***Summer Training Report***

****

Aditya negi (00614902018)

**Objective**

To make a basic game using collision concept.

**Hardware and software utilized**

**Hardware:-**

1. A desktop PC

**Software:-**

1. Operating System

2. Anaconda platform

3. Jupyter notebook IDE

**Tools and Framework used**

1. Python programming language

2. Pygame : It’s an external package used for GUI

**Concept used:**

*Collision concept*: Pygame uses Rect objects to store and manipulate rectangular areas.

“Enemy and player ships are rectangular shapes created using pygame

When position of bullet i.e. A circular object is same as of the enemy ship, u get 1 point.

Nd random was used to get random position of enemy ship spawn position.”

**Music used:**

* Gunfire.mp3 => Bullets
* Burythelight.mp3 => Background Music
* Crashblast.mp3 => When bullets hit

**Source code:**

import pygame

import random

# music and sounds

file = 'burythelight.mp3'

pygame.init()

pygame.mixer.init()

pygame.mixer.music.load(file)

pygame.mixer.music.play(-1)

bullet\_fire = pygame.mixer.Sound("gunfire.mp3")

crash\_sound = pygame.mixer.Sound("crashblast.mp3")

pygame.init()

# screen parameters

screenWidth = 500

screenheight = 500

win = pygame.display.set\_mode((screenheight, screenWidth))

# background loading

pygame.display.set\_caption('Space Game')

# font

font = pygame.font.SysFont(None, 20)

# player ship properties

# player ship class

class Ship(object):

    def \_\_init\_\_(self, x, y, width, height):

        self.x = x

        self.y = y

        self.width = width

        self.height = height

        self.velocity\_ship = 5

# loading player ship and assigning it a rectangular hitbox

    def draw\_ship(self, win):

        shipImg = pygame.image.load("ship.png")

        pygame.draw.rect(win, (0, 191, 255), (self.x, self.y, self.width, self.height), -2)

        rect = shipImg.get\_rect()

        rect.center = (50, 30)

        win.blit(shipImg, [playerShip.x - 11, playerShip.y - 3])

# initializing bullets

class Bullet(object):

    def \_\_init\_\_(self, x, y, radius, color):

        self.x = playerShip.x + 22

        self.y = playerShip.y

        self.radius = radius

        self.color = color

        self.vel = 7

    def draw(self, win):

        pygame.draw.circle(win, self.color, (self.x, self.y), self.radius)

bullets = []

# enemy ship

# loading enemy ship class

enemyship = pygame.image.load("enemy.png")

enemyship = pygame.transform.rotate(enemyship, 180)

class Enemy(object):

    def \_\_init\_\_(self, x, y, width, height):

        self.x = x

        self.y = y

        self.width = width

        self.height = height

        self.vel = 2

# drawing enemy ship and assigning it a rectangular hitbox

    def draw(self, win):

        pygame.draw.rect(win, (0, 255, 0), (self.x, self.y, self.width, self.height), -2)

        win.blit(enemyship, [self.x - 5, self.y])

clock = pygame.time.Clock()

# background image is loaded

backGround = pygame.image.load("bg.jpg").convert()

# drawing window for the game

def screendraw(win):

    win.blit(backGround, [0, 0])

    Dashboard(win)

    playerShip.draw\_ship(win)

    for bullet in bullets:

        bullet.draw(win)

    for enemy in enemies:

        enemy.draw(win)

    pygame.display.update()

# dashboard details - fps , score , Enemies passed

def Dashboard(win):

    showFps = font.render('FPS ' + str(fps), True, (255, 0, 0))

    win.blit(showFps, [0, 0])

    showpoints = font.render('Score = ' + str(score), True, (255, 0, 0))

    win.blit(showpoints, [410, 0])

    enemypass = font.render('enemies pass :  ' + str(points), True, (255, 0, 0))

    win.blit(enemypass, [350, 400])

firetime = 0

enemytime = 0

score = 0

points = 0

enemies = []

run = True

playerShip = Ship(250, 440, 40, 50)

collision = False

# main loop

while run:

    pygame.time.delay(0)                                                         # game speed

    for event in pygame.event.get():

        if event.type == pygame.QUIT:

            run = False

    for bullet in bullets:                                                       # bullet movement

        if (bullet.y <= playerShip.y) and (bullet.y > 0):

            bullet.y -= bullet.vel

        else:

            bullets.pop(bullets.index(bullet))                                   # popping out of bounds bullet

    for enemy in enemies:                                                        # enemy movement

        if enemy.y < screenheight:

            enemy.y += enemy.vel

        else:

            enemies.pop(enemies.index(enemy))                                    # popping out of bounds enemies and incrementing enemies passed value by 1

            points += 1

# checking for collision between enemies and bullets and updating the score by 1

    for enemy in enemies:

        for bullet in bullets:

            if (bullet.x >= enemy.x) and (bullet.x <= (enemy.x + enemy.width)) and (bullet.y <= enemy.y + enemy.height) and (bullet.y >= enemy.y):

                pygame.mixer.Sound.play(crash\_sound)

                enemies.pop(enemies.index(enemy))

                bullets.pop(bullets.index(bullet))

                score += 1

    # collision = (bullet.x >= enemy.x) and (bullet.x <= (enemy.x + enemy.width))

    clock.tick(60)

    if firetime > 0:

        firetime += 1

    if firetime >= 15:

        firetime = 0

    if enemytime > 0:

        enemytime += 1

    if enemytime >= 100:

        enemytime = 0

    fps = int(clock.get\_fps())

# keybinds for player ship and makig sure the player ship doesn't go out of bounds

    keys = pygame.key.get\_pressed()

    if keys[pygame.K\_a] and playerShip.x > playerShip.velocity\_ship - 5:

        playerShip.x -= playerShip.velocity\_ship

    if keys[pygame.K\_d] and playerShip.x < screenWidth - playerShip.width:

        playerShip.x += playerShip.velocity\_ship

    if keys[pygame.K\_w] and playerShip.y > playerShip.velocity\_ship - 5:

        playerShip.y -= playerShip.velocity\_ship

    if keys[pygame.K\_s] and playerShip.y < screenheight - playerShip.height:

        playerShip.y += playerShip.velocity\_ship

    v = playerShip.y

    if keys[pygame.K\_SPACE] and firetime == 0:

        if len(bullets) < 3:

            bullets.append(Bullet(round(playerShip.x + playerShip.width //2), round(playerShip.y), 4, (255, 0, 0)))

            pygame.mixer.Sound.play(bullet\_fire)

        firetime = 1

    if enemytime == 0:

        if len(enemies) < 2:

            enemies.append((Enemy(random.randrange(0, screenWidth - 40, 10), -60, 55, 60)))

        enemytime = 1

    screendraw(win)

# when enemies passed becomes 5 then GAME OVER

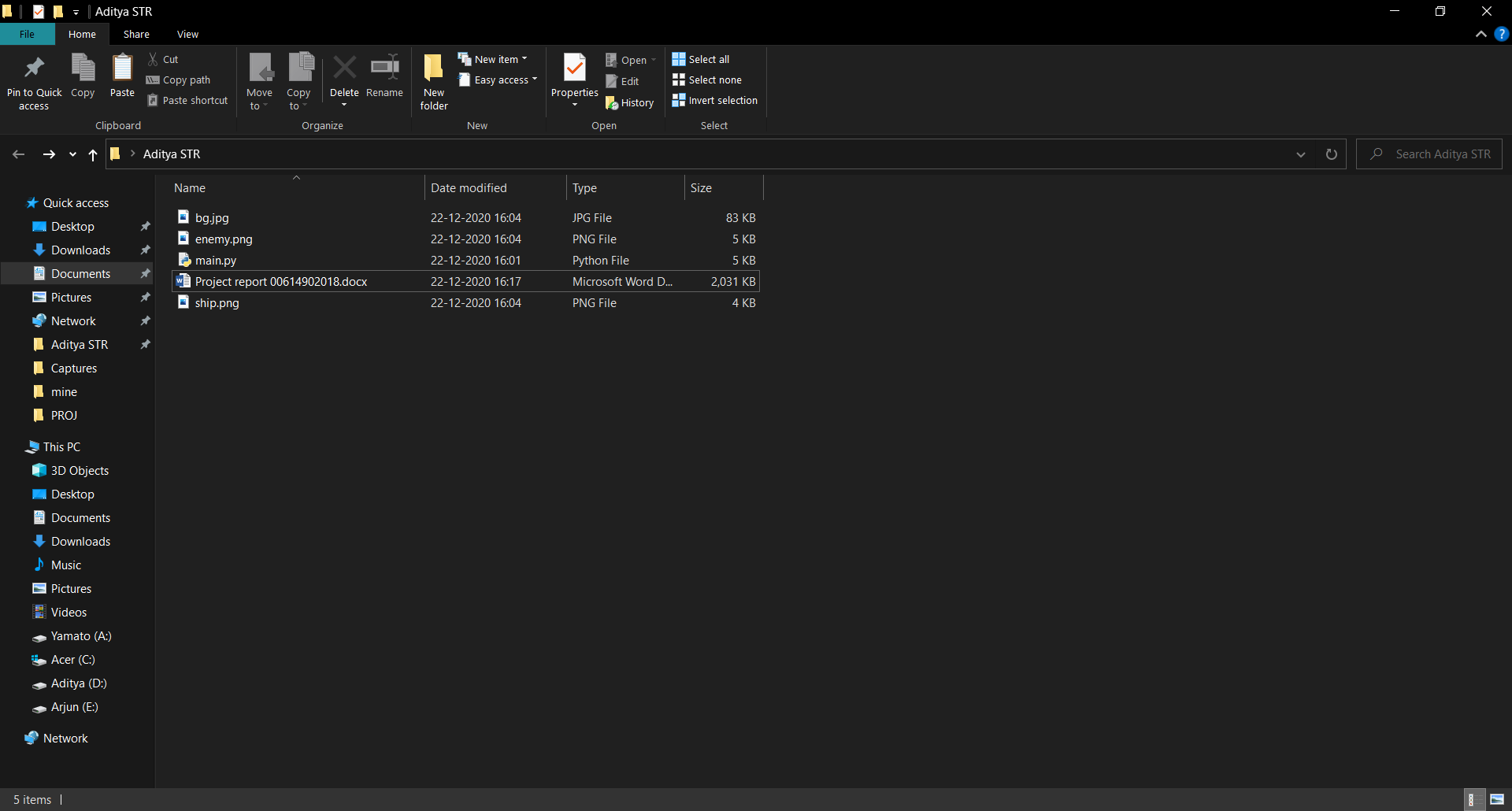
    if points == 5:

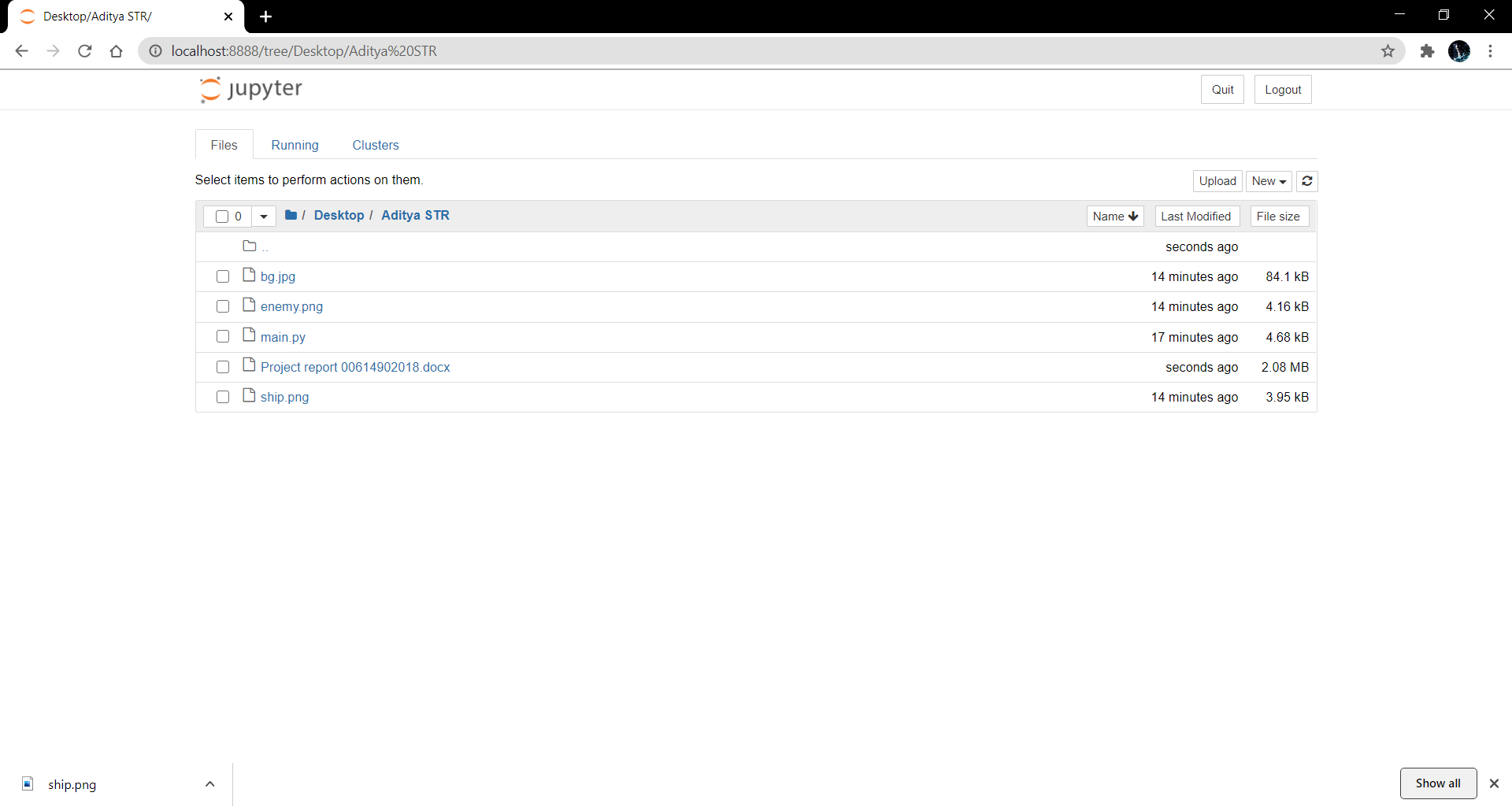
        run = False

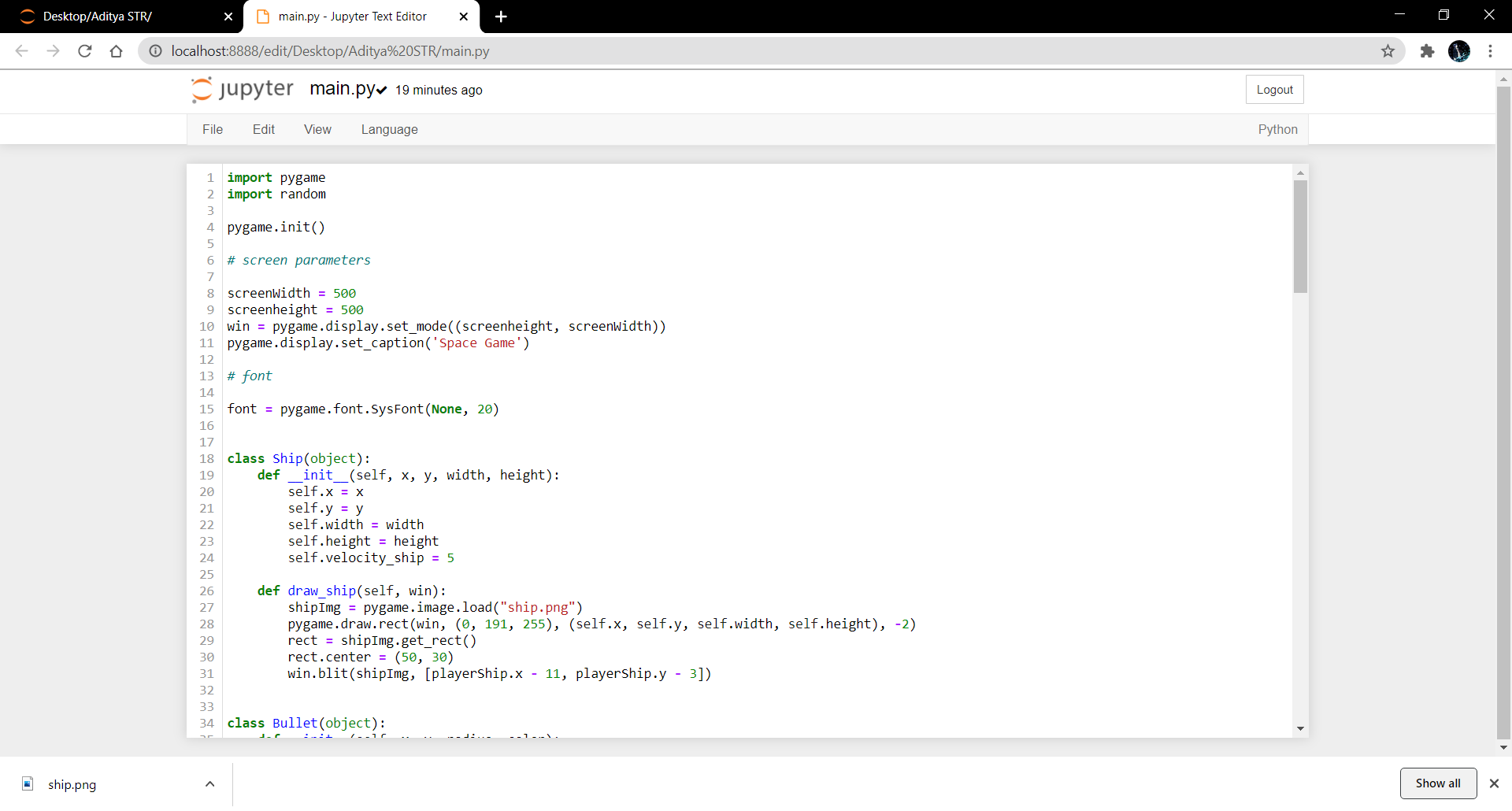
    # win.fill((0, 0, 0))

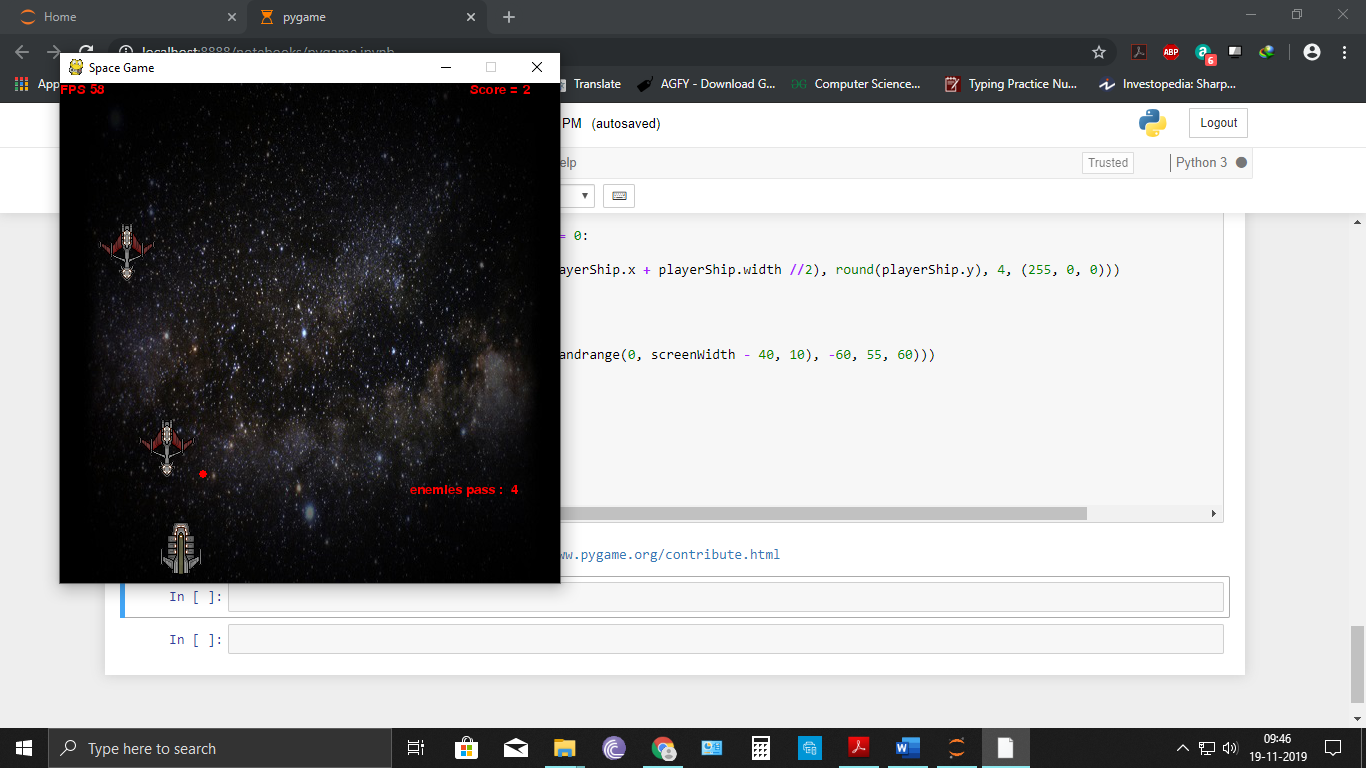
pygame.quit()

**Screenshots:**









**Certificate of training:**

****