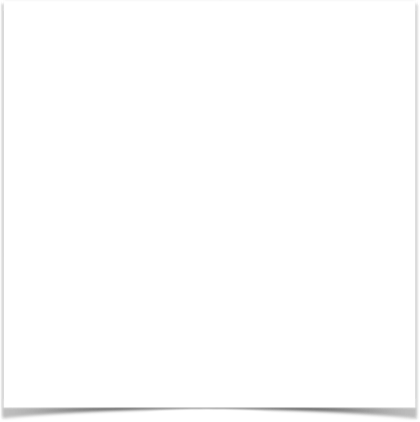
**TITLE:**

**BOTTLE BASH**



Team Members:

PES2UG21CS096 – Arya Umesh

PES2UG21CS391 – Pratham R Prabhal

PES2UG21CS562 – Suyog K

**OVERVIEW**

* Design
* Modules used
* Implementation
* Team roles and description
* Software and Hardware Requirements
* Code

**Table of Contents**

|  |  |
| --- | --- |
| **Page no.** | **Contents** |
| 3 | Design-About Game |
| 3 | Modules Used |
| 3 | Importance of the Game |
| 3 | Implementation |
| 4 | Code |
| 22 | Team Roles  Pratham-GUI  Arya-Game Mechanics  Suyog-Data Handling |
| 23 | Software and Hardware Requirements |
| 23 | Result and Analysis |
| 24 | Conclusions and Future Enhancements |
| 24 | References |

**Design-About Game**

The agenda of this game is that the player has to click on bottles to break them, there are also hidden bombs which are disguised as bottles, if the player clicks on them he loses a life or entire game (depends on difficulty level chosen).

The game has 3 difficulty levels, namely easy, normal, hard.

The user can enter desired username, difficulty level and check high score.

The entire program is designed using python language and its modules.

**Modules Used**

* Pygame: for GUI interface and Game mechanics and user interaction
* Sqlite3: for connecting python with Sqlite3 DBMS
* Time: used to manipulate time allowed in game
* Random: to manipulate bottle and bomb positions and other features
* Sys: to take system font

**Importance of the Game**

* Ever since the COVID-19 pandemic and the lockdowns that followed began, the mental heath of many individuals has greatly deteriorated.
* Due to being locked up in their homes, social interaction has also taken a hit.
* To help make up for it, this game can allow groups of people to compete with one another and in doing so, encourage social interaction.
* This friendly competition can help reduce mental stress as it creates a diversion from any college or office related stress and hence, can improve the mental health of the individual.

**Implementation**

* Programming language used: Python 3
* The GUI uses Pygame module and game uses Pygame, Random and Time modules, the entire program data is handled using Sqlite3
* The program has 3 distinct layers: GUI + Game + DBMS
* GUI allows player to choose his name, difficulty level, get instructions and so on
* Game layer handles game mechanics
* DBMS layer connects python and Sqlite3

**Code**

#BOTTLE BASH

import pygame

import sys

import random

import time

import sqlite3

pygame.init()

#Sqlite connections

cn=sqlite3.connect('game\_stats.db')

c=cn.cursor()

c.execute("SELECT tbl\_name FROM sqlite\_schema WHERE tbl\_name='gstats'")

chk=c.fetchall()

if chk==[]: #checking if gstats table exists and creating one if not, to prevent data redundancy

c.execute('''CREATE TABLE gstats (Player\_Name text,

Score integer,Difficulty text)''')

cn.commit()

cn.close()

def store\_data (pn,hs,df): #pn=player name; hs=highscore; function to input user data into DBMS

cn\_1=sqlite3.connect('game\_stats.db')

c\_1=cn\_1.cursor()

try:

C=chkplayer(pn,hs,df)

if C==True:

exe=("INSERT INTO gstats VALUES(?,?,?)")

c\_1.execute(exe,(pn,hs,df))

cn\_1.commit()

SS='data entered sucessfully'

return SS

else:

OP='welcome back {0}, your new HighScore is {1}! Difficulty level:{2}'.format(pn,C,df)

query="UPDATE gstats SET score=? WHERE Player\_Name=? AND Difficulty=?"

value=C,pn,df

c\_1.execute(query,value)

cn\_1.commit()

return OP

except sqlite3.Error as error:

print('Operation Failed@store\_data:', error)

cn\_1.close()

def high\_score(df): #function to fetch overall HighScore and PlayerName

cn\_2=sqlite3.connect('game\_stats.db')

c\_2=cn\_2.cursor()

try:

Qr='SELECT \* FROM gstats WHERE Difficulty=? ORDER BY Score DESC'

c\_2.execute(Qr,(df,))

HS=c\_2.fetchone()

cn\_2.commit()

return HS

except sqlite3.Error as error:

print('Operation Failed@high\_score:', error)

cn\_2.close()

def clear\_table\_gstats(): #function to clear game stats table

cn\_3=sqlite3.connect('game\_stats.db')

c\_3=cn\_3.cursor()

try:

c\_3.execute('DELETE FROM gstats')

cn\_3.commit()

print('table containing game statistics cleared')

return

except sqlite3.Error as error:

print('Operation Failed@clear\_table\_gstats:', error)

cn\_3.close()

def chkplayer(p\_name,p\_score,p\_diff):

cn\_4=sqlite3.connect('game\_stats.db')

c\_4=cn\_4.cursor()

try:

exe1=('SELECT \* FROM gstats WHERE Player\_Name=? AND Difficulty=?')

c\_4.execute(exe1,(p\_name,p\_diff))

p\_detail=c\_4.fetchone()

cn\_4.commit()

if p\_detail==None:

return True

else:

if p\_detail[1]>p\_score:

return p\_detail[1]

else:

return p\_score

except sqlite3.Error as error:

print('Operation Failed@chkplayer:', error)

##a='NONE'

##b=0

##a1='NONE'

##b1=0

##a2='NONE'

##b2=0

##store\_data(a, b,'easy')

##store\_data(a1, b1,'normal')

##store\_data(a2, b2,'hard')

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

SCREEN = pygame.display.set\_mode((1300, 690))

pygame.display.set\_caption("Menu")

BG = pygame.image.load("project1Background.jpg")

BG = pygame.transform.scale(BG, (1300, 690))

def GAME(dif,Name):

pygame.init()

true = True

name = Name

life = 4 - dif

ti = 120 - 30\*dif

sco = 0

BB = [(100,0,0),(150,0,0),(125,75,0),(0,0,0),(125,125,0)]

def timer(s1,s2):

text3 = font.render("Time: "+str(int(ti-(s2-s1))),True,(0,0,0),(255,255,255))

trect3 = text3.get\_rect()

trect3.center = (1100,100)

screen.blit(text3,trect3)

return int(ti-(s2-s1))

def countdown(): #To give some time for player to get ready

t = 3

while t >= 0:

time.sleep(1)

t -= 1

def score():

text4 = font.render("Your Score: "+str(sco),True,(0,0,0),(255,255,255))

trect4 = text4.get\_rect()

trect4.center = (250,200)

screen.blit(text4,trect4)

def finalscore(string):

t = True

while t == True:

screen.fill((255,255,255))

text11 = font.render("Name: "+str(name),True,(0,0,0),(255,255,255))

trect11 = text11.get\_rect()

trect11.center = (650,50)

text21 = font.render("Your Score: "+str(sco),True,(0,0,0),(255,255,255))

trect21 = text21.get\_rect()

trect21.center = (650,200)

text22 = font.render(string,True,(0,0,0),(255,255,255))

trect22 = text22.get\_rect()

trect22.center = (650,300)

text31 = font.render("Click anywhere to continue",True,(0,0,0),(255,255,255))

trect31 = text31.get\_rect()

trect31.center = (650,500)

screen.blit(text11,trect11)

screen.blit(text21,trect21)

screen.blit(text22,trect22)

screen.blit(text31,trect31)

pygame.display.update()

for event in pygame.event.get():

if event.type == pygame.MOUSEBUTTONUP:

if pygame.mouse.get\_pressed():

pos = pygame.mouse.get\_pos()

if 0 <= pos[0] <= 1300 and 0 <= pos[1] <= 690:

t = False

screen = pygame.display.set\_mode((1300,690))

capt = pygame.display.set\_caption("Bottle Break")

font = pygame.font.SysFont("Arial Rounded MT Bold",40)

text1 = font.render("Name: "+str(name),True,(0,0,0),(255,255,255))

if dif==1:

HSCORE=high\_score('easy')

elif dif==2:

HSCORE=high\_score('normal')

elif dif==3:

HSCORE=high\_score('hard')

HIGHSCORE=("Highest Score:{} , Difficulty:{}").format(HSCORE[1],HSCORE[2])

text2 = font.render(HIGHSCORE,True,(0,0,0),(255,255,255))

trect1 = text1.get\_rect()

trect1.center = (650,50)

trect2 = text2.get\_rect()

trect2.center = (300,100)

countdown()

seconds1 = time.time()

try:

while true == True:

screen.fill((255,255,255))

screen.blit(text1,trect1)

screen.blit(text2,trect2)

score()

bomb = []

if dif == 1:

bomb = [random.choice([0,1,2,3,4])]

else:

blist = [0,1,2,3,4]

b1 = random.choice(blist)

blist.remove(b1)

bomb = [b1,random.choice(blist)]

colour = []

bot = []

for b in range(0,5):

bot.append(pygame.Rect(600+b\*150,random.randint(300,500),50,150))

if b in bomb:

colour.append(random.choice(BB))

else:

colour.append((random.randint(75,205),random.randint(75,205),random.randint(125,205)))

text5 = font.render("Bomb Colours: ",True,(0,0,0),(255,255,255))

trect5 = text5.get\_rect()

trect5.center = (850,200)

screen.blit(text5,trect5)

for bombs in range(0,len(BB)):

pygame.draw.rect(screen,BB[bombs],[1000+50\*bombs,190,30,30])

for bott in range(0,len(bot)):

pygame.draw.rect(screen,colour[bott],bot[bott])

pygame.draw.polygon(screen,colour[bott],[(bot[bott].x,bot[bott].y),(bot[bott].x+15,bot[bott].y-20),(bot[bott].x+35,bot[bott].y-20),(bot[bott].x+49.99,bot[bott].y)])

pygame.draw.rect(screen,colour[bott],[bot[bott].x+18.5,bot[bott].y-35,15,15])

pygame.draw.rect(screen,(0,0,0),[bot[bott].x-50,bot[bott].y+150,150,690-bot[bott].y-150])

pygame.display.flip()

dd = True

broke = 5

while dd == True:

seconds2 = time.time()

zero = timer(seconds1,seconds2)

for event in pygame.event.get():

if event.type == pygame.QUIT:

finalscore("You ended the game!")

if dif==3:

store\_data (Name,sco,'hard')

elif dif==2:

store\_data (Name,sco,'normal')

elif dif==1:

store\_data (Name,sco,'easy')

true = False

screen.exit()

for li in range(0,life):

pygame.draw.ellipse(screen,(200,0,0),[100\*li+40,620,50,50])

pygame.display.flip()

if event.type == pygame.MOUSEBUTTONUP:

if pygame.mouse.get\_pressed():

pos = pygame.mouse.get\_pos()

for loc in bot:

if loc.y <= pos[1] <= loc.y+150:

if (loc.x <= pos[0] <= (loc.x+50)):

if bot.index(loc) in bomb:

pygame.draw.rect(screen,(255,255,255),[loc.x,loc.y-35,50,185])

pygame.draw.ellipse(screen,(255,255,255),[240-(3-life)\*100,620,50,50])

bot[bot.index(loc)].x += 1000

bot[bot.index(loc)].y += 1000

life -= 1

if life == 0:

finalscore("All lives spent!")

if dif==3:

store\_data (Name,sco,'hard')

elif dif==2:

store\_data (Name,sco,'normal')

elif dif==1:

store\_data (Name,sco,'easy')

true = False

screen.exit()

else:

pass

else:

sco += 1

score()

pygame.draw.rect(screen,(255,255,255),[loc.x,loc.y-35,50,100])

pygame.draw.polygon(screen,(255,255,255),[(loc.x,loc.y+65),(loc.x+11,loc.y+80),(loc.x+25,loc.y+65)])

pygame.draw.polygon(screen,(255,255,255),[(loc.x+25,loc.y+65),(loc.x+39,loc.y+80),(loc.x+50,loc.y+65)])

pygame.display.flip()

bot[bot.index(loc)].x += 1000

bot[bot.index(loc)].y += 1000

broke -= 1

if len(bomb) == broke:

break

elif zero <= 0:

finalscore("Time up!")

if dif==3:

store\_data (Name,sco,'hard')

elif dif==2:

store\_data (Name,sco,'normal')

elif dif==1:

store\_data (Name,sco,'easy')

true = False

screen.exit()

else:

pass

except:

None

class Button():

def \_\_init\_\_(self, image, pos, text\_input, font, base\_color, hovering\_color):

self.image = image

self.x\_pos = pos[0]

self.y\_pos = pos[1]

self.font = font

self.base\_color, self.hovering\_color = base\_color, hovering\_color

self.text\_input = text\_input

self.text = self.font.render(self.text\_input, True, self.base\_color)

if self.image is None:

self.image = self.text

self.rect = self.image.get\_rect(center=(self.x\_pos, self.y\_pos))

self.text\_rect = self.text.get\_rect(center=(self.x\_pos, self.y\_pos))

def update(self, screen):

if self.image is not None:

screen.blit(self.image, self.rect)

screen.blit(self.text, self.text\_rect)

def checkForInput(self, position):

if position[0] in range(self.rect.left, self.rect.right) and position[1] in range(self.rect.top, self.rect.bottom):

return True

return False

def changeColor(self, position):

if position[0] in range(self.rect.left, self.rect.right) and position[1] in range(self.rect.top, self.rect.bottom):

self.text = self.font.render(self.text\_input, True, self.hovering\_color)

else:

self.text = self.font.render(self.text\_input, True, self.base\_color)

def get\_font(size): # Returns Press-Start-2P in the desired size

return pygame.font.SysFont("Alferian", size)

def SCORES(dif):

while True:

SCORES\_MOUSE\_POS = pygame.mouse.get\_pos()

SCREEN.fill("white")

if dif==1:

HSCORE=high\_score('easy')

elif dif==2:

HSCORE=high\_score('normal')

elif dif==3:

HSCORE=high\_score('hard')

HIGHSCORE=("PLAYER NAME:{} HIGHEST SCORE:{} DIFFICULTY:{}").format(HSCORE[0],HSCORE[1],HSCORE[2])

OPTIONS\_TEXT = get\_font(45).render(HIGHSCORE, True, "Black")

OPTIONS\_RECT = OPTIONS\_TEXT.get\_rect(center=(640, 260))

SCREEN.blit(OPTIONS\_TEXT, OPTIONS\_RECT)

PLAY\_BACK = Button(image=None, pos=(640, 600),

text\_input="BACK", font=get\_font(75), base\_color="Black", hovering\_color="Green")

PLAY\_BACK.changeColor(SCORES\_MOUSE\_POS)

PLAY\_BACK.update(SCREEN)

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

sys.exit()

if event.type == pygame.MOUSEBUTTONDOWN:

if PLAY\_BACK.checkForInput(SCORES\_MOUSE\_POS):

options()

pygame.display.update()

def start(dif):

base\_font = pygame.font.Font(None, 50)

user\_text = ''

input\_rect = pygame.Rect(200, 150, 900, 50)

colour = pygame.Color('maroon')

while True:

START\_MOUSE\_POS = pygame.mouse.get\_pos()

SCREEN.fill((0, 0, 100))

PLAY\_TEXT = get\_font(45).render("ENTER PLAYER NAME", True, "White")

PLAY\_RECT = PLAY\_TEXT.get\_rect(center=(640, 100))

SCREEN.blit(PLAY\_TEXT, PLAY\_RECT)

PLAY\_TEXT0 = get\_font(25).render("INSTRUCTIONS:", True, "Yellow")

PLAY\_TEXT1 = get\_font(25).render("Click on the rectangular base of the bottle to break it.", True, "White")

PLAY\_TEXT2 = get\_font(25).render("Do not click on a bomb or you lose a life (bomb colours are shown above the bottles).", True, "White")

PLAY\_TEXT3 = get\_font(25).render("You have "+str(4-dif)+" lives in total (shown as red circles in bottom left).", True, "White")

PLAY\_TEXT4 = get\_font(25).render("After all bottles in a given iteration are broken, new bottles (and bombs) will be generated.", True, "White")

PLAY\_TEXT5 = get\_font(25).render("You have a total time of "+str(120 - 30\*dif)+" seconds to break as many bottles as possible.", True, "White")

PLAY\_RECT0 = PLAY\_TEXT1.get\_rect(center=(340, 250))

PLAY\_RECT1 = PLAY\_TEXT1.get\_rect(center=(340, 280))

PLAY\_RECT2 = PLAY\_TEXT1.get\_rect(center=(340, 310))

PLAY\_RECT3 = PLAY\_TEXT1.get\_rect(center=(340, 340))

PLAY\_RECT4 = PLAY\_TEXT1.get\_rect(center=(340, 370))

PLAY\_RECT5 = PLAY\_TEXT1.get\_rect(center=(340, 400))

SCREEN.blit(PLAY\_TEXT0, PLAY\_RECT0)

SCREEN.blit(PLAY\_TEXT1, PLAY\_RECT1)

SCREEN.blit(PLAY\_TEXT2, PLAY\_RECT2)

SCREEN.blit(PLAY\_TEXT3, PLAY\_RECT3)

SCREEN.blit(PLAY\_TEXT4, PLAY\_RECT4)

SCREEN.blit(PLAY\_TEXT5, PLAY\_RECT5)

START\_BUTTON = Button(image=None, pos=(240, 500),

text\_input="START", font=get\_font(75), base\_color="White", hovering\_color="Green")

START\_BUTTON.changeColor(START\_MOUSE\_POS)

START\_BUTTON.update(SCREEN)

PLAY\_BACK = Button(image=None, pos=(1060, 500),

text\_input="BACK", font=get\_font(75), base\_color="White", hovering\_color="Green")

PLAY\_BACK.changeColor(START\_MOUSE\_POS)

PLAY\_BACK.update(SCREEN)

pygame.draw.rect(SCREEN, colour, input\_rect)

text\_surface = base\_font.render(user\_text, True, (255, 255, 255))

SCREEN.blit(text\_surface, (input\_rect.x+5, input\_rect.y+5))

pygame.display.flip()

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

sys.exit()

if event.type == pygame.MOUSEBUTTONDOWN:

if PLAY\_BACK.checkForInput(START\_MOUSE\_POS):

play()

if START\_BUTTON.checkForInput(START\_MOUSE\_POS):

GAME(dif,user\_text)

if event.type == pygame.KEYDOWN:

if event.key == pygame.K\_BACKSPACE:

user\_text = user\_text[:-1]

# Unicode standard is used for string formation

else:

user\_text += event.unicode

pygame.display.update()

def play():

while True:

PLAY\_MOUSE\_POS = pygame.mouse.get\_pos()

SCREEN.fill("black")

EASY\_BUTTON = Button(image=pygame.image.load("Rectangle1.jpg"), pos=(240, 250),

text\_input="EASY", font=get\_font(75), base\_color="#d7fcd4", hovering\_color="White")

EASY\_BUTTON.changeColor(PLAY\_MOUSE\_POS)

EASY\_BUTTON.update(SCREEN)

NORMAL\_BUTTON = Button(image=pygame.image.load("Rectangle1.jpg"), pos=(640, 250),

text\_input="NORMAL", font=get\_font(75), base\_color="#d7fcd4", hovering\_color="White")

NORMAL\_BUTTON.changeColor(PLAY\_MOUSE\_POS)

NORMAL\_BUTTON.update(SCREEN)

HARD\_BUTTON = Button(image=pygame.image.load("Rectangle1.jpg"), pos=(1040, 250),

text\_input="HARD", font=get\_font(75), base\_color="#d7fcd4", hovering\_color="White")

HARD\_BUTTON.changeColor(PLAY\_MOUSE\_POS)

HARD\_BUTTON.update(SCREEN)

PLAY\_BACK = Button(image=None, pos=(640, 460),

text\_input="BACK", font=get\_font(75), base\_color="White", hovering\_color="Green")

PLAY\_BACK.changeColor(PLAY\_MOUSE\_POS)

PLAY\_BACK.update(SCREEN)

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

sys.exit()

if event.type == pygame.MOUSEBUTTONDOWN:

if PLAY\_BACK.checkForInput(PLAY\_MOUSE\_POS):

main\_menu()

if EASY\_BUTTON.checkForInput(PLAY\_MOUSE\_POS):

start(1)

if NORMAL\_BUTTON.checkForInput(PLAY\_MOUSE\_POS):

start(2)

if HARD\_BUTTON.checkForInput(PLAY\_MOUSE\_POS):

start(3)

pygame.display.update()

def options():

while True:

OPTIONS\_MOUSE\_POS = pygame.mouse.get\_pos()

SCREEN.fill("white")

SCORE\_EASY = Button(image=pygame.image.load("Rectangle1.jpg"), pos=(240, 250),

text\_input="Easy Difficulty", font=get\_font(60), base\_color="#d7fcd4", hovering\_color="White")

SCORE\_EASY.changeColor(OPTIONS\_MOUSE\_POS)

SCORE\_EASY.update(SCREEN)

SCORE\_NORMAL = Button(image=pygame.image.load("Rectangle1.jpg"), pos=(640, 250),

text\_input="Normal Difficulty", font=get\_font(60), base\_color="#d7fcd4", hovering\_color="White")

SCORE\_NORMAL.changeColor(OPTIONS\_MOUSE\_POS)

SCORE\_NORMAL.update(SCREEN)

SCORE\_HARD = Button(image=pygame.image.load("Rectangle1.jpg"), pos=(1040, 250),

text\_input="Hard Difficulty", font=get\_font(60), base\_color="#d7fcd4", hovering\_color="White")

SCORE\_HARD.changeColor(OPTIONS\_MOUSE\_POS)

SCORE\_HARD.update(SCREEN)

OPTIONS\_BACK = Button(image=None, pos=(640, 460),

text\_input="BACK", font=get\_font(75), base\_color="Black", hovering\_color="Green")

OPTIONS\_BACK.changeColor(OPTIONS\_MOUSE\_POS)

OPTIONS\_BACK.update(SCREEN)

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

sys.exit()

if event.type == pygame.MOUSEBUTTONDOWN:

if OPTIONS\_BACK.checkForInput(OPTIONS\_MOUSE\_POS):

main\_menu()

if SCORE\_EASY.checkForInput(OPTIONS\_MOUSE\_POS):

SCORES(1)

if SCORE\_NORMAL.checkForInput(OPTIONS\_MOUSE\_POS):

SCORES(2)

if SCORE\_HARD.checkForInput(OPTIONS\_MOUSE\_POS):

SCORES(3)

pygame.display.update()

def main\_menu():

while True:

SCREEN.blit(BG, (0, 0))

MENU\_MOUSE\_POS = pygame.mouse.get\_pos()

MENU\_TEXT = get\_font(100).render("BOTTLE BASH", True, "#b68f40")

MENU\_RECT = MENU\_TEXT.get\_rect(center=(640, 100))

PLAY\_BUTTON = Button(image=pygame.image.load("Rectangle1.jpg"), pos=(640, 250),

text\_input="PLAY", font=get\_font(75), base\_color="#d7fcd4", hovering\_color="White")

OPTIONS\_BUTTON = Button(image=pygame.image.load("Rectangle1.jpg"), pos=(640, 400),

text\_input="HIGH SCORES", font=get\_font(70), base\_color="#d7fcd4", hovering\_color="White")

QUIT\_BUTTON = Button(image=pygame.image.load("Rectangle1.jpg"), pos=(640, 550),

text\_input="QUIT", font=get\_font(75), base\_color="#d7fcd4", hovering\_color="White")

SCREEN.blit(MENU\_TEXT, MENU\_RECT)

for button in [PLAY\_BUTTON, OPTIONS\_BUTTON, QUIT\_BUTTON]:

button.changeColor(MENU\_MOUSE\_POS)

button.update(SCREEN)

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

sys.exit()

if event.type == pygame.MOUSEBUTTONDOWN:

if PLAY\_BUTTON.checkForInput(MENU\_MOUSE\_POS):

play()

if OPTIONS\_BUTTON.checkForInput(MENU\_MOUSE\_POS):

options()

if QUIT\_BUTTON.checkForInput(MENU\_MOUSE\_POS):

pygame.quit()

sys.exit()

pygame.display.update()

main\_menu()

cn.close()

**Team Roles**

* Pratham-GUI
* Arya-Game Mechanics
* Suyog-Data Handling

Pratham-GUI

* Pygame and sys modules are used to create the opening screen
* The functions used are main\_menu (to let the player decide if they want to play or check the high scores), SCORES( to let the player check the highest scores in each difficulty level), play (to let the player enter their name) and start (to start playing the game)
* Buttons are used to direct the player to either check their scores, play the game or quit the program entirely.

Arya-Game Mechanics

* Pygame, time and random modules are used to make the game
* The bottles, lives left and bombs are made using different geometrical objects
* The player’s name, current score, number of lives left and time left are displayed on the screen
* The functions used are timer (to display the time left), score (to display the player’s current score)
* The game (including the above functions) is put in a parent function known as GAME, which takes name and difficulty level as parameters

Suyog-Data Handling

* Sqlite3 module is used to connect python and Sqlite DBMS
* The code has several functions, which makes individual connections to prevent conflict
* There are 4 functions which: checks for old existing player, enters data into the table, clears tables and checks player high score respectively
* The table containing data is created once and for all the first time the program is run in a new computer, the whole process is automated

**Software and Hardware Requirements**

* *Software:*
  + - Python 3
    - Suitable OS
    - Suitable IDE or Terminal
* *Hardware:*
  + - A decently fast computer (Dual core processor or better)
    - Disk space of 10MB
    - Enough RAM for the computer to have minimum functionality

**Result and Analysis**

A picture containing text, device, meter, control panel

Description automatically generated Graphical user interface, website

Description automatically generated

Main Menu Entering Player Details

Chart, histogram

Description automatically generated Graphical user interface, text

Description automatically generated

Game Screen (Hard) Game Over

A screenshot of a computer

Description automatically generated with medium confidence Graphical user interface

Description automatically generated

Highest Score (Hard) Table Containing Names of All Participants

**Conclusion and Future Enhancements**

* This game can be addictive and help reduce mental stress.
* In the future, the game can be modified to allow people to hold competitions online, requiring them to deposit some money to participate. Those with the highest scores will be eligible for cash price and other rewards.

**References**

* [https://www.geeksforgeeks.org/introduction-to-pygame/](https://realpython.com/pygame-a-primer/)
* <https://realpython.com/pygame-a-primer/>
* <https://pythonprogramming.net/pygame-drawing-shapes-objects/>
* <https://coderslegacy.com/python/pygame-mouse-click/>
* <https://www.programiz.com/python-programming/time>