|  |  |  |
| --- | --- | --- |
| STUDENTS NAME | ID\_STUDENT | TASK 3 |
| LARTEY, KEREN KUMA | 11173490 | user management |
| IHONGUI, FABRICE | 11004310 |
| QUAYE, BRUNELLA NAA KOSHIE | 11215194 |
| EDUSEI, DERRICK AKUAMOAH | 11115682 |
| ADDO, NAA SHIDAA ADJORKOR | 11307343 |

**DCIT 206: SYSTEMS ADMINISTRATION**

**END OF SEM GROUP PROJECT**

User management involves creating, modifying, and deleting user accounts and managing their permissions to ensure that all users can access the resources they need while maintaining security.

Steps for User Management:

1. Create User Accounts
2. Set Permissions
3. Configure Authentication Policies
4. Monitor User Activity
5. Update User Profiles
6. Remove Inactive Accounts
7. Document User Management Policies

We are going to build a Bash script to automate all user management steps:

#!/bin/bash

SCRIPT\_PASSWORD="DCIT206"

DEFAULT\_PASSWORD="DCIT206"

# Prompt for script execution password

read -sp "Enter script password: " INPUT\_PASSWORD

echo

if [[ "$INPUT\_PASSWORD" != "$SCRIPT\_PASSWORD" ]]; then

echo "Incorrect password. Exiting."

exit 1

fi

# Function to add a user

add\_user() {

read -p "Enter username: " USERNAME

list\_groups

read -p "Enter group name: " GROUPNAME

# Check if group exists

if ! getent group "$GROUPNAME" > /dev/null; then

echo "Group $GROUPNAME does not exist. Please create it first."

return

fi

sudo useradd -m -g "$GROUPNAME" -s /bin/bash "$USERNAME"

echo "$USERNAME:$DEFAULT\_PASSWORD" | sudo chpasswd

echo "User $USERNAME added with default password."

}

# Function to delete a user

delete\_user() {

read -p "Enter username: " USERNAME

sudo userdel -r "$USERNAME"

echo "User $USERNAME deleted."

}

# Function to list users

list\_users() {

cut -d: -f1 /etc/passwd

}

# Function to create a group

create\_group() {

read -p "Enter group name: " GROUPNAME

sudo groupadd "$GROUPNAME"

echo "Group $GROUPNAME created."

}

# Function to delete a group

delete\_group() {

read -p "Enter group name: " GROUPNAME

sudo groupdel "$GROUPNAME"

echo "Group $GROUPNAME deleted."

}

# Function to list groups

list\_groups() {

cut -d: -f1 /etc/group

}

# Function to modify user

modify\_user() {

read -p "Enter username to modify: " USERNAME

list\_groups

read -p "Enter new group name: " NEW\_GROUPNAME

# Check if group exists

if ! getent group "$NEW\_GROUPNAME" > /dev/null; then

echo "Group $NEW\_GROUPNAME does not exist. Please create it first."

return

fi

sudo usermod -g "$NEW\_GROUPNAME" "$USERNAME"

echo "User $USERNAME's group changed to $NEW\_GROUPNAME."

}

# Function to monitor user activity

monitor\_user\_activity() {

last

}

# Main menu

while true; do

echo "User Management Menu"

echo "1. Add User"

echo "2. Delete User"

echo "3. List Users"

echo "4. Modify User"

echo "5. Monitor User Activity"

echo "6. Create Group"

echo "7. Delete Group"

echo "8. List Groups"

echo "9. Exit"

read -p "Choose an option: " OPTION

case $OPTION in

1) add\_user ;;

2) delete\_user ;;

3) list\_users ;;

4) modify\_user ;;

5) monitor\_user\_activity ;;

6) create\_group ;;

7) delete\_group ;;

8) list\_groups ;;

9) exit ;;

\*) echo "Invalid option. Please try again. ”;;

esac

done

**Explanation:**

* **Password Protection:** The script prompts for a password before execution. If the password doesn't match, the script exits.
* **Group Management:** Added options to create, delete, and list groups.
* **User Addition:** When adding a user, you can choose from existing groups. If the chosen group does not exist, the script prompts you to create it first.

**Benefits:**

The benefits of automating system administration tasks in Linux are multifaceted, encompassing time-saving efficiency, reduced errors, and enhanced consistency and security. Automation streamlines repetitive tasks, such as user and group management, through batch processing and standard procedures, which minimizes human error and ensures consistent execution. With features like password protection and default password settings, security is bolstered, while interactive menus and guided input simplify usage, making user management intuitive even for those unfamiliar with complex command syntax. Additionally, modular functions and monitoring capabilities offer flexibility and insights into user activity, making this approach scalable for managing environments of any size, from educational institutions to corporate settings, by ensuring efficiency, security, and ease of use.

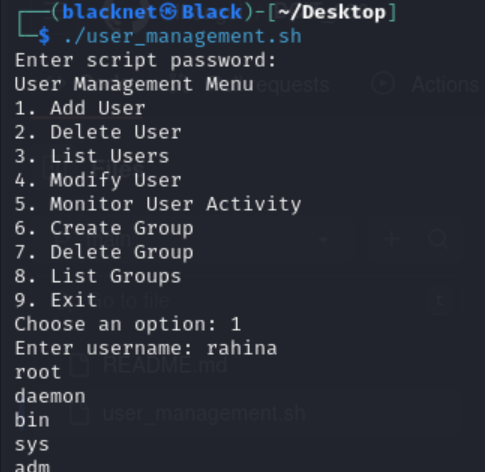
Provide a screenshot of how the script works:

* **First step executes ./user\_management.sh**

A screenshot of a computer

Description automatically generated

* **Step 2 if we chose Option 1 add user we have:**

A screenshot of a computer

Description automatically generated

Before continue create an user we need to chose a group where we need our user be