System Administrator Helper Tool

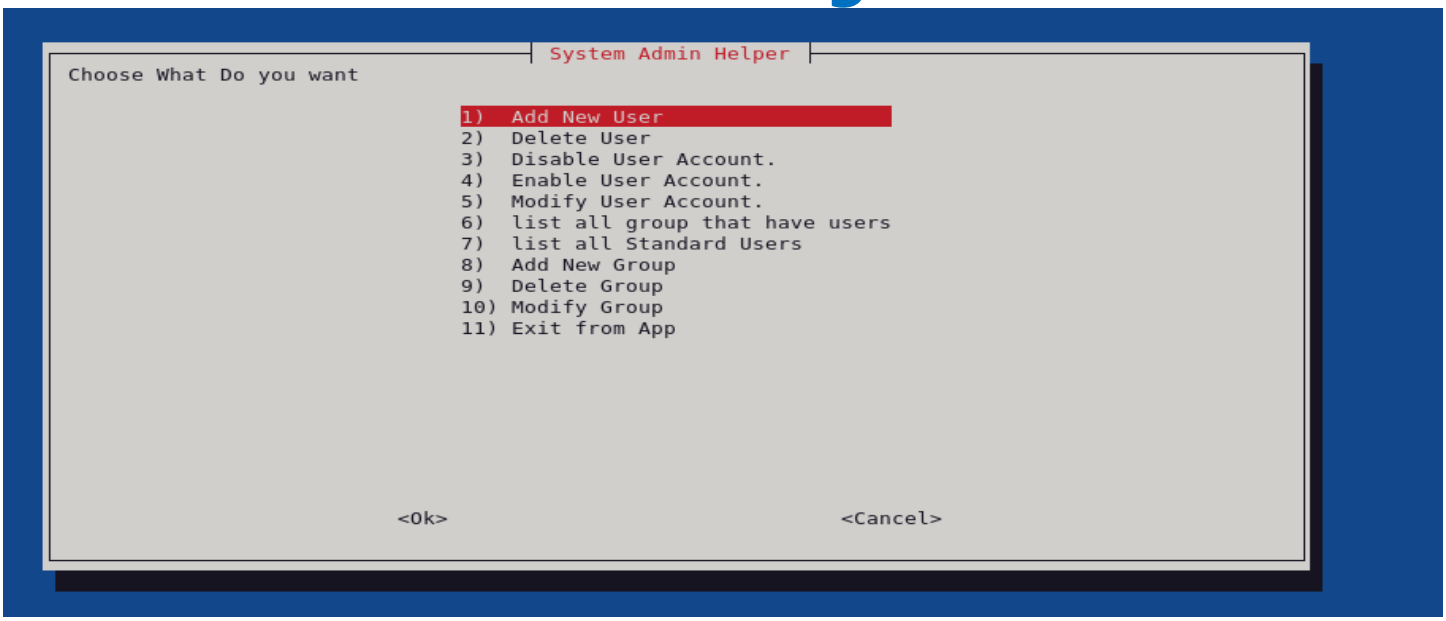
ITI Bash Script by Instructor
Eng : romany Nageh

Bash Project Create by
Eng : Soliman Ali Soliman

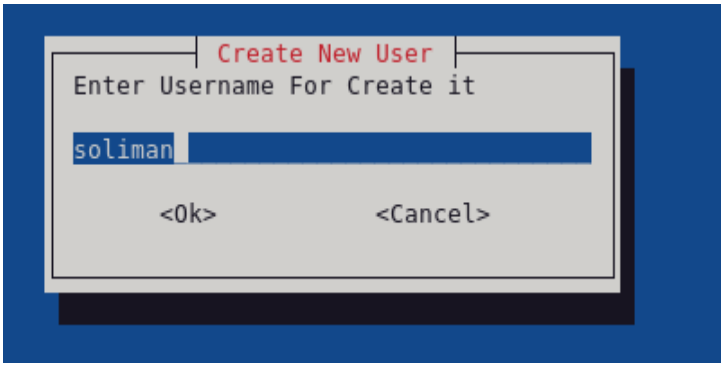
Project make some feature in menu using whiptail Features by using whiptail in bash script make menu and the change affect in real system .

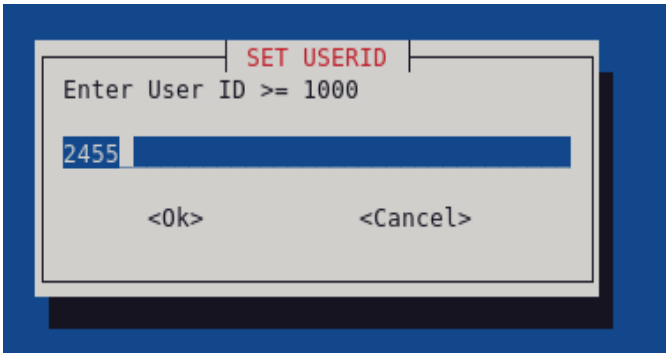
Contents List

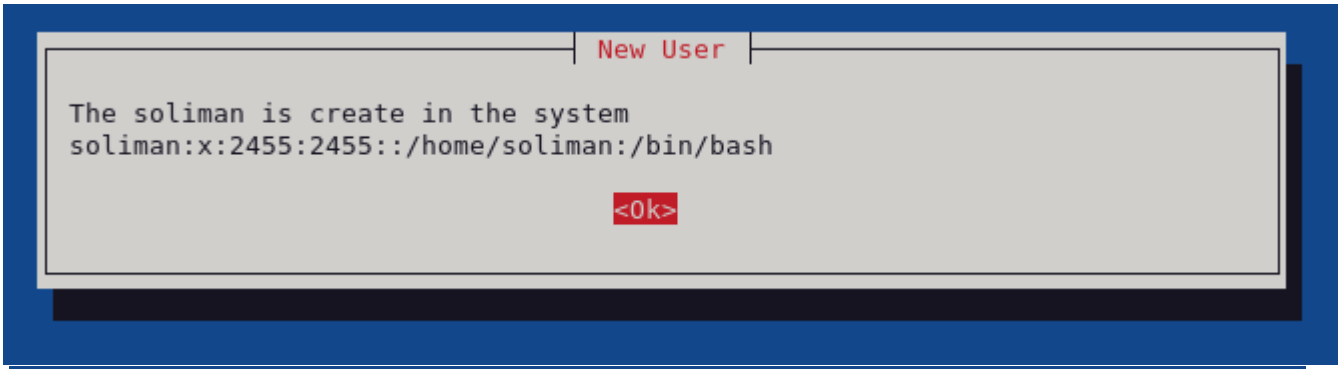
Menu Page

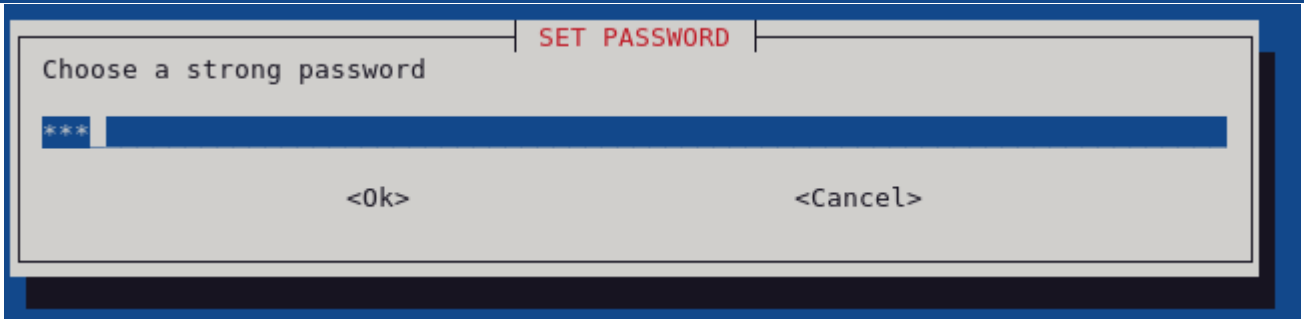


1. Add New User .





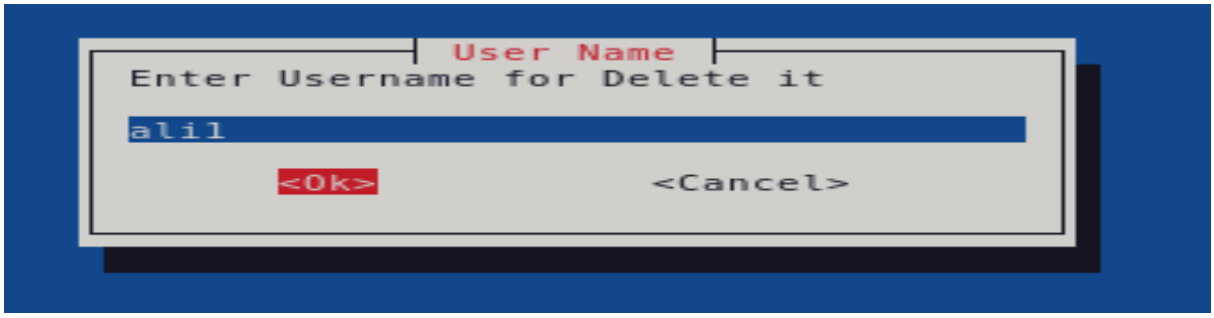


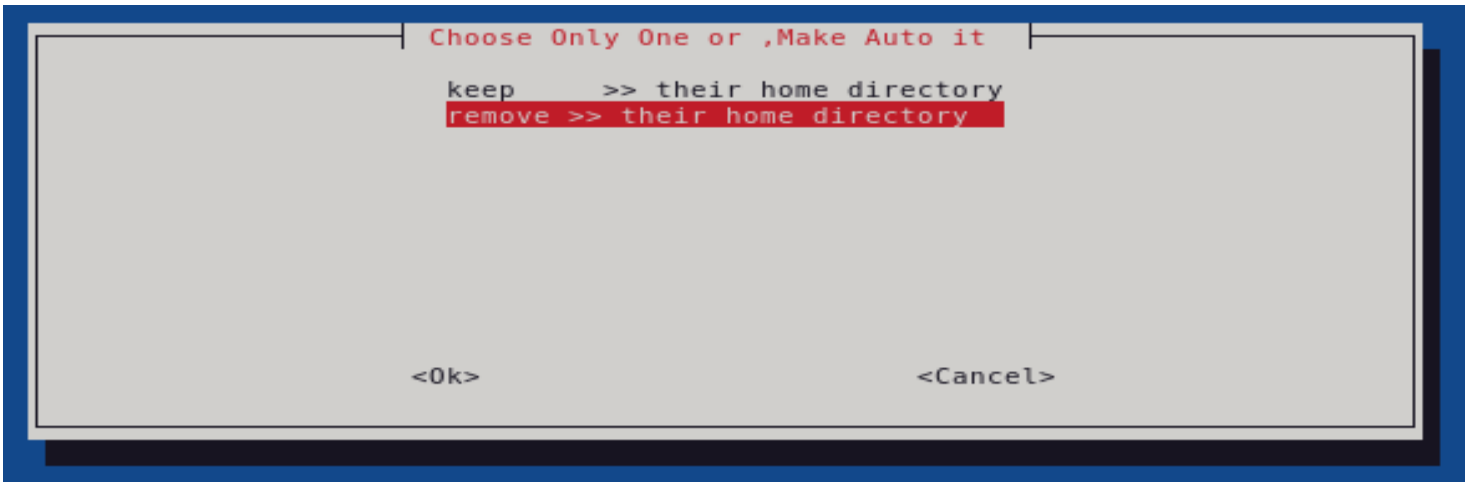


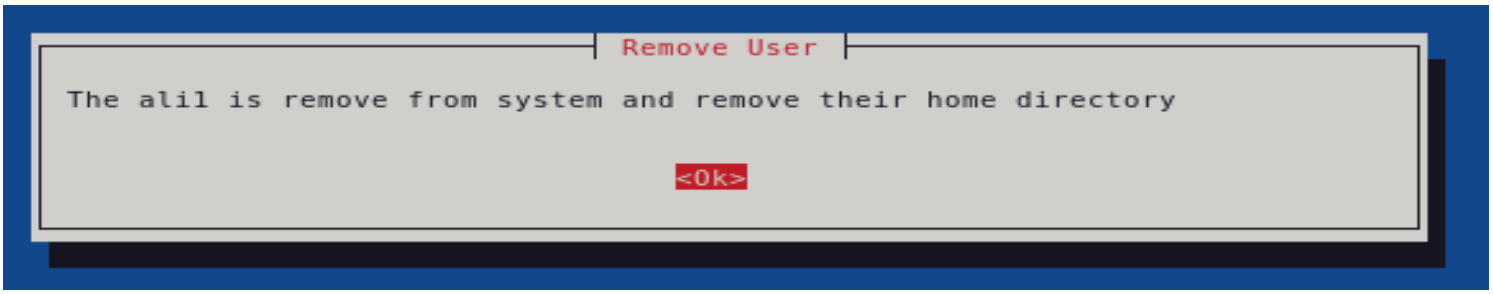
Source code : Create User

```
##************************** for create new user ********************************
function adduser(){
#check user exists or not
    getent passwd $1  > /dev/null 2&>1
    if [ $? -eq 0 ]; then
       if (whiptail --title "Try Again" --yesno "The $1 is exists in the system , please try again...." 8 78 ); then
           adduser
       fi
    else
       USERID=$(whiptail --title "SET USERID" --inputbox "Enter User ID >= 1000" 8 40 3>&1 1>&2 2>&3)
       if [ $? -eq 0 ]; then
#for check that userid  is empty
       if [ -z $USERID ];then
          useradd   $1
          whiptail --title "New User" --msgbox "The $1 is create in the system  \n$(tail -1 /etc/passwd | grep $1 ) " 8 78
       else
          useradd  --uid $(echo $USERID)  $1
          whiptail --title "New User" --msgbox "The $1 is create in the system  \n$(tail -1 /etc/passwd | grep $1 ) " 8 78
       fi
      fi
    fi
}
##############################  function set passwd ***************************************************************
function set_passwd(){
     PASSWORD=$(whiptail --title "SET PASSWORD" --passwordbox "Choose a strong password" 8 78 3>&1 1>&2 2>&3)
     if [ $? -eq 0 ]; then
#  for check that password  is empty
     if [ -z $PASSWORD ];then
         whiptail --title "No data entered" --msgbox "Please, Try again Enter PassWord !!........" 8 78
         set_passwd
     fi
     echo $PASSWORD | passwd --stdin $1  >/dev/null2&>1
    # echo $PASSWORD  "this is pass for " $1
    fi
}

#********************************** end fun  set passwd ***************************************************************

#************************************ create user ********************************************************************
function create_user(){
    NEW_USER=$(whiptail --title "Create New User" --inputbox "Enter Username For Create it" 8 40 3>&1 1>&2 2>&3)
    if [ $? -eq 0 ]; then
## for check that input not empty
    if [ -z $NEW_USER ];then
       whiptail --title "No data entered" --msgbox "please try again enter username .... " 8 78
       create_user
    else
       adduser $NEW_USER
       set_passwd $NEW_USER
    fi
    fi
}

#*********************************end create user ************************************************
```

2.Delete User .



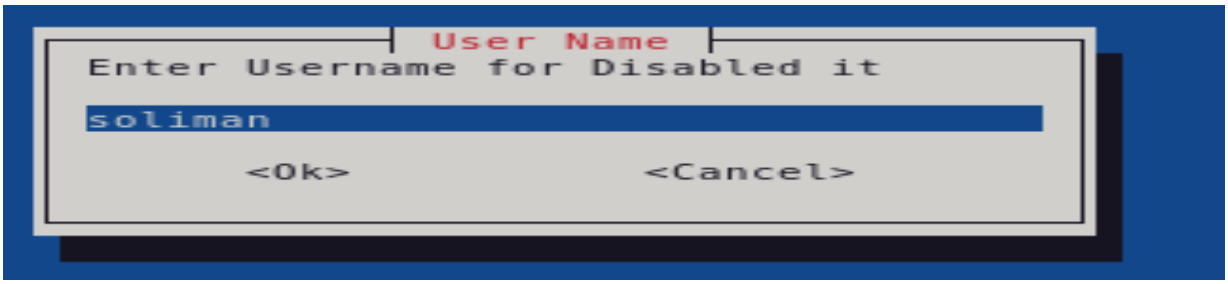


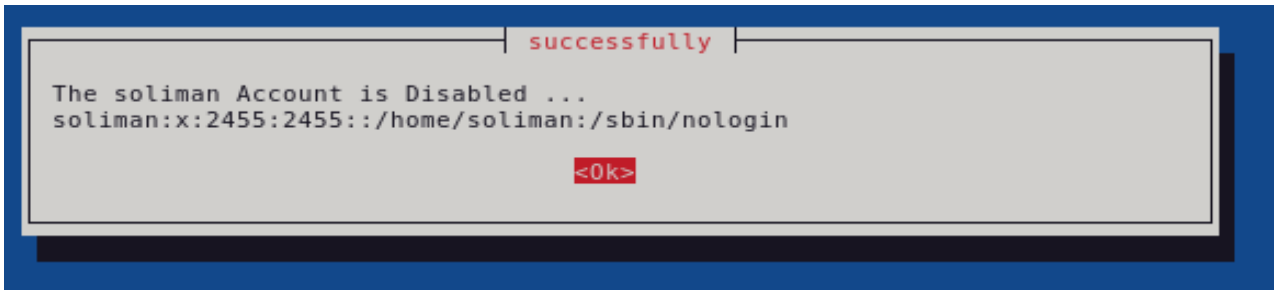


Source code : Delete User

```
#****************************** for delete user exist ******************************************
function delete_u(){
#check user exists or not
     getent passwd $1  > /dev/null 2&>1
     if [ $? -eq 0 ]; then
# keep or remove their home directory.
        SELECTED=$(whiptail --title "Choose Only One or ,Make Auto it " --menu  "List of choose ..... " 16 80 9 \
                 "keep" "  >> their home directory" \
                 "remove" ">> their home directory" 3>&1 1>&2 2>&3)
        case $SELECTED in
              "keep")  userdel  $1
                       whiptail --title "Remove User" --msgbox "The $1 is remove from system and keep their home directory" 8 78
                      ;;
            "remove")  userdel -r  $1
                       whiptail --title "Remove User" --msgbox "The $1 is remove from system and remove their home directory " 8 78
                      ;;
                  *)   userdel -r  $1
                       whiptail --title "Remove User" --msgbox "This is Defualt ....\nthe $1 is removed with their home directory .... " 8 78
                      ;;
        esac
   else
        whiptail --title "unavailable" --msgbox "the $1 does not exist in the system" 8 78
   fi}
#****************************** delete user ************************************************************
function delete_user(){
         remove_user=$(whiptail --title "User Name" --inputbox "Enter Username for Delete it" 8 40 3>&1 1>&2 2>&3)
         if [ $? -eq 0 ]; then
## for check that input not empty
         if [ -z $remove_user ];then
             whiptail --title "No data entered" --msgbox "please try again enter username .... " 8 78
             delete_user
         else
             delete_u $remove_user
     fi  fi
}
#**********************************  end delete user ***************************************************
```

3.Disable User Account

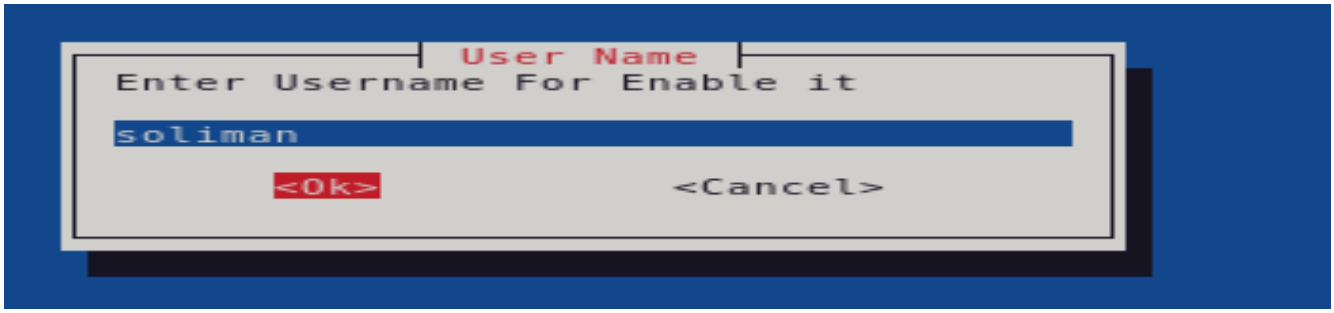


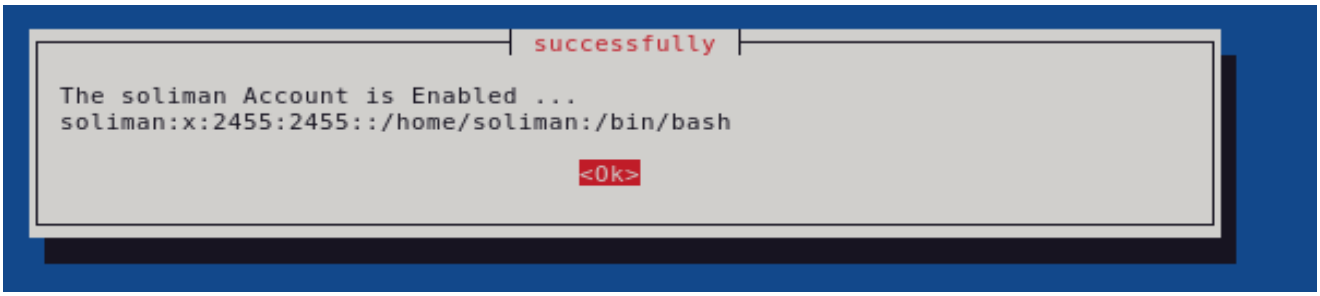


Source code : Disable account

```
#******************************** Disable  User Account *****************************************

function disable_user(){
     DIS_USER=$(whiptail --title "User Name" --inputbox "Enter Username for Disabled it" 8 40 3>&1 1>&2 2>&3)
    if  [ $? -eq 0 ]; then
## for check that input not empty
    if [ -z $DIS_USER ];then
        whiptail --title "No data entered" --msgbox "please try again enter username .... " 8 78
        disable_user
    else
       usermod -L -s /sbin/nologin -e 1 $DIS_USER
       whiptail --title "successfully" --msgbox "The $DIS_USER Account is Disabled ... \n$(tail -1 /etc/passwd | grep $DIS_USER ) " 8 78
    fi
   fi
}
#********************************end disable user ***********************************************************
```

4.Enable User Account

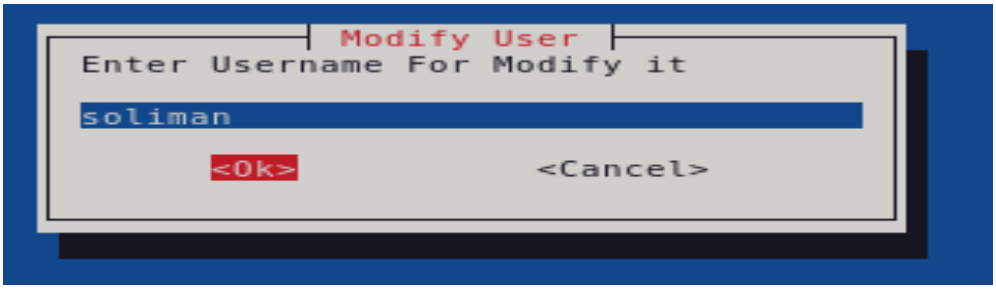


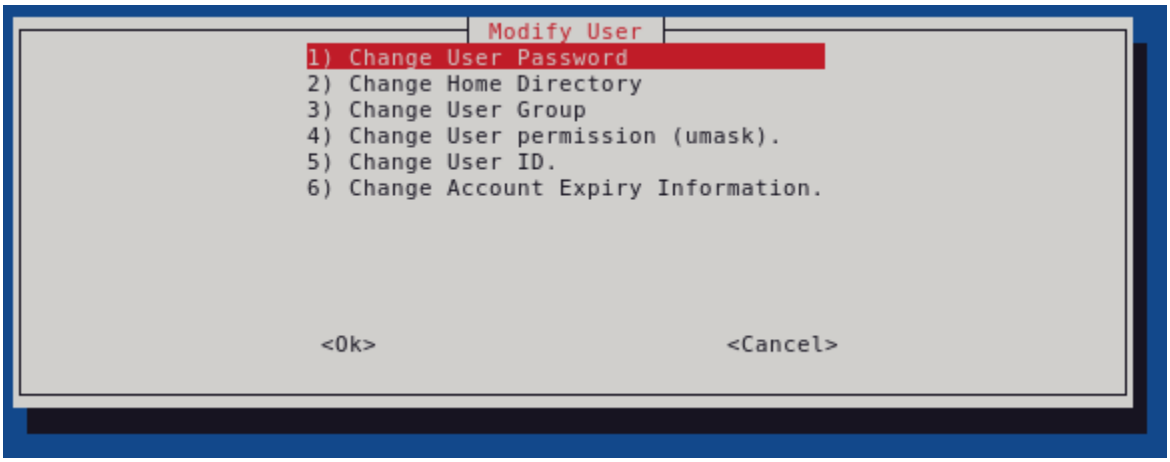


Source code : Enable account

```
#********************************* enable  User Account *********************************************
function enable_user(){
      EN_USER=$(whiptail --title "User Name" --inputbox "Enter Username For Enable it" 8 40 3>&1 1>&2 2>&3)
      if [ $? -eq 0 ]; then
## for check that input not empty
      if [ -z $EN_USER ];then
          whiptail --title "No data entered" --msgbox "please try again enter username .... " 8 78
          enable_user
      else
          usermod -U -s /bin/bash -e -1 $EN_USER
          whiptail --title "successfully" --msgbox "The $EN_USER Account is Enabled ... \n$(tail -1 /etc/passwd | grep $EN_USER ) " 8 78
      fi
      fi
}
#********************************end enable user ***********************************************************
```

5.Modify User Account

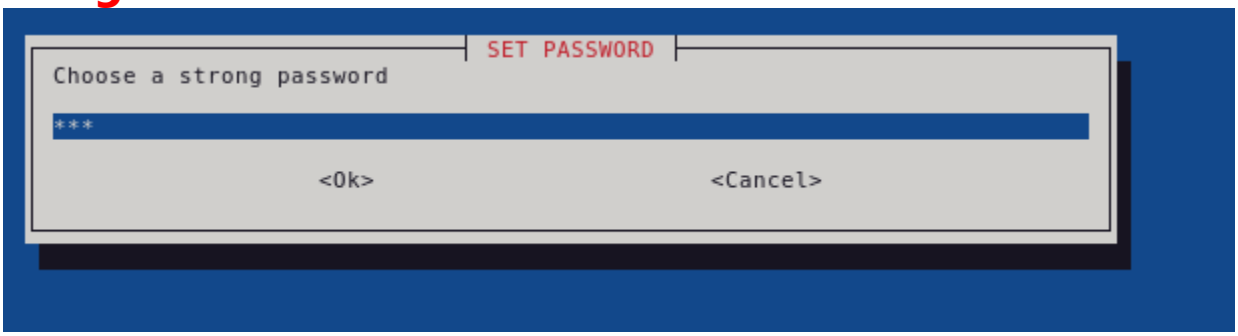




Source code : Modify User Account

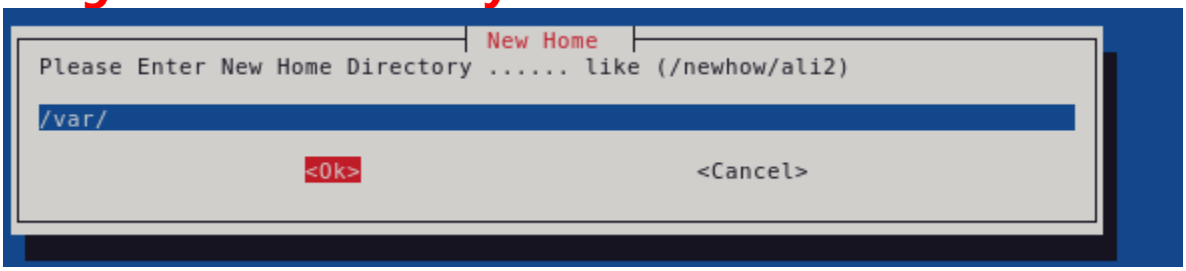
```
#
#********************************* function for edit user ***************************************************************************************

function edit_user(){
       EN_USER=$(whiptail --title "Modify User" --inputbox "Enter Username For Modify it " 8 40 3>&1 1>&2 2>&3)
       if [ $? -eq 0 ]; then
## for check that input not empty
       if [ -z $EN_USER ];then
           whiptail --title "No data entered" --msgbox "please try again enter username .... " 8 78
           edit_user
       else
          getent passwd $EN_USER  > /dev/null 2&>1
       if [ $? -eq 0 ]; then
             CHOICE=$(whiptail --title "Modify User" --menu "Choose from List...." 15 80 9 \
                   "1)" "Change User Password"    \
                   "2)" "Change Home Directory"  \
                   "3)" "Change User Group " \
                   "4)" "Change User permission (umask)." \
                   "5)" "Change User ID." \
                   "6)" "Change Account Expiry Information."  3>&2 2>&1 1>&3  )

    case $CHOICE in
         "1)") set_passwd $EN_USER  ;;
         "2)")  ch_home  $EN_USER   ;;
         "3)")  ch_group $EN_USER    ;;
         "4)") ch_umask ch_umask     ;;
         "5)") userid $EN_USER       ;;
         "6)") u_pass_expiry $EN_USER ;;
           *) edit_user ;;
  esac
   else
      whiptail --title "unavailable" --msgbox "the $1 does not exist in the system" 8 78
   fi
fi
fi
}
```

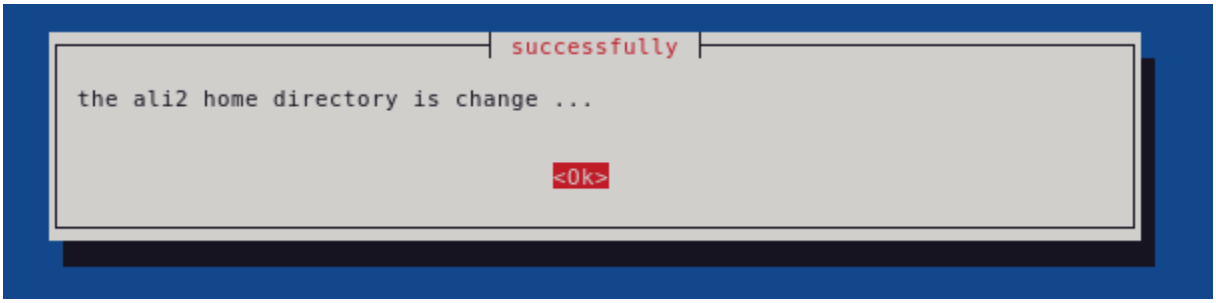
5.1.Change User Password




```
############################## function set passwd ***************************************************************
function set_passwd(){
     PASSWORD=$(whiptail --title "SET PASSWORD" --passwordbox "Choose a strong password" 8 78 3>&1 1>&2 2>&3)
     if [ $? -eq 0 ]; then
#  for check that password  is empty
     if [ -z $PASSWORD ];then
         whiptail --title "No data entered" --msgbox "Please, Try again Enter PassWord !!........" 8 78
         set_passwd
     fi
     echo $PASSWORD | passwd --stdin $1  >/dev/null2&>1
    # echo $PASSWORD  "this is pass for " $1
    fi
}

#********************************** end fun  set passwd ***************************************************************
```

5.2. Change Home Directory

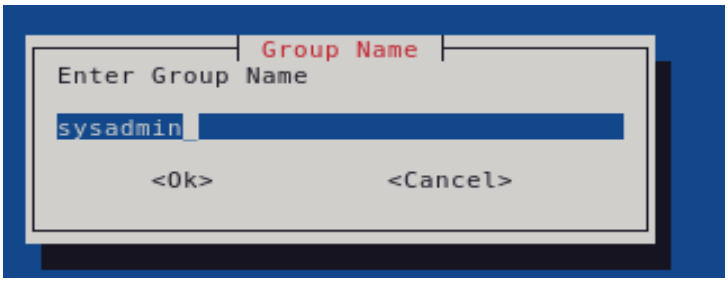


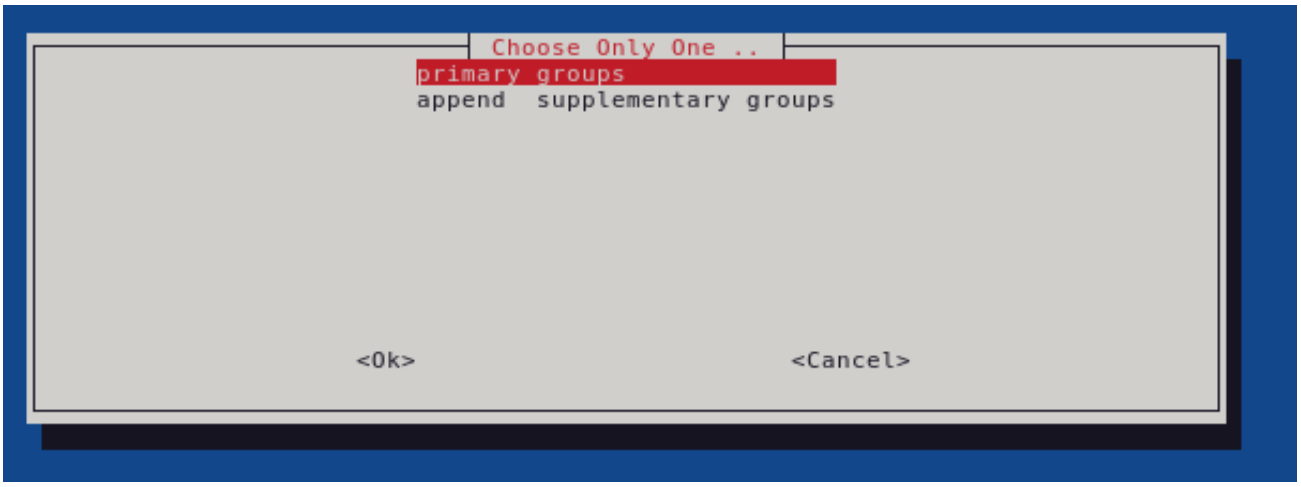


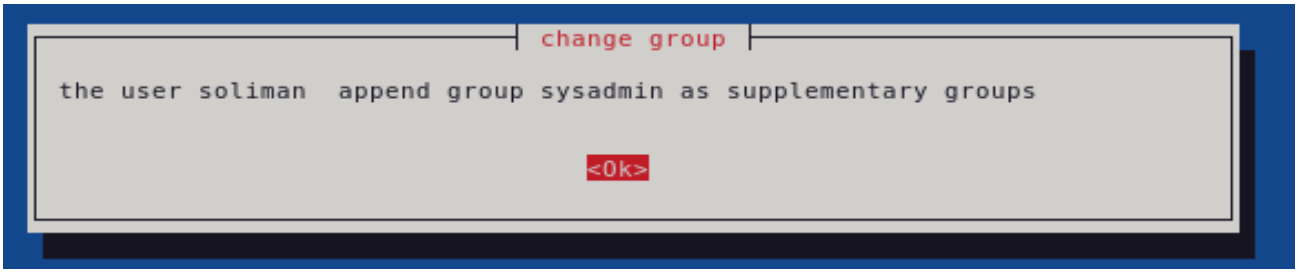
Source code

```
#************************************** For change home dir ***************************************************************
function  ch_home(){
          NEW_HOME=$(whiptail --title "New Home " --inputbox "Please Enter New Home Directory ..... like (/newhow/$EN_USER)" 8 78 3>&1 1>&2 2>&3)
          if [ $? -eq 0 ];then
               if [ -z $NEW_HOME ];then
                 whiptail --title "No data entered" --msgbox "Please try again enter Home Directory .... " 8 78
                 ch_home
              fi
              usermod  -d $NEW_HOME  $1
              whiptail --title "successfully" --msgbox "the $1 home directory is change ... \n$(tail -1 /etc/passwd | grep $EN_USER ) " 8 78
           fi
}
```

5.3.Change User Group .



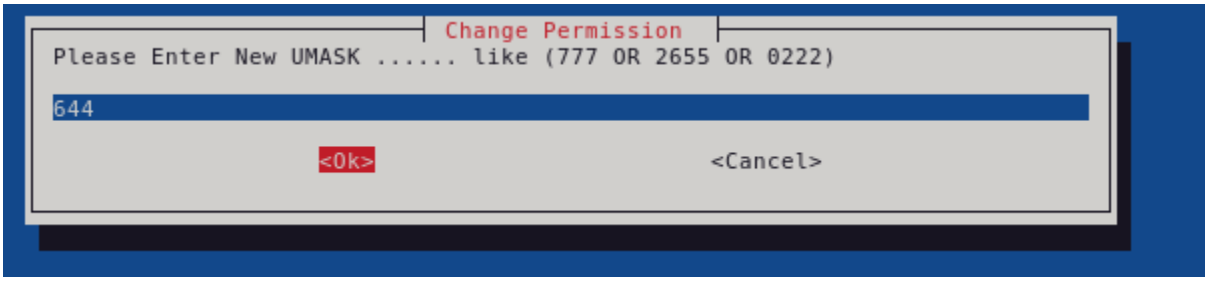


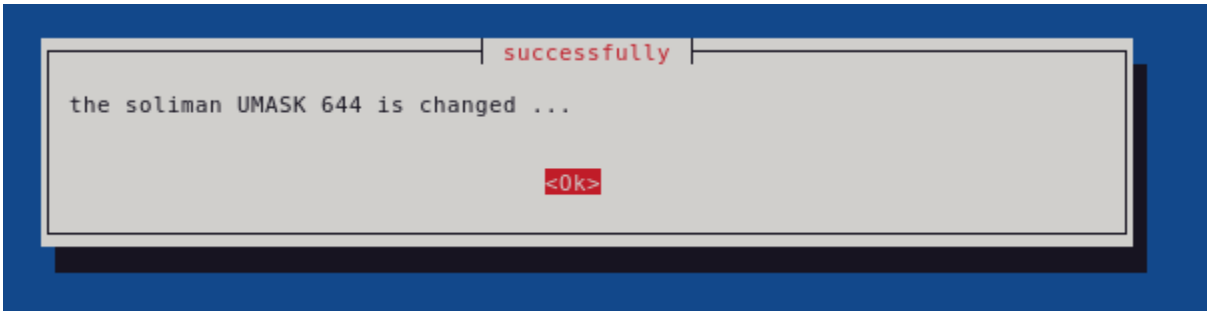


Source code

```
#************************************** Function  Change Group ************************************
function ch_group(){
   GROUP=$(whiptail --title "Group Name" --inputbox "Enter Group Name" 8 40 3>&1 1>&2 2>&3)
  if [ $? -eq 0 ]; then
## for check that input not empty
   if [ -z $GROUP ];then
     whiptail --title "No data entered" --msgbox "please try again enter Group Name .... " 8 78
     ch_group
   else

      SELECTED=$(whiptail --title "Choose Only One .." --menu  "List of choose ..... " 15 80 9  \
              "primary" "groups" \
              "append" "supplementary groups" 3>&1 1>&2 2>&3)

      case $SELECTED in
           "primary")  usermod -g $GROUP  $1
                       whiptail --title "change group" --msgbox "the user $1  change group $GROUP as primary group " 8 78
                    ;;
            "append")  usermod  -a -G  $GROUP  $1
                       whiptail --title "change group" --msgbox "the user $1  append group $GROUP as supplementary groups" 8 78
                    ;;
                 *)    usermod  -a -G  $GROUP  $1
                       whiptail --title "change group" --msgbox "the user $1 change group $GROUP as supplementary groups" 8 78
                    ;;
      esac
fi
fi
}
```

5.4. Change User permission (umask).

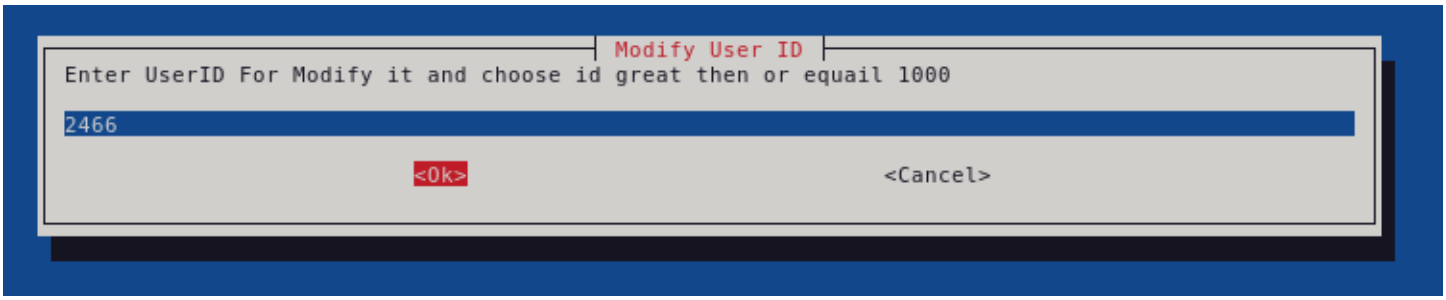


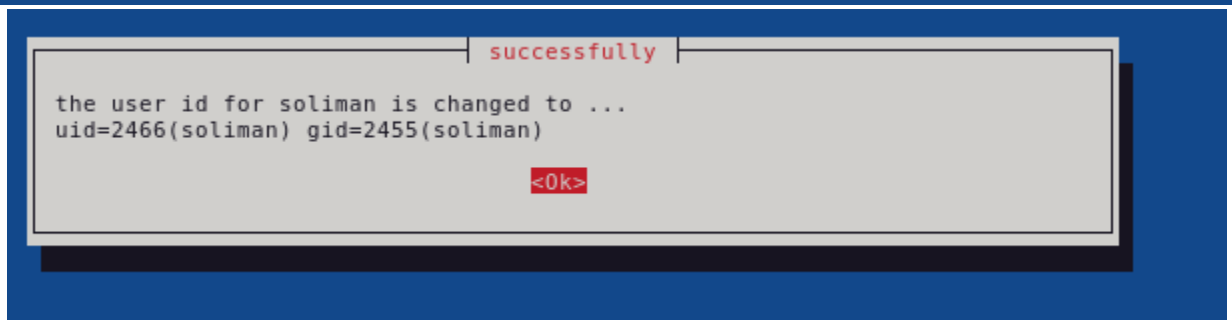


Source code

```
#************************************ for chane umask ***********************************************************************
function ch_umask(){
         UMASK=$(whiptail --title "Change Permission " --inputbox "Please Enter New UMASK ..... like (777 OR 2655 OR 0222)" 8 78 3>&1 1>&2 2>&3)
         if [ $? -eq 0 ];then
             if [ -z $UMASK ];then
                 whiptail --title "No data entered" --msgbox "Please try again enter umask .... " 8 78
                 ch_umask
              fi
              echo "umask" $UMASK >> /home/$1/.bashrc
              whiptail --title "successfully" --msgbox "the $1 UMASK $UMASK is changed ... " 8 78
         fi
}
```

5.5 .Change User ID

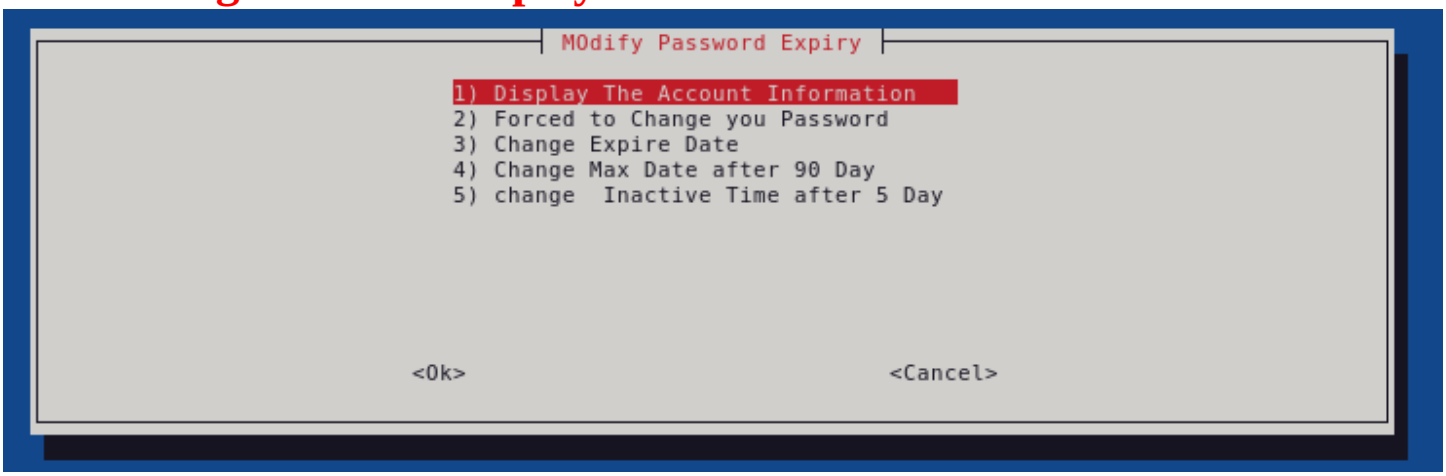




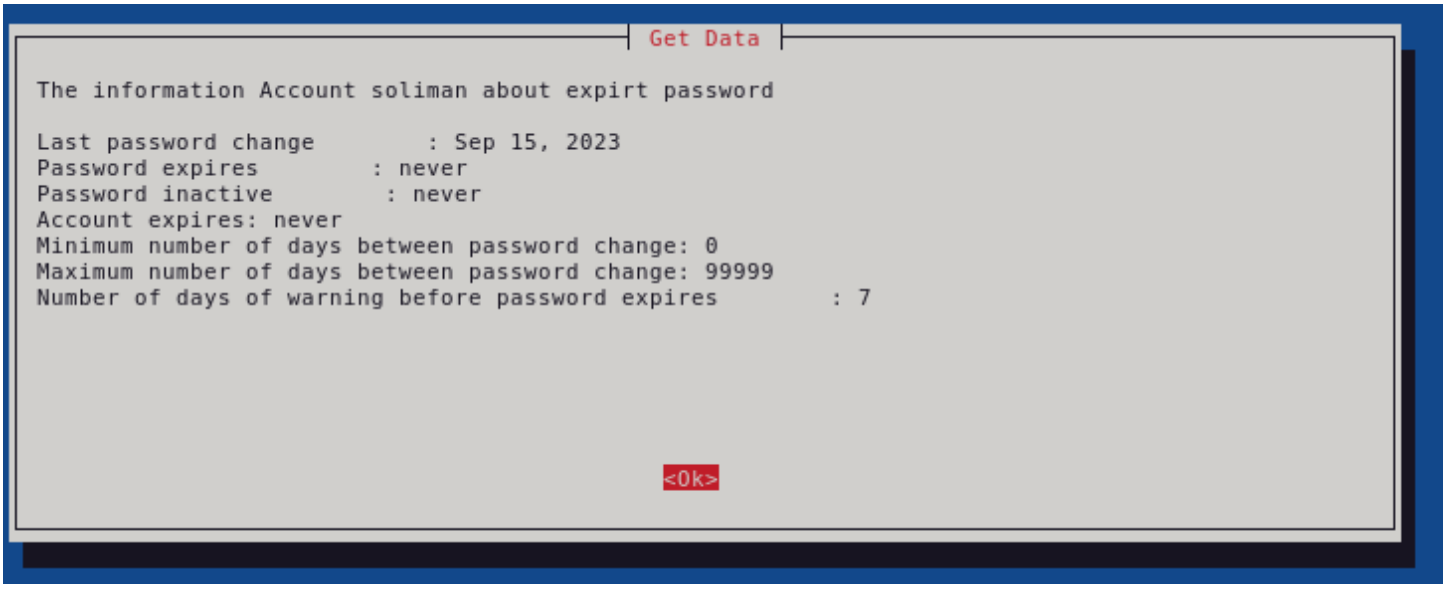
Source code

```
##******************************** FUNCTION CHANGE USERID
function userid(){
      USERID=$(whiptail --title "Modify User ID" --inputbox "Enter UserID For Modify it and choose id great then or equail 1000 " 8 100  3>&1 1>&2 2>&3)
      if [ $? -eq 0 ]; then
## for check that input not empty
      if [ $USERID >= 1000 ];then
         whiptail --title "No data entered" --msgbox "please try again enter userid.... " 8 78
         userid
      else
         usermod -u $USERID  $1
         whiptail --title "successfully" --msgbox "the user id for $1 is changed to ... \n$(id $1 |grep  uid ) " 8 78
      fi
      fi
}
#************************************************END FUNCTION ****************************************
```

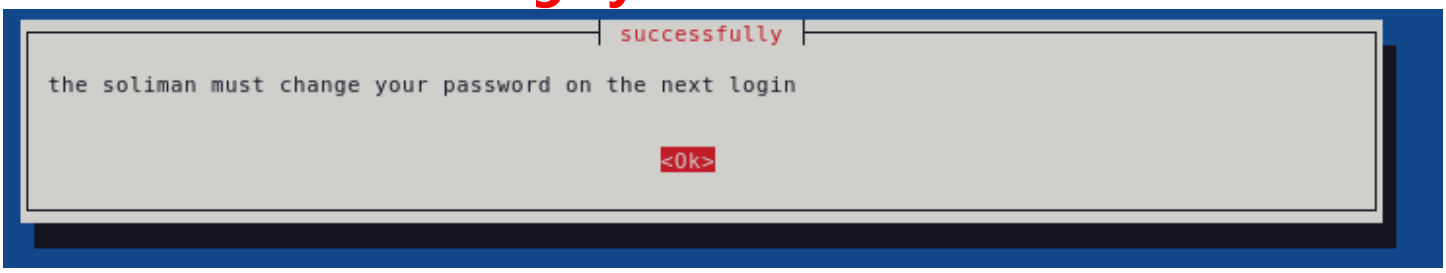
5.6. Change Account Expiry Information



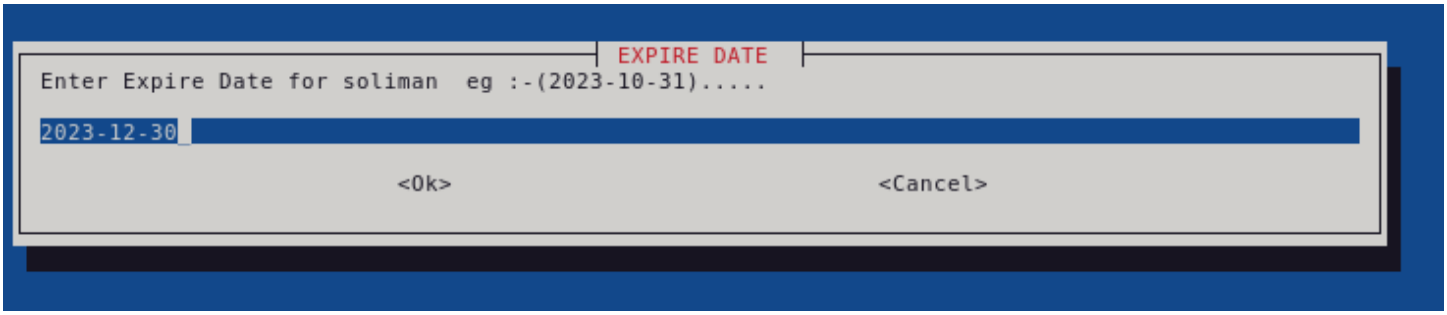
5.6.1. Display the Account Information

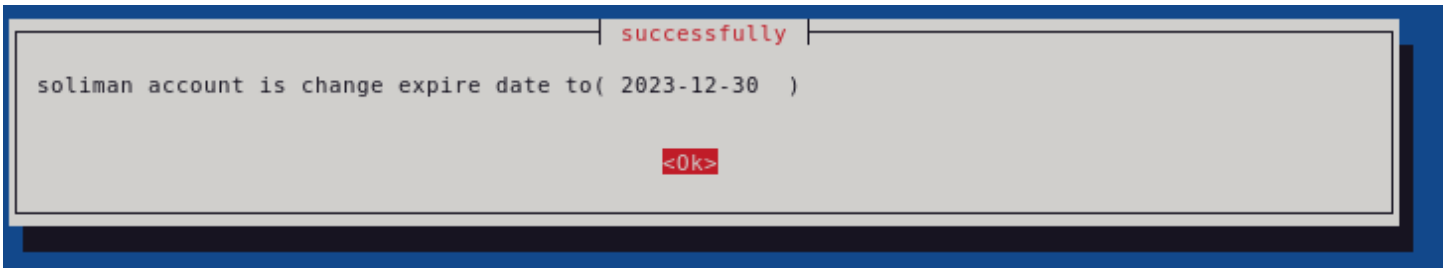


5.6.2. Forced to Change you Password

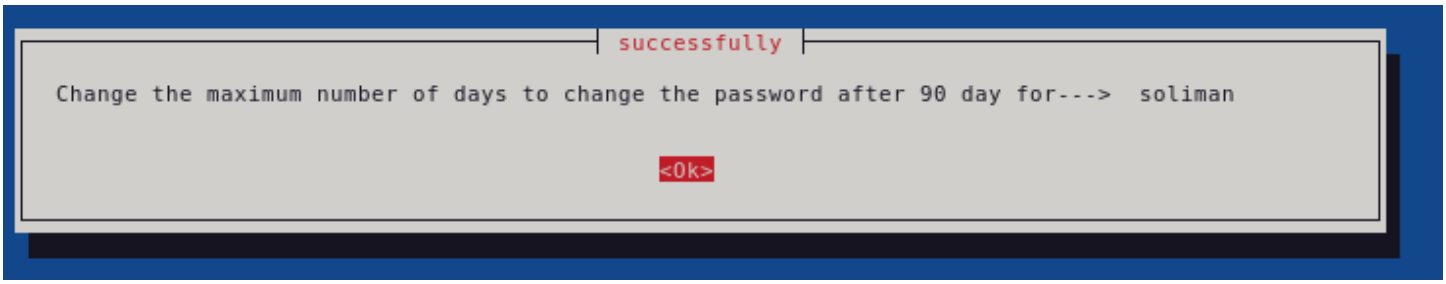


5.6.3. Change Expire date

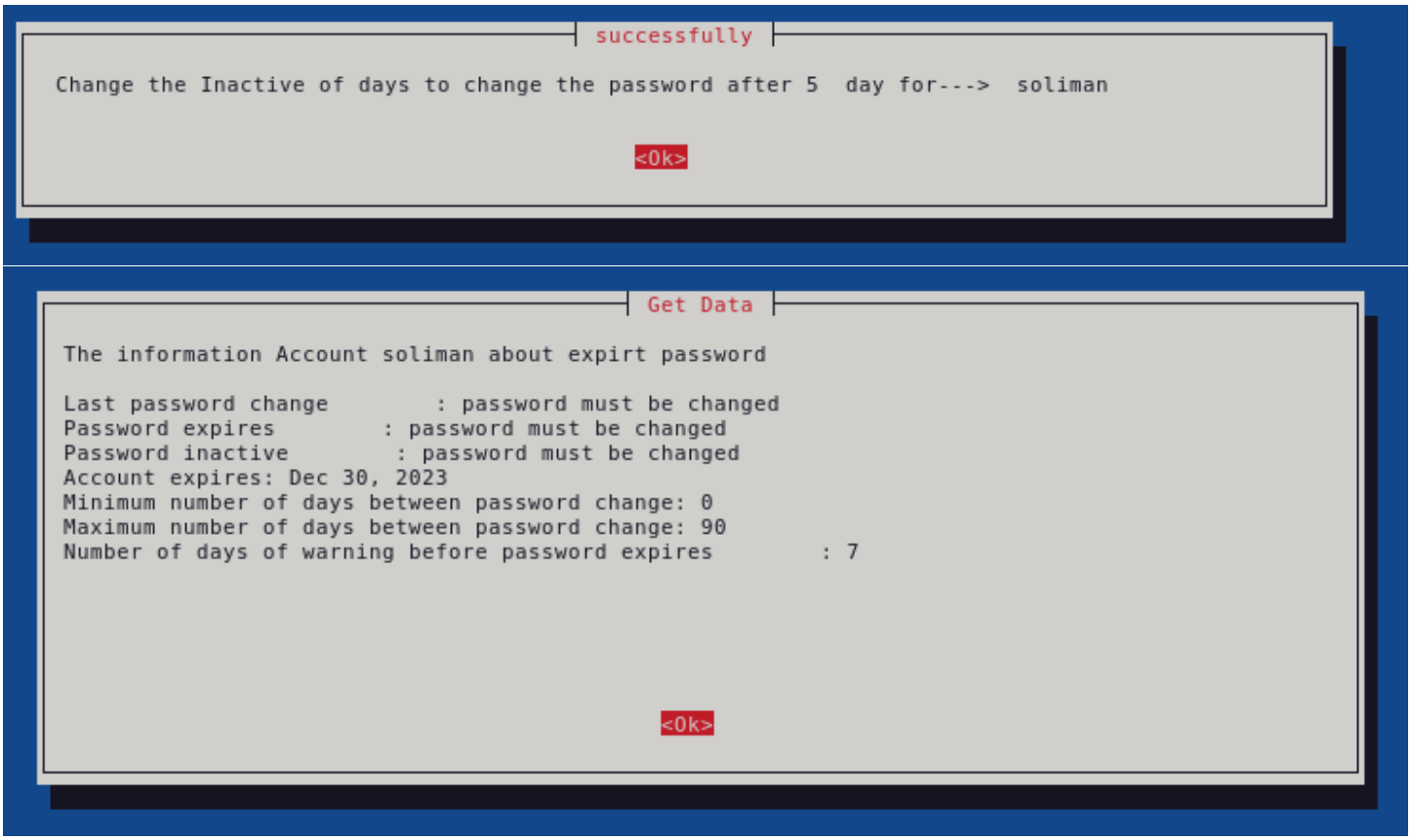




5.6.4. Change Max Date after 90 Day



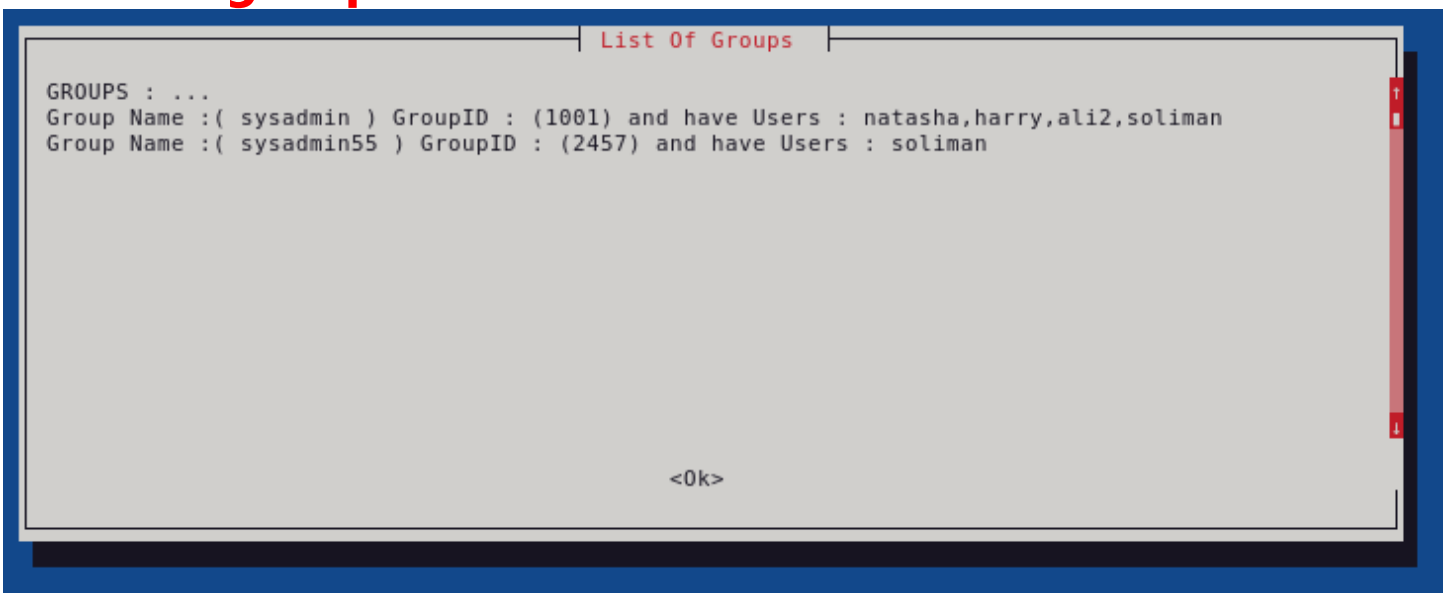
5.6.5. Change Inactive Time after 5 Day



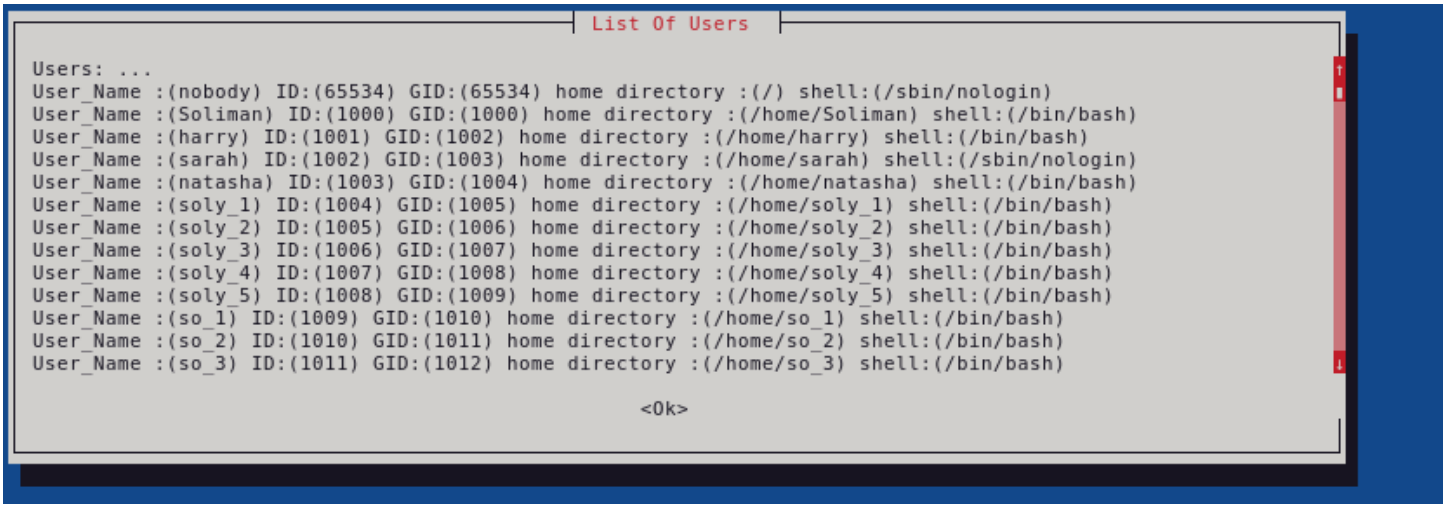
source code

```
#*******************************************change user password expiry information *****************************************************
function  u_pass_expiry(){
       CHOICE=$(whiptail --title "MOdify Password Expiry" --menu "Modify User Password Expiry Information" 16 100 9 \
         "1)" "Display The Account Information"   \
         "2)" "Forced to Change you Password"  \
         "3)" "Change Expire Date  " \
         "4)" "Change Max Date after 90 Day " \
         "5)" "change  Inactive Time after 5 Day " 3>&2 2>&1 1>&3  )

    case $CHOICE in
       "1)")whiptail --title "Get Data" --msgbox "The information Account $1 about expirt password \n\n$(chage -l  $1)" 20  100 9
            ;;
       "2)") chage -d 0 $1
             whiptail --title "successfully" --msgbox "the $1 must change your password on the next login " 8 100
            ;;
       "3)") EXPIREDATE=$(whiptail --title "EXPIRE DATE " --inputbox "Enter Expire Date for $1  eg :-(2023-10-31)..... " 8 40 3>&1 1>&2 2>&3)
             if [ $? -eq 0 ]; then
             ## for check that input not empty
              if [ -z $EXPIREDATE  ];then
                 whiptail --title "No data entered" --msgbox "please try again enter expire date.... " 8 100
                 EXPIREDATE=$(whiptail --title "EXPIRE DATE " --inputbox "Enter Expire Date for $1  eg :-(2023-10-31)..... " 8 40 3>&1 1>&2 2>&3)
              else
               chage -E $EXPIREDATE   $1
                whiptail --title "successfully" --msgbox "$1 account is change expire date to( $EXPIREDATE  ) " 8 100
              fi
              fi          ;;
       "4)") chage -M 90   $1
             whiptail --title "successfully" --msgbox " Change the maximum number of days to change the password after 90 day for--->  $1 " 8 100
             ;;
       "5)") chage -I 5 $1
             whiptail --title "successfully" --msgbox " Change the Inactive of days to change the password after 5  day for--->  $1 " 8 100
              ;;
        *) u_pass_expiry ;;
       esac

}
```

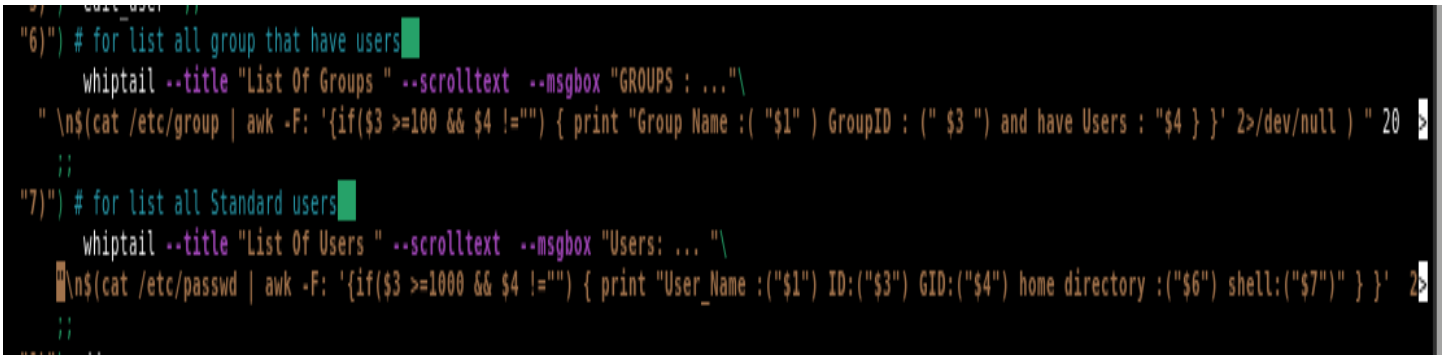
6. list all group that have users



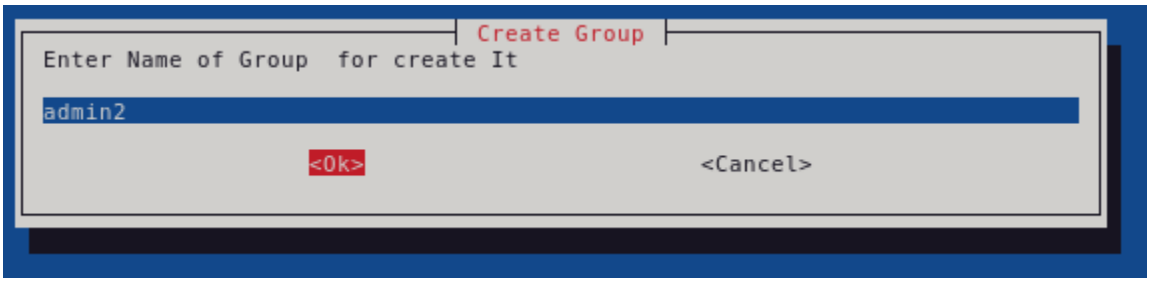
7. list all Standard Users .

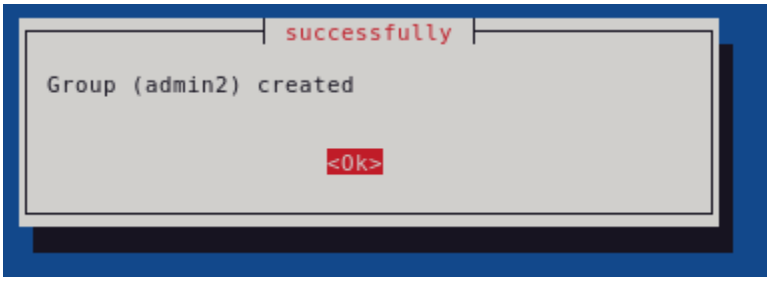


Code



8.Add New Group .

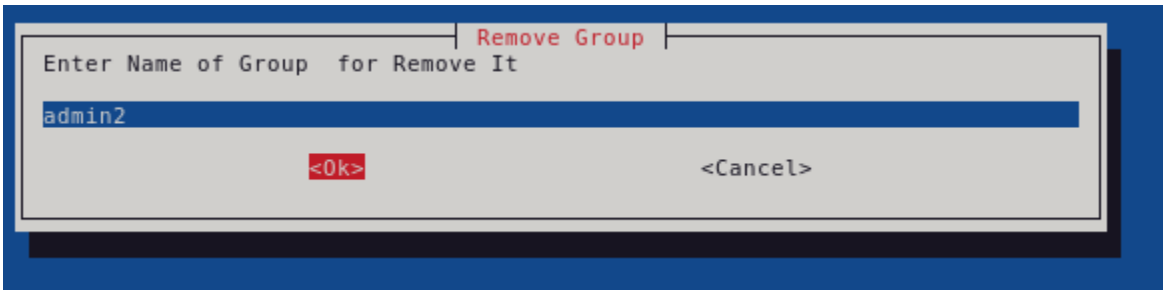


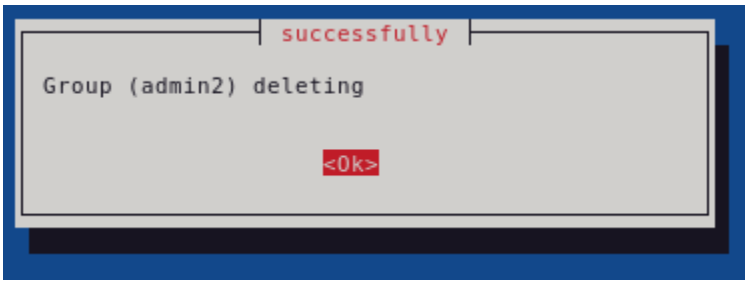


Code

```
#************************************************for add group ***************************************************************************
function add_group(){
    GROUP_NAME=$(whiptail --title "Create Group" --inputbox "Enter Name of Group  for create It" 8 78  3>&1 1>&2 2>&3)
    if [ $? -eq 0 ]; then
## for check that input not empty
    if [ -z $GROUP_NAME ];then
       whiptail --title "No data entered" --msgbox "please try again to enter group name .... " 8 78
       add_group
    else
## for check group existed or not
       getent group $GROUP_NAME  > /dev/null 2&>1
       if [ $? -eq 0 ]; then
           whiptail --title "available" --msgbox "the $GROUP_NAME group does exist in the system" 8 78
       else
           groupadd  $GROUP_NAME
           whiptail --title "successfully" --msgbox "Group ($GROUP_NAME) created " 8 50
       fi
   fi
   fi
}

#################################################end adding group ###########################################################################
```

9.Delete Group.

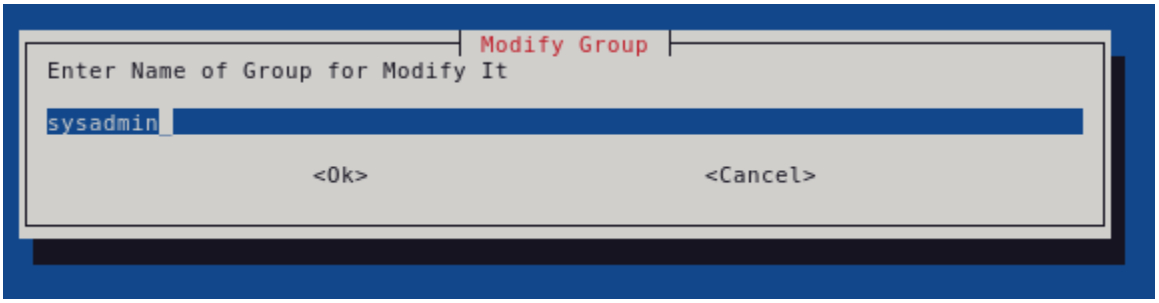


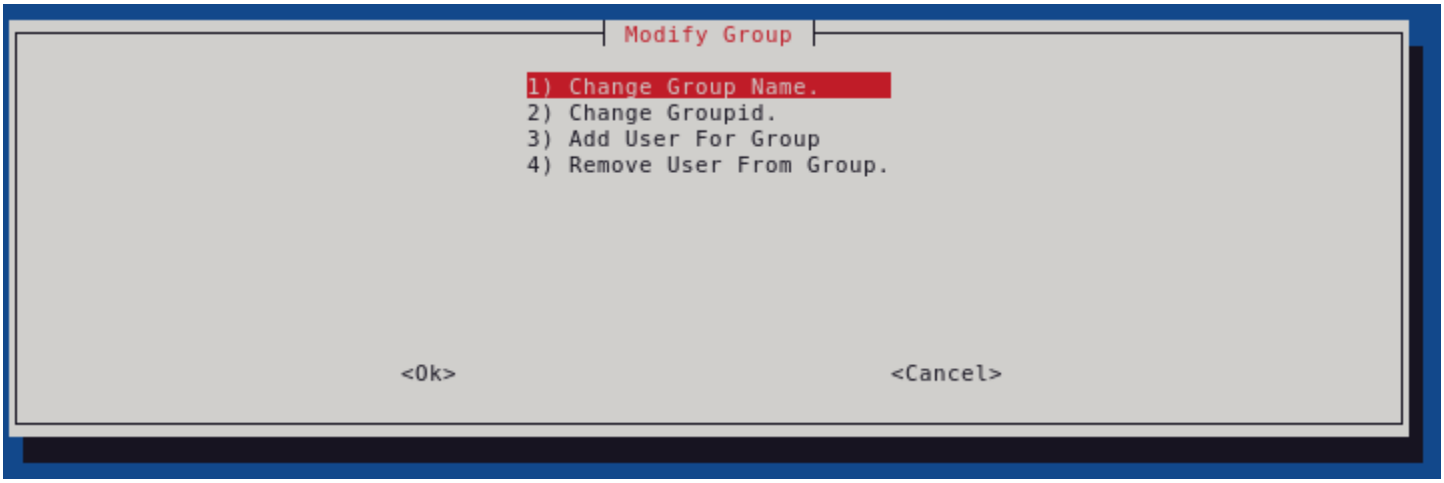


Code

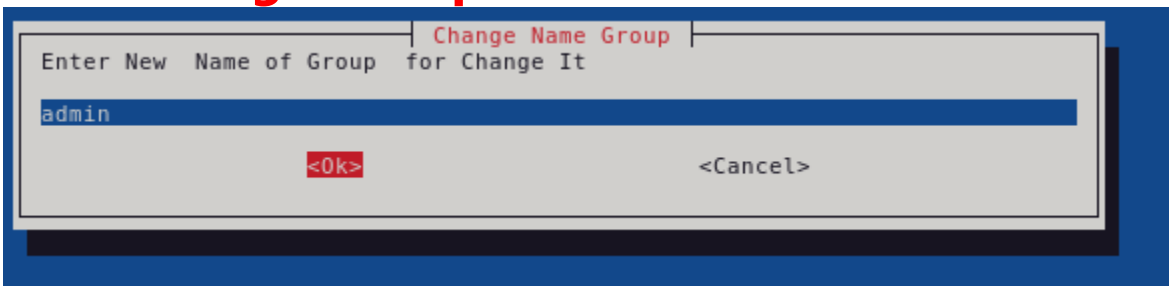
```
function del_group(){
    GROUP_del=$(whiptail --title "Remove Group" --inputbox "Enter Name of Group  for Remove It" 8 78  3>&1 1>&2 2>&3)
    if [ $? -eq 0 ]; then
## for check that input not empty
    if [ -z $GROUP_del ];then
       whiptail --title "No data entered" --msgbox "please try again to enter group name .... " 8 78
       del_group
    else
## for check group existed or not
       getent group $GROUP_del  > /dev/null 2&>1
       if [ $? -eq 0 ]; then
           groupdel  $GROUP_del
           whiptail --title "successfully" --msgbox "Group ($GROUP_del) deleting  " 8 50
       else
           whiptail --title "available" --msgbox "the $GROUP_del group does not exist in the system" 8 78
       fi

   fi
   fi
}


#*********************************************************************
```

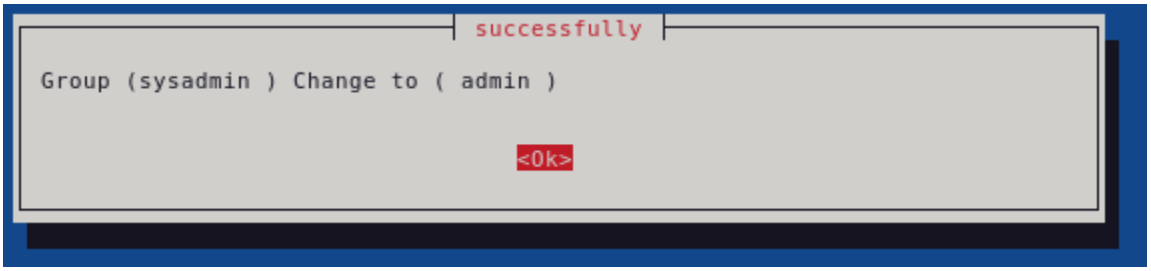
10.Modify Group .





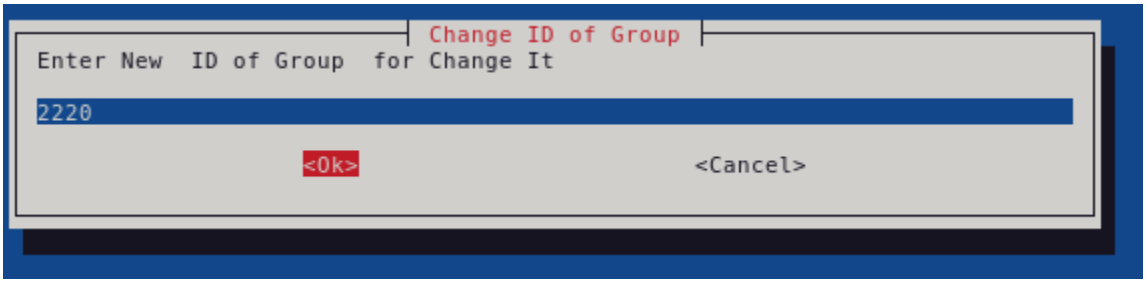
10.1.Change Group Name.

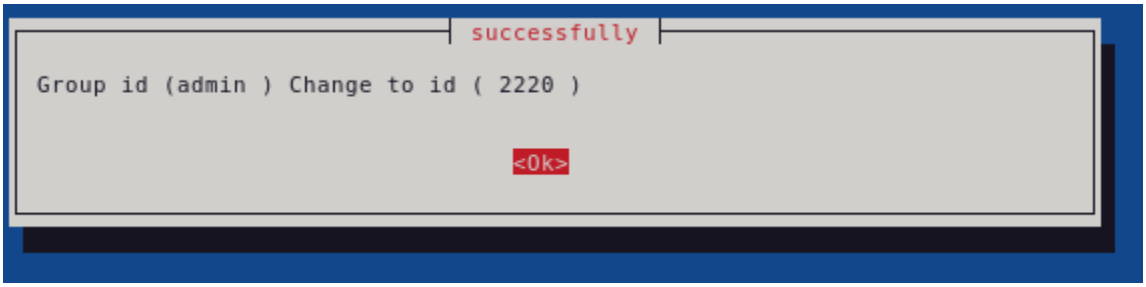





```
#******************************* function Change Group Name  ***************************************
function ch_name_group(){
       new_name=$(whiptail --title "Change Name Group" --inputbox "Enter New  Name of Group  for Change It" 8 78  3>&1 1>&2 2>&3)
       if [ $? -eq 0 ]; then
## for check that input not empty
      if [ -z $new_name ];then
         whiptail --title "No data entered" --msgbox "please try again to enter group name .... " 8 78
         ch_name_group
      else
         groupmod -n $new_name  $1
         whiptail --title "successfully" --msgbox "Group ($1 ) Change to ( $new_name ) " 8 78
      fi
      fi
}
```

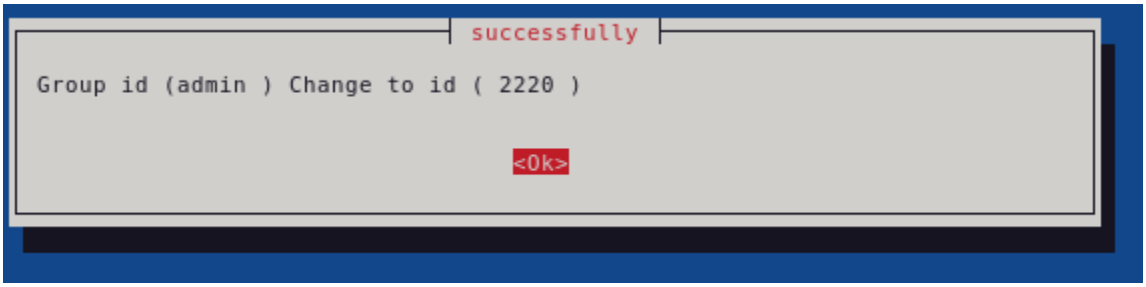
10.2. Change Groupid.





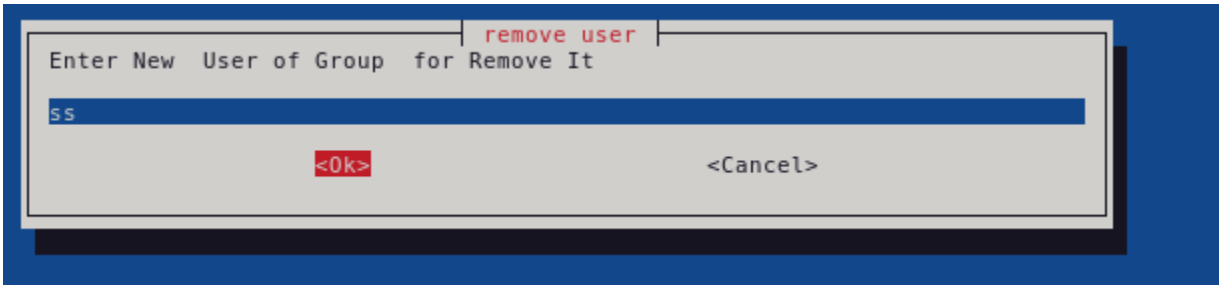

```
#******************************** function Change Group ID  ***************************************
function ch_groupid(){
       new_id=$(whiptail --title "Change ID of Group" --inputbox "Enter New  ID of Group  for Change It" 8 78  3>&1 1>&2 2>&3)
       if [ $? -eq 0 ]; then
## for check that input not empty
       if [ $new_id >= 1000 ];then
          whiptail --title "No data entered" --msgbox "please try again to enter group ID Great then 1000 .... " 8 78
          ch_groupid
       else
          groupmod -g $new_id  $1
          whiptail --title "successfully" --msgbox "Group id ($1 ) Change to id ( $new_id ) " 8 78

       fi
       fi
}
```

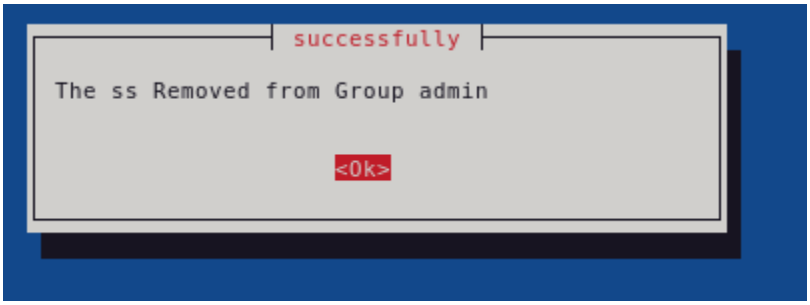
10.3. Add User For Group



```
#********************************* function add User For Group ***************************************
function add_user(){
        user=$(whiptail --title "Add user" --inputbox "Enter Username for Add it to Group " 8 78  3>&1 1>&2 2>&3)
        if [ $? -eq 0 ]; then
## for check that input not empty
       if [ -z $user ];then
          whiptail --title "No data entered" --msgbox "please try again to enter user name  .... " 8 78
          add_user
       else
          getent group $1  | grep -w $user  # > /dev/null 2&>1
          if [ $? -eq 0 ]; then
             whiptail --title "User Exist" --msgbox "the $user is exists at Group $1 ..... " 8 78
          else
              gpasswd  -a $user  $1
              whiptail --title "successfully" --msgbox "the $user Add for group $1  " 8 50
          fi
       fi
       fi
}
```

10.4. Remove User From Group.






```
#********************************* function Remove User From Group ***************************************
function remove_user(){
       user=$(whiptail --title "remove user" --inputbox "Enter New  User of Group  for Remove It" 8 78  3>&1 1>&2 2>&3)
       if [ $? -eq 0 ]; then
## for check that input not empty
      if [ -z $user ];then
         whiptail --title "No data entered" --msgbox "please try again to enter user name  .... " 8 78
         remove_user
      else
         getent group $1  | grep -w $user  # > /dev/null 2&>1
         if [ $? -eq 0 ]; then
            gpasswd  -d $user  $1
            whiptail --title "successfully" --msgbox "The $user Removed from Group $1  " 8 50
         else
             whiptail --title "unavailable" --msgbox "the $user is not exists at Group $1 ..... " 8 78
         fi
      fi
      fi
}
```

Source code

```
##************************** fuction for modify groups ***********************

function modify_group(){
       GROUP_MOD=$(whiptail --title "Modify Group" --inputbox "Enter Name of Group for Modify It" 8 78  3>&1 1>&2 2>&3)
       if [ $? -eq 0 ]; then
## for check that input not empty
       if [ -z $GROUP_MOD ];then
          whiptail --title "No data entered" --msgbox "please try again to enter group name .... " 8 78
          modify_group
       else
## for check group existed or not
          getent group $GROUP_MOD  > /dev/null 2&>1
          if [ $? -eq 0 ]; then

             CHOICE=$(whiptail --title "Modify Group" --menu "Make your choice" 16 100 9 \
                    "1)" "Change Group Name."   \
                    "2)" "Change Groupid."  \
                    "3)" "Add User For Group"  \
                    "4)" "Remove User From Group." 3>&2 2>&1 1>&3  )


              case $CHOICE in
                 "1)") ch_name_group $GROUP_MOD ;;
                 "2)") ch_groupid $GROUP_MOD ;;
                 "3)") add_user $GROUP_MOD ;;
                 "4)") remove_user $GROUP_MOD ;;
                 *) modify_group ;;
              esac
          else
             whiptail --title "GROUP NOT FOUND" --msgbox "this $GROUP_MOD group is not exists ..... " 8 78
          fi

      fi
      fi
}
```

11. Exit from App.

```
##############################  MAIN  ###########################################################
while [ 1 ]
do
CHOICE=$( whiptail --title "System Admin Helper" --menu "Choose What Do you want " 25 100 15 \
       "1)" "Add New User "   \
       "2)" "Delete User "  \
       "3)" "Disable User Account." \
       "4)" "Enable User Account." \
       "5)" "Modify User Account." \
       "6)" "list all group that have users" \
       "7)" "list all Standard Users" \
       "8)" "Add New Group " \
       "9)" "Delete Group " \
       "10)" "Modify Group " \
       "11)" "Exit from App " 3>&2 2>&1 1>&3 )

case $CHOICE in
       "1)")  create_user ;;
       "2)")  delete_user ;;
       "3)")  disable_user ;;
       "4)")  enable_user ;;
       "5)")  edit_user  ;;
       "6)") # for list all group that have users
             whiptail --title "List Of Groups " --scrolltext  --msgbox "GROUPS : ...\n$(cat /etc/group | awk -F: '{if($3 >=100 && $4 !="") { print "Group Name :
           ;;
       "7)") # for list all Standard users
             whiptail --title "List Of Users " --scrolltext  --msgbox "Users: ... \n$(cat /etc/passwd | awk -F: '{if($3 >=1000 && $4 !="") { print "User_Name :(
          ;;
       "8)") add_group ;;
       "9)") del_group ;;
      "10)") modify_group ;;
      "11)") exit ;;
esac
done
exit
```

Thanks