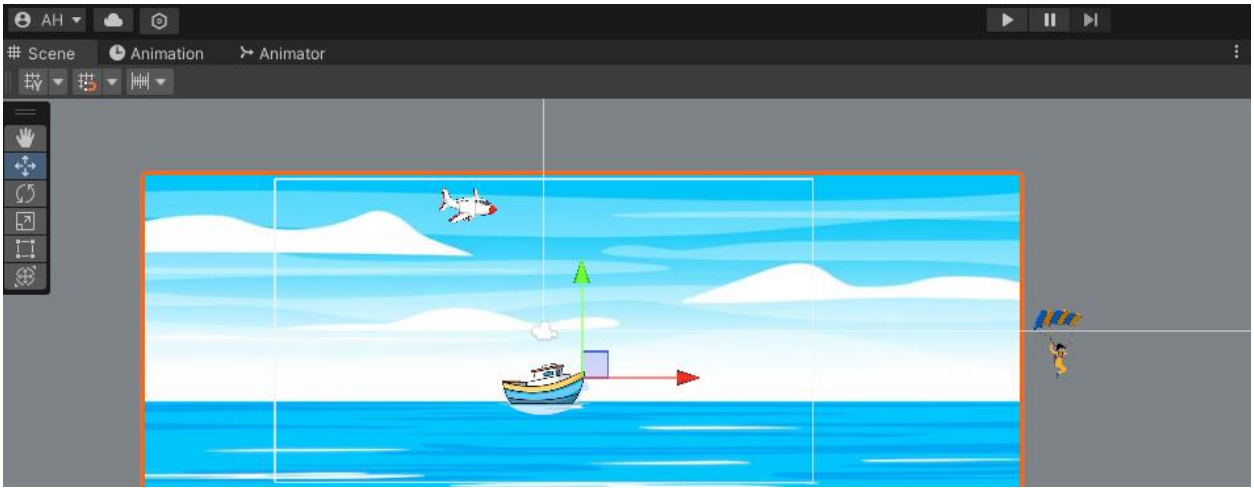
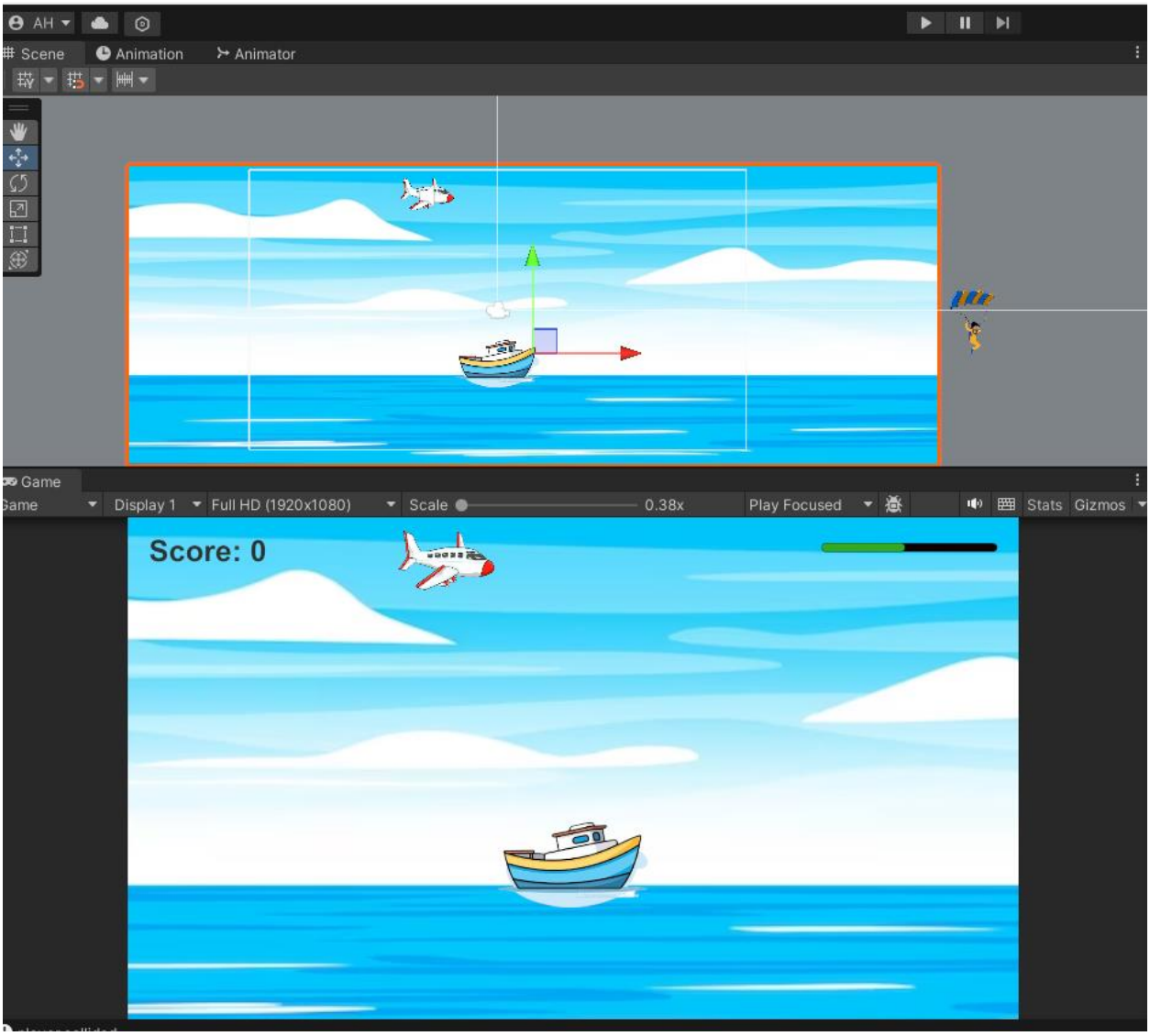
CSC496 - Game Development Terminal Lab Exam

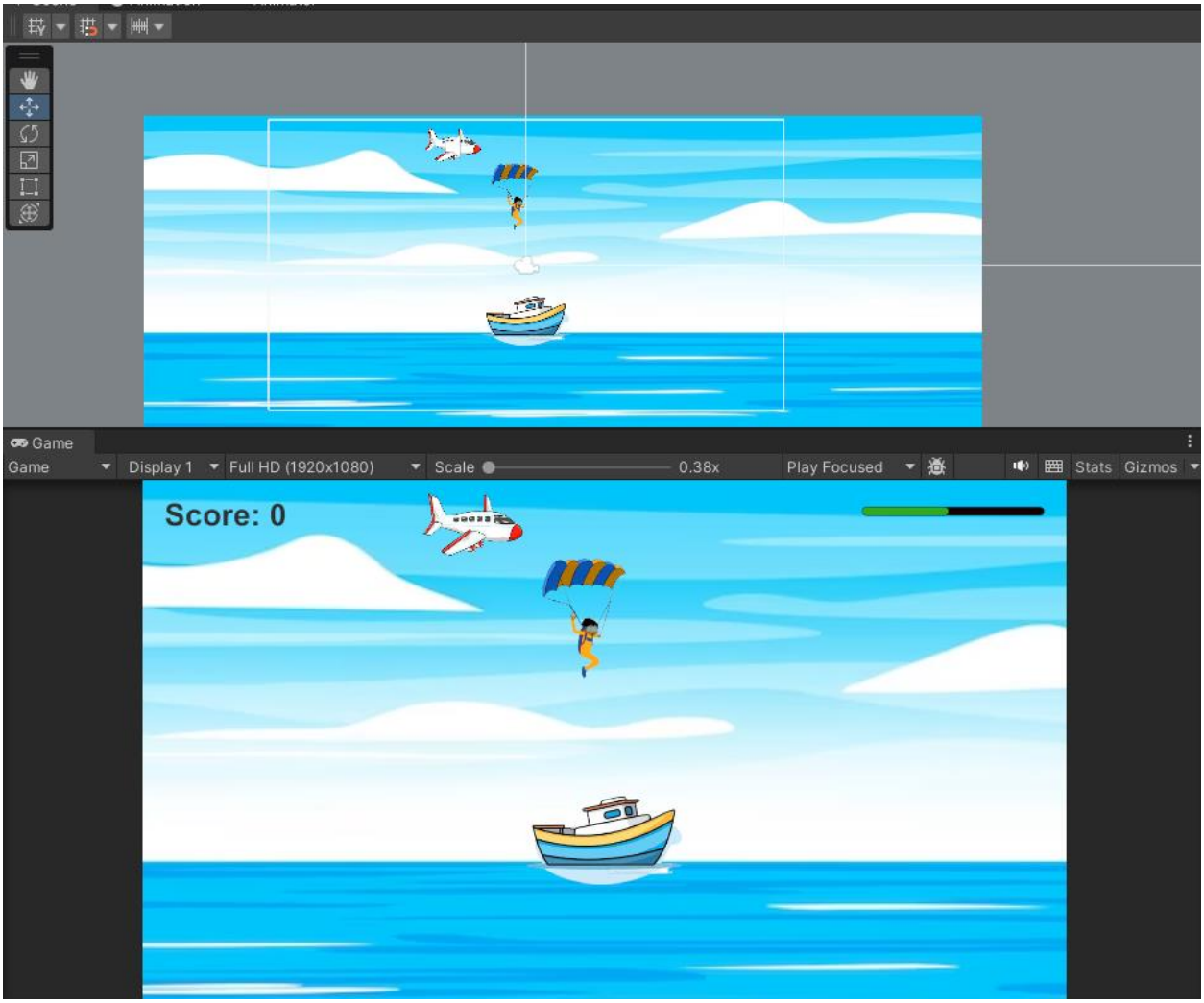
Name: Mr. Aoun Haider

ID: FA21-BSE-133




AH
Scene
Animation
Animator
Game
Display 1
Full HD (1920x1080)
Scale
0.38x
Play Focused
Stats
Gizmos
Score: 0


Score: 0



Scripts:

./AeroplaneScript.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class AeroplaneScript : MonoBehaviour
{
    public float speed = 5f;
    public float leftEdge;
    public float rightEdge;

    public float InstantiateTime = 3.0f;
    public GameObject paratrooper;
```

```
    public bool isMovingRight = true;

    void Update()
    {
        CreatePrefab();
        Move();
        CheckEdgesAndFlip();
    }
    void CreatePrefab()
    {
        InstantiateTime -= Time.deltaTime;
        if(InstantiateTime <= 0)
        {
            Instantiate(paratrooper,transform.position,Quaternion.identity);
            InstantiateTime = Random.Range(2,4);
        }
    }
    public void Move()
    {
        float moveDirection = isMovingRight ? 1 : -1;
        transform.Translate(Vector2.right * moveDirection * speed *
Time.deltaTime);
    }

    private void CheckEdgesAndFlip()
    {
        if (transform.position.x >= rightEdge && isMovingRight)
        {
            isMovingRight = false;
            FlipSprite();
        }
        else if (transform.position.x <= leftEdge && !isMovingRight)
        {
            isMovingRight = true;
            FlipSprite();
        }
    }
    private void FlipSprite()
    {
        Vector3 scale = transform.localScale;
        scale.x *= -1;
        transform.localScale = scale;
    }
}
```

./boatScript.cs

```
using UnityEngine;
using UnityEngine.UI;

public class BoatScript : MonoBehaviour
{
    [Header("Movement Settings")]
    public float speed = 5f; // Speed at which the boat moves
    public float leftEdge = -10f; // Left boundary
    public float rightEdge = 10f; // Right boundary
    private int score = 0; // Starting score
    public Text scoreText;

    public Text GameWinText;
    private bool isFacingRight = true; // To keep track of the boat's facing
direction

    private void Start()
    {
        // Initialize the score text
        UpdateScoreText();
    }

    private void Update()
    {
        if(score>50)
        {
            GameWinText.text = "You Won the Game!";
        }
        MoveBoat();
    }

    // Function to move the boat left or right based on input
    private void MoveBoat()
    {
        float horizontalInput = Input.GetAxis("Horizontal");

        if (horizontalInput > 0 && transform.position.x < rightEdge)
        {
            // Move right and flip sprite to the right
            transform.Translate(Vector2.right * speed * Time.deltaTime);
            if (!isFacingRight) FlipSprite();
        }
        else if (horizontalInput < 0 && transform.position.x > leftEdge)
```

```
        {
            // Move left and flip sprite to the left
            transform.Translate(Vector2.left * speed * Time.deltaTime);
            if (isFacingRight) FlipSprite();
        }
    }

    // Function to flip the boat's sprite
    private void FlipSprite()
    {
        isFacingRight = !isFacingRight;
        Vector3 localScale = transform.localScale;
        localScale.x *= -1; // Invert the x scale to flip the sprite
        transform.localScale = localScale;
    }

    // Function to detect collision with other objects
    private void OnCollisionEnter2D(Collision2D collision)
    {
        // Increase score by 5 when collided with any object
        score += 5;
        UpdateScoreText();
        Destroy(collision.transform.gameObject);
    }

    // Function to update the score text
    private void UpdateScoreText()
    {
        scoreText.text = "Score: " + score.ToString();
    }
}
```

./playerScript.cs

```
using UnityEngine;
using UnityEngine.UI;

public class PlayerScript : MonoBehaviour
{
    public float speed = 2f;
    public float destroyYPosition = -3.93f;

    public Slider healthSlider;
```

```
    public Text scoreText;
    public Text GameOverText, GameWinText;
    private int health = 50;
    public int score;

    private void Start()
    {
        // Find the Slider in the scene and assign it to healthSlider
        GameObject sliderObject = GameObject.Find("Slider");
        if (sliderObject != null)
        {
            healthSlider = sliderObject.GetComponent<Slider>();
        }

        if (healthSlider == null)
        {
            Debug.LogError("Health Slider is not assigned or found!");
            return;
        }

        // Find the score text in the scene and assign it to scoreText
        GameObject scoreTextObject = GameObject.Find("ScoreText");
        if (scoreTextObject != null)
        {
            scoreText = scoreTextObject.GetComponent<Text>();
        }

        if (scoreText == null)
        {
            Debug.LogError("Score Text is not assigned or found!");
            return;
        }

        // Find the GameOver text in the scene and assign it to GameOverText
        GameObject gameOverTextObject = GameObject.Find("GameOverText");
        if (gameOverTextObject != null)
        {
            GameOverText = gameOverTextObject.GetComponent<Text>();
        }

        if (GameOverText == null)
        {
            Debug.LogError("Game Over Text is not assigned or found!");
            return;
        }
```

```
        // Find the GameWin text in the scene and assign it to GameWinText
        GameObject gameWinTextObject = GameObject.Find("WinText");
        if (gameWinTextObject != null)
        {
            GameWinText = gameWinTextObject.GetComponent<Text>();
        }

        if (GameWinText == null)
        {
            Debug.LogError("Game Win Text is not assigned or found!");
            return;
        }

        healthSlider.value = health;
    }

    void Update()
    {
        MoveDownward();

        if (health <= 0)
        {
            GameOverText.text = "You Lose the Game!";
            Time.timeScale = 0;
        }

        if (score >= 50)
        {
            GameWinText.text = "You Won the Game!";
            Time.timeScale = 0;
        }
    }

    private void MoveDownward()
    {
        transform.Translate(Vector2.down * speed * Time.deltaTime);

        if (transform.position.y <= destroyYPosition)
        {
            health -= 5;
            if (healthSlider != null)
            {
                healthSlider.value = health;
            }
```

```
            Destroy(gameObject);
        }
    }

    private void OnCollisionEnter2D(Collision2D collision)
    {
        health += 5;
        if (healthSlider != null)
        {
            healthSlider.value = health;
        }
        print("player collided");
        score += 5;
        UpdateScoreText();
    }

    private void UpdateScoreText()
    {
        if (scoreText != null)
        {
            scoreText.text = "Score: " + score.ToString();
        }
    }
}
```