localUserGroupMgr

```
#!/bin/bash

# Author : Abhijeet K

# Defining Variables

mapfile -t users < <(awk 'NR>1 && $1 != "" {print $1}' /opt/UserPassword)
mapfile -t existing < <(awk -F: '$1 != "" {print $1}' /etc/passwd)

mapfile -t group_file < <(awk 'NR>1 && $1 != "" {print $1}' /opt/groups.txt)
mapfile -t group_name < <(awk '{print $1}' /etc/group)

# Function to add User

useradd_code() {
      useradd --badname -m "$1"

      if [[ $? -eq 0 ]];then
         echo "Users Created...!"
      else
         echo "error"
      fi
}

#Checking Users are exist or not

check_user() {

   for agent in "${users[@]}"
   do
      found=false
```

```
        for agent_exist in "${existing[@]}"
        do
            if [[ "$agent" == "$agent_exist" ]];then
                echo "User $agent is already present into system"
                found=true
                break
            fi
        done

        if ! $found;then
            useradd_code "$agent"
        fi
    done

}

# Check group exist or not

check_group() {

    for grp in "${group_file[@]}"
    do
        found=false
        for grp_name in "${group_name[@]}"
        do
            if [[ "$grp" == "$grp_name" ]];then
                echo "Group $grp is already present into system"
                found=true
                break
            fi
        done
        if ! $found;then
            groupadd_code "$grp"
        fi
    done
}

groupadd_code() {
```

```
    if [[ -z "$1" ]];then
        echo "Group is empty"
        return
    fi

    if groupadd "$1" 2>/tmp/groupadd_error;then
        echo "group created...!"
    else
        :
    fi
}

change_password () {

    mapfile -t userpass < <(awk 'NR>1 && $1 != "" {print $1}' /opt/UserPassw
ord)
    mapfile -t pass < <(awk 'NR>1 && $1 != "" {print $2}' /opt/UserPassword)

    for (( i=1; i<${#userpass[@]}; i++ ))
    do
        username="${userpass[i]}"
        password="${pass[i]}"

        if id "$username" &> /dev/null; then
              echo "$username:$password" | chpasswd
              chage -d 0 "$username"
           else
             echo "User not found : $username"
           fi
    done
}

change_group () {

    mapfile -t usergroups < <(awk 'NR>1 && $1 != "" {print $2}' /opt/UserGro
ups)
    mapfile -t users < <(awk 'NR>1 && $1 != "" {print $1}' /opt/UserGroups)
```

```
    for (( i=1; i<${#usergroups[@]}; i++ ))
    do
        username="${users[i]}"
        groups="${usergroups[i]}"

        if id "$username" &> /dev/null; then
            usermod -aG "$username" "$groups"
          else
            echo "User Not found : $username"
          fi
    done

}

remove_users_from_group () {

    mapfile -t users < <(awk 'NR>1 && $1 != "" {print $1}' /opt/RemoveUsers
FromGroup)
    mapfile -t groups < <(awk 'NR>1 && $2 != "" {print $2}' /opt/RemoveUse
rsFromGroup)

    echo "$users"
    echo "$groups"
    for (( i=0; i<${#groups[@]}; i++ ))
    do
        username="${users[i]}"
        groupname="${groups[i]}"

        if id "$username" &> /dev/null;then
            gpasswd -d "$username" "$groupname"
        fi
    done
}

delete_users () {

    mapfile -t users < <(awk 'NR>1 && $1 != "" {print $1}' /opt/DeleteUsers)
```

```
    for (( i=0; i<${#users[@]}; i++ ))
    do
        username="${users[i]}"
        if id "$username" &> /dev/null;then
            userdel -r "$username"
        fi
    done
}

get_users_uuid () {

    mapfile -t users < <(awk 'NR>1 && $1 != "" {print $1}' /opt/Usersforuuid)

    for (( i=0; i<${#users[@]}; i++ ))
    do
        useruuid="${users[i]}"

        uuidofuser=$(id -u $useruuid)
        echo -e "$useruuid\t$uuidofuser" >> /opt/uuidofusers
    done
}

echo ""
echo "List of work :⟹>"
echo ""
echo "1. Add users"
echo "2. Add Groups"
echo "3. Change Password for User"
echo "4. Change group for User"
echo "5. Remove user from Group"
echo "6. Delete users"
echo "7. Get Users and its UID"
read -p "Enter your choice : " option

case $option in
    1)
        check_user
```

```
    ;;
    2)
        check_group
    ;;
    3)
        change_password
    ;;
    4)
        change_group
    ;;
    5)
        remove_users_from_group
    ;;
    6)
        delete_users
    ;;
    7)
        get_users_uuid
    ;;
    *)
        echo "you have entered Invalid option"
    ;;
esac
```