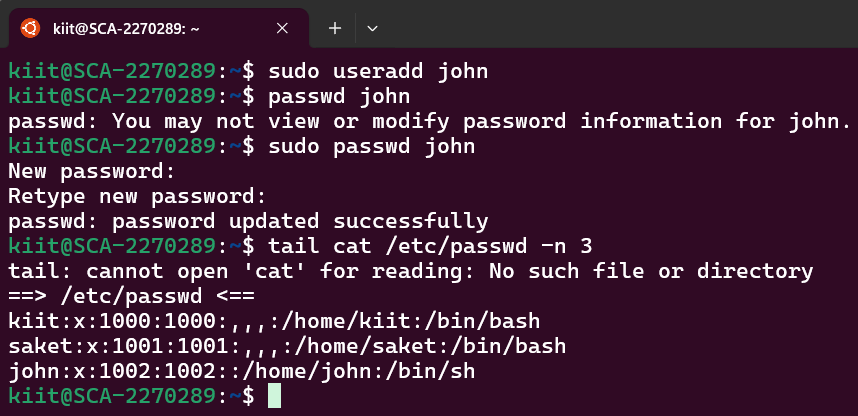
1. Use the useradd command to create a new user, e.g., john.

2. Set a password for the new user using passwd.

3. Verify the new user by checking the /etc/passwd file.

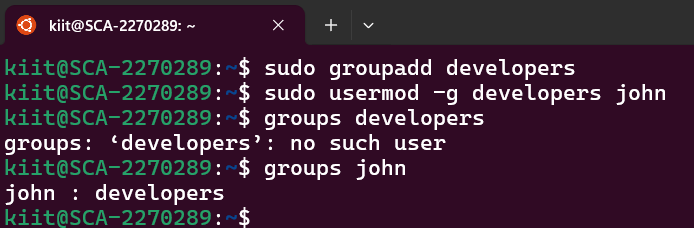
**Solution :**

****

1. Create a new group (e.g., developers) using groupadd.

2. Add an existing user (e.g., john) to the group using usermod.

3. Verify that the user is added to the group by using the groups command.

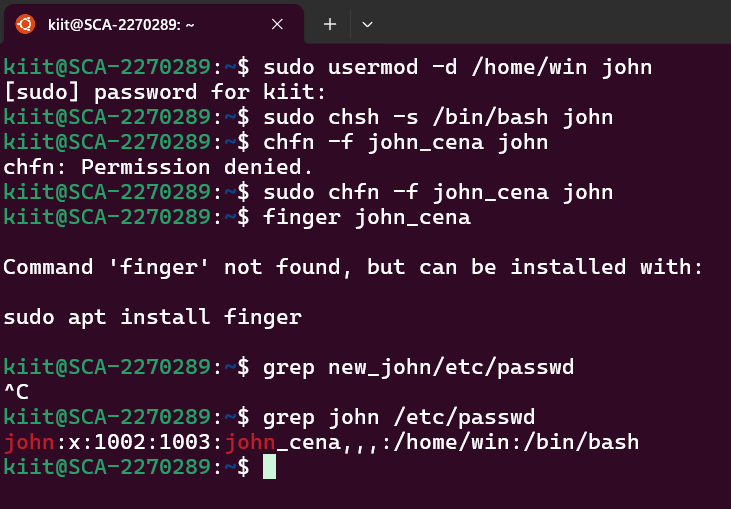
****

1. Modify the home directory for user john using usermod.

2. Change the default shell for john to /bin/bash.

3. Change the user’s full name using the chfn command.

4. Verify the changes using grep john /etc/passwd.

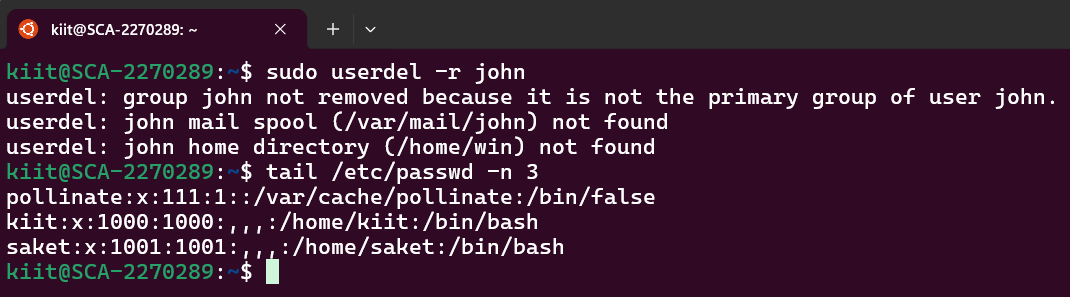
****

1. Delete the user john using the userdel command.

2. Ensure the user's home directory and files are removed by using userdel -r.

3. Verify the deletion by checking the /etc/passwd file.

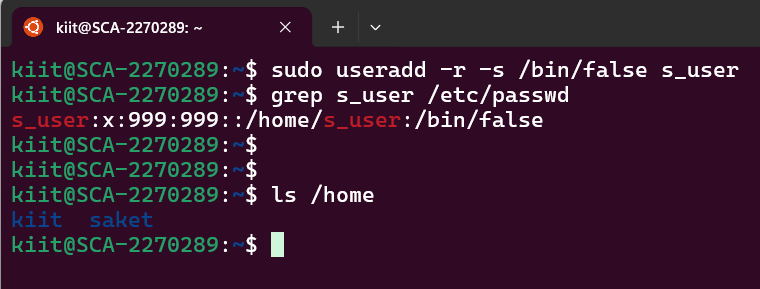
**Solution :**

****

1. Create a system user for an application (e.g., www-data for web server users).

2. Ensure that the system user has no login shell and that no home directory is created by using useradd -r.

3. Verify the user is created with no login shell by inspecting /etc/passwd.

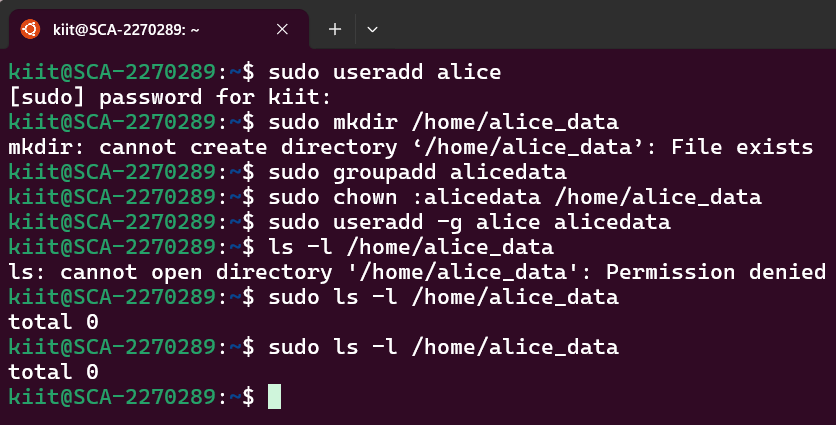


1. Create a new user alice.

2. Create a directory /home/alice\_data and set it as rw for the owner, r for the group, and no permissions for others.

3. Add alice to the group that has access to this directory.

4. Verify the permissions using ls -l.

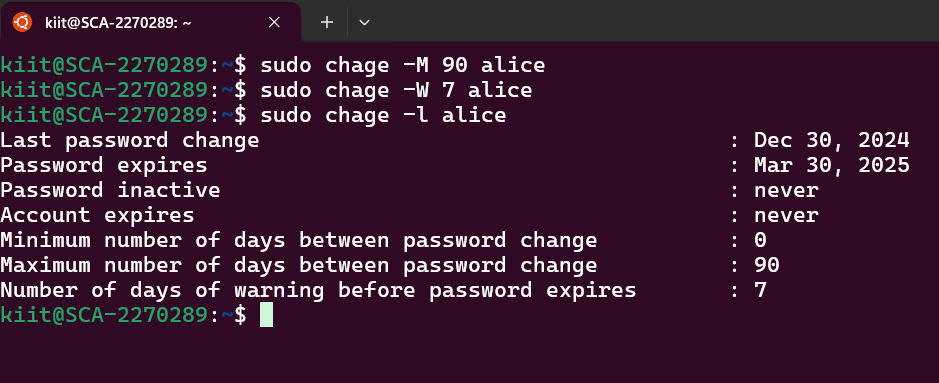


1. Set a password expiration period of 90 days for user alice using chage.

2. Set a warning period to notify the user 7 days before the password expires.

3. Verify the changes using chage -l alice.

**Solution :**

****

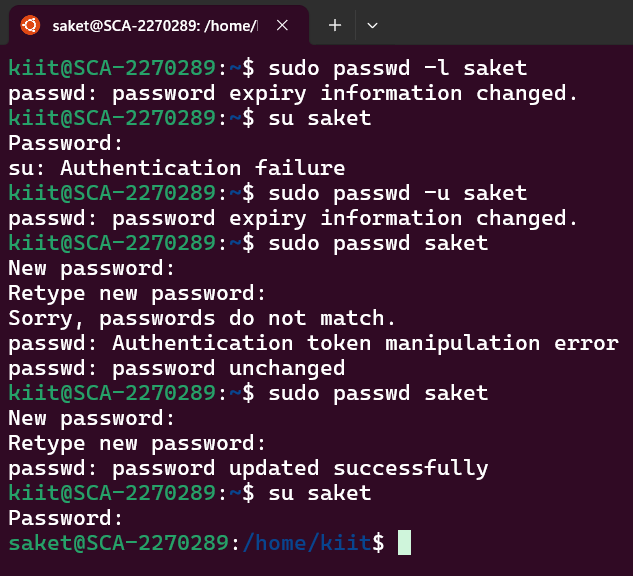
**Objective:** Learn how to lock and unlock user accounts.

1. Lock the user account alice by using the passwd -l command.

2. Verify that the account is locked by trying to log in as alice.

3. Unlock the account using the passwd -u command.

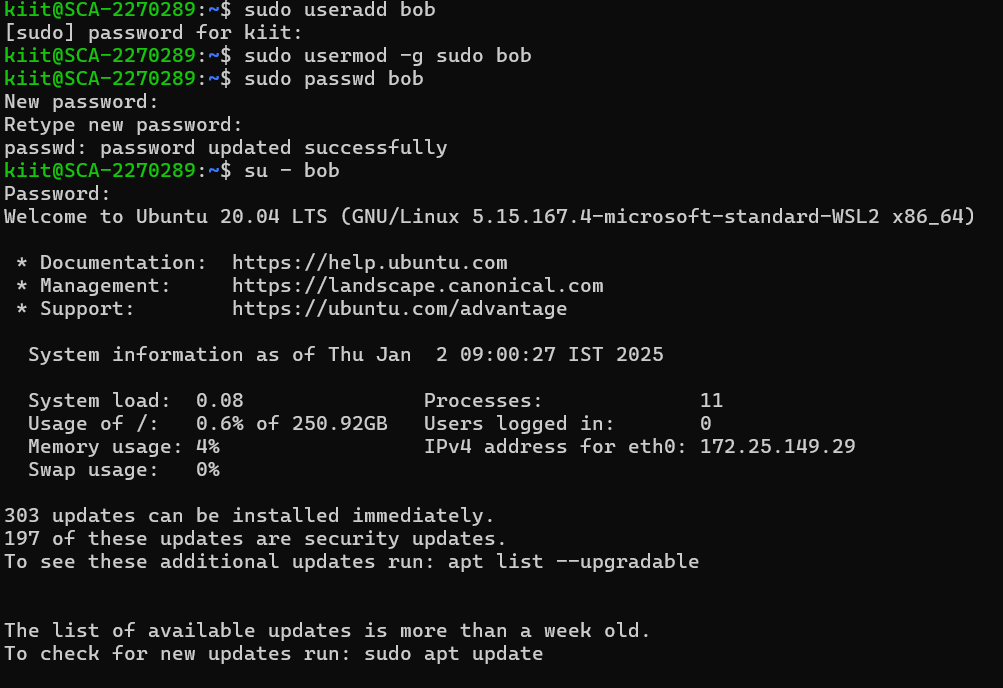
4. Verify the account is unlocked by trying to log in again.

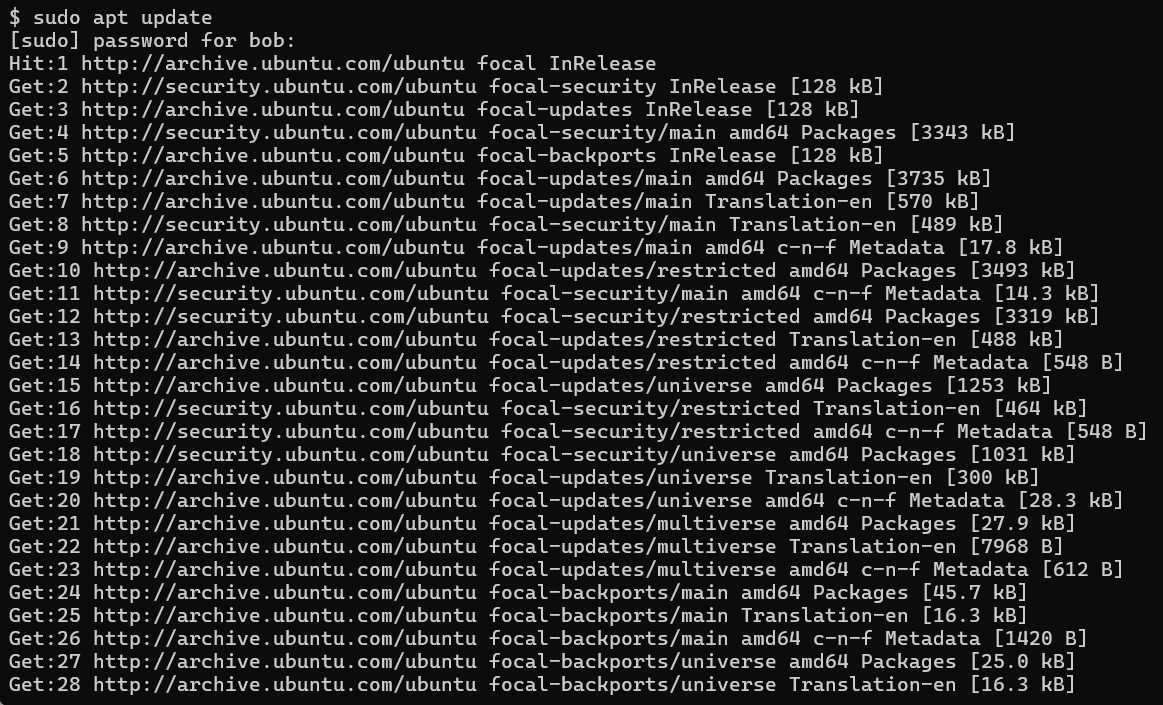


1. Add a user bob to the sudo group, allowing bob to execute commands as root.

2. Test by logging in as bob and running a command with sudo.

3. Optionally, restrict bob’s sudo access by editing the /etc/sudoers file using visudo (e.g., allow only apt-get commands).



****

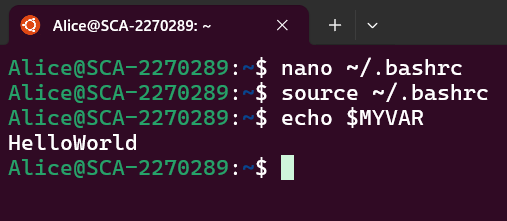
**Objective :** Learn how to set and customize user environment variables.

• **Expected Outcome**: You will practice setting environment variables that affect user sessions.

1. Modify the .bashrc file for a user (alice) to set a custom environment variable (e.g., MYVAR=HelloWorld).

2. Have the user log out and log back in, then check the environment variable using echo $MYVAR.

Solution :



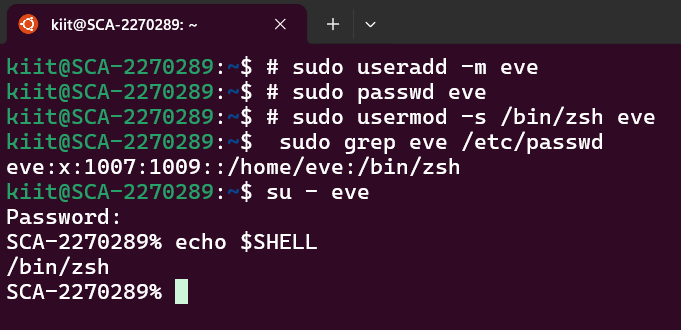
**Objective** : Learn how to configure a specific shell for users.

1. Create a user eve and set their default shell to /bin/zsh using usermod -s /bin/zsh.

2. Verify that eve’s default shell is set to Zsh by checking /etc/passwd.

3. Log in as eve and confirm the shell is now Zsh.

• Expected Outcome: You will learn how to change a user’s login shell and verify the change.



**Objective** : Write a script to automate the creation of users and groups.

• **Expected Outcome:** You will automate user and group management tasks, which is useful for system administration.

1. Write a Bash script that takes a username and a group as input.

2. Create the user, create the group if it does not exist, and add the user to the group.

3. Set a default password for the new user and notify the administrator by email.

Solution : The main code which I saved in create create\_user\_group.sh in kiit directory is :

#!/bin/bash

# Accept username and group as input

echo "Enter the username:"

read username

echo "Enter the group name:"

read groupname

# Check if the group exists, if not, create it

if ! getent group "$groupname" > /dev/null 2>&1; then

echo "Group '$groupname' does not exist. Creating group."

sudo groupadd "$groupname"

else

echo "Group '$groupname' already exists."

fi

# Check if the user exists

if id "$username" &>/dev/null; then

echo "User '$username' already exists."

else

# Create the user and add them to the group

sudo useradd -m -g "$groupname" "$username"

echo "User '$username' created and added to group '$groupname'."

fi

# Set a default password for the user

default\_password="default123"

echo "$username:$default\_password" | sudo chpasswd

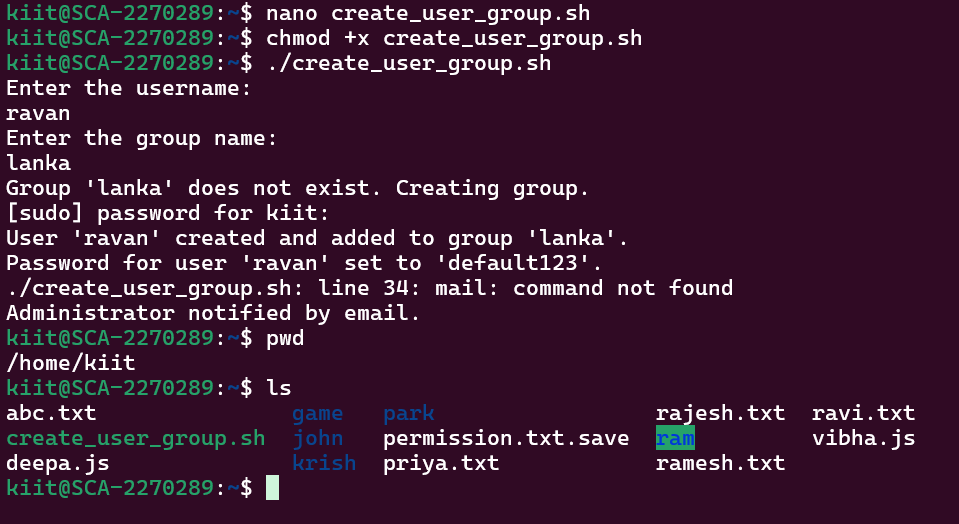
echo "Password for user '$username' set to '$default\_password'."

# Notify the administrator by email (assumes 'mail' is set up on the system)

admin\_email="admin@example.com"

echo "User '$username' has been created and added to group '$groupname' with a default password." | mail -s "New User Created: $username" "$admin\_email"

echo "Administrator notified by email."



**Objective :** Learn how to audit user account

1. Write a script to list all users who have not logged in for the past 90 days.

2. Optionally, send an email alert for these inactive accounts.

3. Disable inactive accounts by locking them (passwd -l).

• Expected Outcome: You will practice auditing user activity and manage inactive accounts.

**Solution :**

#!/bin/bash

# Set threshold days (90 days in this case)

threshold\_days=90

# Get the current date in Unix timestamp format

current\_date=$(date +%s)

# Specify the administrator email for sending alerts (change as needed)

admin\_email="admin@example.com"

# Temporary file to store inactive users

inactive\_users\_file="/tmp/inactive\_users.txt"

# Clear the temporary file before starting

> "$inactive\_users\_file"

# Header for the output file

echo "Inactive Users Report - Not Logged In for the Last 90 Days" > "$inactive\_users\_file"

echo "----------------------------------------------------------" >> "$inactive\_users\_file"

echo -e "Username\t\tLast Login\n" >> "$inactive\_users\_file"

# Use lastlog command with sudo to get the last login of all users and check for inactivity

sudo lastlog | awk -v threshold="$threshold\_days" -v cutoff="$current\_date" '

BEGIN {

# Calculate the cutoff timestamp for 90 days ago

cutoff = cutoff - (threshold \* 86400)

}

{

# Skip the first line (headers)

if (NR > 1) {

# Extract the username and last login date

username = $1

last\_login = $4 " " $5 " " $6 " " $7

if (last\_login != "Never") {

# Convert the last login date to Unix timestamp

login\_timestamp = mktime(last\_login " 00 00 00")

if (login\_timestamp < cutoff) {

# If the last login is older than the cutoff, add to the file

printf "%-15s\t%s\n", username, last\_login >> "'$inactive\_users\_file'"

}

}

}

}

'

# Check if the file contains any inactive users

if [ -s "$inactive\_users\_file" ]; then

# Output the inactive users to the console

echo "The following users have not logged in for the past 90 days:"

cat "$inactive\_users\_file"

# Send an email alert to the administrator using sudo

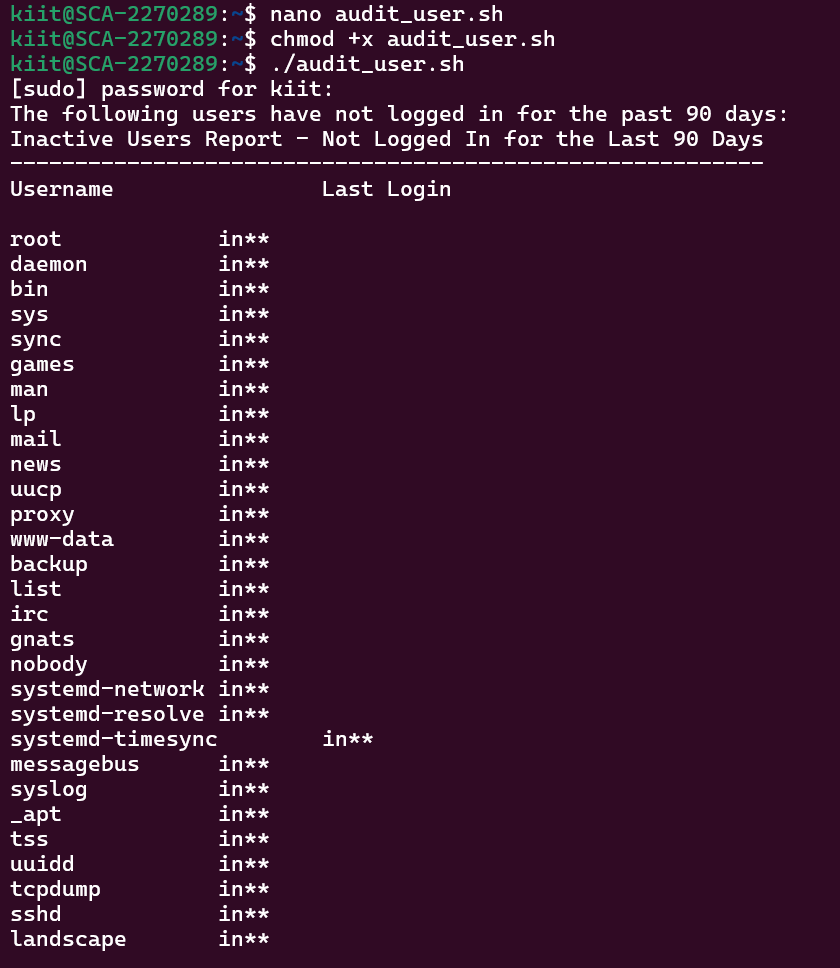
echo -e "The following users have not logged in for the past 90 days:\n\n$(cat "$inactive\_users\_file")" | sudo mail -s "Inactive Users Alert" "$admin\_email"

echo "Email alert sent to $admin\_email"

else

echo "No inactive users found."

fi





Objective : Learn to manage file permissions and ownership for users.

• Expected Outcome: You will gain experience in changing file ownership and setting permissions.

1. Create a file /home/alice/important\_file.txt.

2. Change the ownership of the file to the user alice using chown.

3. Set the file permissions so that only alice has read and write access, while others have no access.

4. Verify the permissions using ls -l.

Solution:

