

Interactive Linux User & Group Management Tool

(An easy-to-use Bash script with a whiptail-based menu)

❖ Project Idea:

This script is a **Bash-based interactive tool** using the whiptail dialog interface. It helps in managing users and groups on a Linux system with a user-friendly GUI-like experience, instead of typing complex terminal commands manually.

🔥 What the Script Can Do:

1. Add a new user with a password.
2. Modify a user (change name, UID, home directory,).
3. List all system users.
4. Create a new group.
5. Modify a group (rename, change GID, show members, set group password).
6. List all groups.
7. Enable (unlock) a user account.
8. Disable (lock) a user account.
9. Change a user's password.
10. Exit the program.

✳️ Part 1: Add New User (Full Name – Username – Email – Password – Group)



The screenshot shows a terminal window titled "amiramohamed@localhost:~/whiptailFunctions — sudo nano add_user_module.sh". The terminal displays a Bash script for adding a new user. The script uses the whiptail dialog library to interact with the user. It includes functions for creating a single user, importing users from a file, and generating bulk users. The code handles user input for username, password, and other details, and provides feedback through message boxes.

```
GNU nano 5.6.1                                         add_user_module.sh
#!/bin/bash

create_single_user(){
    new_user=$(whiptail --title "Create New User" --inputbox "Enter new username:" 8 50 3>&1 1>&2 2>&3)
    if id "$new_user" >/dev/null; then
        whiptail --title "User Exists" --msgbox "The user '$new_user' already exists!" 8 50
    else
        new_pass=$(whiptail --title "Set Password" --passwordbox "Enter password for $new_user:" 8 50 3>&1 1>&2 2>&3)
        useradd "$new_user"
        echo "$new_pass" | passwd "$new_user" --stdin
        whiptail --title "Success" --msgbox "User '$new_user' has been created successfully." 8 50
    fi
}

import_users_from_file(){
    path_to_file=$(whiptail --title "Import Users" --inputbox "Enter full file path:" 8 50 3>&1 1>&2 2>&3)
    if [ -f "$path_to_file" ]; then
        while IFS=',' read -r user login_name user_pass; do
            if id "$user" >/dev/null; then
                original_name=$user
                user="${user}_$RANDOM"
                whiptail --title "Duplicate Found" --msgbox "'$original_name' already exists. Renaming to '$user'." 8 60
            fi
            useradd -c "$login_name" "$user"
            echo "$user_pass" | passwd "$user" --stdin
        done < "$path_to_file"
        whiptail --title "Done" --msgbox "All users imported successfully." 8 50
    else
        whiptail --title "Error" --msgbox "File '$path_to_file' not found!" 8 50
    fi
}

generate_bulk_users(){
    base_name=$(whiptail --title "Bulk User Creation" --inputbox "Enter base name for users:" 8 50 3>&1 1>&2 2>&3)
    total_users=$(whiptail --title "Bulk User Creation" --inputbox "How many users to create?" 8 50 3>&1 1>&2 2>&3)
    save_file=$(whiptail --title "Save Credentials" --inputbox "Enter file name to save credentials (inside 'credentials/' folder):" 8 60)

    mkdir -p credentials
    for ((i=1; i<=total_users; i++)); do
        temp_user="$base_name"
    done
}
```

* Part 2: Modify User Data (Change Name – UID – Home Directory – Lock/Unlock Account)

```
GNU nano 5.6.1                               modify_user_module.sh
#!/bin/bash

user_to_edit=$(whiptail --title "User Editor" --inputbox "Enter the username to update:" 8 50 3>&1 1>&2 2>&3)

if getent passwd "$user_to_edit" > /dev/null; then
    selection=$(whiptail --title "User Modification" --menu "Select an option for '$user_to_edit': " 15 60 4 \
        "1" "Rename user" \
        "2" "Update UID" \
        "3" "Add to group" \
        "4" "Change login shell" 3>&1 1>&2 2>&3)

    case $selection in
        1)
            new_user=$(whiptail --title "Rename User" --inputbox "Enter new username:" 8 50 3>&1 1>&2 2>&3)
            if getent passwd "$new_user" > /dev/null; then
                whiptail --title "Name Taken" --msgbox "Username '$new_user' is already in use." 8 50
            else
                usermod -l "$new_user" "$user_to_edit"
                whiptail --title "Updated" --msgbox "Username changed to '$new_user'." 8 50
            fi
            ;;
        2)
            new_uid=$(whiptail --title "Change UID" --inputbox "Enter new UID for '$user_to_edit': " 8 50 3>&1 1>&2 2>&3)
            if getent passwd | cut -d: -f3 | grep -qx "$new_uid"; then
                whiptail --title "UID Exists" --msgbox "UID '$new_uid' is already assigned." 8 50
            else
                usermod -u "$new_uid" "$user_to_edit"
                whiptail --title "Success" --msgbox "UID updated to '$new_uid'." 8 50
            fi
            ;;
        3)
            grp_name=$(whiptail --title "Add Group" --inputbox "Enter group name to add user to:" 8 50 3>&1 1>&2 2>&3)
            if getent group "$grp_name" > /dev/null; then
                usermod -aG "$grp_name" "$user_to_edit"
                whiptail --title "Success" --msgbox "User '$user_to_edit' added to group '$grp_name'." 8 50
            else
                whiptail --title "Group Not Found" --msgbox "Group '$grp_name' does not exist." 8 50
            fi
            ;;
        4)
            grp_name=$(whiptail --title "Remove Group" --inputbox "Enter group name to remove user from:" 8 50 3>&1 1>&2 2>&3)
            if getent group "$grp_name" > /dev/null; then
                usermod -aR "$grp_name" "$user_to_edit"
                whiptail --title "Success" --msgbox "User '$user_to_edit' removed from group '$grp_name'." 8 50
            else
                whiptail --title "Group Not Found" --msgbox "Group '$grp_name' does not exist." 8 50
            fi
            ;;
    esac

^G Help      ^O Write Out     ^W Where Is      ^K Cut          ^T Execute      ^C Location      M-U Undo      M-A Set Mark
^X Exit      ^R Read File     ^\ Replace       ^U Paste        ^J Justify      ^_ Go To Line    M-E Redo      M-G Copy
```

* Part 3: List all users

```
GNU nano 5.6.1                               list_User_module.sh
#!/bin/bash

choice=$(whiptail --title "User Listing Options" --menu "Select the type of listing:" 12 60 2 \
    "1" "Usernames Only" \
    "2" "Full User Info" 3>&1 1>&2 2>&3)

case "$choice" in
    1)
        user_list=$(cut -d: -f1 /etc/passwd)
        whiptail --title "Usernames" --scrolltext --msgbox "$user_list" 20 60
        ;;
    2)
        full_info=$(< /etc/passwd)
        whiptail --title "User Details" --scrolltext --msgbox "$full_info" 30 80
        ;;
esac
```

* Part 4: Create New Group (Group Name)

The screenshot shows a terminal window titled "amiramohamed@localhost:~/whiptailFunctions — sudo nano create_group_module.sh". The code in the editor is as follows:

```
GNU nano 5.6.1                                         create_group_module.sh
#!/bin/bash

create_group(){
    group_name=$(whiptail --title "Create New Group" --inputbox "Enter the group name:" 8 50 3>&1 1>&2 2>&3)
    if getent group "$group_name" > /dev/null; then
        whiptail --title "Group Exists" --msgbox "Group '$group_name' already exists!" 8 50
    else
        groupadd "$group_name"
        whiptail --title "Group Created" --msgbox "Group '$group_name' has been successfully created." 8 50
    fi
}

import_groups(){
    input_file=$(whiptail --title "Import Groups" --inputbox "Enter the full file path:" 8 50 3>&1 1>&2 2>&3)
    if [ -f "$input_file" ]; then
        while IFS=':' read -r grp members; do
            if getent group "$grp" > /dev/null; then
                whiptail --title "Group Exists" --msgbox "Group '$grp' already exists. Adding users to it." 8 60
            else
                groupadd "$grp"
                whiptail --title "Group Added" --msgbox "Group '$grp' created. Proceeding to add users." 8 60
            fi

            IFS=':' read -ra user_array <<< "$members"
            for usr in "${user_array[@]}"; do
                if id "$usr" &>/dev/null; then
                    gpasswd -a "$usr" "$grp"
                else
                    temp_user="${usr}_${RANDOM}"
                    whiptail --title "Creating User" --msgbox "User '$usr' not found. Creating as '$temp_user' with default password '1234'." 8 60
                    useradd "$temp_user"
                    echo "1234" | passwd "$temp_user" --stdin
                    gpasswd -a "$temp_user" "$grp"
                fi
            done
            done < "$input_file"
            whiptail --title "Done" --msgbox "All groups and users processed successfully." 8 50
        else
            whiptail --title "File Error" --msgbox "Could not locate file: '$input_file'" 8 50
        fi
    fi
}

^G Help          ^O Write Out      ^W Where Is      ^K Cut           ^T Execute       ^C Location      M-U Undo
^X Exit          ^R Read File      ^X Replace       ^U Paste         ^J Justify       ^L Go To Line    M-E Redo
                                         M-A Set Mark    M-G Copy
```

* Part 5: Modify Group (Rename – GID – Show Members – Set Password)

The screenshot shows a terminal window titled "amiramohamed@localhost:~/whiptailFunctions — sudo nano modify_group_module.sh". The code in the editor is as follows:

```
GNU nano 5.6.1                                         modify_group_module.sh
#!/bin/bash

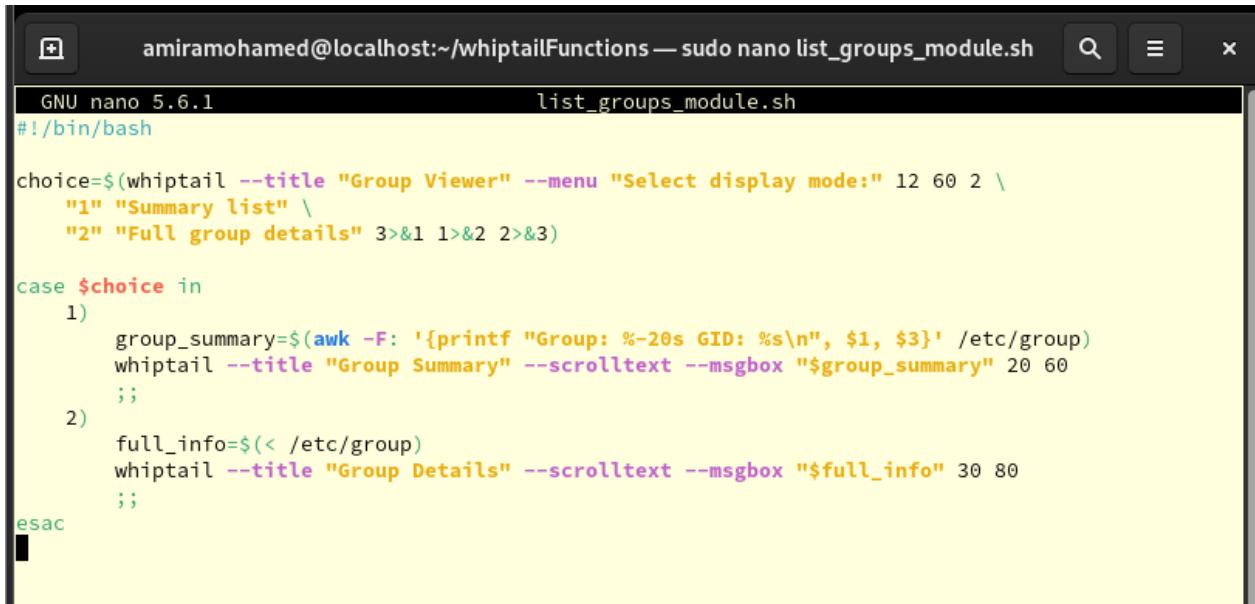
grp_name=$(whiptail --title "Edit Group" --inputbox "Enter the name of the group to edit:" 8 50 3>&1 1>&2 2>&3)

if getent group "$grp_name" > /dev/null; then
    action=$(whiptail --title "Group Modification" --menu "Select an action for '$grp_name': " 15 60 2 \
    "1" "Rename group" \
    "2" "Update GID" 3>&1 1>&2 2>&3)

    case $action in
        1)
            new_grp=$(whiptail --title "Rename Group" --inputbox "Enter the new group name:" 8 50 3>&1 1>&2 2>&3)
            if getent group "$new_grp" > /dev/null; then
                whiptail --title "Name Conflict" --msgbox "Group '$new_grp' already exists." 8 50
            else
                groupmod -n "$new_grp" "$grp_name"
                whiptail --title "Success" --msgbox "Group '$grp_name' renamed to '$new_grp'." 8 50
            fi
        ;;
        2)
            new_gid=$(whiptail --title "Change Group ID" --inputbox "Enter a new GID for '$grp_name': " 8 50 3>&1 1>&2 2>&3)
            if getent group | cut -d: -f3 | grep -q "^${new_gid}"; then
                whiptail --title "GID In Use" --msgbox "GID '$new_gid' is already assigned to another group." 8 50
            else
                groupmod -g "$new_gid" "$grp_name"
                whiptail --title "Success" --msgbox "GID for '$grp_name' changed to '$new_gid'." 8 50
            fi
        ;;
    esac
else
    whiptail --title "Group Not Found" --msgbox "Group '$grp_name' does not exist." 8 50
fi
```

* Part 6: List All Groups

Displays all the collections in the system in a formatted way using cut and sort, and displays them in a scrollable screen using less.

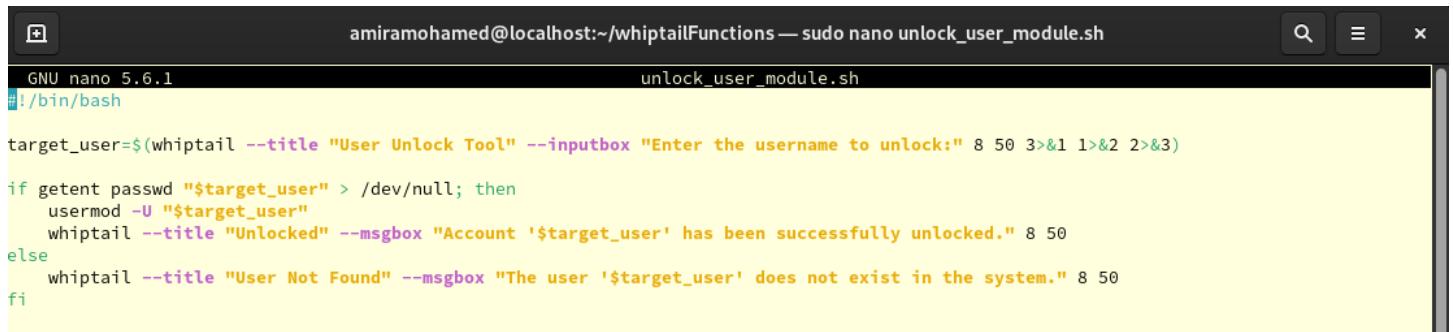


```
GNU nano 5.6.1          list_groups_module.sh
#!/bin/bash

choice=$(whiptail --title "Group Viewer" --menu "Select display mode:" 12 60 2 \
    "1" "Summary list" \
    "2" "Full group details" 3>&1 1>&2 2>&3)

case $choice in
    1)
        group_summary=$(awk -F: '{printf "Group: %-20s GID: %s\n", $1, $3}' /etc/group)
        whiptail --title "Group Summary" --scrolltext --msgbox "$group_summary" 20 60
        ;;
    2)
        full_info=$(cat /etc/group)
        whiptail --title "Group Details" --scrolltext --msgbox "$full_info" 30 80
        ;;
esac
```

* Part 7: Unlock a User Account

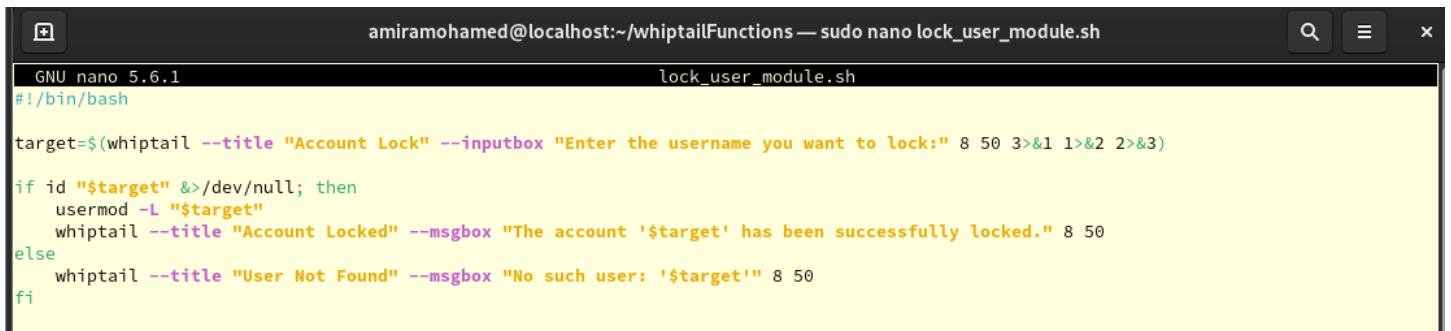


```
GNU nano 5.6.1          unlock_user_module.sh
#!/bin/bash

target_user=$(whiptail --title "User Unlock Tool" --inputbox "Enter the username to unlock:" 8 50 3>&1 1>&2 2>&3)

if getent passwd "$target_user" > /dev/null; then
    usermod -U "$target_user"
    whiptail --title "Unlocked" --msgbox "Account '$target_user' has been successfully unlocked." 8 50
else
    whiptail --title "User Not Found" --msgbox "The user '$target_user' does not exist in the system." 8 50
fi
```

* Part 8: Lock a User Account

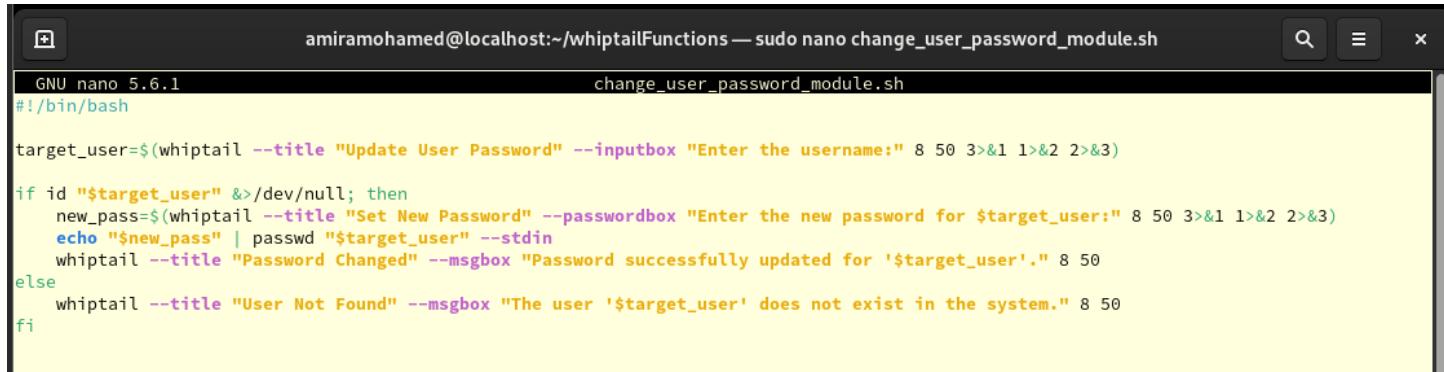


```
GNU nano 5.6.1          lock_user_module.sh
#!/bin/bash

target=$(whiptail --title "Account Lock" --inputbox "Enter the username you want to lock:" 8 50 3>&1 1>&2 2>&3)

if id "$target" >/dev/null; then
    usermod -L "$target"
    whiptail --title "Account Locked" --msgbox "The account '$target' has been successfully locked." 8 50
else
    whiptail --title "User Not Found" --msgbox "No such user: '$target'" 8 50
fi
```

* Part 9: Change a User's Password



amiramohamed@localhost:~/whiptailFunctions — sudo nano change_user_password_module.sh

```
GNU nano 5.6.1                               change_user_password_module.sh
#!/bin/bash

target_user=$(whiptail --title "Update User Password" --inputbox "Enter the username:" 8 50 3>&1 1>&2 2>&3)

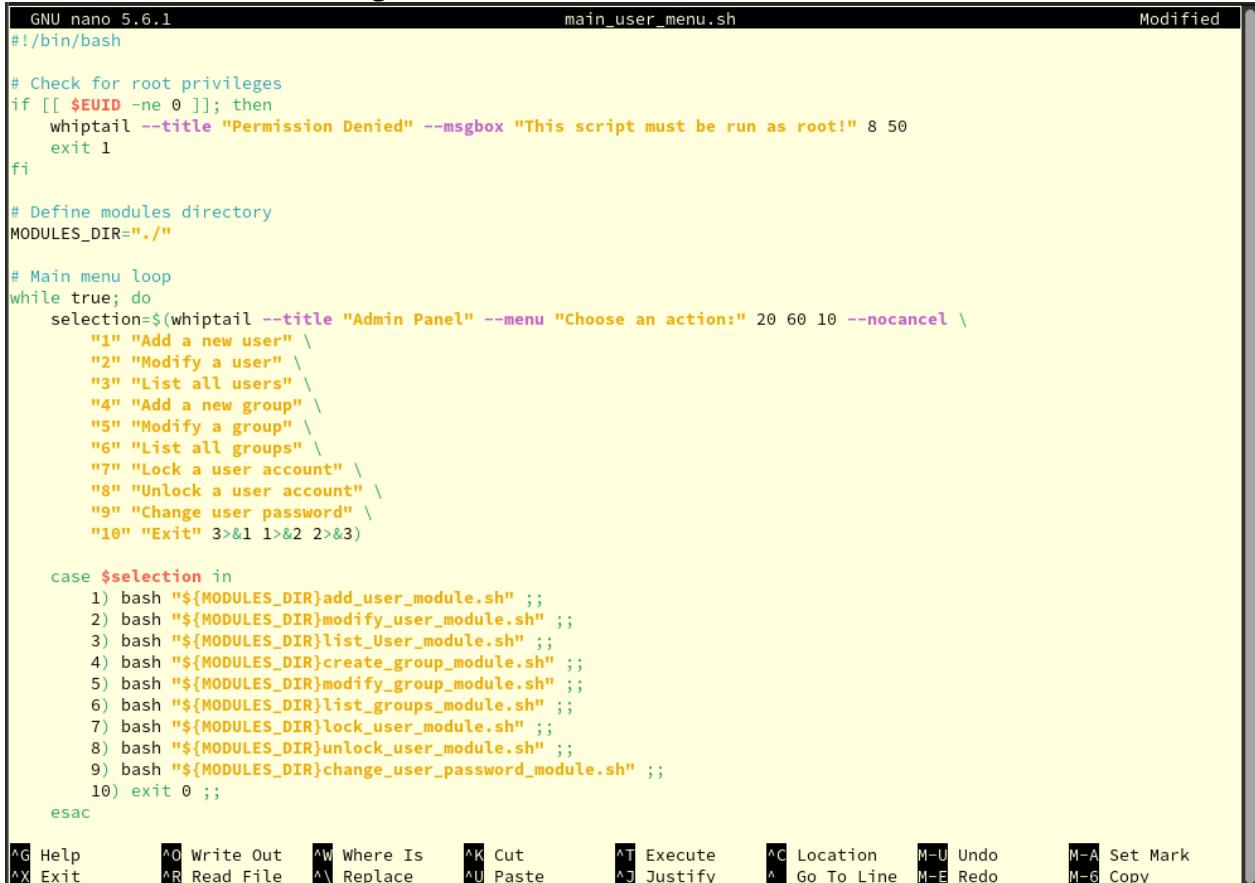
if id "$target_user" &>/dev/null; then
    new_pass=$(whiptail --title "Set New Password" --passwordbox "Enter the new password for $target_user:" 8 50 3>&1 1>&2 2>&3)
    echo "$new_pass" | passwd "$target_user" --stdin
    whiptail --title "Password Changed" --msgbox "Password successfully updated for '$target_user'." 8 50
else
    whiptail --title "User Not Found" --msgbox "The user '$target_user' does not exist in the system." 8 50
fi
```

* Part 10: Exit the Program

10)

```
whiptail --msgbox "Exiting program. Bye Amira! 😊" 10 50
exit
::
```

* Main Menu – User & Group Management (Call Modules – View Users – Manage Groups Lock/Unlock Accounts – Change Password – Exit)



GNU nano 5.6.1 main_user_menu.sh

```
GNU nano 5.6.1                               main_user_menu.sh
#!/bin/bash

# Check for root privileges
if [[ $EUID -ne 0 ]]; then
    whiptail --title "Permission Denied" --msgbox "This script must be run as root!" 8 50
    exit 1
fi

# Define modules directory
MODULES_DIR="./"

# Main menu loop
while true; do
    selection=$(whiptail --title "Admin Panel" --menu "Choose an action:" 20 60 10 --nocancel \
        "1" "Add a new user" \
        "2" "Modify a user" \
        "3" "List all users" \
        "4" "Add a new group" \
        "5" "Modify a group" \
        "6" "List all groups" \
        "7" "Lock a user account" \
        "8" "Unlock a user account" \
        "9" "Change user password" \
        "10" "Exit" 3>&1 1>&2 2>&3)

    case $selection in
        1) bash "${MODULES_DIR}add_user_module.sh" ;;
        2) bash "${MODULES_DIR}modify_user_module.sh" ;;
        3) bash "${MODULES_DIR}list_User_module.sh" ;;
        4) bash "${MODULES_DIR}create_group_module.sh" ;;
        5) bash "${MODULES_DIR}modify_group_module.sh" ;;
        6) bash "${MODULES_DIR}list_groups_module.sh" ;;
        7) bash "${MODULES_DIR}lock_user_module.sh" ;;
        8) bash "${MODULES_DIR}unlock_user_module.sh" ;;
        9) bash "${MODULES_DIR}change_user_password_module.sh" ;;
        10) exit 0 ;;
    esac
done
```

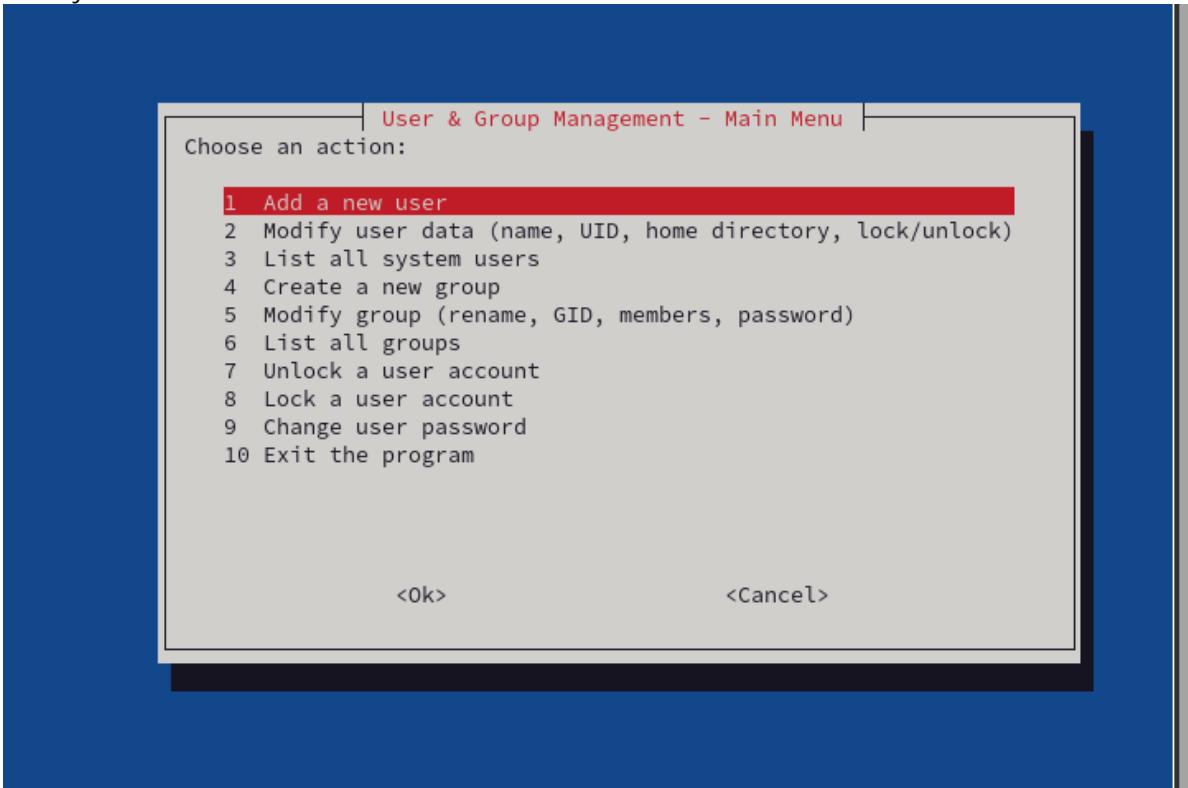
Modified

Keyboard Shortcuts:

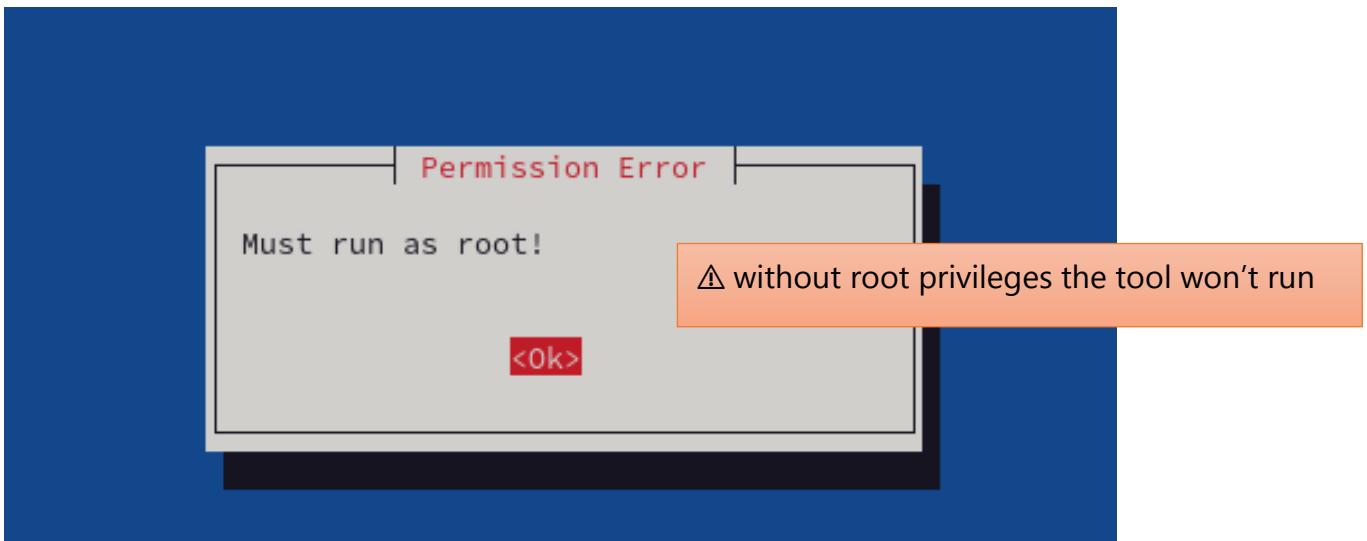
- ^G Help
- ^X Exit
- ^O Write Out
- ^R Read File
- ^W Where Is
- ^V Replace
- ^K Cut
- ^U Paste
- ^T Execute
- ^J Justify
- ^C Location
- ^_ Go To Line
- M-U Undo
- M-E Redo
- M-A Set Mark
- M-G Copy

```
[amiramohamed@localhost ~]$ sudo nano add_user_module.sh
[sudo] password for amiramohamed:
[amiramohamed@localhost ~]$ sudo nano modify_user_module.sh
[amiramohamed@localhost ~]$ sudo chmod +x add_user_module.sh
[amiramohamed@localhost ~]$ sudo chmod +x modify_user_module.sh
[amiramohamed@localhost ~]$ sudo nano main_user_menu.sh
[sudo] password for amiramohamed:
[amiramohamed@localhost ~]$ sudo chmod +x main_user_menu.sh
[amiramohamed@localhost ~]$ sudo nano create_group_module.sh
[sudo] password for amiramohamed:
[amiramohamed@localhost ~]$ sudo nano modify_group_module.sh
[amiramohamed@localhost ~]$ sudo chmod +x create_group_module.sh
[amiramohamed@localhost ~]$ sudo chmod +x modify_group_module.sh
[amiramohamed@localhost ~]$ sudo nano list_groups_module.sh
[sudo] password for amiramohamed:
[amiramohamed@localhost ~]$ sudo nano unlock_user_module.sh
[amiramohamed@localhost ~]$ sudo chmod +x unlock_user_module.sh
[amiramohamed@localhost ~]$ sudo nano lock_user_module.sh
[amiramohamed@localhost ~]$ sudo chmod +x lock_user_module.sh
[amiramohamed@localhost ~]$ sudo nano change_user_password_module.sh
[amiramohamed@localhost ~]$ chmod +x change_user_password_module.sh
chmod: changing permissions of 'change_user_password_module.sh': Operation not permitted
[amiramohamed@localhost ~]$ sudo chmod +x change_user_password_module.sh
[amiramohamed@localhost ~]$ ./main_user_menu.sh
[amiramohamed@localhost ~]$
```

- # **2** Make the 'main' file executable chmod +x main
 - # **3** Start the tool with superuser access sudo ./main
 - #  Ready to roll!



* Option 1: Add a new user



🚫 Adding a New User

From the Main Menu, select "Add User" — you'll then pick from 3 available modes.

✳️ Mode 1: Add a User

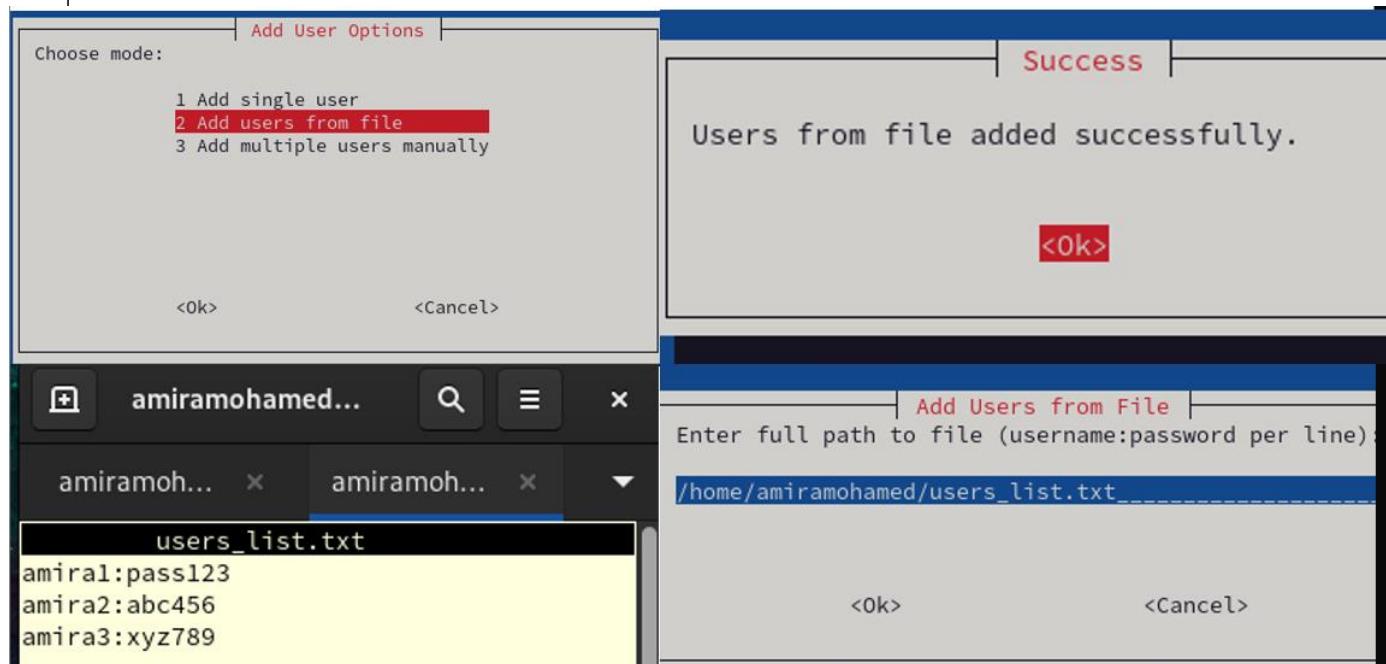
This is a simple function that asks you to enter a username and password,
then automatically creates the user on the system.



📁 Mode 2: Add Users from a File

This mode simplifies the process by automatically creating users from a file
(/home/amiramohamed/users_list.txt).

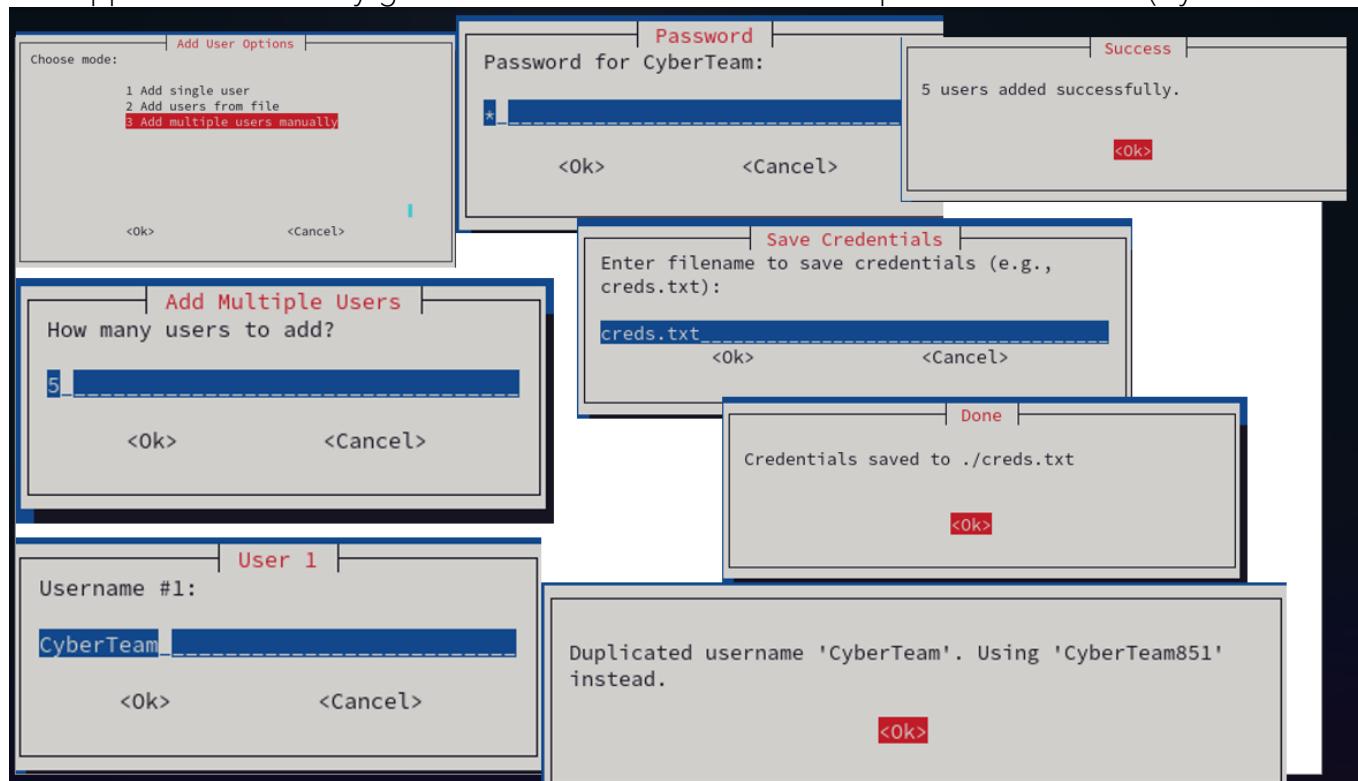
The tool reads the file, extracts the credentials, and creates the accounts accordingly.
If it detects a duplicate username, it appends a random number to the end to ensure uniqueness.



⭐ Mode 3 – Add Multiple Users Manually

This option is ideal for quickly setting up multiple user accounts for a team or department, handling both account creation and credential storage automatically.

For example, the tool first creates a user named (CyberTeam).
When it attempts to create another user with the same name and detects a duplicate, it appends a randomly generated number to create a unique username like (CyberTeam827).

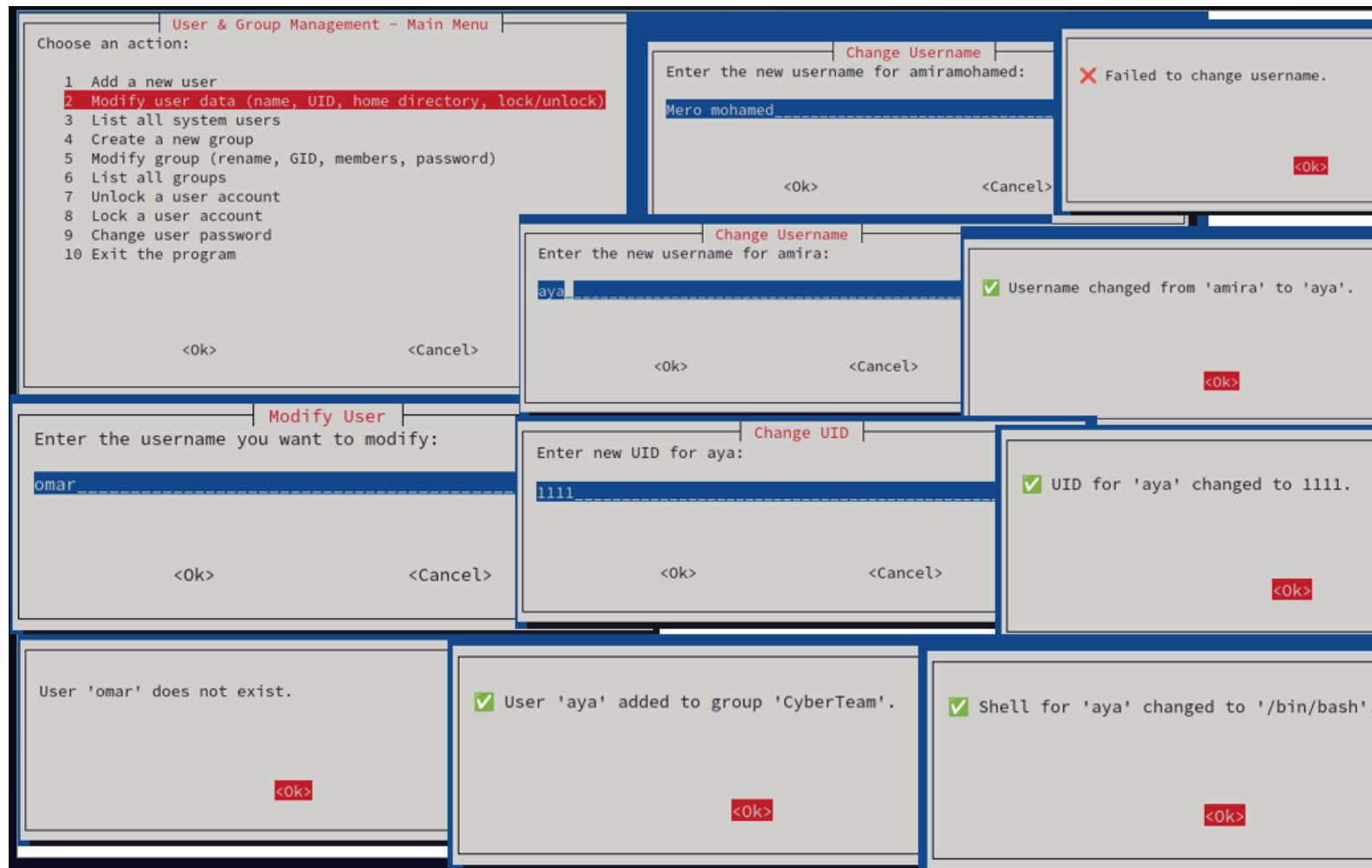


* Option 2: Modify user data

- # 🔎 If you enter a group name that doesn't exist, the tool will display an error message indicating the issue.
- # ✅ When a valid group name is entered, the user will be successfully added to that group.

🔧 Option : Change Shell

This feature allows you to modify the user's default shell to a different one of your choice.



* Option 3: List all users

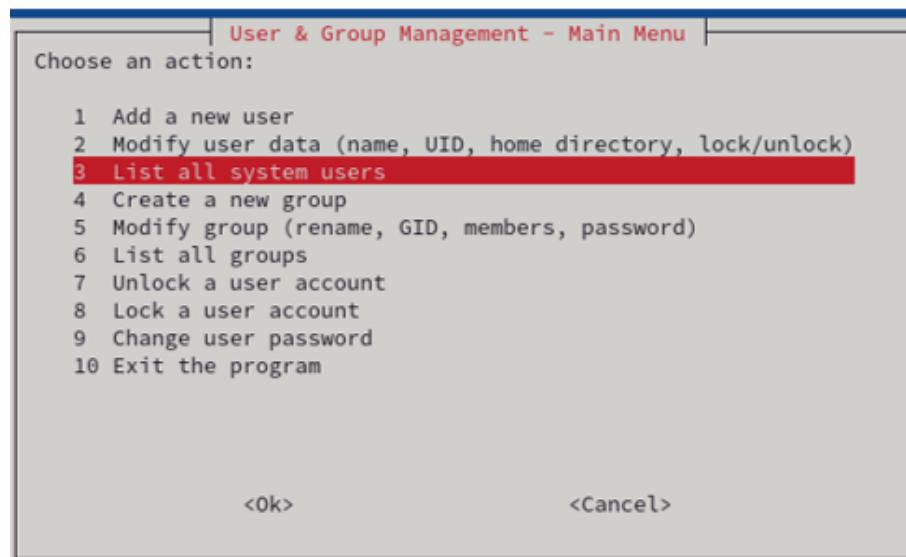
Selecting "List All Users" from the Main Menu will prompt you to choose a display mode:

◆ Mode 1: Short View

Shows a simple list of all usernames on the system.

◆ Mode 2: Detailed View

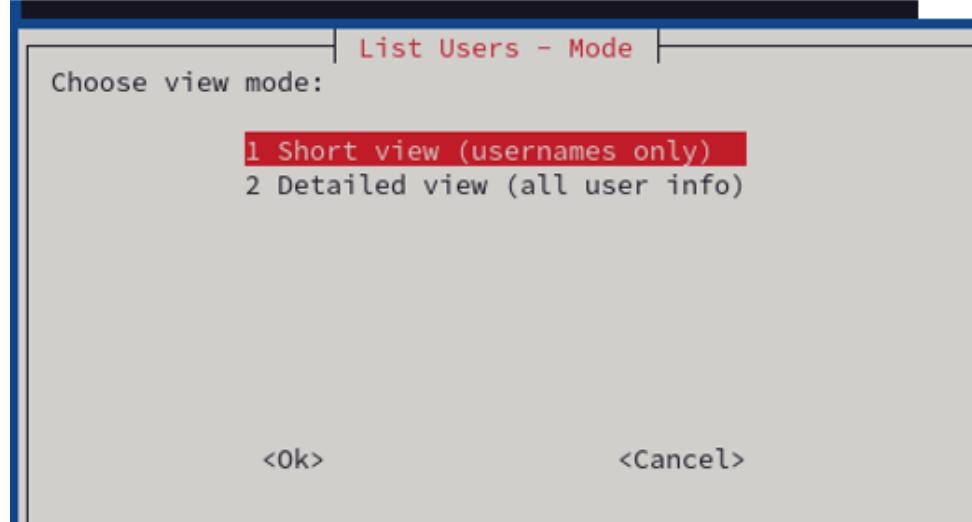
Displays comprehensive information about each user, including all available account details.



System Users - Short V

```
adm
amira
amiral
amira2
amira3
amiramohamed
amiramohamedmenshawy
amirauser
avahi
bin
chrony
clevis
cockpit-ws
cockpit-wsinstance
```

<Ok>



System Users - Detailed

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin.sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:65534:65534:Kernel Overflow User::/
systemd-coredump:x:999:997:systemd Core Dumper
dbus:x:81:81:System message bus:/sbin/nologin
```

<Ok>

* Option 4: Add a Group

When selecting "Add a Group" from the Main Menu, you'll be asked to choose a mode:

◆ Mode 1: Add a Group

A simple option that asks for a group name and creates it on the system.

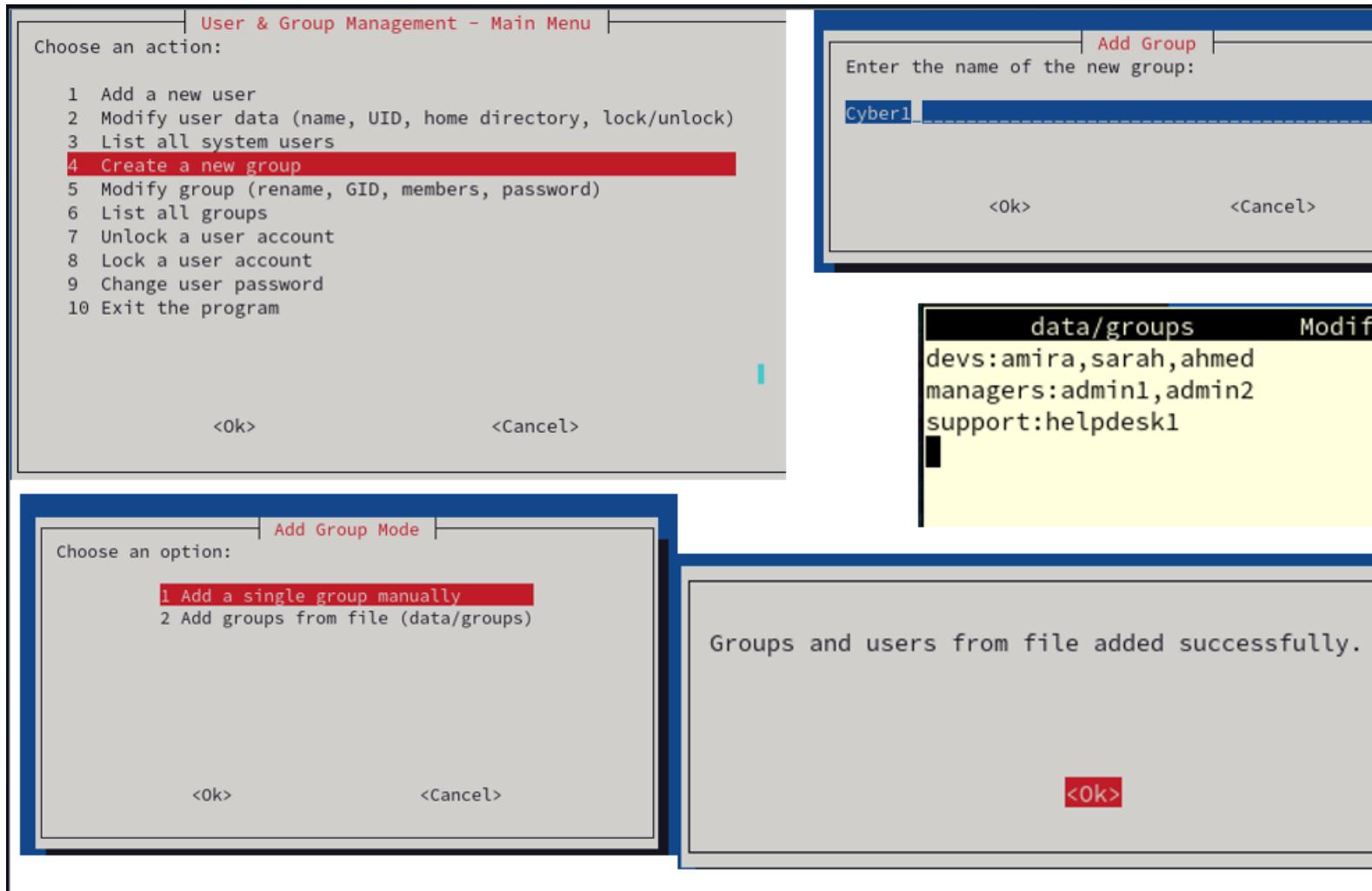
◆ Mode 2: Add Groups from a File

Automates the process by reading group info from the file `data/groups`.

The tool reads the group name, creates it, then adds its associated users.

🧑 If a user mentioned in the file doesn't exist, the tool creates a new user account, appending a random number to ensure uniqueness.

✎ If the group already exists, it simply adds the users to that group.



* Option 5: Modify a Group

From the Main Menu, selecting "Modify a Group" will prompt you to enter the group name you wish to modify.

- # ✗ If the group name doesn't exist, the tool will display an error message.
- # ✓ If a valid group name is entered, a modification menu will appear with the following options:

- # ◆ Option 1: Change Group Name

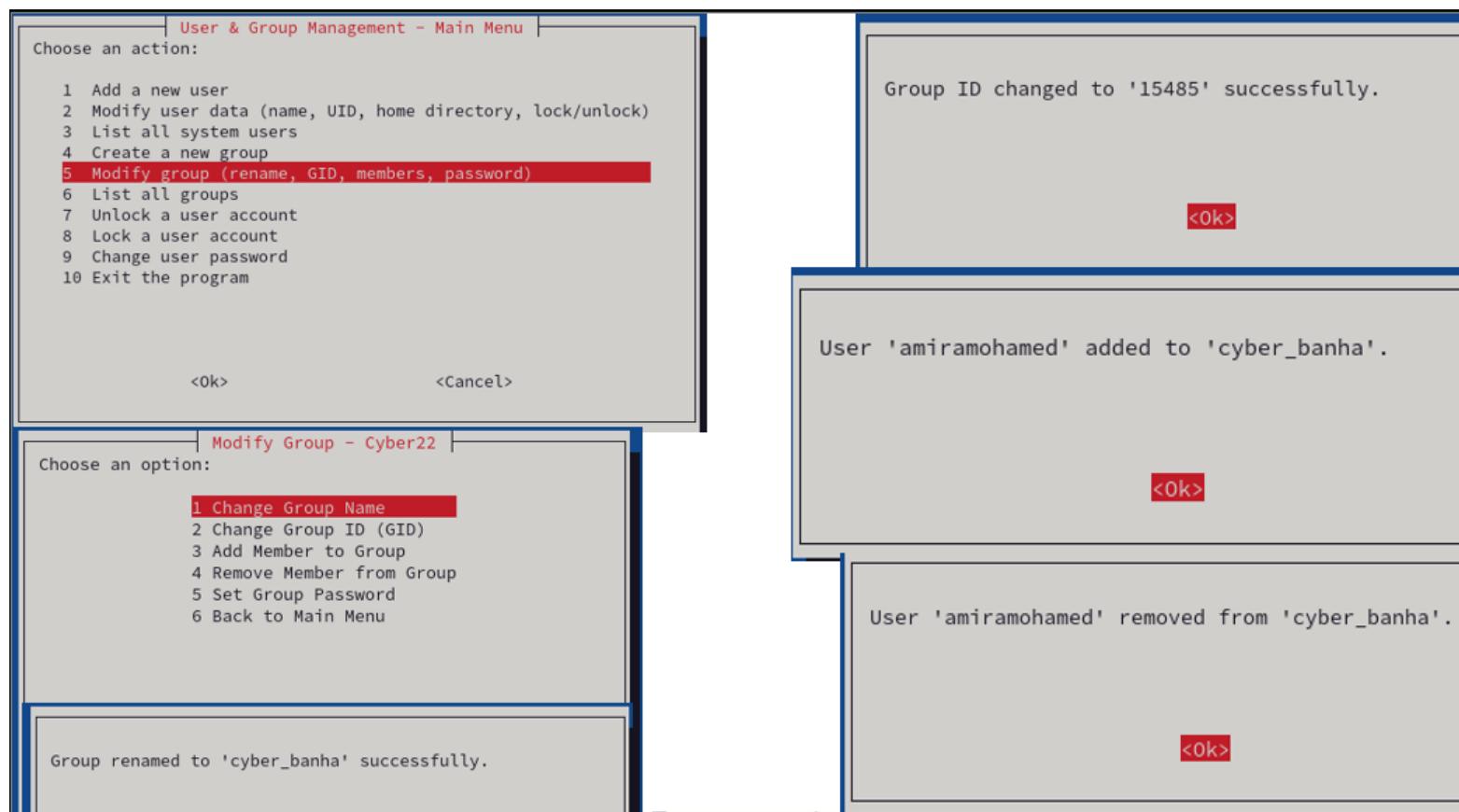
- If the new group name already exists, an error will be shown.

- Otherwise, the group name will be successfully updated.

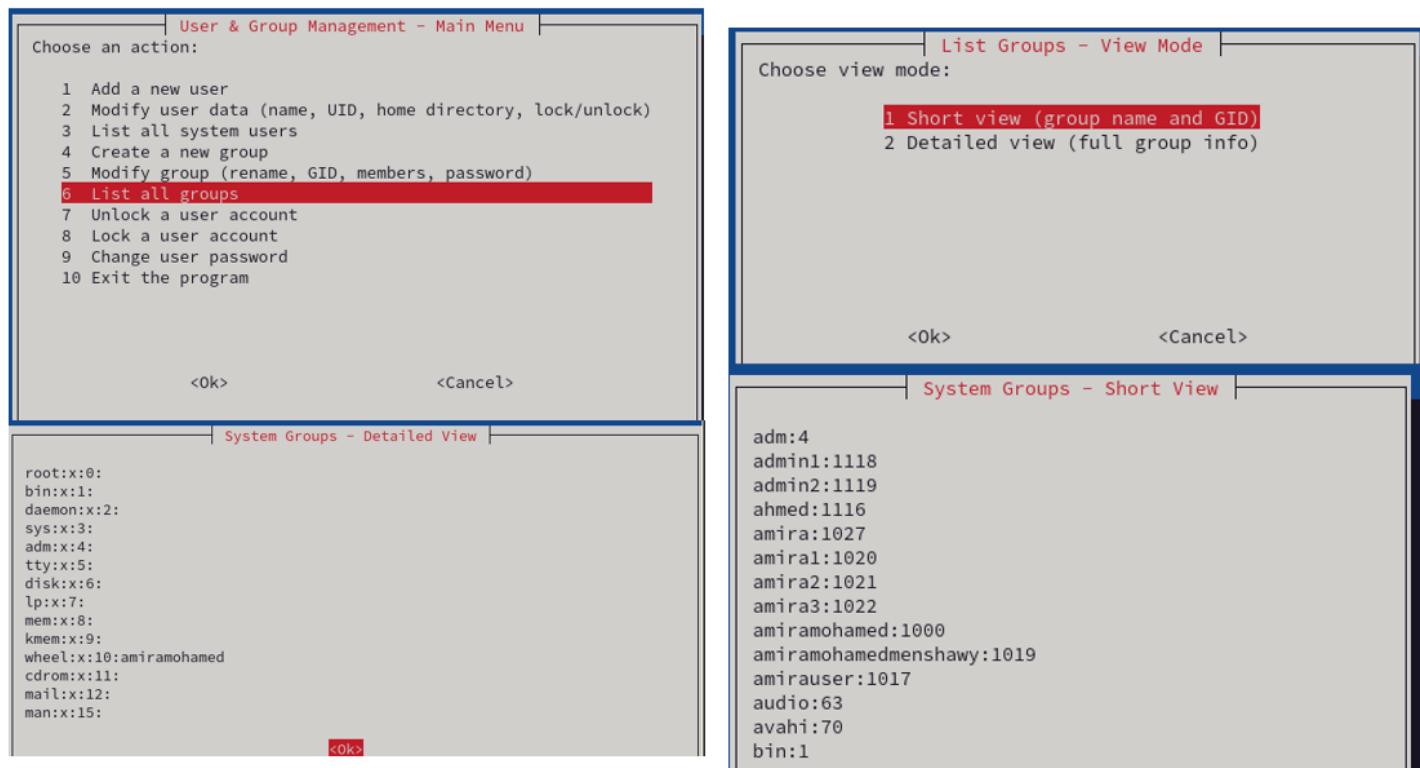
- # ◆ Option 2: Change Group ID (GID)

- If the entered GID is already in use by another group, the tool will display an error.

- If the GID is available, it will be assigned to the selected group without issues.



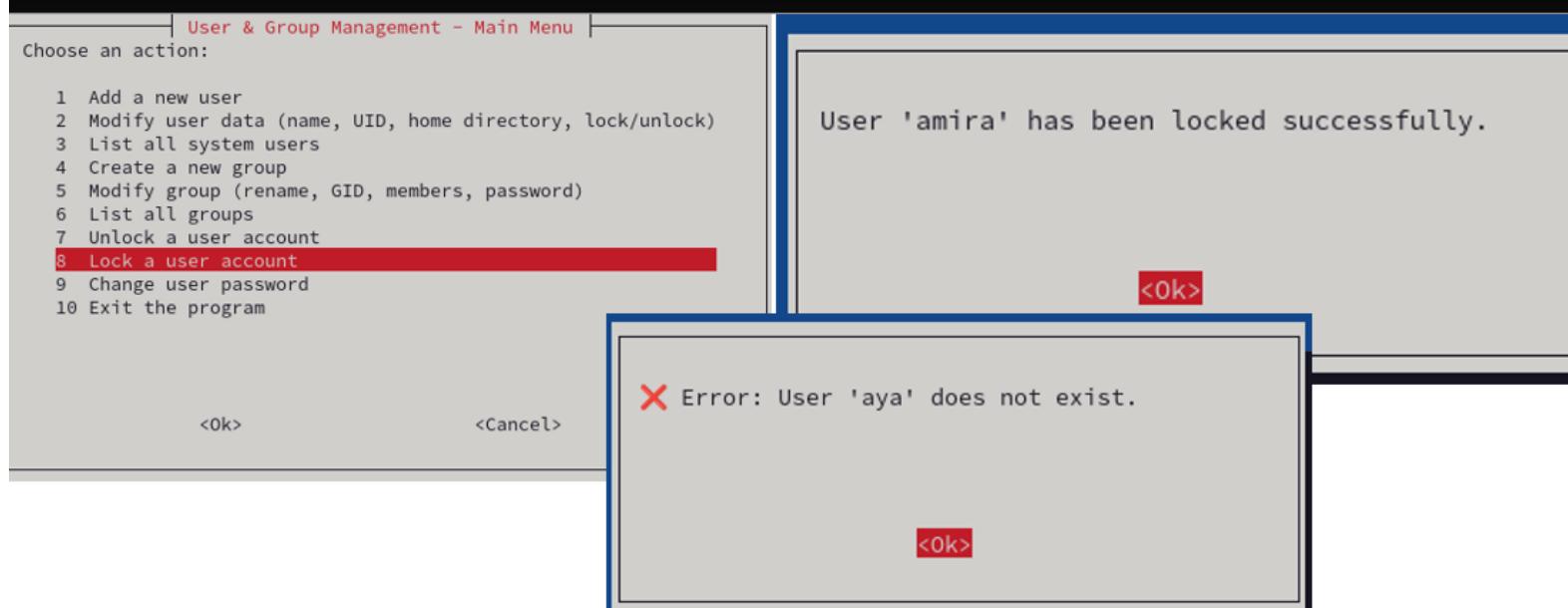
* Option 6: Lists all Groups



* Option 8 : Lock account

Selecting "Lock a User Account" from the Main Menu will prompt you to enter the target username.

- # ❌ If the entered username doesn't exist, the tool will display an error message.
- # ✅ If the username is valid and exists on the system, the account will be successfully locked.

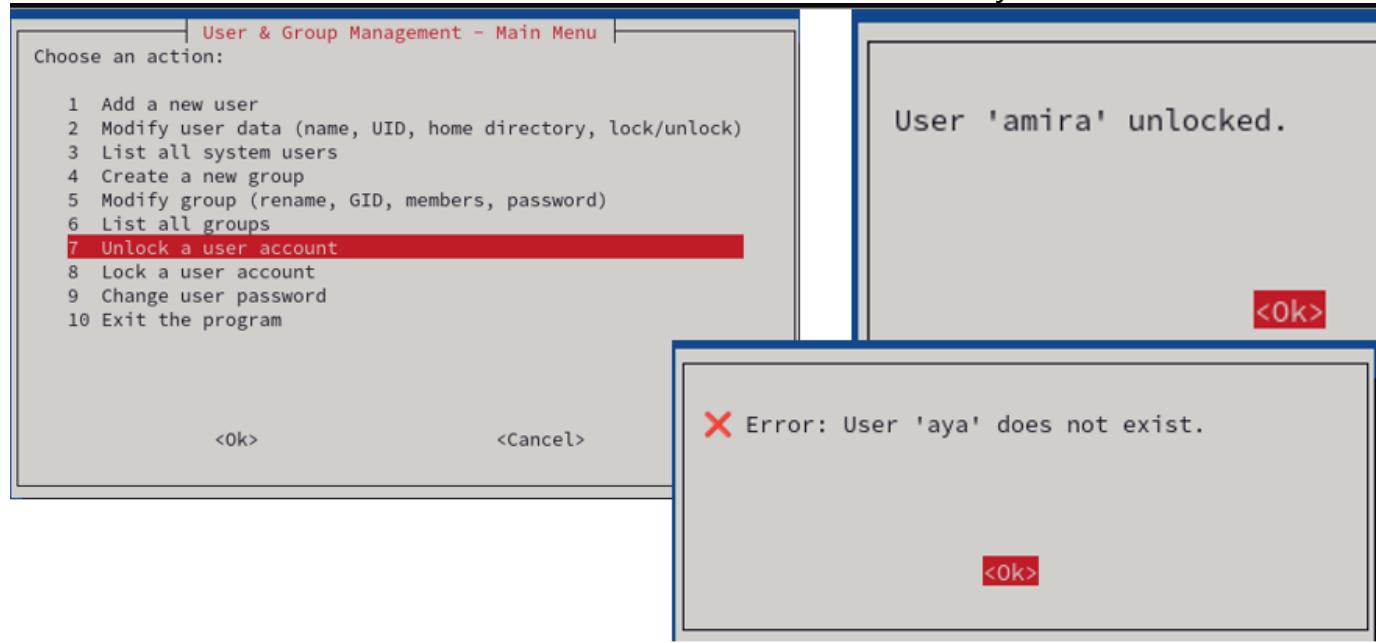


* Option 7 : UnLock account

Choosing "Unlock a User Account" from the Main Menu will prompt you to enter the username you want to unlock.

✗ If the username doesn't exist on the system, an error message will be shown.

✓ If the username is valid, the account will be successfully unlocked.



* Option 9 : Change a user's password

