COMPUTER PROJECT

MADE BY:-

- *NAME:* *KABIR GULATI & AKASHAT KUMAR*
- *CLASS:* *XII-'A'*
- *SCHOOL:* ***Air Force Golden Jubilee Institute***

INDEX

PARTICULARS	PAGE NO.
TITLE	1
INDEX	2
ACKNOWLEDGEMENT	3
CERTIFICATE	4
INTRODUCTION	5
BLOCK DIAGRAM	7
HARDWARE AND SOFTWARE USED	8
MODULES AND THEIR BUILT IN FUNCTIONS USED	9
SOURCE CODE	10
OUTPUT SCREEN	24
BIBLIOGRAPHY	27

Acknowledgement

I take this opportunity to express deep gratitude and sincere thanks to my respected computer science teacher Mrs. MOHINI ARORA who guided me to complete this project successfully. I would like to thank her for her invaluable guidance, constant encouragement, constructive attitude and immense motivation which has sustained my effort at all stages of this project work.

Last but not the least; I extend my sincere thanks to the computer lab assistant Mr. Surender Singh for helping me complete this project.

Kabir Gulati

XII ‘A’

Certificate

This is to certify that the computer science project on ‘Gaming Evolution’ has been submitted by the candidate Kabir Gulati of class XII, roll no. _____ for the class XII practical examination conducted by the Central Board of Secondary Education (C.B.S.E.) in the academic year 2020-21.

Mrs. Mohini Arora

HOD Computer Science

Air Force Golden Jubilee Institute

Introduction

The project that you are about to view in the next few pages is based on the topic " Gaming Evolution " and is created by Kabir Gulati and Akashat Kumar of class 12th A . The program has not been limited to only the back hand part of software development but also front hand. We have used the tkinter module for its better smooth graphic display. It's an attractive user friendly, password protected program. The program is easy to use, secure and accessible with the graphic-user interface part of the program.

The program that we have created is a library of various games. It contains both statistical detail as well as the overview details. This app can be used by both game developers and game users.

For Developer:

1) Enter the details about your game, So that the world could know about your wonderful creation. The data will be stored permanently into your hard drive in an organised manner.

2) Entered some wrong data? or need to update. No need to enter the record again. We have the modify feature for you. Modify details about your game in an easy accessible way.

3) There are options for deletion of single or complete records. But The process is password protected, Therefore the security of data is completely safe.

For User:

1) Need to know the specification of a game. We will show you the statistical details of the game.

2) Want to surf about statistics of various games. We have the complete display (statistical) feature.

3) Want to know the plot and other details of the game.We have the complete display (Overview) feature.

4) A SPECIAL FEATURE has also been added wherein you can compare the specs of your PC with the required specs of the game so that you can know the percentage of compatibility of your computer with the game.'

HARDWARE AND SOFTWARE USED

Hardware

Processor	Intel Core i3 1st Gen 380M (2nd Gen) 2.5 Ghz
Chipset	Intel HM55 Express
Graphic Processor	Intel HD Graphics
Capacity	4 GB
RAM type	DDR3
HDD Capacity	640 GB
HDD Speed(RPM)	5400 RPM
HDD type	SATA

Software

Python 3.7.0 (64 Bit), Python 3.8.0 (64 Bit)

MODULES & THEIR BUILT IN FUNCTIONS USED

PICKLE	• Dump • Load
ttk	• Optionmenu
TKINTER	• Toplevel • Messagebox • Label • Entry • Optionmenu • Button • Text • Radiobutton • Canvas • Labelframe
TIME	• Sleep
OS	• Rename • Remove

SOURCE CODE

```
import pickle
import time
from tkinter import *
from tkinter import messagebox
from tkinter import ttk
import os
def name():
   Games=[]
   f=open('Specification.dat','rb+')
   try:
      while True:
         x=pickle.load(f)
         Games.append(x['Game'])
   except EOFError:
      f.close()
   return Games
def empty(x,y):
   Label(root,text="                    ").grid(row=x,column=y)
#Lists
menu_list=['1.Data Entry.','2.Complete Display.','3.Selective
Display','4.Modification','5.Deletion.']
#------------------------------------------- DATA ENTRY
-------------------------------------------------------------------------
def option1():

   for x in range(1):
      top=Toplevel()
      top.title('Data Entry Window')

      top.configure(bg='#1f6f8b')
      def output(D):
         C1=open("Specification.dat","ab+")
         C2=open("Backup.dat","ab+")
         for x in D:
            try:
               D[x]=D[x].get()
            except AttributeError:
               pass
         D["Overview"]=OverviewInput.get(0.0,'end')
         f='Data entry of '+D['Game']+' was successful.'
         messagebox.showinfo('Confirmation', f)
```

```
            pickle.dump(D,C1)
            pickle.dump(D,C2)
            C1.close()
            C2.close()
            top.destroy()
            option1()

        #Labels-----------------------------------------------------------------------------------------------
        L_text=["Enter Name of the Game : ","Enter the Year in which the Game Launched
:","Enter the Latest Version of the Game :","Enter an overview of the game:","Enter the
owner of the Game:","Enter the Creator of the Game:","Choose your Operating System
:","Enter the 'MINIMUM' Processor Requirements for proper functioning of Game
:","Enter the 'MINIMUM' RAM (Random Access Memory) of PC to run the
Game:","Enter the minimum required VRAM:","What is the Size of Your Game:"]
        L_row=[1,2,3,4,9,10,14,15,16,17,18]
        for j in L_text:
            Label(top,text=j,font="Helvetica 12
bold",bg='#1f6f8b',fg='white').grid(row=L_row[L_text.index(j)],column=1,sticky=W)
        Label(top,text="GHz",bg='#1f6f8b',fg='white').grid(row=15,column=3,sticky=W)
        for x in range(3):

Label(top,text="GB",bg='#1f6f8b',fg='white').grid(row=16+x,column=3,sticky=W)
        #Entry Box
Variables--------------------------------------------------------------------------------------------
        VarGame=StringVar()
        VarYear=StringVar()
        VarProcess=DoubleVar()
        VarRAM=IntVar()
        VarVRAM=IntVar()
        VarSize=IntVar()
        choosed=StringVar()
        VarVersion=StringVar()
        VarOwner=StringVar()
        VarCreator=StringVar()
        #Entry
Boxes------------------------------------------------------------------------------------------------
        GameEntry=Entry(top,borderwidth=3,textvariable=VarGame)
        GameEntry.grid(row=1,column=2,sticky=W,padx=10,pady=10,ipadx=100)

        Year=Entry(top,borderwidth=3,textvariable=VarYear)
        Year.grid(row=2,column=2,sticky=W,padx=10,pady=10,ipadx=100)

        VersionInput=Entry(top,borderwidth=3,textvariable=VarVersion)
        VersionInput.grid(row=3,column=2,sticky=W,padx=10,pady=10,ipadx=100)
```

```
OverviewInput=Text(top,wrap=WORD,width=42,height=5,bg='white',fg='black',relief=
SOLID)
    OverviewInput.grid(row=4,column=2)

    OwnerInput=Entry(top,borderwidth=3,textvariable=VarOwner)
    OwnerInput.grid(row=9,column=2,sticky=W,padx=10,pady=10,ipadx=100)

    CreatorInput=Entry(top,borderwidth=3,textvariable=VarCreator)
    CreatorInput.grid(row=10,column=2,sticky=W,padx=10,pady=10,ipadx=100)

    os_types=["Windows","Linux","Android","macOS","iOS","MS-DOS","Windows"]
    choosed.set('Select Your OS')
    dropdown=ttk.OptionMenu(top,choosed,*os_types)
    dropdown.grid(row=14,column=2,sticky=W,padx=10,pady=10,ipadx=100)

    ProcessorInput=Entry(top,borderwidth=3,textvariable=VarProcess)
    ProcessorInput.grid(row=15,column=2,sticky=W,padx=10,pady=10,ipadx=100)

    RAMInput=Entry(top,borderwidth=3,textvariable=VarRAM)
    RAMInput.grid(row=16,column=2,sticky=W,padx=10,pady=10,ipadx=100)

    VRAMInput=Entry(top,borderwidth=3,textvariable=VarVRAM)
    VRAMInput.grid(row=17,column=2,sticky=W,padx=10,pady=10,ipadx=100)

    GamesizeInput=Entry(top,borderwidth=3,textvariable=VarSize)
    GamesizeInput.grid(row=18,column=2,sticky=W,padx=10,pady=10,ipadx=100)

    D={}
    D["Game"]=VarGame
    D["Year"]=VarYear
    D["Version"]=VarVersion
    D["Creator"]=VarCreator
    D["Owner"]=VarOwner
    D["Operating System"]=choosed
    D["Processor"]=VarProcess
    D["RAM"]=VarRAM
    D["VRAM"]=VarVRAM
    D["Size"]=VarSize
    #Buttons-------------------------------------------------------------------------------------------
    def e1_delete():
        top.destroy()
        option1()
```

```
submit=Button(top,text='SUBMIT',borderwidth=3,command=lambda:output(D),bg='w
hite',fg='#1f6f8b',font="Times 12 italic
bold").grid(row=19,column=1,ipadx=100,sticky=E)

done=Button(top,text='EXIT',borderwidth=3,command=lambda:top.destroy(),bg='whit
e',fg='#1f6f8b',font="Times 12 italic bold").grid(row=21,column=1,ipadx=110,sticky=E)
#---------------------------------------------------- COMPLETE DISPLAY
------------------------------------------------------------
def option2():
   displaywin=Toplevel()
   displaywin.geometry('790x230')
   displaywin.configure(bg='white')
   displaywin.title('Complete Display')
   Games=name()
   f=open('Specification.dat','rb+')

text=Text(displaywin,wrap=WORD,width=55,height=13,bg='pink',fg='white',font='Ver
dana 12 bold italic')
   text.grid(row=0,column=0,rowspan=3)
   def info_disp(x=0):
      if x<(len(Games)):
         f.seek(0)
         for z in range(len(Games)):
            c=pickle.load(f)
            if c["Game"]==Games[x]:
               b=c
               y=x
               break
         string='Game Name :'+b['Game']+'\n'+'Year of Release :'+b['Year']+'\n'+'Version
of Game :'+b['Version']+'\n'+'Name of Creator of Game :'+b['Creator']+'\n'+'Name of
Owner of Game :'+b['Owner']+'\n'+'Size of Game :'+str(b['Size'])+' GB'+'\n'+'Operating
System Requirements :'+b["Operating System"]+'\n'+'RAM required :'+str(b['RAM'])+'
GB'+'\n'+'VRAM Required :'+str(b['VRAM'])+' GB'+'\n'+'Processor Requirements
:'+str(b['Processor'])+' GHz'+'\n'
         text.configure(state=NORMAL)
         text.insert(INSERT,string)
         text.configure(state=DISABLED)
         def add(x):
            text.configure(state=NORMAL)
            text.delete(0.0,'end')
            text.configure(state=DISABLED)
            x=x+1
            info_disp(x)
```

```
        def sub(x):
            text.configure(state=NORMAL)
            text.delete(0.0,'end')
            text.configure(state=DISABLED)
            x=x-1
            info_disp(x)

nex=Button(displaywin,text='Next',borderwidth=3,command=lambda:add(x),activebac
kground='yellow',width=20,fg='red',bg='white',font=('Tempus Sans ITC', 10, 'bold'))
        nex.grid(row=0,column=1,sticky=W)

prev=Button(displaywin,text='Previous',borderwidth=3,command=lambda:sub(x),activ
ebackground='yellow',width=20,fg='red',bg='white',font=('Tempus Sans ITC', 10,
'bold'))
        prev.grid(row=1,column=1,sticky=W)
        if x==0:
            prev.configure(state=DISABLED)
    else:
        messagebox.showerror('Error','No more Data to Show !')
  info_disp()

Button(displaywin,text='Exit',borderwidth=3,command=lambda:displaywin.destroy(),a
ctivebackground='yellow',width=20,fg='red',bg='white',font=('Tempus Sans ITC', 10,
'bold')).grid(row=2,column=1,sticky=W)
  return
#------------------------------------------------------ SELECTIVE DISPLAY
------------------------------------------------------------
def option3():
  try:
    displaywin=Toplevel()
    displaywin.geometry('650x450')
    displaywin.configure(bg='#f7d1ba')
    displaywin.title('Selective Display')
    Games=name()
    def info_disp(a):
        text.configure(state=NORMAL)
        text.delete(0.0,END)
        f=open('Specification.dat','rb+')
        for z in range(len(Games)):
            b=pickle.load(f)
            if(b['Game']==a.get()):
                string='Game Name                    :'+b['Game']+'\n'+'Year of Release
:'+b['Year']+'\n'+'Version of Game                :'+b['Version']+'\n'+'Name of Creator
of Game          :'+b['Creator']+'\n'+'Name of Owner of Game
```

```
:'+b['Owner']+'\n'+'Size of Game                    :'+str(b['Size'])+'
GB'+'\n'+'Operating System Requirements :'+b["Operating System"]+'\n'+'RAM
required                    :'+str(b['RAM'])+' GB'+'\n'+'VRAM Required
:'+str(b['VRAM'])+' GB'+'\n'+'Processor Requirements        :'+str(b['Processor'])+'
GHz'+'\n'+'-'*20+'\n'
                text.configure(state=NORMAL)
                text.configure(font="Times 12 bold")
                text.insert(INSERT,string)
                text.configure(state=DISABLED)
        select=StringVar()
        select.set('Choose The Game')
        Label(displaywin,text='Select the Name of the Game whose data you want to
Display :',bg='#f7d1ba',fg='black').grid(row=10,column=0,columnspan=2,sticky=E)

menu=ttk.OptionMenu(displaywin,select,*Games).grid(row=10,column=2,sticky=W,pa
dx=10,pady=10)#,ipadx=70)
        Label(displaywin,text='OUTPUT BOX',bg='#f7d1ba',fg='black').grid(row=0)

text=Text(displaywin,wrap=WORD,width=85,height=15,bg='#557571',fg='white',state=
DISABLED)
        text.grid(row=1,column=0,columnspan=3)
        Label(displaywin,text='       ',bg='#f7d1ba').grid(row=2)
        SHOW=Button(displaywin,text='Show
Data',borderwidth=3,command=lambda:info_disp(select),activebackground='yellow')
        SHOW.grid(row=3,column=0,ipadx=70,sticky=E)

Button(displaywin,text='Exit',borderwidth=3,command=lambda:displaywin.destroy(),a
ctivebackground='yellow').grid(row=3,column=2,ipadx=100,sticky=W)

    except TypeError:
        displaywin.destroy()
        messagebox.showinfo('Empty!', 'ALL THE DATA HAS BEEN REMOVED')
#---------------------------------------------------- MODIFICATION
-----------------------------------------------------------------------
def option4():
    modify=Toplevel()
    modify.title('Modification Window')
    #Labels-------------------------------------------------------------------------------------------------
    Label(modify,text='Select the Name of the Game You want to Modify
:').grid(row=0,sticky=W)
    empty(1,0)
    Label(modify,text='Choose the Option You Want to Modify').grid(row=2,sticky=W)
    #Entry Boxes--------------------------------------------------------------------------------------------
    p=name()
```

```
   modifygame=StringVar()
   modifygame.set('Select Game to Modify')
   ModifyGame=ttk.OptionMenu(modify,modifygame,*p)
   ModifyGame.grid(row=0,column=1,sticky=W)
   #Radio Button----------------------------------------------------------------------------------------
   f=open("Specification.dat",'rb+')
   temp=open("Clone.dat","wb+")
   b=StringVar()
   def change(val):
      empty(10,0)
      val1=val
      if val not in ["Owner","Size"]:
         val1=val+" requirement"
      str='Enter the New '+ val1 +'of the Game :'
      Label(modify,text=str).grid(row=11,sticky=W)
      new=Entry(modify,borderwidth=3)
      new.grid(row=11,column=1,sticky=W)
      def make_change():
         try:
            while True:
               i=pickle.load(f)
               if(i['Game']==modifygame.get()):
                  messagebox.showinfo('Modify','Changes To The Selected Game are
applied')
                  pre_owner=i[val]
                  i[val]=new.get()
                  pickle.dump(i,temp)
               else:
                  pickle.dump(i,temp)
         except EOFError:
            pass
         f.close()
         temp.close()
         os.remove('Specification.dat')
         os.rename('Clone.dat','Specification.dat')
      Button(modify,text='Click To Make
Changes',borderwidth=3,command=make_change).grid(row=13,column=1,padx=50,sti
cky=W)
   radios=['Owner Name','Operating System Requirements','Processor
Requirements','RAM required','VRAM required','Size of Game']
   radio_vals=['Owner',"Operating System","Processor","RAM","VRAM","Size"]
   for rr in range(6):
```

```
Radiobutton(modify,text=radios[rr],variable=b,value=radio_vals[rr]).grid(row=3+rr,stic
ky=W)
   b.set('Owner')

submit1=Button(modify,text='SUBMIT',borderwidth=3,command=lambda:change(b.get
())).grid(row=9,sticky=W)
#-----------------------------------------------DELETION-----------------------------------------------
----------------
def option5():
   delwin=Toplevel()
   delwin.title('Deletion Window')
   delwin.configure(bg='grey')
   Label(delwin,text="    Select the name of the game whose data is to be deleted
",bg='grey',fg='white',font=('Fixedsys',18,'bold italic')).pack()
   games=name()
   game_name=StringVar()
   game_name.set('Entry to be deleted')
   e=ttk.OptionMenu(delwin,game_name,*games,)
   e.pack(ipadx=20)

   def DELETE(DELNAME):
      if messagebox.askyesno('Verify', 'Are you Sure ?'):
         B1=open("Specification.dat","ab+")
         C1=open("Specification1.dat","wb+")
         Games=name()
         B1.seek(0)
         for z in range(len(Games)):
            try:
               dic=pickle.load(B1)
               if(dic["Game"]==DELNAME):
                  messagebox.showinfo('Deleted', "The Selected Entry Has Been Deleted
Successfully !!!")
               if(dic['Game']!=DELNAME):
                  pickle.dump(dic,C1)
            except EOFError:
               pass
         C1.close()
         B1.close()
         os.remove("Specification.dat")
         os.rename("Specification1.dat","Specification.dat")
   Label(delwin,text=" ",bg='grey').pack()
```

```
Button(delwin,text="DELETE",command=lambda:DELETE(game_name.get()),width=1
8).pack()
   Button(delwin,text="EXIT",command=lambda:delwin.destroy(),width=18).pack()
def OverviewOption():
   F=open("Specification.dat",'rb+')
   root1 = Toplevel()
   root1.title('Overview of All Games')
   T = Text(root1, height=30, width=60,bg='black')
   T.pack()
   try:
      s='Use Cursor keys to scroll'+'\n'+'through complete document!'+'\n'+'\n'
      T.insert(END,s,'big')
      while True:
         a=pickle.load(F)
         s=a['Game']+':'+'\n'
         length=len(a['Game'])
         T.insert(END,s,'big')
         T.insert(END,a['Overview']+'\n'+'-'*53+'\n','colour')
         T.tag_configure('big', font=('Verdana', 15, 'bold'),foreground='white')
         T.tag_configure('colour',foreground="lightblue",font=('Tempus Sans ITC', 12,
'bold'))
   except EOFError:
      pass
   F.close()
   Button(root1,text="EXIT",command=lambda:root1.destroy()).pack()

#-------------------------------SYSTEM
COMPATIBILITY-----------------------------------------------
def compatibility():
   comp=Toplevel()
   comp.title('Compatability Test')
   comp.configure(bg='#4f8a8b')
   comp.geometry('1200x700')
   games=name()
   Label(comp,text='To check the Compatability of Your PC with a Game we need to
know the specs your PC
:',bg='#4f8a8b',fg='white',font=(None,17)).grid(row=0,column=0,sticky=W)
   def proceed():
      labels=['         Your Operating System :','         Your Processor :','         Your RAM :','
Your VRAM :','Choose the Game Whose Compatibility You want to check with Your PC
:']
      rows=[2,3,4,5,7]
      for g in labels:
```

```
Label(comp,text=g,bg='#4f8a8b',fg='white',font=(None,17)).grid(row=rows[labels.index
(g)],sticky=W)
        os_types=["Windows","Linux","Android","macOS","iOS","MS-DOS","Windows"]
        RAMs=[1,2,3,4,8,16,1]
        VRAMs=[2,4,8,16,2]
        os=StringVar()
        ram=IntVar()
        vram=IntVar()
        gamevar=StringVar()
        for i in [os,ram,vram,gamevar]:
            i.set('Make Your Choice')
        ttk.OptionMenu(comp,os,*os_types).grid(row=2,column=1,ipadx=30,sticky=W)
#--------Operating Systems
        pro=Entry(comp,borderwidth=3,width=22)
        pro.grid(row=3,column=1,sticky=W)
        Label(comp,text="GHz",fg='white',bg='#4f8a8b').grid(row=3,column=2,sticky=W)
        ttk.OptionMenu(comp,ram,*RAMs).grid(row=4,column=1,ipadx=50,sticky=W)
#----------RAM's
        Label(comp,text="GB",fg='white',bg='#4f8a8b').grid(row=4,column=2,sticky=W)
        ttk.OptionMenu(comp,vram,*VRAMs).grid(row=5,column=1,ipadx=50,sticky=W)
#----------VRAM's
        Label(comp,text="GB",fg='white',bg='#4f8a8b').grid(row=5,column=2,sticky=W)
        Label(comp,text='',bg='#4f8a8b').grid(row=6,sticky=W)

ttk.OptionMenu(comp,gamevar,*games).grid(row=7,column=1,ipadx=30,sticky=W)
        def check():
            f=open('Specification.dat','rb+')
            while True:
                u=pickle.load(f)
                if(u['Game']==gamevar.get()):
                    count_percent=0
                    a=eval(pro.get())
                    b=float(u["Processor"])
                    if(b<a):
                        count_percent=count_percent+25
                    else:
                        i=(a/b)*25
                        count_percent=count_percent+i
                    if (int(u['RAM'])<ram.get()):
                        count_percent=count_percent+25
                    else:
                        i=(ram.get()/int(u['RAM']))*25
                        count_percent=count_percent+i
```

```
                if (u['VRAM']<vram.get()):
                    count_percent=count_percent+25
                else:
                    i2=(vram.get()/u['VRAM'])*25
                    count_percent=count_percent+i2
                if(u['Operating System']==os.get()):
                  count_percent=count_percent+25
                break
            arc_extent=(count_percent/100)*360
            if(arc_extent==360):
               arc_extent=359
            Label(comp,text=' ',bg='#4f8a8b').grid(row=10)
            canvas=Canvas(comp,height=300,width=500,bg='black')
            arc=canvas.create_arc(50,50,250,250,extent=arc_extent,fill='yellow')
            ball=canvas.create_oval(140,100,160,120,fill='black')
            ball=canvas.create_oval(280,130,320,170,fill='blue')
            ball=canvas.create_oval(350,130,390,170,fill='blue')
            ball=canvas.create_oval(420,130,460,170,fill='blue')

            canvas.grid(row=11)
            Label(comp,text=' ',bg='#4f8a8b').grid(row=12)
            t=Text(comp,width=60,height=1)
            string=str(count_percent)+"% is your PC's Compatibility"
            t.insert(0.0,string)
            t.grid(row=13)
            canvas.grid(row=11)
        Label(comp,text=' ',bg='#4f8a8b').grid(row=8)
        Button(comp,text='Check
Compatibility',borderwidth=3,command=check).grid(row=9)

Button(comp,text='Proceed',borderwidth=3,command=proceed).grid(row=0,column=1,
sticky=W,ipadx=30)

Button(comp,text='Exit',borderwidth=3,command=lambda:comp.destroy()).grid(row=0,
column=2,sticky=E,ipadx=20)
    Label(comp,text='       ',bg='#4f8a8b').grid(row=1)
#-----------------------------PASSWORD CHECK DEFINATION
-------------------------------------------
def password(pswrd,y):
    def submit():
            if(PASSWORD.get()==pswrd):
                passwin.destroy()
                y()
            else:
```

```
            messagebox.showerror('ERROR !','The Password you entered is incorrect')
    passwin=Toplevel()
    passwin.title('PASSWORD')
    PASSWORD=StringVar()
    pass_lbl=Label(passwin,text='Enter the Password to continue
:').grid(row=0,column=0)

password_entry=Entry(passwin,textvariable=PASSWORD,borderwidth=3,show='*').gri
d(row=0,column=1)

pass_submit=Button(passwin,text='Continue',command=submit).grid(row=1,column=1
,sticky=W)
#-------------------------------------------------------------------------------------------------


root=Tk()
root.title("Gaming Evolution")
root.geometry('500x490')
empty(0,0)
canvas=Canvas(root,height=150,width=500)
canvas.grid(row=1,columnspan=2,sticky=E)

ball=canvas.create_oval(0,15,110,135,fill='black')
ball1=canvas.create_oval(390,15,500,135,fill='black')

canvas.create_text(250,75,fill='#f0f0ed',font="Verdena 40 italic bold",text="Welcome")
canvas.create_text(50,75,fill='#f0f0ed',font="Times 12 italic bold",text="    BY\n
AKASHAT")
canvas.create_text(450,75,fill='#f0f0ed',font="Times 12 italic bold",text="BY\nKABIR")

xspeed=10
yspeed=0

#**************************************************************************************************
**********

empty(2,0)
a=IntVar()
frame=LabelFrame(root,text='Authorised Access
Options',padx=5,pady=5,bg='lightblue',fg='black')
frame.grid(row=3,column=0)
frame1=LabelFrame(root,text='Outputs',padx=5,pady=5,bg='lightblue',fg='black')
frame1.grid(row=3,column=1,sticky=W)
def selected(val,x=0):
```

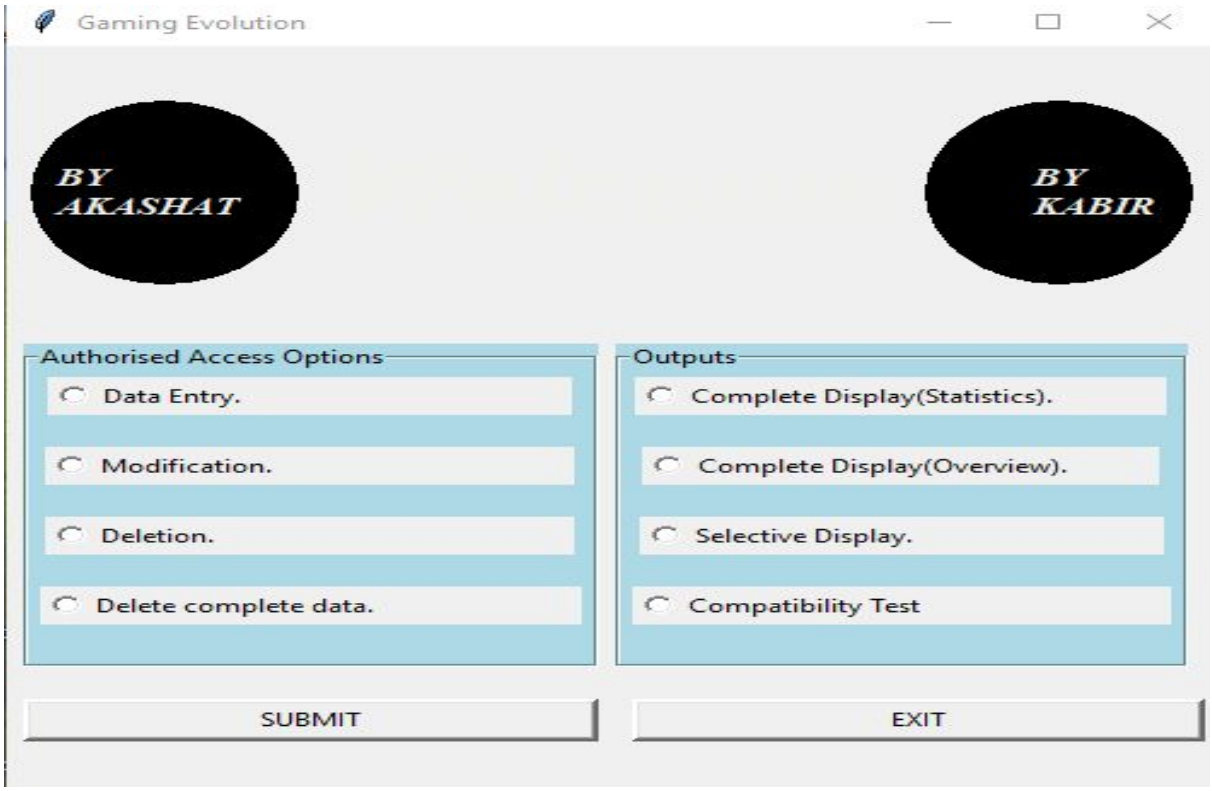
```
    x=val
    if(x==1):
        password('python',option1)
    if(x==2):
        password('python',option4)
    if(x==3):
        password('python',option5)
    if(x==4):
        def correct():
            if messagebox.askyesno('Verify', 'Do you wish to clear all data?'):
                if messagebox.askyesno('Verify', 'You can not get it afterwards!'):
                    F=open("Specification.dat",'wb+')
                    F.close()
                    F=open("Games.txt",'w+')
                    F.close()
                    messagebox.showinfo('Delete', 'Complete data is deleted!')
        password('factoryreset',correct)
    if(x==5):
        option2()
    if(x==6):
        OverviewOption()
    if(x==7):
        option3()
    if(x==8):
        compatibility()

radio_authorised=['Data Entry.                                    ','Modification.
','Deletion.                                    ','Delete complete data.                    ']
radio_output=['Complete Display(Statistics).          ','Complete Display(Overview).
','Selective Display.                          ','Compatibility Test                          ']
g=4
for d in range(len(radio_authorised)):
    Radiobutton(frame,text=radio_authorised[d],variable=a,value=d+1,fg='black').pack()
    Label(frame,text="                ",bg='lightblue').pack()
for d in range(len(radio_output)):
    Radiobutton(frame1,text=radio_output[d],variable=a,value=g+d+1,fg='black').pack()
    Label(frame1,text="                ",bg='lightblue').pack()
empty(5,0)
submit=Button(root,text='SUBMIT',borderwidth=3,command=lambda:selected(a.get())).
grid(row=6,column=0,ipadx=92)
Button(root,text='EXIT',borderwidth=3,command=lambda:exit()).grid(row=6,column=1
,ipadx=101)
#*****************************************************************************************
*****************************************
```

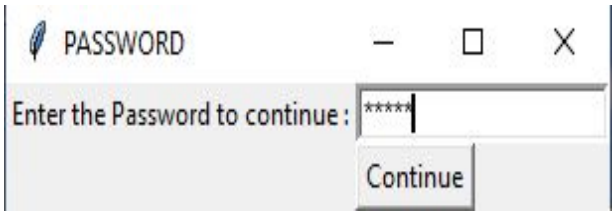
```
while True:
    canvas.move(ball,xspeed,yspeed)
    canvas.move(ball1,-xspeed,yspeed)
    time.sleep(0.1)
    pos=canvas.coords(ball)   #[left,top,right,bottom]
    if pos[2]==300:
        xspeed=-xspeed
    if pos[0]==0:
        xspeed=xspeed
        time.sleep(0.2)
    pos1=canvas.coords(ball1)   #[left,top,right,bottom]
    if pos1[2]==500:
        xspeed=-xspeed
        time.sleep(0.2)
    if pos1[0]==200:
        xspeed=xspeed
    root.update()

root.mainloop()
```

OUTPUT SCREENS

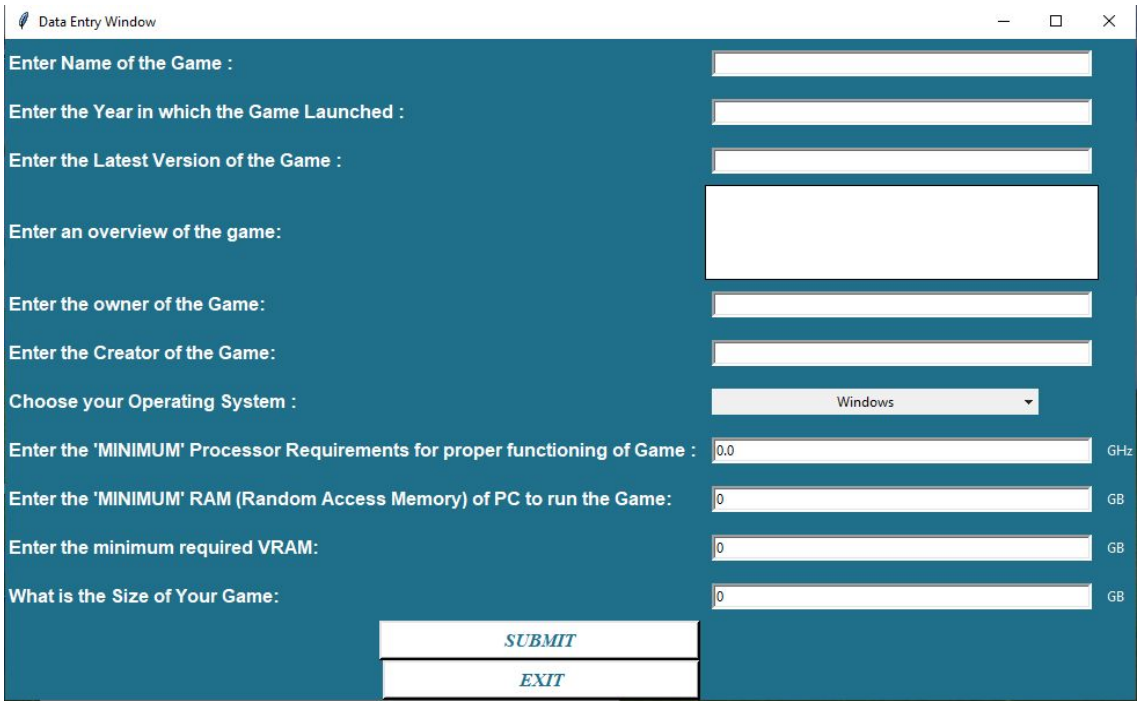


MENU

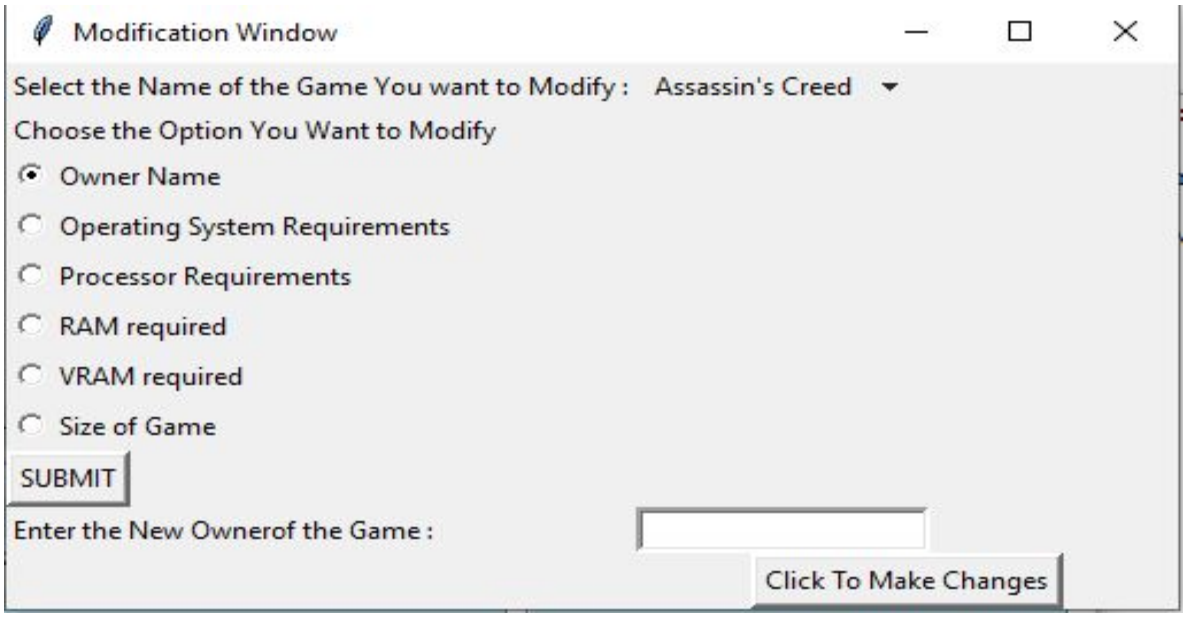


PASSWORD BOX

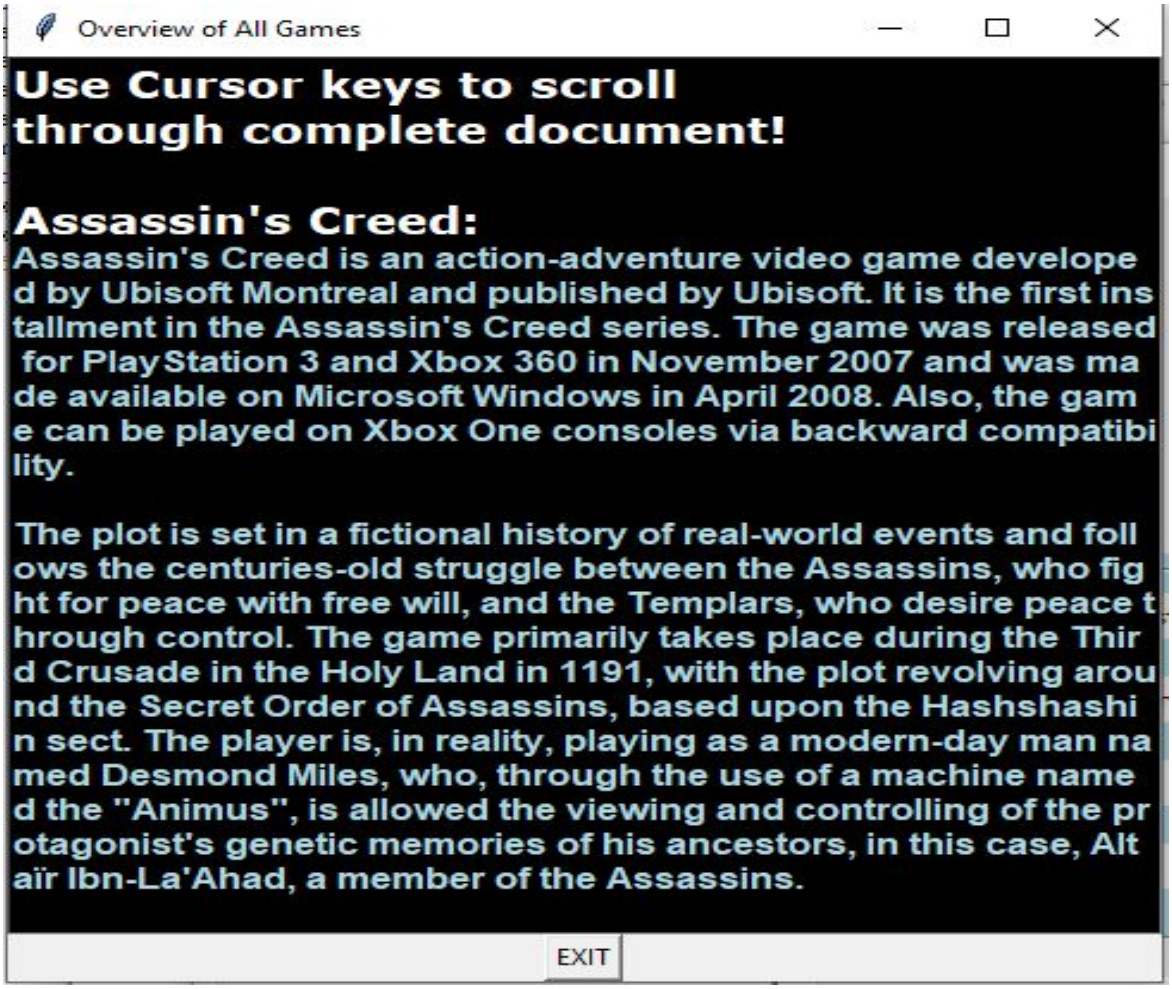
DELETION WINDOW



Data Entry Window

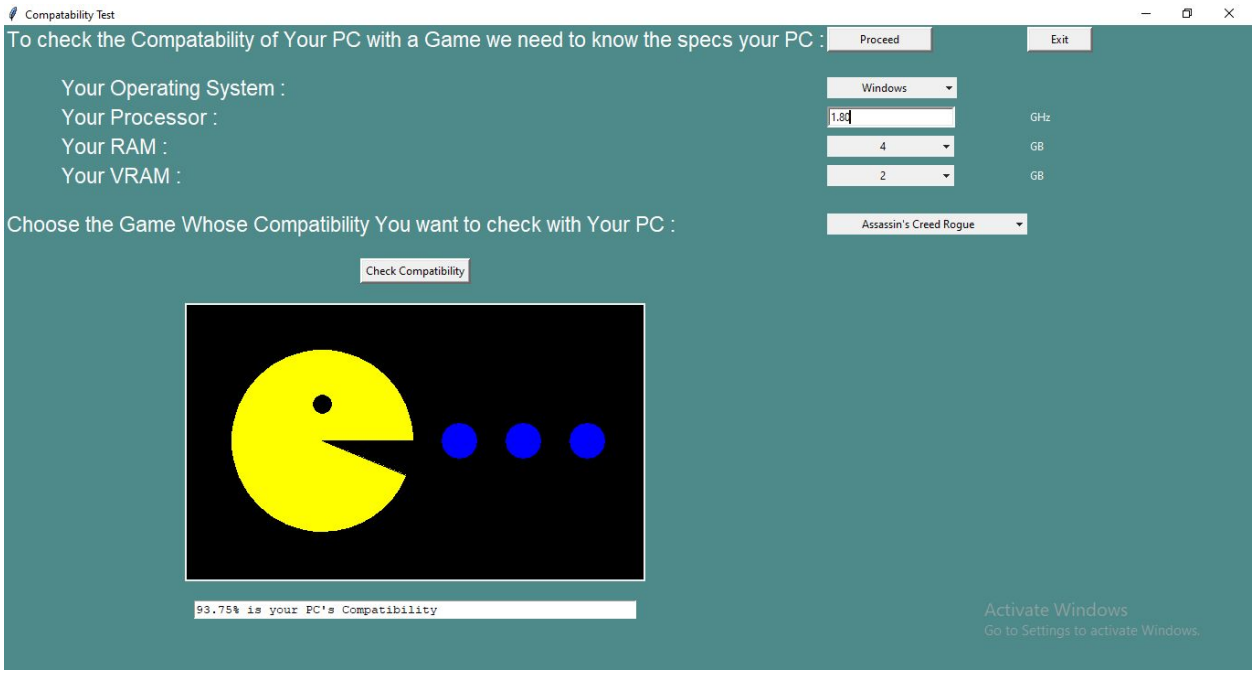


Modification Window

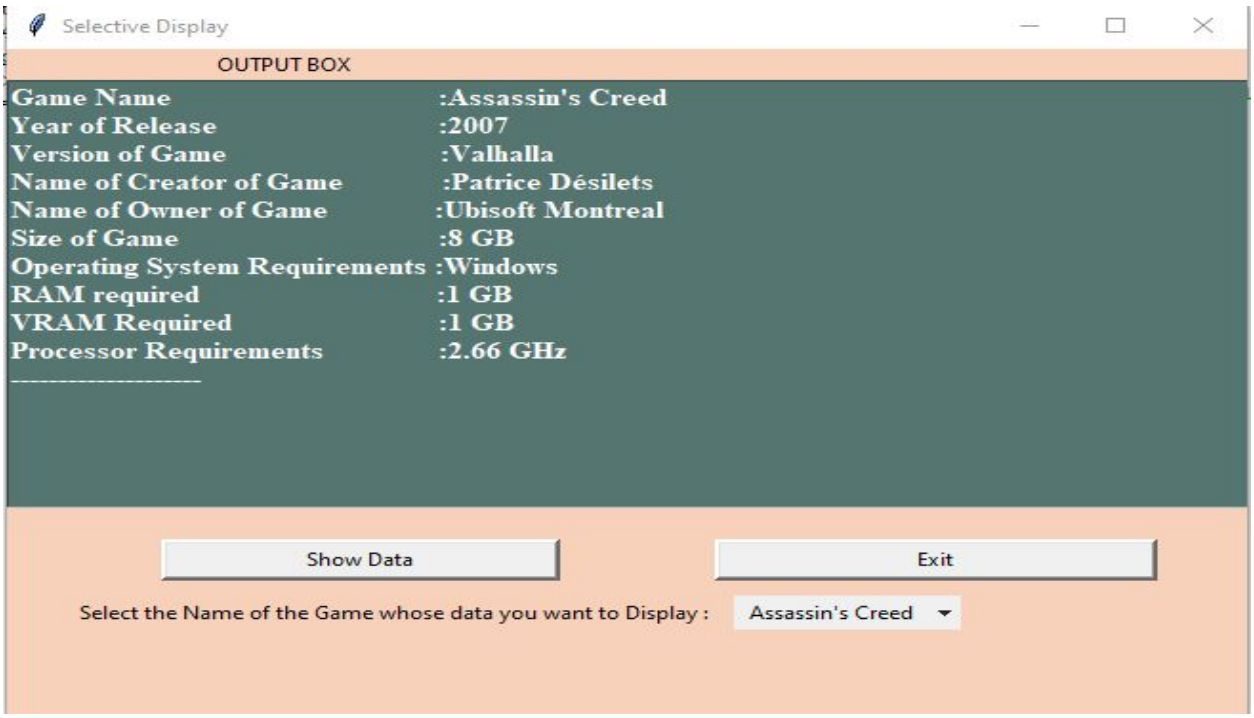


Complete Display (overview)

Complete Display (Statistics)



Compatibility Tester



Selective display

BIBLIOGRAPHY

Tkinter Course - Create Graphic User Interfaces in Python Tutorial
https://www.youtube.com/watch?v=YXPyB4XeYLA

Python Course:
https://www.python-course.eu/python_tkinter.php