Index

Chapter 1 : Introduction to project

The project opted by me as a part of institutional training is a **RockPaperScissors Game** constructed in python language using **tkinter** library. The game is in GUI.

Components of the game :-

- Log In/ Sign Up Page
- Welcome Page
- Game page
- Leaderboard page
- Result page

Features :-

Log In/ Sign Up page – This page contains 2 buttons to switch to either page. The page is connected to a database. The data entered in Sign Up form is stored in a database table. When a person tries to log In, the data entered is then checked, if it is present in the database. If the username and password exists for a user then the welcome page is loaded else an error dialog box is shown stating username or password is incorrect.

Welcome Page – This page welcomes the user by his/her name on the top. Also the page has two photos one of which represents the computer and the other one is either a male or a female depending on the gender of the user. **Game Page** – This page contains 3 buttons each with an image(Rock, Paper Scissors). There is a Scoreboard which keeps count of the score of the user and the computer. There are 2 buttons, one to reset the state of the buttons and the other one to view the leaderboard page. When the user clicks on any one of the button(Rock, Paper, Scissors) then the images of all the 3 buttons changes reflecting the result.

After the game is over a result page appears and the buttons freezes. The result is stored in the database.

Leaderboard Page – This page retrieves its data from the database. It shows all the users who have signed up; their name, matches played, matches won, matches lost, matches tied and winning rate. There are 3 buttons at the bottom. Two are of Light and Dark theme by which we can change the theme. The other one is Exit, to exit the game. **Result Page** – This page announces the result of the game, if you won or lose or tied. There are 2 buttons; one is of exit and the other one of play again. When Play again button is clicked the game again loads for the same user. The user doesn't need to login again to play.

Libraries used :-

1. tkinter
2. random
3. PIL
4. sqlite3

Chapter 2 : Project Requirements

Software Requirements :

- Client environment may be Windows, macOS or Linux.
- Python 3 should be installed.
- IDE for writing and running the program, e.g. Pycharm or Anaconda Navigator
- Libraries such as tkinter, sqlite should be preinstalled before running the program.

Hardware Requirements :

- Physical server or virtual machine. **CPU**: 2 x 64-bit, 2.8 GHz, 8.00 GT/s CPUs
- or better. **Memory**: minimum RAM size of 32 GB, or 16 GB RAM with 1600
- MHz DDR3 installed, for a typical installation with 50 regular users.
- **Storage**: Recommended minimum of 100 GB, or 300 GB if you are planning to mirror both Anaconda Repository, which is approximately 90 GB, and the PyPI repository, which is approximately 100 GB, or at least 1 TB for an air gapped environment. Additional space is recommended if Repository is used to store packages built by your organization.
- Internet access to download the files from Anaconda Cloud, or a USB drive containing all of the files you need with alternate instructions for air gapped installations.

Chapter 3 : Implementation Details

Algorithm for Login/Signup

Step1 : Start

Step2 : Firstly, run the function create() to create the USERS database where the data of signed up users will be stored.

Step3 : When the user is signing up, collect the data and store them in the database.

Step4 : When the user is logging in , take the username and password, check if the data exists in the USERS database in a particular record.

If it exists then load another page or continue

Else show an error dialog box stating incorrect username or password.

Step5 : Stop

Basic Algorithm for Rock-Paper-Scissors Game

```
Winning Rules as follows :


Rock vs paper-> paper wins

Rock vs scissor-> Rock wins

paper vs scissor-> scissor wins.
```

Step1 : Start

Step2 : Read the inputs of Player and computer (Rock or Paper or Scissors)

Step3 : Declare the winner based on the winning rules of the game mentioned above.

Step4 : Stop

CODE :-

```
""" INSTRUCTIONS :- If you are running the project for the first time on your system,
then  call  the  functions  create()  and  LeaderboardCreate()  to  create  the  required
database for the project. This should create .db files in the repository same as this
(.py) file is in. After you have run these 2 functions for once comment the function
calls as the required .db files have already been created. Now you can run this .py
file any number of times. The database files needs to created only once.

create() - 326

LeaderboardCreate() - 347

Note : Do remember to comment these two function calls after you have called them

once. These functions are to be run for only once.
"""

from tkinter import *

import sqlite3
from PIL import ImageTk, Image
from tkinter import messagebox
import random

sqlite3.paramstyle = 'named'

root = Tk()
root.title("Login/Signup Form")
root.geometry("400x400")
root.resizable(width=False, height=False)
lframe = LabelFrame(root, padx = 15, pady = 15, borderwidth=5)
sframe = LabelFrame(root, padx=15, pady=15, borderwidth=5)
USER = ""
UserGender = ""
userWin = 0
compWin = 0
```

```
counter = 0
turns = 5  # After how many turns the game will end
rHandButton = '' pHandButton = '' sHandButton = ''

#----------------------------------------

# Function creating the whole ROCK-PAPER-SCISSORS Game
def play():


    global rHandButton, pHandButton, sHandButton, userWin, compWin, Scoreboard,
resetButton, rockLabel, paperLabel, scissorLabel, buttonHolder, LeaderBoardBtn
    rockLabel = Label(root, text='Rock', bg='#238f02', fg='white', width=35, padx=10,
pady=10)
    paperLabel = Label(root, text='Paper', bg='#de9a03', fg='white', width=35,
padx=10, pady=10)
    scissorLabel = Label(root, text='Scissors', bg='#c20c0c', fg='white', width=35,
padx=10, pady=10)
    rockLabel.grid(row=0, column=0, padx=5, pady=7)
    paperLabel.grid(row=0, column=1, padx=5, pady=5)
    scissorLabel.grid(row=0, column=2, padx=5, pady=5)
    rHandButton = Button(root, image=rHandPhoto, command=lambda: youPick('rock'))
    pHandButton = Button(root, image=pHandPhoto, command=lambda: youPick('paper'))
    sHandButton = Button(root, image=sHandPhoto, command=lambda: youPick('scissors'))
    rHandButton.grid(row=1, column=0)
    pHandButton.grid(row=1, column=1)
    sHandButton.grid(row=1, column=2)
    Scoreboard = Label(root, text="SCORE \n\n    " + USER.upper() + " - " +
str(userWin) + "\t\tCOMPUTER - " + str(compWin), bg='blue',
                       fg='white', padx=10, pady=20)
    Scoreboard.config(font=("Times", 15))
    Scoreboard.grid(row=2, column=0, columnspan=2, sticky=W + E, padx=10, pady=10)
    buttonHolder = Frame(root)
    buttonHolder.grid(row=2, column=2)
    resetButton = Button(buttonHolder, text='RESET', fg='white', command=lambda:
reset_frame(), bg='green', width=30,
                         pady=10)
    resetButton.pack(pady=5)
    LeaderBoardBtn = Button(buttonHolder, text='Leader Board', fg='white',
command=lambda: getLeaderboard(), bg='black',
                            width=30, pady=10)
    LeaderBoardBtn.pack(pady=5)

# Computer randomly picks a choice
def computerPick():
    choice = random.choice(['rock', 'paper', 'scissors'])
    return choice

# Function to play the game again after it is finished once
def playAgain():
    global counter, userWin, compWin
    counter = 0
    userWin = 0
    compWin = 0
    # top.quit()
    start()
```

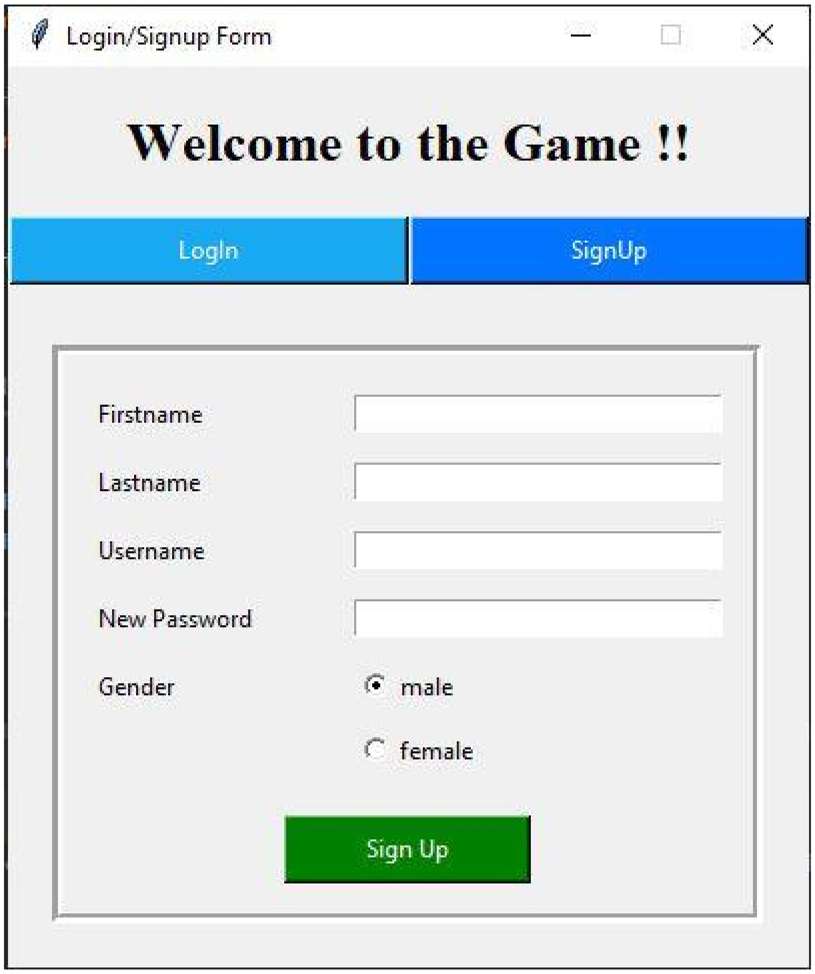
```python
    return

# Function containing the whole logic of won-lose-tie in the game. Decision maker :)
def youPick(yourChoice):
    global click, userWin, compWin, Scoreboard, rockImage, tieImage, paperImage,
loseImage, scissorImage, winImage, compPick, rockLabel, paperLabel, scissorLabel,
counter, turns, top
    compPick = computerPick()
    if click:
        counter += 1
        if yourChoice == 'rock':
            rHandButton.configure(image=rockImage)
            rockLabel.configure(text='Rock')
            if compPick == 'rock':
                pHandButton.configure(image=rockImage)
                sHandButton.configure(image=tieImage)
                paperLabel.configure(text='Rock')
                scissorLabel.configure(text='Tie')
                click = False
            elif compPick == 'paper':
                pHandButton.configure(image=paperImage)
                sHandButton.configure(image=loseImage)
                paperLabel.configure(text='Paper')
                scissorLabel.configure(text='Lose')
                compWin += 1
                click = False
            else:
                pHandButton.configure(image=scissorImage)
                sHandButton.configure(image=winImage)
                paperLabel.configure(text='Scissors')
                scissorLabel.configure(text='Win')
                userWin += 1
                click = False
        elif yourChoice == 'paper':
            rHandButton.configure(image=paperImage)
            rockLabel.configure(text='Paper')
            if compPick == 'rock':
                pHandButton.configure(image=rockImage)
                sHandButton.configure(image=winImage)
                paperLabel.configure(text='Rock')
                scissorLabel.configure(text='Win')
                userWin += 1
                click = False
            elif compPick == 'paper':
                pHandButton.configure(image=paperImage)
                sHandButton.configure(image=tieImage)
                paperLabel.configure(text='Paper')
                scissorLabel.configure(text='Tie')
                click = False
            else:
                pHandButton.configure(image=scissorImage)
                sHandButton.configure(image=loseImage)
                paperLabel.configure(text='Scissors')
                scissorLabel.configure(text='Lose')
                compWin += 1
                click = False
        else:
            rHandButton.configure(image=scissorImage)
            rockLabel.configure(text='Scissors')
            if compPick == 'rock':
                pHandButton.configure(image=rockImage)
                sHandButton.configure(image=loseImage)
                paperLabel.configure(text='Rock')
                scissorLabel.configure(text='Lose')
                compWin += 1 click = False

            elif compPick == 'paper':
                pHandButton.configure(image=paperImage)
                sHandButton.configure(image=winImage)
                paperLabel.configure(text='Paper')
                scissorLabel.configure(text='Win')
                userWin += 1 click = False

            else:
                pHandButton.configure(image=scissorImage)
                sHandButton.configure(image=tieImage)
                paperLabel.configure(text='Scissors')
                scissorLabel.configure(text='Tie')  click  =
                False
    else:
        if yourChoice == 'rock' or yourChoice == 'paper' or yourChoice == 'scissors':
            rHandButton.configure(image=rHandPhoto)
            pHandButton.configure(image=pHandPhoto)
            sHandButton.configure(image=sHandPhoto)
            rockLabel.configure(text='Rock')
            paperLabel.configure(text='Paper')
            scissorLabel.configure(text='Scissors')
            click = True

    Scoreboard = Label(root, text="SCORE \n\n     "+ USER.upper() +" - " +
str(userWin) + "\t\tCOMPUTER - " + str(compWin), bg='blue',
                       fg='white', padx=10, pady=20)
    Scoreboard.config(font=("Times", 15))
    Scoreboard.grid(row=2, column=0, columnspan=2, sticky=W + E, padx=10, pady=10)
    if counter == turns:
    message = ''
    if userWin > compWin:
            message = 'You Won!!'
        elif userWin < compWin:
            message = 'You Lose!!'
        else:
            message = 'You Tied!!'
        insertToLeaderBoard()   top   =   Toplevel()
        top.geometry('300x300')     confetiImg      =
        PhotoImage(file="confeti.gif")   confetiLabel
        =         Label(top,         image=confetiImg)
        confetiLabel.image         =         confetiImg
        confetiLabel.grid(row=0,             column=0)
        messageFrame           =           Frame(top)
        messageFrame.grid(row=0, column=0)  message  =
        Label(messageFrame, text=message)
        message.config(font=("Times", 30,'bold'))
        message.pack()
        exitButton=  Button(messageFrame,text="Exit", bg='red', fg='white',
width=10, padx=3, pady=3,
                            command=root.quit)
        exitButton.config(font=("Times",12))
        exitButton.pack(pady=3, padx=3)
        playAgainBtn =Button(messageFrame,text="PlayAgain", bg='#8953ff',
fg='white', width=10,padx=3,pady=3,
                            command=playAgain)
        playAgainBtn.config(font=("Times",  12))
        playAgainBtn.pack(pady=3,        padx=3)
        rHandButton.configure(state="disabled")
        pHandButton.configure(state="disabled")
        sHandButton.configure(state="disabled")


# Reseting the frame to original startingpictures
def reset_frame():
    global                             click
    rHandButton.configure(image=rHandPhoto)
    pHandButton.configure(image=pHandPhoto)
    sHandButton.configure(image=sHandPhoto)
    click = True

 click =''

# Function creatingthe  GAME   windowreadingthe images
def start():
    global root, click, rHandPhoto,pHandPhoto,  sHandPhoto, userWin, compWin,
rockImage, paperImage,scissorImage, loseImage,  winImage, tieImage
    root.destroy()
    root= Tk()
    root.title('Rock Paper ScissorsGame')
    root.resizable(width=False, height=False)
    click = True
    userWin = 0
    compWin = 0

    # ----------------Image set----------------
    rHandPhoto = PhotoImage(file='rHand.png')
    pHandPhoto = PhotoImage(file='pHand.png')
    sHandPhoto = PhotoImage(file='sHand.png')
    rock= Image.open("Rockimg.jpg")
    rockImage= ImageTk.PhotoImage(rock)
    paper = Image.open("Paperimg.jpg")
    paperImage = ImageTk.PhotoImage(paper)
    scissors= Image.open("Scissorsimg.jpg")
    scissorImage =ImageTk.PhotoImage(scissors)
    win= Image.open("YouWin.jpg")
    winImage= ImageTk.PhotoImage(win)
    lose= Image.open("YouLose.jpg")
    loseImage= ImageTk.PhotoImage(lose)
    tie= Image.open("YouTie.jpg")
    tieImage= ImageTk.PhotoImage(tie)
    #  ------------------------------------------
    play() return


# Function creatingthe WELCOME User Page
def welcomeUserPage():
    global root
    root.destroy()
    root=   Tk()
    root.title("Welcome ^_^ ")
    root.geometry("400x400")
    root.resizable(width=False, height=False)
    name= "Welcome" + USER
    welcomeUser = Label(root,text=   name, pady = 20, width= 25)
    welcomeUser.config(font=("Times", 20,"bold"))
    welcomeUser.grid(row = 0,column  = 0,columnspan  = 2)

    if UserGender == "Male":
        maleimg = ImageTk.PhotoImage(Image.open("Male.jpg"))
        maleLabel =Label(root ,image=maleimg)
        maleLabel.image = maleimg
        maleLabel.grid(row=1,column=0)
        compimg = PhotoImage(file ="computer.png")
        compLabel =Label(root, image=compimg)
        compLabel.image = compimg
        compLabel.grid(row=1,column=1)
    else:
        femaleimg =ImageTk.PhotoImage(Image.open("Female.jpg"))
        femaleLabel=Label(root, image =femaleimg)
        femaleLabel.image = femaleimg
        femaleLabel.grid(row=1, column =0)
        compimg = PhotoImage(file="computer.png")
        compLabel =Label(root, image=compimg)
        compLabel.image = compimg
        compLabel.grid(row=1,column=1)
        text="Start
    StartBtn=  Button(root,             Now", pady=10, width=27, bg='green',
fg='white', command =start)
    StartBtn.grid(row=2, column=0,columnspan  = 2, pady=(30,0))
    return



# Function tocheckifthe    usernameand  password ispresent in the database and is
correct
def check():
    global username, password,USER, UserGender
    conn= sqlite3.connect('Users.db')
    c =conn.cursor()
    c.execute("SELECT*  from Userswhereusername   =:user and password = :pass",
              {
                  'user': username.get(),
                  'pass': password.get()
              }
              )
    data= c.fetchone()
    conn.commit()
    conn.close()

    if username.get()==  "" orpassword.get()  ==  ""or data == None:
        messagebox.showerror("Try   Again","Usernameor password is incorrect.")
        # clear thetextboxes
        username.delete(0, END)
        password.delete(0, END)
    else:
        USER+= data[0]
        UserGender= data[4]
        welcomeUserPage()
    return
  Function
#         tocreate  USERS TABLE ->  To be run onlyoncein   the beginning
def create():
conn= sqlite3.connect('Users.db')
c =conn.cursor()
c.execute("""
CREATE TABLE USERS(
firstnametext,
lastnametext,
usernametext,
passwordtext,
gender text
)
""")
conn.commit()
conn.close()
return
# create()

Function
#         tocreate  LeaderBoard Table ->  To be runonlyonce in the beginning
def LeaderboardCreate():
    conn= sqlite3.connect('LeaderBoard.db')
    c =conn.cursor()
    c.execute("""
    CREATE TABLE LEADERBOARD(
    firstnametext,
    GamesWonint,
    GamesLostint,
    GamesTiedint,
    GamesPlayed int,
    WiningRate real
    )
    """)
    conn.commit()
    conn.close()
    return



# LeaderboardCreate()

# Function to INSERT data in the Leaderboard table
def insertToLeaderBoard():
    conn = sqlite3.connect('LeaderBoard.db')
    c =conn.cursor()
    c.execute("SELECT*  FROM LEADERBOARDWHERE firstname = :USER", {
        'USER' : USER
    })             data=
    c.fetchone()    won=
    data[1]        lost=
    data[2]        tied=
    data[3]           if
    userWin>compWin:
        won += 1
    elifuserWin<compWin:
        lost+= 1
    else:
        tied+= 1
    played =data[4] + 1
    rate= won/played

    c.execute("""

        UPDATE LEADERBOARD SET
        GamesWon =?,
        GamesLost =?,
        GamesTied =?,
        GamesPlayed=?,
        WiningRate= ?
        WHEREfirstname  = ?;""",(won, lost,tied, played, rate, USER))
    conn.commit()
    conn.close()
    return


# Function tochange  to LIGHTthemein   LeaderBoard
def changeLight():
  boardb.configure(bg='white')
  board.configure(bg='white')
  light.configure(bg='black', fg='white')
  dark.configure(bg='black',fg='white')
  return

  Function
#         tochange  to DARK themein  LeaderBoard
def changeDark():
    boardb.configure(bg='black')
    board.configure(bg='black')
    light.configure(bg='white', fg='black')
    dark.configure(bg='white',fg='black')
    return

# Function toexitallthe existingwindows
def exitall():
    boardb.quit()
    root.quit()

# Function toconstruct the LeaderBoard
def getLeaderboard():
    global boardb,board, light, dark
    boardb =Tk()
    boardb.title("Leader Board:)")
```

```python
    boardb.configure(bg = 'white')
    board = Frame(boardb, bg ='white', padx = 10, pady = 10)
    board.grid(row = 0, column = 0, columnspan = 3)
    conn = sqlite3.connect('LeaderBoard.db')
    c = conn.cursor()
    c.execute("SELECT * FROM LEADERBOARD ORDER BY GamesWon DESC")
    data = c.fetchall()
    rowno = 2
    Heading = Label(board, text = "Leaderboard", bg = '#dd2c00', fg = 'white', width
= 45, pady = 10, padx = 10)
    Heading.config(font=("Times", 20, "bold"))
    Heading.grid(row = 0, column = 0, columnspan = 6, padx = 5, pady  = 5)

    name = Label(board, text="Name", width=15, bg='#0069b3', fg='white', padx=5,
pady=5)
    name.grid(row=1, column=0, padx=2, pady=2)
    won = Label(board, text="Games Won", width=15, bg='#a617ff', fg='white', padx=5,
pady=5)
    won.grid(row=1, column=1, padx=2, pady=2)
    lost = Label(board, text="Games Lost", width=15, bg='#e01717', fg='white',
padx=5, pady=5)
    lost.grid(row=1, column=2, padx=2, pady=2)
    tied = Label(board, text="Games Tied", width=15, bg='#e77c00', fg='white',
padx=5, pady=5)
    tied.grid(row=1, column=3, padx=2, pady=2)
    played = Label(board, text="Games Played", width=15, bg='#30b000', fg='white',
padx=5, pady=5)
    played.grid(row=1, column=4, padx=2, pady=2)
    rate = Label(board, text="Winning Rate", width=15, bg='#ff1f60', fg='white',
padx=5, pady=5)
    rate.grid(row=1, column=5, padx=2, pady=2)
    for record in data:

        name = Label(board, text = record[0], width = 15, bg = '#3d7eac', fg =
'white', padx = 5, pady = 5)
        name.grid(row = rowno, column = 0, padx = 2, pady = 3)
        won = Label(board, text=record[1], width = 15, bg = '#b846ff', fg = 'white',
padx = 5, pady = 5)
        won.grid(row = rowno, column=1, padx = 2, pady = 3)
        lost = Label(board, text=record[2], width = 15, bg = '#e33f3f', fg = 'white',
padx = 5, pady = 5)
        lost.grid(row = rowno, column=2, padx = 2, pady = 3)
        tied = Label(board, text=record[3], width = 15, bg = '#e38d2a', fg = 'white',
padx = 5, pady = 5)
        tied.grid(row = rowno, column=3, padx = 2, pady = 3)
        played = Label(board, text=record[4], width = 15, bg = '#62be40', fg =
'white', padx = 5, pady = 5)
        played.grid(row = rowno, column=4, padx = 2, pady = 3)
        rate = Label(board, text=record[5], width = 15, bg = '#ff5385', fg = 'white',
padx = 5, pady = 5)
        rate.grid(row = rowno, column=5, padx = 2, pady = 3)
        rowno += 1

    conn.commit()
    conn.close()
    light = Button(boardb, text = "Light Theme", pady = 7, command = changeLight, bg
= 'black', fg= 'white', width = 30)
    light.grid(row= 1, column = 0,pady= (0, 20))
    dark=  Button(boardb, text="Dark Theme",pady  =7,  command = changeDark, bg =
'black',fg ='white',width =30)
       dark.grid(row=1, column=1,pady = (0,20))
    exit=  Button(boardb, text="Exit",  pady=7,command=exitall, bg='red', fg='white',
width=30)
    exit.grid(row=1, column=2,pady=(0, 20))


# To print the recordson  theGUI  -> To check the recordsentered in the database
def printdata():
    global fname, lname, username,password
    conn= sqlite3.connect('Users.db')
    c =conn.cursor()
    c.execute("SELECT* FROM USERS")
    data= c.fetchall()
    records =Label(root, text = data)
    records.grid(row =3, column =0, columnspan = 2)
    conn.commit()
    conn.close()

# printdata()

# Function toinsert  data in the USERS database -> calledfrom signup
def insert():
    global fname, lname, username,password,gender
    conn1 = sqlite3.connect('Users.db')
    c =conn1.cursor()
    c.execute("INSERTINTO  USERS VALUES(:fname, :lname, :username, :password,
:gender)",
              {
                  'fname':fname.get(),
                  'lname':lname.get(),
                  'username':username.get(),
                  'password':password.get(),
                  'gender' : gender.get()
              }
              )
    conn1.commit()
    conn1.close()
    conn2 = sqlite3.connect('LeaderBoard.db')
    c2 =conn2.cursor()
    c2.execute("""
            INSERTINTO  LEADERBOARDVALUES(:name,  0,0,  0, 0, 0)""",{'name' :
fname.get()})
    conn2.commit()
    conn2.close()
    # clear the textboxes
    fname.delete(0,END)
    lname.delete(0,END)
    username.delete(0,END)
    password.delete(0,END)
    login()
    return
```

```python
loginbool = False
signupbool = False

# Login page creation
def login():
    global signupbool, sframe, loginbool, LoginBtn, SignupBtn, username, password,
root
    if signupbool == True:
        sframe.destroy()
        signupbool = False
    loginbool=True
    root.geometry("400x400")
    LoginBtn.configure(bg = '#0074ff')
    SignupBtn.configure(bg='#19a8f2')
    lframe = LabelFrame(root, padx = 15, pady = 15, borderwidth=5)
    lframe.grid(row = 2, column = 0, columnspan = 2,padx = 5, pady = (30,5))

    usernameLabel = Label(lframe , text = "Username", pady = 5, anchor=W, width = 10)
    usernameLabel.grid(row = 0, column = 0, sticky=W+E, padx= (0,20))
    passwordLabel = Label(lframe, text="Password", pady=5, anchor = W, width = 10)
    passwordLabel.grid(row=1, column=0, sticky=W+E, padx= (0,20))
    username = Entry(lframe, width = 30)
    username.grid(row = 0, column=1)
    password = Entry(lframe, width = 30)
    password.grid(row = 1, column=1)
    Login = Button(lframe, text = "Log In", width = 15, padx = 5, pady = 4,


bg='green', fg='white', command = check)
    Login.grid(row = 2, column = 0, columnspan = 2, padx = 10, pady = (10,0))
    return

# Sign Up Page creation
def signup():
    global sframe, loginbool, signupbool, SignupBtn, LoginBtn, fname, lname,
username, password, root, gender
    if loginbool==True:
        lframe.destroy()
        loginbool = False
    signupbool = True
    root.geometry("400x450")
    SignupBtn.configure(bg='#0074ff')
    LoginBtn.configure(bg='#19a8f2')
    sframe = LabelFrame(root, padx=15, pady=15, borderwidth=5)
    sframe.grid(row=2, column=0, columnspan=2, padx=5, pady=(30, 5))

    fnameLabel = Label(sframe, text = "Firstname", pady=5, anchor=W, width=15)
    fnameLabel.grid(row = 0, column = 0, sticky=W + E, padx=(0, 20), pady=(0,5))
    lnameLabel = Label(sframe, text="Lastname", pady=5, anchor=W, width=15)
    lnameLabel.grid(row=1, column=0, sticky=W + E, padx=(0, 20), pady=(0,5))
    usernameLabel = Label(sframe, text="Username", pady=5, anchor=W, width=15)
    usernameLabel.grid(row=2, column=0, sticky=W + E, padx=(0, 20), pady=(0,5))
    passwordLabel = Label(sframe, text="New Password", pady=5, anchor=W, width=15)
    passwordLabel.grid(row=3, column=0, sticky=W + E, padx=(0, 20), pady=(0,5))
    genderLabel = Label(sframe, text="Gender", pady=5, anchor=W, width=15)
    genderLabel.grid(row=4, column=0, sticky=W + E, padx=(0, 20), pady=(0, 5))

    fname = Entry(sframe, width=30)
    fname.grid(row = 0, column=1, pady=(0,5))
    lname = Entry(sframe, width=30)
    lname.grid(row=1, column=1, pady=(0,5))
    username = Entry(sframe, width=30)
    username.grid(row=2, column=1, pady=(0,5))
    password = Entry(sframe, width=30)
    password.grid(row=3, column=1, pady=(0,5))
    gender = StringVar()
    gender.set("Male")
    Radiobutton(sframe, text = "male", variable = gender, value = "Male", anchor =
W).grid(row = 4, column = 1, pady=(0,5), sticky = W+E)
    Radiobutton(sframe, text="female", variable=gender, value="Female", anchor =
W).grid(row=5, column=1, pady=(0, 5), sticky = W+E)

    Signup = Button(sframe, text="Sign Up", width=15, padx=5, pady=5, bg='green',
fg='white', command = insert)
    Signup.grid(row=6, column=0, columnspan=2, padx=10, pady=(15, 0))
    return


# WELCOME LABEL
WelcomeLabel = Label(root, text = "Welcome to the Game !!", pady = 20)
WelcomeLabel.config(font=("Times", 20, "bold"))
WelcomeLabel.grid(row=0, column=0, columnspan = 2)

# LOGIN BUTTON
LoginBtn = Button(root, text = "LogIn", command = login, pady = 5, width=27,
bg='#19a8f2', fg='white')
LoginBtn.grid(row = 1, column = 0, padx = 1)
# SIGN UP BUTTON
SignupBtn = Button(root, text = "SignUp", command = signup, pady = 5, width=27,
bg='#19a8f2', fg='white')
SignupBtn.grid(row = 1, column = 1)

root.mainloop()
```

Chapter 4 : Output Analysis

SignUp page – Here you can sign up to play the game. After you sign up you will be redirected to login page.

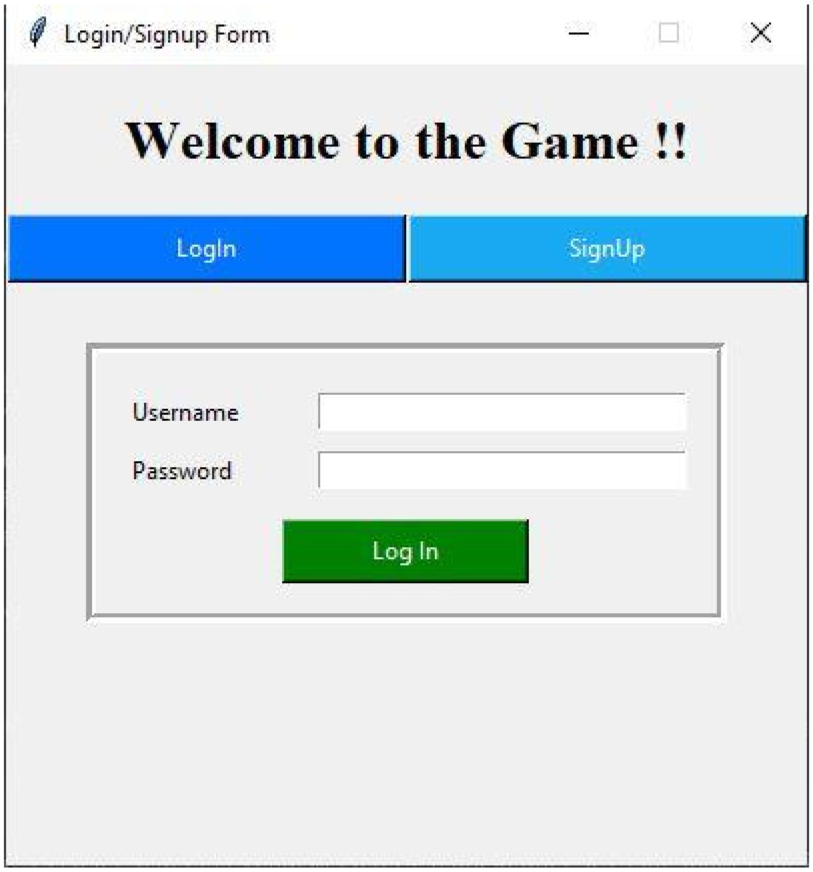


Login/Signup Form

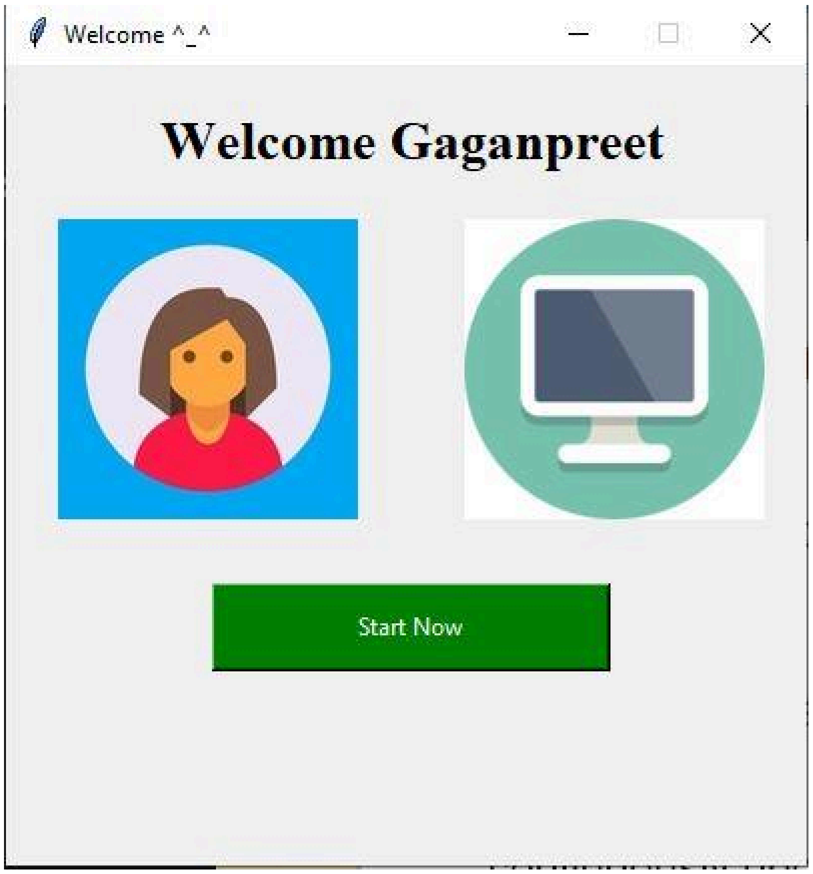
Welcome to the Game !!

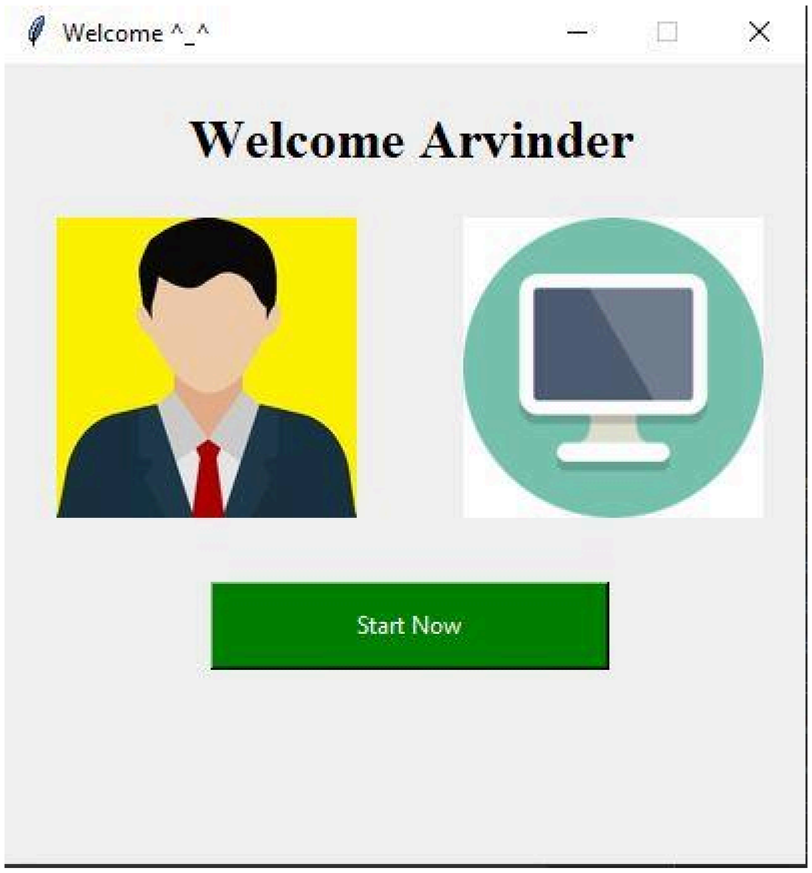
Firstname	
Lastname	
Username	
New Password	
Gender	[✓ male] female

Login page – Here you can login with your username and pass
username or password is incorrect, an error dialog will appear stating username or password is incorrect.

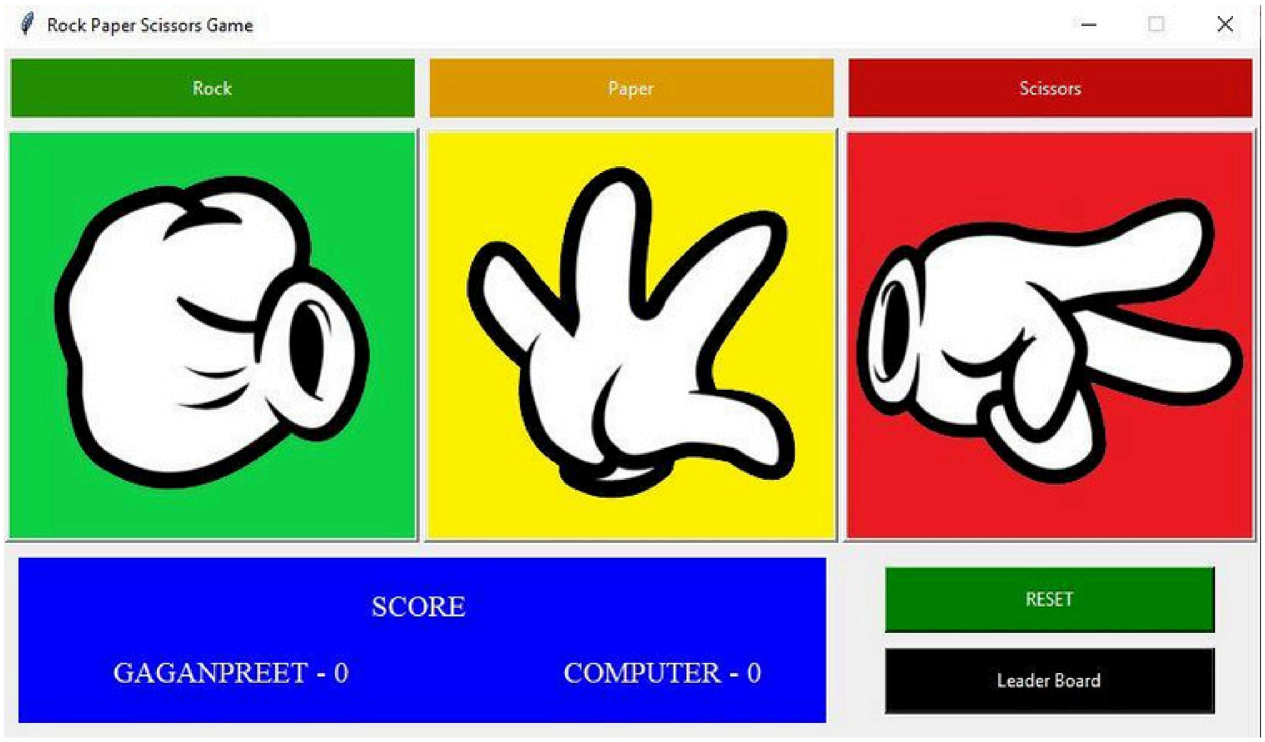


Welcome Page :- The welcome page welcomes you with your registered name. Italsoplacestheimage based on the user’s gender (male or female).

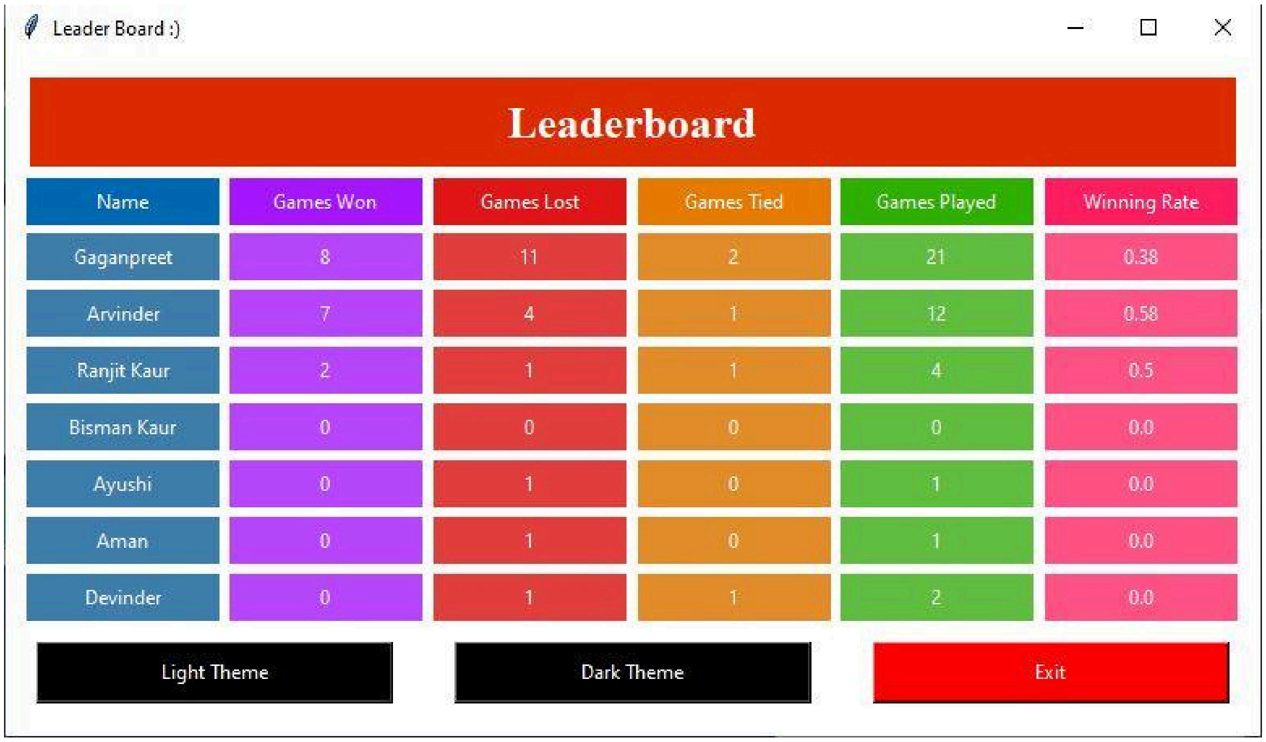

Welcome ^_^
Welcome Gaganpreet
Start Now


Welcome ^_^
Welcome Arvinder
Start Now

Game Page – This is the main game play page. The user clicks on a
button(Rock, Paper, Scissors). After that, the images on the buttons changes reflecting the result.



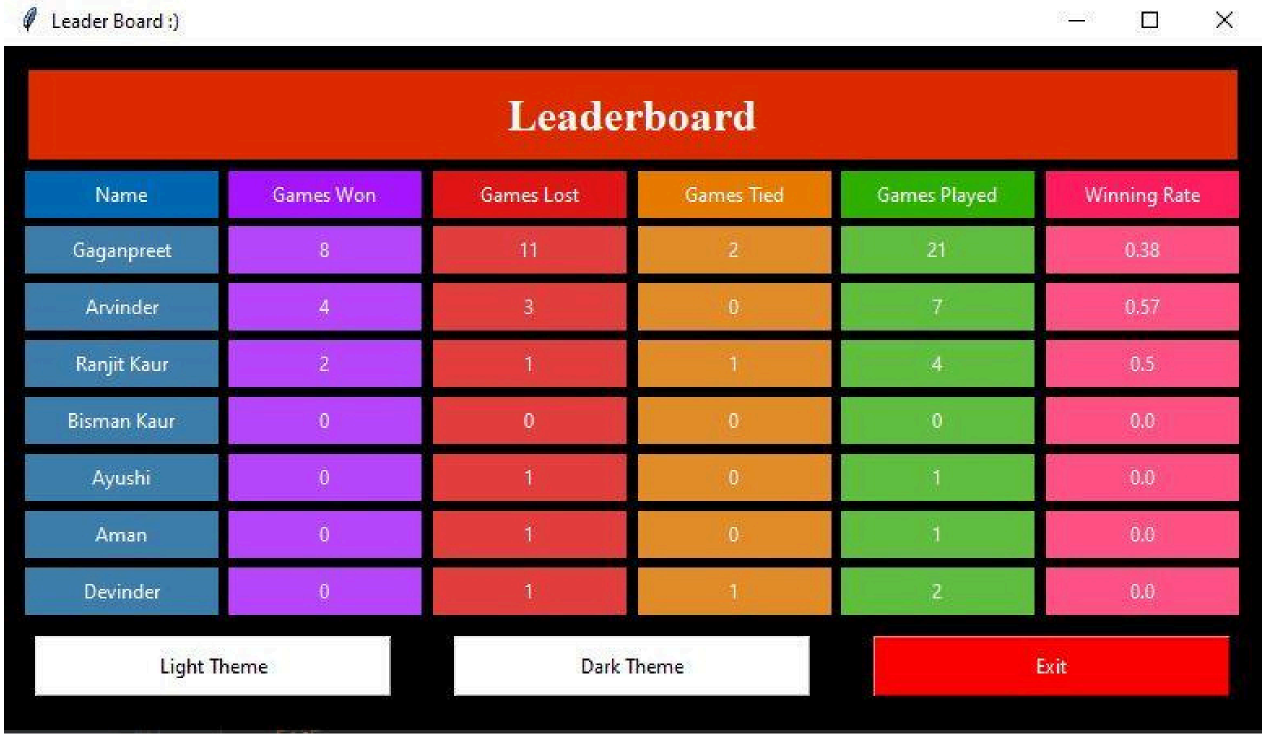
Leaderboard Page – The data present in the database about the user's play is
reflected in this page(Games played, Games won, Games lost, Games Tied, etc.). There are also 2 theme buttons. You can change the theme according to your suitability.



Leader Board :)

Leaderboard

Name	Games Won	Games Lost	Games Tied	Games Played	Winning Rate
Gaganpreet	8	11	2	21	0.38
Arvinder	7	4	1	12	0.58
Ranjit Kaur	2	1	1	4	0.5
Bisman Kaur	0	0	0	0	0.0
Ayushi	0	1	0	1	0.0
Aman	0	1	0	1	0.0
Devinder	0	1	1	2	0.0

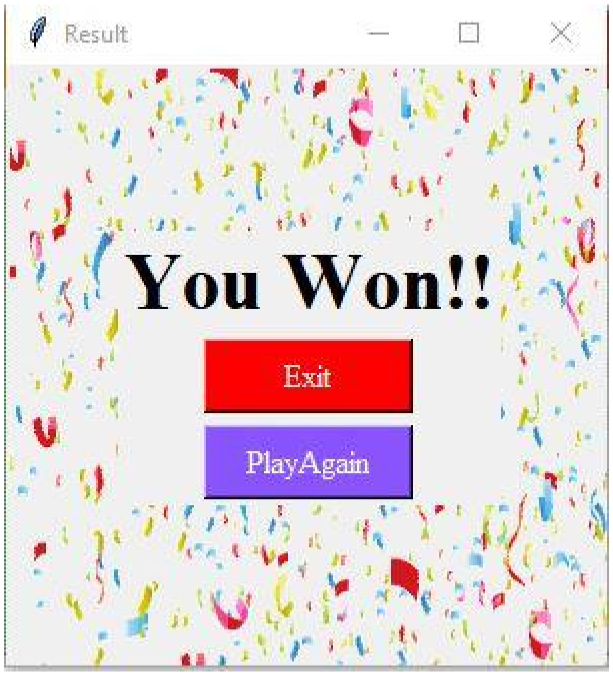


Leader Board :)

Leaderboard

Name	Games Won	Games Lost	Games Tied	Games Played	Winning Rate
Gaganpreet	8	11	2	21	0.38
Arvinder	4	3	0	7	0.57
Ranjit Kaur	2	1	1	4	0.5
Bisman Kaur	0	0	0	0	0.0
Ayushi	0	1	0	1	0.0
Aman	0	1	0	1	0.0
Devinder	0	1	1	2	0.0

Result Page :- This tells you the result of the game, if it was a Won Won situation for you or not. There is a PlayAgain button through which you can play the game without logging in again.



<u>ThankYou!</u>