra información del sistema.
- Si se ingresa "fyh", ejecuta el script 'fechayhora.sh' que muestra la fecha y hora actual.
- Si se ingresa "buscar", ejecuta el script 'buscar.sh' que probablemente realiza una búsqueda de archivos.
- Si se ingresa "creditos", ejecuta el script 'creditos.sh' que muestra los créditos de los programadores.
- Si se ingresa "juego", ejecuta el script 'juego.sh' que inicia un juego en la terminal.

- Si se ingresa "musica", ejecuta el script 'musica.sh' que reproduce música en la terminal.
- En caso de que no se ingrese alguno de los comandos anteriores, la funcion eval evalúa si la opción ingresada es un comando de Linux y lo ejecuta
- Si la opción ingresada no es ningún comando reconocido por la terminal, se vuelve a pedir que ingrese otra opción que sí sea válida.
- El bucle continúa hasta que el usuario elige "salir", donde redirige al usuario al sistema de inicio de sesión nuevamente.

```
Bienvenido al menu de opciones, escribe ayuda para ver las opciones disponibles
joshuaqm:/home/joshuaqm/Documentos/GitHub/ProyectoLinux> ayuda
ayuda: Muestra la lista de comandos disponibles en la terminal PREBE
infosys: Muestra la informacion del sistema
fyh: Muestra la fecha y hora actual
buscar: Busca un archivo en la carpeta especificada por el usuario
creditos: Muestra los creditos de los programadores
juego: Inicia un juego en la terminal
musica: Reproduce musica en la terminal
salir: Redirige al usuario al sistema de login
```

2.3 Información del Sistema

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

```
echo "
```

```
"
```

```
echo " | - - - - - Informacion del sistema - - - - -
      - - - - - | "
```

```
echo "
```

```
"
```

```
echo " | Nombre del dispositivo: - - - - - | -$(uname -n) - - - - -
      - - - - - | "
```

```
echo "
```

```
"
```

```
echo " | Arquitectura: - - - - - | -$(uname -p) - - - - -
      - - - - - | "
```

```
echo "
```

```
"
```

```
echo " | Sistema Operativo: - - - - - | -$(uname -o) - - - - -
      - - - - - | "
```

```
echo '
,
```

```
,
```

```
echo " | Version del OS: ----- | $(grep -oP '
    VERSION="\K[^"]+' /etc/os-release) ----- | "
echo "
```

```
echo "
```

```
echo " | Memoria Total | Memoria Usada | Memoria Libre |
    Intercambio Total | Intercambio Usado | Intercambio
    Libre | "
echo "
```

```
# Ejecutar el comando free -h y mostrar los resultados en
    la tabla
free -h | awk 'NR==2{printf " | ---%s----- | ---%s----- | -
    ---%s----- | ", - $2, - $3, - $4} NR==3{printf " -----%s-----
    --- | -----%s----- | -----%s----- |\n", - $2, - $3, - $4}
    ,
echo "
```

```
Bienvenido al menu de opciones, escribe ayuda para ver las opciones disponibles
joshuaqm:/home/joshuaqm/Documentos/GitHub/ProyectoLinux> infosys
La informacion del sistema es:
-----
|                               Información del sistema                               |
|-----|-----|-----|-----|-----|-----|
|Nombre del dispositivo: | m1PC |
|Arquitectura: | x86_64 |
|Sistema Operativo: | GNU/Linux |
|Versión del OS: | 22.04.3 LTS (Jammy Jellyfish) | | | | |
|---|---|---|---|---|---|
|Memoria Total | Memoria Usada | Memoria Libre | Intercambio Total | Intercambio Usado | Intercambio Libre|
| 3,8G | 2,2G | 178M | 3,9G | 465M | 3,4G |
|-----|-----|-----|-----|-----|-----|
```

Este comando principalmente muestra información del sistema, incluyendo detalles sobre el nombre del dispositivo, arquitectura, sistema operativo, versión del sistema operativo y estadísticas de memoria en forma de tabla.

- Comienza mostrando una cabecera para la sección de "Información del sistema."
- Imprime el nombre del dispositivo utilizando el comando 'uname -n'.

- Imprime la arquitectura del sistema utilizando el comando ‘uname -p’.
- Imprime el nombre del sistema operativo utilizando el comando ‘uname -o’.
- Utiliza ‘grep’ para leer el archivo ‘/etc/os-release’ y extraer el valor de ‘VERSION’, que representa la versión del sistema operativo.
- Imprime una cabecera para la sección de ”Memoria Total — Memoria Usada — Memoria Libre — Intercambio Total — Intercambio Usado — Intercambio Libre.”
- Ejecuta el comando ‘free -h’, que muestra estadísticas de memoria, y utiliza ‘awk’ para formatear la salida en una tabla.

2.4 Fecha y hora

```
awk 'BEGIN {now=systime()-/proc/uptime; print strftime("
Estamos a :%Y-%m-%d - y la hora es : %H:%M:%S", now)}' /
proc/uptime
```

Lo que hacemos con este comando es leer el tiempo desde /proc/uptime, este es el tiempo encendido del sistema y lo resta al tiempo actual del sistema que se obtiene desde systime(), luego usamos strftime para dar formato al resultado con una cadena

```
Bienvenido al menu de opciones, escribe ayuda para ver las opciones disponibles
joshuaqm:/home/joshuaqm/Documentos/GitHub/ProyectoLinux> fyh
La fecha y hora actual es:
Estamos a :2023-09-22 y la hora es: 22:02:12
```

2.5 Buscar

El comando comienza solicitando al usuario que ingrese el nombre del archivo que desea buscar y la ruta absoluta del directorio en el que desea realizar la búsqueda.

Luego, utiliza el comando find para buscar archivos dentro del directorio especificado (\$directorio) que coincidan con el nombre proporcionado (\$archivo). La opción -type f indica que se deben buscar solo archivos regulares (no directorios ni enlaces simbólicos).

Los resultados de la búsqueda se almacenan en una variable llamada busqueda.

Posteriormente, el script muestra en pantalla los resultados de la búsqueda, que son los nombres de los archivos encontrados.

A continuación, utiliza una estructura condicional if para verificar si la variable busqueda está vacía. Si está vacía, significa que no se encontraron archivos que coincidan con el nombre y la ruta proporcionados.

Si no se encontraron archivos, el comando muestra un mensaje que indica que el archivo no se encontró en la ruta especificada.

Si se encontraron archivos, el comando muestra un mensaje que indica que el archivo se encontró y luego muestra la lista de archivos encontrados, que incluyen sus rutas absolutas.

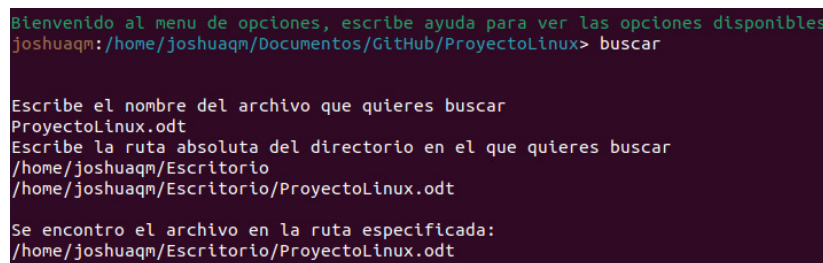
```
#!/bin/bash

echo -e "\n\nEscribe el nombre del archivo que quieres buscar"
read archivo

echo "Escribe la ruta absoluta del directorio en el que quieres buscar"
read directorio

busqueda=$(find "$directorio" -type f -name "$archivo")
echo $busqueda

if [ "$busqueda" == "" ];then
    echo -e "\nNo se encontro el archivo\n"
else
    echo -e "\nSe encontro el archivo en la ruta especificada:"
    echo $busqueda
fi
```



```
Bienvenido al menu de opciones, escribe ayuda para ver las opciones disponibles
joshuaqm:/home/joshuaqm/Documentos/GitHub/ProyectoLinux> buscar

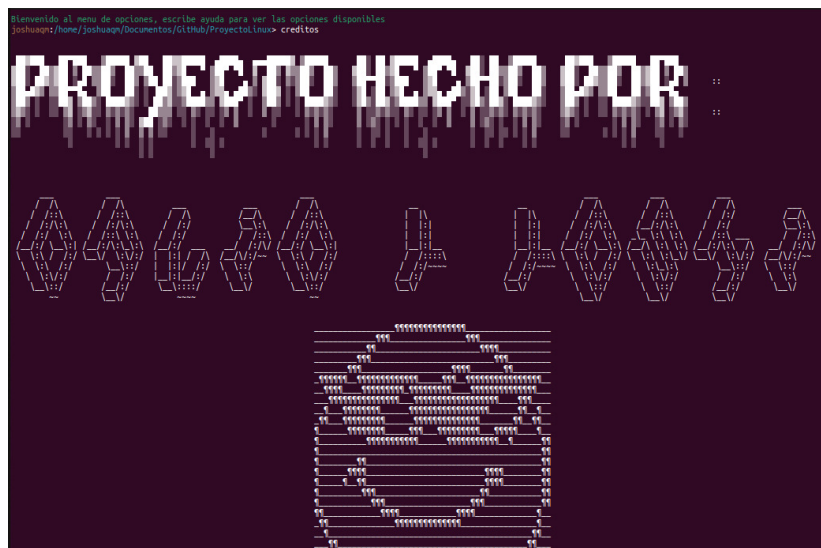
Escribe el nombre del archivo que quieres buscar
ProyectoLinux.odt
Escribe la ruta absoluta del directorio en el que quieres buscar
/home/joshuaqm/Escritorio
/home/joshuaqm/Escritorio/ProyectoLinux.odt

Se encontro el archivo en la ruta especificada:
/home/joshuaqm/Escritorio/ProyectoLinux.odt
```

Este comando realiza una búsqueda de archivos en un directorio específico basado en el nombre de archivo proporcionado por el usuario. Luego, muestra la ruta completa de los archivos encontrados que coinciden con el nombre especificado. Si no se encuentra ningún archivo, muestra un mensaje indicando que el archivo no se encontró en la ruta especificada.

2.6 Credits

Usando Char-art hicimos una bonita presentacion de los integrantes del equipo.



2.7 Juego

Explicación del Juego de Ahorcado

Este código implementa un juego de ahorcado a partir de la línea de comandos. A continuación, se proporciona una explicación detallada de cada parte del código:

Lista de Palabras

Se crea una lista de palabras (frutas) llamada "palabras" que el jugador puede adivinar. Estas palabras están almacenadas en un array.

Selección Aleatoria de la Palabra

Una palabra se selecciona aleatoriamente de la lista de palabras utilizando la variable \$RANDOM. Esto garantiza que el juego sea diferente cada vez que se juega.

Palabra Adivinada

Se crea una cadena llamada "palabra_adivinada" que inicialmente está vacía. Esta cadena representará la palabra que el jugador está tratando de adivinar, con guiones bajos "_" para ocultar las letras no adivinadas.

Número de Intentos

Se establece un límite de 6 intentos para el jugador. El jugador debe adivinar la palabra antes de agotar estos intentos.

Funciones "mostrar_juego" y "dibujar_ahorcado"

Se definen dos funciones. "mostrar_juego" muestra el estado actual del juego, incluyendo la palabra a adivinar y el número de intentos restantes. "dibujar_ahorcado" se encarga de dibujar un gráfico del ahorcado en función del número de intentos restantes, utilizando caracteres ASCII.

Inicialización de "palabra_adivinada"

Un bucle "for" se utiliza para inicializar la variable "palabra_adivinada" con

```

Bienvenido al menu de opciones, escribe ayuda para ver las opciones disponibles
joshuaqn:/home/joshuaqn/Documents/GitHub/ProyectoLinux> juego
¡Bienvenid@ a ahorcado!, la palabra que tienes que adivinar es una fruta

Palabra a adivinar: _____
Intentos restantes: 6
Ingresa una letra: s

  O

Palabra a adivinar: _____
Intentos restantes: 5
Ingresa una letra: m

  O

Palabra a adivinar: m_____
Intentos restantes: 5
Ingresa una letra: a

  O

Palabra a adivinar: ma__a_a
Intentos restantes: 5
Ingresa una letra: n

  O

Palabra a adivinar: man_ana
Intentos restantes: 5
Ingresa una letra: z

  O

Palabra a adivinar: manzana
Intentos restantes: 5
¡Felicidades! Adivinaste la palabra: manzana

```

guiones bajos "_" según la longitud de la palabra secreta. Esto oculta todas las letras al comienzo del juego.

Bucle de Juego

Se inicia un bucle "while" que permite al jugador realizar intentos para adivinar la palabra. En cada iteración, se muestra el estado actual del juego, se recoge la letra ingresada por el jugador y se verifica si está en la palabra secreta. Si la letra es correcta, se revela en la palabra oculta y se muestra el dibujo actualizado del ahorcado. Si la letra es incorrecta, se disminuye en uno el número de intentos y se muestra el dibujo actualizado del ahorcado. El juego continúa hasta que el jugador adivina la palabra o se queda sin intentos.

Fin del Juego

Después de salir del bucle, se muestra el estado final del juego con "mostrar_juego", y se imprime un mensaje que indica si el jugador adivinó la palabra o se quedó sin intentos, revelando la palabra secreta en este último caso.

```
#!/bin/bash

echo "¡Bienvenid@ a ahorcado!, la palabra que tienes que adivinar es una fruta"
# Palabras para el juego
palabras=("mango" "manzana" "melon" "pera" "kiwi" "sandia" "platano")

palabra_secreta=${palabras[$RANDOM % ${#palabras[@]}]}

palabra_adivinada=""
intentos=6

# Función que muestra el estado actual del juego
mostrar_juego() {
    echo -e "\nPalabra a adivinar: $palabra_adivinada"
    echo "Intentos restantes: $intentos"
}

# Funcion que dibuja al ahorcado
dibujar_ahorcado() {
    case $1 in
        5)
            echo -e "  0"
            ;;
        4)
            echo -e "  0"
            echo -e "  I"
            ;;
        3)
            echo -e "  0"
            echo -e "--I"
            ;;
        2)
            echo -e "  0"
            echo -e "--I--"
            ;;
        1)
            echo -e "  0"
            echo -e "--I--"
            echo -e " i"
            ;;
        0)
            echo -e "  0"
            echo -e "--I--"
            echo -e " i i"
            ;;
        *)
    
```

```

        echo "Número de intentos no válido: $1"
        ;;
    esac
}

for ((i=0; i<${#palabra_secreta}; i++)); do
    palabra_adivinada+="_"
done

# Bucle que contabiliza intentos e imprime el dibujo del ahorcado
while [ $intentos -gt 0 ]; do
    mostrar_juego

    read -p "Ingresa una letra: " letra
    echo -e "\n"

    if [[ $palabra_secreta == *"$letra"* ]]; then
        for ((i=0; i<${#palabra_secreta}; i++)); do
            if [ "${palabra_secreta:i:1}" == "$letra" ]; then
                palabra_adivinada="${palabra_adivinada:0:$i}$letra${palabra_adivinada:$i+1}"
            fi
        done

        dibujar_ahorcado $intentos

    else
        intentos=$((intentos-1))
        dibujar_ahorcado $intentos
    fi

    if [ "$palabra_adivinada" == "$palabra_secreta" ]; then
        mostrar_juego
        echo "¡Felicidades! Adivinaste la palabra: $palabra_secreta"
        exit 0
    fi
done

mostrar_juego
echo "Te quedaste sin intentos. La palabra era: $palabra_secreta"

```

2.8 Reproductor MP3

Desactivación de CTRL+C

Nuevamente implementamos la función para este menú, el cual desactiva el comando CTRL+C que interrumpe la ejecución del programa.

Verificación de mpg123 instalado

A través de una estructura condicional se verifica si la dependencia de mpg123 está instalada en la computadora, en caso de que no, se le pregunta al usuario si quiere instalarla para continuar con la ejecución del reproductor de mp3. En caso de que si esté instalada, el programa continua normalmente.

Menú del reproductor

Se muestra un nuevo menú dentro de este script de reproductor, con las siguientes opciones: ayuda, reproducir, todo y salir.

Ayuda

El comando 'ayuda' muestra los comandos disponibles dentro del menu principal del reproductor de música

Reproducir

El comando 'reproducir' reproduce un solo archivo de audio a la vez, le pide al usuario que ingrese la ruta absoluta donde se encuentra su archivo y también el nombre del mismo. Se muestra el mensaje "Presiona 'q' para dejar de reproducir" y la reproducción inicia.

Todo

El comando 'todo' reproduce todos los archivos de audio que se encuentren en la ruta indicada por el usuario, se muestran los mensajes "Presiona 'q' para dejar de reproducir", "Presiona 'f' para reproducir la siguiente canción", "Presiona 'd' para reproducir la canción anterior" y los archivos de audio son introducidos en una cola de reproducción.

Salir

El menú del reproductor se encuentra dentro de un bucle, en donde hasta que se escriba 'salir', saldrá del reproductor hacia el menú principal.

```
#!/bin/bash
# Funcion para manejar la senal SIGINT (Ctrl+C)
ctrl_c_handler() {
    echo "Ctrl+C-desactivado"
}

# Asociar la funcion al manejo de la senal SIGINT
trap ctrl_c_handler SIGINT

#/dev/null hace que no se imprima la ruta, 2>&1 descarta
mensaje de error
if which mpg123 > /dev/null 2>&1;then
    echo mpg123 instalado
else
    echo -e "\nParece que no tienes instalado mpg123, ¿
   Quieres instalarlo?-(s/n)"
    read respuesta
    if [ "$respuesta" == "s" ];then
        sudo apt-get install mpg123
    else
```

```

        echo -e "\nNo se instalo mpg123, no se puede
        reproducir musica\n"
        exit 1
    fi
fi

opcion=""

echo -e "\n Bienvenid@-a-Espotifai!\n-Escribe-'ayuda'-
para-ver-los-comandos-disponibles\n"
while [ "$opcion" != "salir" ];do
    echo -e -n "\n>"
    read opcion

    if [ "$opcion" == "ayuda" ];then
        echo ayuda:      Muestra la lista de comandos
                        disponibles en el reproductor Espotifai
        echo reproducir:  Reproduce la cancion
                        especificada
        echo todo:        Reproduce todas las canciones
                        de la carpeta especificada
        echo salir:       Regresa al menu principal
    fi

    if [ "$opcion" == "reproducir" ];then
        echo -e "\n\nEscribe-la-ruta-absoluta-del-
                directorio-donde-se-encuentran-tus-canciones"
        read directorio

        echo -e "\n\nEscribe-el-nombre-del-archivo-que-
                quieres-reproducir"
        read archivo

        echo -e "\nPresiona-'q'-para-dejar-de-reproducir"

        mpg123 "$directorio"/"$archivo"

    fi

    if [ "$opcion" == "todo" ];then
        echo -e "\n\nEscribe-la-ruta-absoluta-del-
                directorio-donde-se-encuentran-tus-canciones"
        read directorio

        echo -e "\nPresiona-'q'-para-dejar-de-reproducir"
        echo -e "\nPresiona-'f'-para-reproducir-la-

```

```

        siguiente -cancion"
echo -e "\nPresiona -'d'- para reproducir la -
cancion -anterior"

for archivo in "$directorio"/*;do
    mpg123 "$archivo"
done

fi

done

```

```

Bienvenido al menu de opciones, escribe ayuda para ver las opciones disponibles
joshuaqm:/home/joshuaqm/Documentos/GitHub/ProyectoLinux> musica
mpg123 instalado

;Bienvenid@ a Espotifai!
Escribe 'ayuda' para ver los comandos disponibles

>ayuda
ayuda: Muestra la lista de comandos disponibles en el reproductor Espotifai
reproducir: Reproduce la cancion especificada
todo: Reproduce todas las canciones de la carpeta especificada
salir: Regresa al menu principal

>reproducir

Escribe la ruta absoluta del directorio donde se encuentran tus canciones
/home/joshuaqm/Escritorio

Escribe el nombre del archivo que quieres reproducir
Intro.mp3

Presiona 'q' para dejar de reproducir
High Performance MPEG 1.0/2.0/2.5 Audio Player for Layers 1, 2 and 3
version 1.29.3; written and copyright by Michael Hipp and others
free software (LGPL) without any warranty but with best wishes

Directory: /home/joshuaqm/Escritorio/

Terminal control enabled, press 'h' for listing of keys and functions.

Playing MPEG stream 1 of 1: Intro.mp3 ...

MPEG 1.0 L III vbr 48000 j-s

Title: Intro
Album: xx
Genre: Indie; Alternative; Electronic

>ayuda
ayuda: Muestra la lista de comandos disponibles en el reproductor Espotifai
reproducir: Reproduce la cancion especificada
todo: Reproduce todas las canciones de la carpeta especificada
salir: Regresa al menu principal

>salir

Bienvenido al menu de opciones, escribe ayuda para ver las opciones disponibles
joshuaqm:/home/joshuaqm/Documentos/GitHub/ProyectoLinux>

```


3 Conclusiones

Quintero Montero Francisco Joshua:

Gracias a la realización del proyecto pude reforzar mis conocimientos en bash, un lenguaje de programación que era totalmente nuevo para mí. Además, al investigar, aprendí más cosas que se pueden realizar en Linux, tal como agregar texto de colores, reproducir archivos de audio a través de la terminal y a utilizar nuevas funciones. Me pareció muy interesante el proceso de realizar un programa con ayuda de GitHub y también utilizar Latex para documentar el desarrollo de scripts.

Muñoz Norberto David Julian:

En conclusión, con ayuda de este proyecto logramos cumplir nuestros objetivos iniciales, crear una interfaz de usuario intuitiva y eficiente, tratamos de mantener una sólida documentación a lo largo del proceso. Gracias a las pruebas exhaustivas, pudimos entregar el producto a tiempo. La comunicación efectiva y la colaboración en equipo fueron esenciales para poder concluir el proyecto. Además, estamos abiertos a futuras mejoras y continuaremos trabajando en base a los comentarios de los becarios para poder seguir aprendiendo. Fué un poco pesado pero se logró terminar.