Sri Adichunchanagiri Shikshana Trust ®

BGS NATIONAL PUBLIC SCHOOL

(Affiliated to Central Board of Secondary Education, New Delhi)
Hulimavu, Bannerghatta Road, Bengaluru - 560 076

Computer Science Investigatory Project

Year: 2022 –2023

Topic: Balloon Shooter game

Name: Viswak Ggautham

Class: 12B

Exam Roll No: 12B28

CERTIFICATE

This is to certify that **Kasak Agarwal** of class XII of BGS National Public School has successfully completed the Investigatory Project in Computer Science for **ALL INDIA SENIOR SECONDARY CERTIFICATE EXAMINATION (AISSCE)** prescribed by CBSE in the year 2022-2023.

Date:

Principal Sign

External Examiner Internal Examiner

ACKNOWLEDGEMENTS

I would like to express my gratitude to our Respected **Academic Director Dr Sri. S A Nair**, **Principal Ms. Sreekala G Kumar**, **and Vice Principal Ms Savitha Suverna** for being a constant pillar of support.

I would then like to thank our **Computer Science teacher, Ms. Babitha E Z** for helping me with this project and guiding me throughout.

I would also like to thank God Almighty and my parents for always being by my side and my fellow mates who were always ready to help.

TABLE OF CONTENTS

INTRODUCTION

The simple plot in this game is that there will be balloons moving all over the screen. The player will have a target like an arrow and with the help of a mouse, the player has to bust those moving balloons. There will be a counter for busted balloons and on busting each moving balloon successfully, the busting score will increase. For the development of the Balloon Shooter game using Python PyGame, we will use various pygame modules to add different functionalities to the game. We have to code for continuous movement of balloons , a shooting functionality, and updating the source every time the balloon is busted. All these functions can be done using various modules like draw, mouse, render, etc.

Game Features:

- It is a single player game.
- It will keep track of total number of balloons busted.

PROJECT SELECTION

We adopted this idea to provide an interactive interface between users and the game. The application program provides a quality experience to the user using data files. Concepts in Python and MySQL were used to develop the application.

The admin creates a database consisting of a table called login. Users can play the game using a simple and efficient interface. All this has been achieved through the efficient extraction from and injection into the database.

WORKING ENVIRONMENT

OPTIMUM REQUIREMENTS

- **Operating System** – Windows 10
- **Processor** – Must be clocked over 1.5 GHz
- **Graphics Driver** – Intel Integrated Graphics
- **RAM** – 4 GB or more.
- **Hard Disk** – 1 TB
- **Python interpreter** – Python IDLE 3.6
- **MySQL**

LIBRARIES & MODULES

Libraries	Purpose
PyGame	We used various PyGame modules to add different functionalities to the game. For the balloon's color and shape, we used draw module for shapes like ellipses, circles, etc.
Random	To add functionality that is based on random events
Math	To use functions like sin
Sys	To interact with the system and to quit game and resume the game

DATA DICTIONARY

USER DEFINED FUNCTIONS

Functions	PURPOSE
__init()__	Function to assign the value for speed
move()	Function to move the balloon
show()	Function to display the balloons on the screen
burst()	Function to check if the balloon has busted or not
reset()	Function to reset balloon's position
pointer()	Function to show location of balloon on the screen
lowerPlatform()	Function to display lower box with score
Close()	Function to quit the game
game()	Main Function to start the game
open1()	Function to open the csv file
view()	Function to view csv file
insert()	Function to receive values in email,password spaces
recupdate()	Function to update username,email and password
admin()	Function to create the admin interface
delrec()	Function to delete record where username is given

SOURCE CODE

```
import pygame
import sys
import random
import os
from math import *
import csv
pygame.init()

width = 700
height = 600
abc = (0, 0, 0)
display = pygame.display.set_mode((width, height))
display.fill(abc)
pygame.display.flip()
pygame.display.set_caption("abc")
clock = pygame.time.Clock()

margin = 100
lowerBound = 100

score = 0

white = (230, 230, 230)
lightBlue = (4, 27, 96)
red = (231, 76, 60)
lightGreen = (25, 111, 61)
darkGray = (40, 55, 71)
darkBlue = (10, 0, 2)
green = (35, 155, 86)
yellow = (244, 208, 63)
blue = (46, 134, 193)
purple = (155, 89, 182)
```

```
orange = (243, 156, 18)

font = pygame.font.SysFont("Arial", 25)



class Balloon:
    def __init__(self, speed):
        self.a = random.randint(30, 40)
        self.b = self.a + random.randint(0, 10)
        self.x = random.randrange(margin, width + self.a + margin)
        self.y = height + lowerBound
        self.angle = 90
        self.speed = -speed
        self.proPool = [-1, -1, -1, 0, 0, 0, 0, 1, 1, 1]
        self.length = random.randint(100, 100)
        self.color = random.choice([red, green, purple, orange, yellow,
blue])

    def move(self):
        direct = random.choice(self.proPool)

        if direct == -1:
            self.angle += -10
        elif direct == 0:
            self.angle += 0
        else:
            self.angle += 10

        self.y += self.speed*sin(radians(self.angle))
        self.x += self.speed*cos(radians(self.angle))

        if (self.x + self.a > width) or (self.x < 0):
            if self.y > height/5:
                self.x -= self.speed*cos(radians(self.angle))
            else:
                self.reset()
        if self.y + self.b < 0 or self.y > height + 30:
```

```
            self.reset()

    def show(self):
        pygame.draw.line(display, darkBlue, (self.x + self.a/2, self.y +
                         self.b), (self.x + self.a/2, self.y + self.b +
self.length))
        pygame.draw.ellipse(display, self.color,
                            (self.x, self.y, self.a, self.b))
        pygame.draw.ellipse(display, self.color, (self.x +
                            self.a/2 - 5, self.y + self.b - 3, 10, 10))

    def burst(self):
        global score
        pos = pygame.mouse.get_pos()

        if isonBalloon(self.x, self.y, self.a, self.b, pos):
            score += 1
            self.reset()

    def reset(self):
        self.a = random.randint(30, 40)
        self.b = self.a + random.randint(0, 10)
        self.x = random.randrange(margin, width - self.a - margin)
        self.y = height - lowerBound
        self.angle = 90
        self.speed -= 0.002
        self.proPool = [-1, -1, -1, 0, 0, 0, 0, 1, 1, 1]
        self.length = random.randint(50, 100)
        self.color = random.choice([red, green, purple, orange, yellow,
blue])


balloons = []
noBalloon = 10
for i in range(noBalloon):
    obj = Balloon(random.choice([1, 1, 2, 2, 2, 2, 3, 3, 3, 4]))
    balloons.append(obj)
```

```
def readstat(name, mode):
    global score
    f = open('highscore.csv', 'r+', newline='')
    r = csv.reader(f)

    print(name)
    if mode == 'r':
        for i in r:
            if i[0] == name:
                return int(i[1])
        return 0
    elif mode == 'w':
        f2 = open('temp.csv', "w+", newline="")
        w = csv.writer(f2)
        for i in r:
            if i[0] != name:
                w.writerow(i)
        w.writerow([name, score])
        f.close()
        f2.close()
        os.remove("highscore.csv")
        os.rename("temp.csv", "highscore.csv")

    f.close()


def isonBalloon(x, y, a, b, pos):
    if (x < pos[0] < x + a) and (y < pos[1] < y + b):
        return True
    else:
        return False


def highscore():
    return score
```

```
def pointer():
    pos = pygame.mouse.get_pos()
    r = 25
    l = 20
    color = green
    for i in range(noBalloon):
        if isonBalloon(balloons[i].x, balloons[i].y, balloons[i].a,
balloons[i].b, pos):
            color = white
    pygame.draw.ellipse(display, color, (pos[0] - r/2, pos[1] - r/2, r,
r), 4)
    pygame.draw.line(
        display, color, (pos[0], pos[1] - l/2), (pos[0], pos[1] - l), 4)
    pygame.draw.line(
        display, color, (pos[0] + l/2, pos[1]), (pos[0] + l, pos[1]), 4)
    pygame.draw.line(
        display, color, (pos[0], pos[1] + l/2), (pos[0], pos[1] + l), 4)
    pygame.draw.line(
        display, color, (pos[0] - l/2, pos[1]), (pos[0] - l, pos[1]), 4)


def lowerPlatform():
    pygame.draw.rect(display, darkGray, (0, height -
                      lowerBound, width, lowerBound))


def showScore():
    scoreText = font.render("Balloons Bursted : " + str(score), True,
white)
    display.blit(scoreText, (150, height - lowerBound + 50))


def close(name):
    global old_score
    old_score = readstat(name, 'r')
    if score > old_score:
```

```
        readstat(name, 'w')
    pygame.quit()
    sys.exit()


def game(name):
    global score
    loop = True

    while loop:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                close(name)
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_q:
                    close()
                if event.key == pygame.K_r:
                    score = 0
                    game()

            if event.type == pygame.MOUSEBUTTONDOWN:
                for i in range(noBalloon):
                    balloons[i].burst()

        display.fill(abc)

        for i in range(noBalloon):
            balloons[i].show()

        pointer()

        for i in range(noBalloon):
            balloons[i].move()
        lowerPlatform()
        showScore()
        pygame.display.update()
        clock.tick(60)
```

```
# tkinter login
import tkinter as tkin
import customtkinter as tk
from tkinter.constants import *
from tkinter import ttk
from tkinter import *
from PIL import Image, ImageTk
import os
import csv
from balloon import *
import mysql.connector as sql
import pygame as py


class end():
    user = ''
    end = True


class window():
    def __init__(self):
        with open("highscore.csv", "a+") as w:
            print("Created Highscore")
        self.main = tkin.Tk()
        self.l = False
        # self.image = Image.open('tkproj.png')
        # self.copy_of_image = self.image.copy()
        # self.photo = ImageTk.PhotoImage(self.image)
        # self.label = tk.CTkLabel(self.main, image = self.photo)
        # self.label.bind('<Configure>', self.resize_image)
        # self.label.pack(fill=BOTH, expand = YES)
        self.main.title('LOGIN')
        self.main.geometry('900x650')
        tk.CTkLabel(self.main, text='Username',
                    text_color='black').place(x=180, y=40)
        tk.CTkLabel(self.main, text='Password',
                    text_color='black').place(x=180, y=80)
```

```
        tk.CTkLabel(self.main, text='Dont have an account? Sign in Now',
                    text_color='black', width=300).place(x=300, y=470)
        self.user = tk.CTkEntry(self.main)
        self.user.place(x=280, y=40)
        self.pswd = tk.CTkEntry(self.main, show='*')
        self.pswd.place(x=280, y=80)
        tk.CTkButton(master=self.main, text='Login', width=100,
text_color='black',
                     fg_color='grey', bg='white',
command=self.login).place(x=280, y=110)
        tk.CTkButton(master=self.main, text='Sign Up', width=100,
text_color='black',
                     fg_color='grey', bg='white',
command=self.signup).place(x=375, y=500)
        tk.CTkButton(master=self.main, text='Update', width=100,
text_color='black',
                     fg_color='grey', bg='white',
command=self.update).place(x=280, y=140)
        tk.CTkButton(master=self.main, text='Admin', width=100,
text_color='black',
                     fg_color='grey', bg='white',
command=self.chkadmin).place(x=375, y=200)
        tk.CTkButton(master=self.main, text='Instructions', width=100,
text_color='black',
                     fg_color='grey', bg='white',
command=self.sample).place(x=500, y=400)
        self.tgbutton = tk.CTkButton(master=self.main, text='Show
Password', text_color='black',
                                     width=200, fg_color='grey',
bg='white', command=self.togglepas)
        tk.CTkButton(master=self.main, text='VIEW', width=100,
text_color='black',
                     fg_color='grey', bg='white',
command=self.view).place(x=375, y=400)
        self.tgbutton.place(x=500, y=80)
        tk.CTkButton(master=self.main, text='highscore', width=100,
text_color='black',
                     fg_color='grey', bg='white',
command=self.open1).place(x=375, y=400)
        self.main.mainloop()


    def open1(self):
```

```
        os.system('highscore.csv')

    def view(self):
        f = os.system(r'view.csv')
        f = open('view.csv', 'a+', newline='')
        r = csv.reader(f)
        w = csv.writer(f)
        db = sql.connect(host='localhost', user='root',
                         database='project', password='password')
        cursor = db.cursor()
        cursor.execute('select username from login;')
        user = cursor.fetchall()
        w.writerow(user)
        db.commit()
        db.close()

    def sample(self):
        f = os.system(r'instructions.txt')

    def togglepas(self):
        if self.l:
            self.l = not self.l
            self.pswd.configure(show='')
            self.tgbutton.configure(text='Hide Password')
        else:
            self.l = not self.l
            self.pswd.configure(show='*')
            self.tgbutton.configure(text='Show Password')

    def insert(self, user, pswd, email):
        db = sql.connect(host='localhost', user='root',
                         database='project', password='password')
        cursor = db.cursor()
        cursor.execute(
            'insert into login values("%s","%s","%s");' % (email, user,
pswd))
        readstat(name=user, mode='w')
```

```
        db.commit()
        db.close()

    def recupdate(self, user, pswd, email):
        db = sql.connect(host='localhost', user='root',
                         database='project', password='password')
        cursor = db.cursor()
        cursor.execute(
            'update login set username="%s" where email_id="%s";' %
(user, email))
        cursor.execute(
            'update high set username="%s" where email_id="%s";' %
(user, email))
        cursor.execute(
            'update login set password="%s" where email_id="%s";' %
(pswd, email))
        db.commit()
        db.close()

    def admin(self):
        global trv
        self.adm = Toplevel()
        self.adm.geometry('800x500')
        self.adm.title('ADMIN')
        Label(self.adm, text=' ADMIN - changes can be
done').place(x=100, y=20)

        #
upd=tk.CTkButton(self.adm,text='Update',width='10',fg='blue',text_color=
'black',bg='white',command=self.uprec).place(x=200,y=150)
        tk.CTkLabel(self.adm, text=' User Deletion ',
                    text_color='black').place(x=70, y=250)
        tk.CTkLabel(self.adm, text=' User Search ',
                    text_color='black').place(x=70, y=350)
        dele = tk.CTkButton(self.adm, text='Delete', width=10,
fg_color='blue',
                            bg_color='white',
command=self.delrec).place(x=200, y=250)
        ser = tk.CTkButton(self.adm, text=' Search', width=10,
```

```
                            fg_color='blue',
bg_color='white').place(x=200, y=350)
        add = tk.CTkButton(self.adm, text='Add', width=10,
fg_color='blue',
                            bg_color='white', text_color='black',
command=self.addrec).place(x=200, y=85)
        tk.CTkLabel(self.adm, text=' Username').place(x=300, y=85)
        tk.CTkLabel(self.adm, text=' Email').place(x=400, y=85)
        tk.CTkLabel(self.adm, text=' Password').place(x=450, y=85)
        self.nuser = tkin.Entry(self.adm)
        self.nuser.place(x=300, y=100)
        self.nemail = tkin.Entry(self.adm)
        self.nemail.place(x=400, y=100)
        self.npswd = tkin.Entry(self.adm)
        self.npswd.place(x=450, y=100)

    # tree function

        db = sql.connect(host='localhost', user='root',
                         database='project', password='password')
        cursor = db.cursor()
        cursor.execute('SELECT*FROM LOGIN;')
        al = cursor.fetchall()
        trv = ttk.Treeview(self.adm, selectmode='browse')

        trv.place(x=300, y=200)
        verbar = ttk.Scrollbar(self.adm, orient="vertical",
command=trv.yview)
        # verbar.pack(side ='right', fill ='x')
        verbar.place(x=570, y=200)

        trv.configure(xscrollcommand=verbar.set)

        trv["columns"] = ("1", "2", "3")

        trv['show'] = 'headings'

        trv.column("1", width=80, anchor='c')
```

```
        trv.column("2", width=105, anchor='c')
        trv.column("3", width=80, anchor='c')

        trv.heading("1", text="username")
        trv.heading("2", text="Email")
        trv.heading("3", text="Password")
        for i in al:
            trv.insert("", 'end', iid=i[0],
                       text=i[0], values=(i[1], i[0], i[2]))

    def delrec(self):
        sel = trv.selection()
        for i in sel:
            trv.delete(sel)
        db = sql.connect(host='localhost', user='root',
                         database='project', password='password')
        cursor = db.cursor()
        cursor.execute(f"delete from login where username='{sel[0]}'")

        db.commit()
        db.close()

    def chkadmin(self):
        if self.user.get() == 'admin' and self.pswd.get() == '12345':
            self.admin()
        else:
            Label(self.main, text=' not admin ').place(x=200, y=250)

    def addrec(self):
       trv.insert("", 'end', iid=0, text='', values=(
           self.nuser.get(), self.nemail.get(), self.npswd.get()))

       db = sql.connect(host='localhost', user='root',
                        database='project', password='password')
       cursor = db.cursor()
       cursor.execute('insert into login values("%s","%s","%s");' %
```

```
                        (self.nuser.get(), self.nemail.get(),
self.npswd.get()))
       db.commit()
       db.close()


    def uprec(self):
        '''sel=trv.selection()

db=sql.connect(host='localhost',user='root',database='project',password=
'password')
        cursor=db.cursor()
        cursor.execute('update login set
("%s","%s","%s");'%(self.nuser.get(),self.nemail.get(),self.npswd.get())
)
        db.commit()
        db.close()'''


        self.uprec = Toplevel()
        self.uprec.geometry('800x500')
        self.uprec.title('ADMIN')
        Label(self.uprec, text='UPDATE RECORD').place(x=100, y=20)
        Label(self.uprec, text='user').place(x=100, y=200)
        Label(self.uprec, text='email').place(x=100, y=300)
        Label(self.uprec, text='pswd').place(x=100, y=400)
        self.user1 = tkin.Entry(self.uprec)
        self.user1.place(x=100, y=300)
        self.email1 = tkin.Entry(self.uprec)
        self.pswd1 = tkin.Entry(self.uprec)
        self.email1.place(x=100, y=400)
        self.pswd1.place(x=100, y=500)


    def login(self):
        db = sql.connect(host='localhost', user='root',
                         database='project', password='password')
        cursor = db.cursor()
        cursor.execute('SELECT*FROM LOGIN;')
        u = self.user.get()
        p = self.pswd.get()
```

```
        w = cursor.fetchall()

        def check():
            for i in w:
                if i[1] == u and i[2] == p:
                    end.end = True
                    end.user = i[1]
                    self.main.destroy()
                    game(u)

        tk.CTkButton(master=self.main, text='Login', width=100,
                     fg_color='grey', bg='white',
command=check).place(x=280, y=110)
        db.close()

    def update(self):
        db = sql.connect(host='localhost', user='root',
                         database='project', password='password')
        cursor = db.cursor()
        user1 = self.user.get()
        pswd1 = self.pswd.get()
        cursor.execute('SELECT*FROM LOGIN;')
        w = cursor.fetchall()
        db.close()
        new1 = Toplevel()
        new1.geometry('1000x800')
        new1.title('Update')
        tk.CTkLabel(new1, text='UPDATE YOUR CREDENTIALS').place(x=100,
y=10)
        tk.CTkLabel(new1, text='Username').place(x=180, y=40)
        tk.CTkLabel(new1, text='Email').place(x=180, y=70)
        tk.CTkLabel(new1, text='Password').place(x=180, y=100)
        tk.CTkLabel(new1, text='confirm Password').place(x=180, y=130)
        g1 = tk.CTkEntry(new1)
        g2 = tk.CTkEntry(new1)
        g3 = tk.CTkEntry(new1)
        g4 = tk.CTkEntry(new1)
```

```
        g1.place(x=500, y=40)
        g2.place(x=500, y=70)
        g3.place(x=500, y=100)

        g4.place(x=500, y=130)

        print(g3)
        # def check1():

        def rec():
            if g3.get() == g4.get():
                print('y')
                self.recupdate(g1.get(), g3.get(), g2.get())
                new1.destroy()
            elif g3.get() != g4.get():
                tk.CTkLabel(self.new, text='Your password doesnt
match').place(
                    x=200, y=250)
            elif g3.get() == '' or g4.get == '':
                tk.CTkLabel(self.new, text='No password').place(x=200,
y=250)
        tk.CTkButton(new1, text='Update', width=10, fg_color='blue',
                     bg='white', command=rec).place(x=250, y=160)

    def signup(self):
        new = Toplevel()
        new.geometry('1000x800')
        new.title('Sign Up')
        tk.CTkLabel(new, text='SIGN IN NOW WITH YOUR CREDENTIALS',
                    text_color='black').place(x=100, y=10)
        tk.CTkLabel(new, text='Username',
                    text_color='black').place(x=180, y=40)
        tk.CTkLabel(new, text='Email', text_color='black').place(x=180,
y=70)
        tk.CTkLabel(new, text='Password',
                    text_color='black').place(x=180, y=100)
        tk.CTkLabel(new, text='Confirm Password',
                    text_color='black').place(x=150, y=130)
```

```
        e1 = tk.CTkEntry(new)
        e2 = tk.CTkEntry(new)
        e3 = tk.CTkEntry(new)
        e4 = tk.CTkEntry(new)
        e1.place(x=300, y=40)
        e2.place(x=300, y=70)
        e3.place(x=300, y=100)
        e4.place(x=300, y=130)

        def validate():
            global l
            pswd = e3.get()
            conf = e4.get()
            user = e1.get()
            email = e2.get()
            if pswd != conf:
                tk.CTkLabel(new, text='Your password doesnt
match').place(
                    x=200, y=250)
            elif pswd == '' or conf == '':
                tk.CTkLabel(new, text='SOMETHING IS MISSING').place(
                    x=200, y=300)
            elif len(e1.get()) > 15:
                tk.CTkLabel(new, text='Exceeded Limit').place(x=200,
y=400)
            else:
                self.insert(user, pswd, email)
                new.destroy()

        tk.CTkButton(master=new, text='Sign Up', width=100,
fg_color='grey',
                     bg='white', command=validate).place(x=375, y=200)

    def resize_image(self, event):
        new_width = event.width
        new_height = event.height
        image = self.copy_of_image.resize((new_width, new_height))
        photo = ImageTk.PhotoImage(image)
```

```
        self.label.config(image=photo)
        self.label.image = photo


a = window()
```

LOG OF PROJECT

LOG-1:15/6/2022

- Ideation
- Discussion of topic – Balloon Shooter

LOG-2:27/6/2022

- Searched for various modules available to make the game
- Learning to work with Tkinter

LOG-3:10/7/2022

- Decision to use pygame module to make the game

LOG-4:23/7/2022

- Created SQL table login
- Worked on Home screen
- Worked on sign up and login options using Python-SQL Connectivity

LOG-5:24/7/2022

- Fixed bugs and errors

LOG-6:26/7/2022

- Imported modules and worked on the main game

LOG-7:30/7/2022

- Worked on admin options

LOG-8:3/8/2022

- Worked on the main game program, added background

LOG-9:4/8/2022

- Worked on main game program, added all features and positioned the lower platform

LOG-10:5/8/2022

- Created login page

LOG-11:6/8/2022

- Worked on login page

LOG-12:7/8/2022

- Worked on admin options: Integrating admin options(Update and Delete)

LOG-13:14/8/2022

- Worked on user options: Integrating user options

LOG-14:1/9/2022

- Created text file

LOG-15:8/9/2022

- Created csv file

LOG-16:12/9/2022

- Worked on the main game program and completed it

LOG-17:6/11/2022

- Completed login page

LOG-18:9/11/2022

- Worked on integrating all the program files together

LOG-19:12/11/2022

- Worked on various bug fixes and improvements

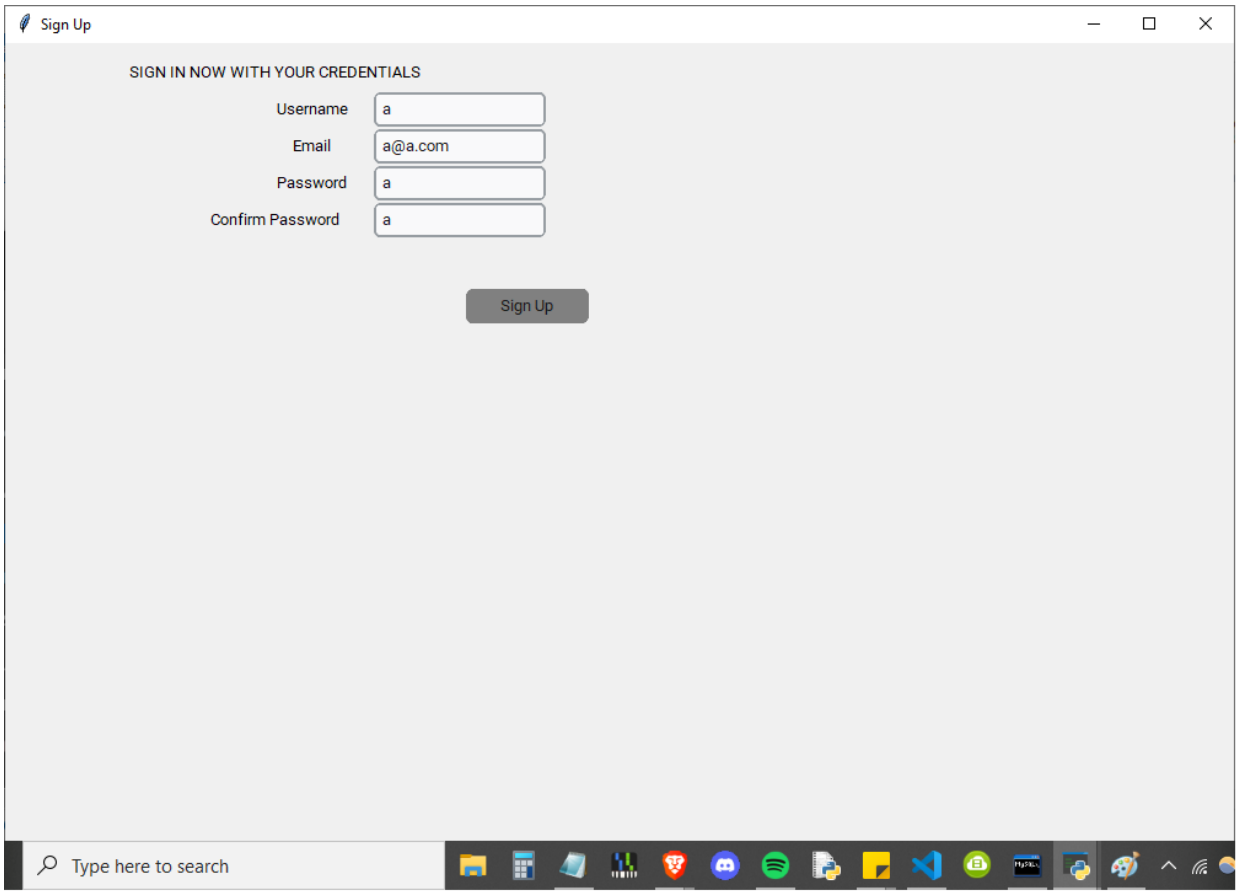
LOG-20:28/11/2022

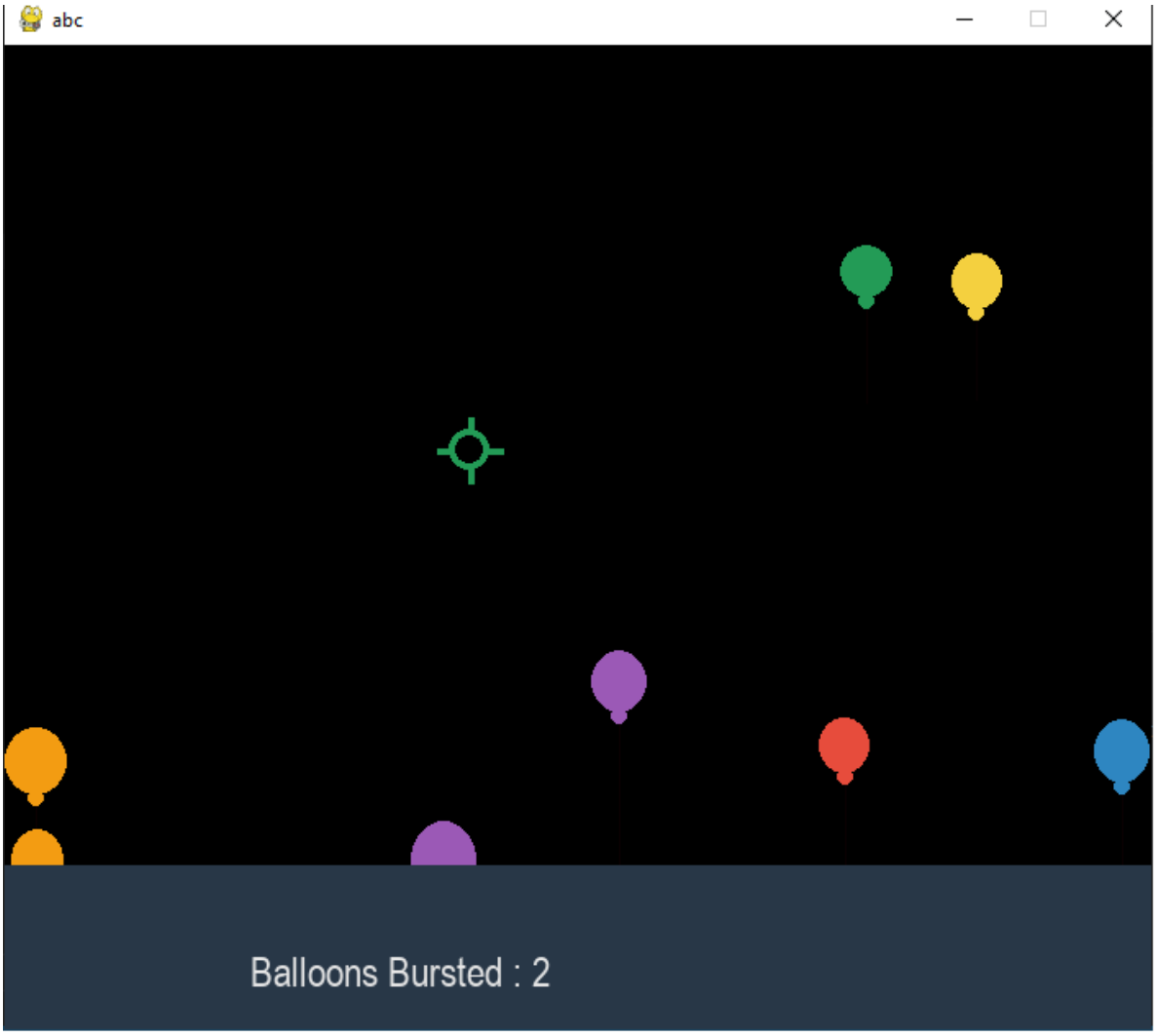
- Improvements in the design of GUI interface and layout
- Worked with color combinations and backgrounds

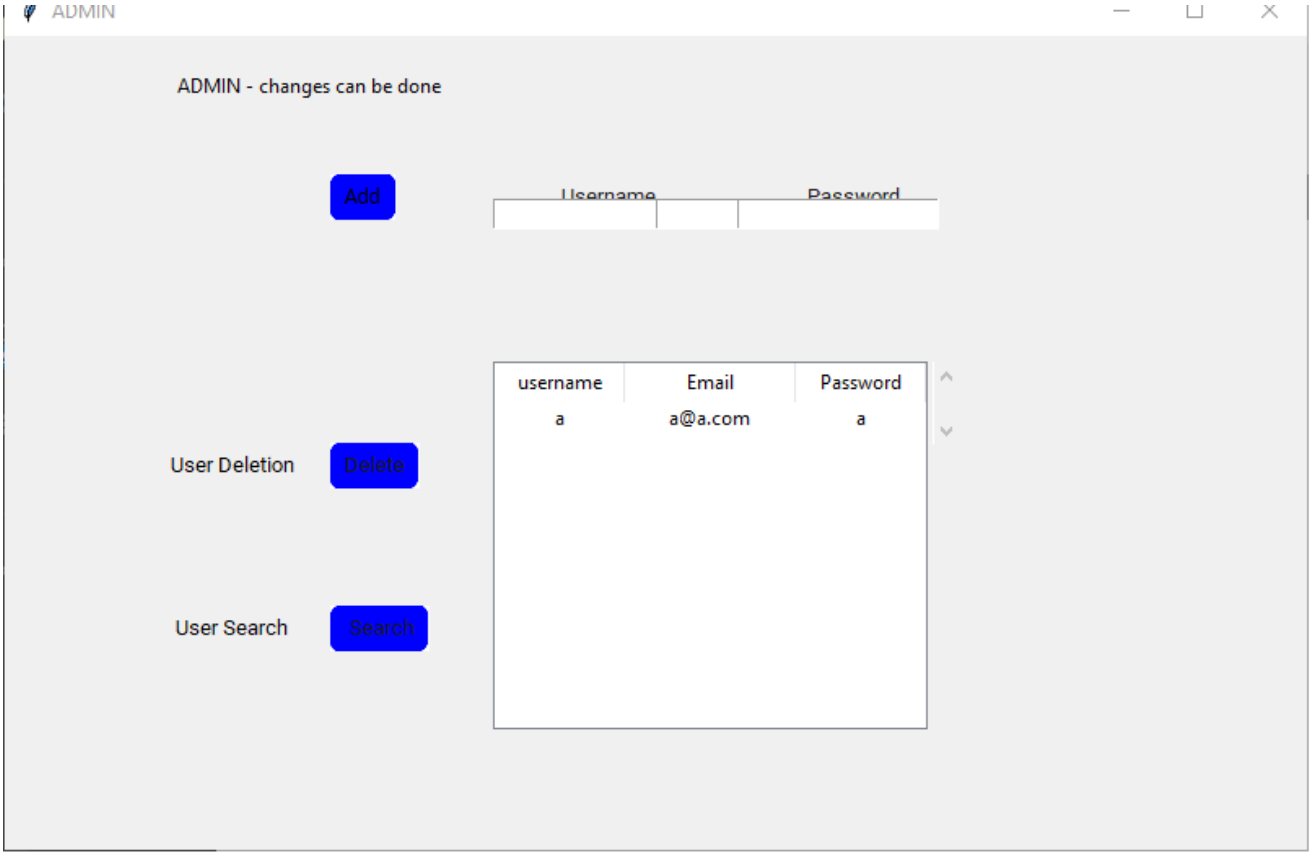
LOG-21:15/12/2022

- Completion of Project and Submission

SAMPLE OUTPUT




abc
Balloons Bursted : 2


ADMIN
ADMIN - changes can be done
Add
Username
Password
User Deletion
Delete
User Search
Search
username
Email
Password
a
a@a.com
a

BIBLIOGRAPHY

- https://www.tutorialspoint.com/python/
- https://www.geeksforgeeks.org
- http://stackoverflow.com/
- https://www.javatpoint.com
- https://www.google.co.in/
- https://pythonexamples.org
- https://effbot.org/