Lab # 11

OPEN ENDED LAB



Fall 2024-25

**CSE-411L Intro to Game Development Lab**

Submitted by: MUHAMMAD SADEEQ

Registration No.: 21PWCSE2028

Section: C

“On my honor, as a student of the University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work”

Submitted to:

Engr. Abdullah Hamid

(30 Jan 2025)

Department of Computer systems engineering

University of Engineering and Technology, Peshawar

1. **Scene Setup**
   1. Created a new Unity 3D project and designed a dojo-themed environment with a textured plane as the ground.
   2. Imported props from the Unity Asset Store and arranged them to enhance the scene’s realism.
   3. Adjusted lighting (directional light) and shadows to match the dojo aesthetic.
2. **Character Setup**
   1. Downloaded "Knight" (player) and "Ninja" (enemy) models from Mixamo.com. Imported them into Unity.
   2. Assigned animations:

**Player**: Idle, Walking, Punch, Kick (configured in Animator Controller).

**Enemy**: Idle, Walking, Punch.

1. Verified Humanoid Rigging compatibility using the Rig tab in the model Import Settings.
2. **Player Controls**

using Unity.VisualScripting;

using UnityEngine;

using UnityEngine.EventSystems;

using UnityEngine.UI;

public class PlayerController11 : MonoBehaviour

{

Animator animator;

float speed = 2.0f;

public float rotationSpeed = 5f;

public GameObject W, A, S, D;

private Vector3 movementDirection = Vector3.zero;

AudioSource punchSound;

public float playerHealth = 100;

public Image Healthbar;

public GameObject gameOverPanel;

int walkHash, punchHash, defeatHash;

Quaternion targetRotation;

public delegate void PlayerActionHandler();

public static event PlayerActionHandler OnPunch;

public static event PlayerActionHandler OnKick;

// Start is called before the first frame update

void Start()

{

punchSound = GetComponent<AudioSource>();

animator = GetComponent<Animator>();

walkHash = Animator.StringToHash("Walk");

punchHash = Animator.StringToHash("Punch");

defeatHash = Animator.StringToHash("PlayerHealth");

EventTrigger W\_ET = W.AddComponent<EventTrigger>();

EventTrigger S\_ET = S.AddComponent<EventTrigger>();

EventTrigger A\_ET = A.AddComponent<EventTrigger>();

EventTrigger D\_ET = D.AddComponent<EventTrigger>();

EventTrigger.Entry Forwardwalk = new EventTrigger.Entry();

EventTrigger.Entry Backwardwalk = new EventTrigger.Entry();

EventTrigger.Entry Leftwalk = new EventTrigger.Entry();

EventTrigger.Entry Rightwalk = new EventTrigger.Entry();

EventTrigger.Entry release = new EventTrigger.Entry();

Forwardwalk.eventID = EventTriggerType.PointerDown;

Backwardwalk.eventID = EventTriggerType.PointerDown;

Leftwalk.eventID = EventTriggerType.PointerDown;

Rightwalk.eventID = EventTriggerType.PointerDown;

release.eventID = EventTriggerType.PointerUp;

release.callback.AddListener(upPointer);

Forwardwalk.callback.AddListener(forward);

Backwardwalk.callback.AddListener(backward);

Leftwalk.callback.AddListener(left);

Rightwalk.callback.AddListener(right);

W\_ET.triggers.Add(Forwardwalk);

W\_ET.triggers.Add(release);

S\_ET.triggers.Add(Backwardwalk);

S\_ET.triggers.Add(release);

A\_ET.triggers.Add(Leftwalk);

A\_ET.triggers.Add(release);

D\_ET.triggers.Add(Rightwalk);

D\_ET.triggers.Add(release);

}

// Update is called once per frame

void Update()

{

transform.Translate(movementDirection \* speed \* Time.deltaTime);

if (Input.GetMouseButtonDown(0) && Input.GetKey(KeyCode.Q))

{

OnPunch = TriggerPunchAnimation;

OnPunch();

OnPunch -= TriggerPunchAnimation;

}

// Trigger kick animation (Mouse0 + W)

if (Input.GetMouseButtonDown(0) && Input.GetKey(KeyCode.W))

{

OnKick = TriggerKickAnimation;

OnKick();

OnKick = TriggerKickAnimation;

}

transform.rotation = Quaternion.Lerp(transform.rotation, targetRotation, Time.deltaTime \* rotationSpeed);

}

private void TriggerPunchAnimation()

{

animator.SetTrigger("Punch");

}

private void TriggerKickAnimation()

{

animator.SetTrigger("Kick");

}

public void forward(BaseEventData x)

{

animator.SetTrigger(walkHash);

movementDirection = Vector3.forward;

targetRotation = Quaternion.Euler(0, 0, 0);

}

public void backward(BaseEventData x)

{

animator.SetBool(walkHash, true);

movementDirection = Vector3.forward;

targetRotation = Quaternion.Euler(0, 180, 0);

}

public void left(BaseEventData x)

{

animator.SetBool(walkHash, true);

movementDirection = Vector3.forward;

targetRotation = Quaternion.Euler(0, -90, 0);

}

public void right(BaseEventData x)

{

animator.SetBool(walkHash, true);

movementDirection = Vector3.forward;

targetRotation = Quaternion.Euler(0, 90, 0);

}

public void upPointer(BaseEventData x)

{

animator.SetBool(walkHash, false);

movementDirection = Vector3.zero;

}

public void PunchSound()

{

punchSound.Play();

}

private void OnTriggerEnter(Collider other)

{

if (other.gameObject.CompareTag("EDamage"))

{

TakeDamage(10);

animator.SetFloat(defeatHash,playerHealth);

}

}

void TakeDamage(float Damage )

{

playerHealth -= Damage;

Healthbar.fillAmount -= Damage/100;

}

public void PlayerDied()

{

gameOverPanel.SetActive(true);

Time.timeScale = 0;

}

}

1. **Enemy Behavior**

using System.Collections;

using UnityEditor;

using UnityEngine;

using UnityEngine.AI;

using UnityEngine.UI;

public class EnemyController11 : MonoBehaviour

{

public NavMeshAgent Agent;

public GameObject Player;

Animator animator;

private int walkHash, punchHash,defeatHash;

public float enemyHealth = 100;

public AudioSource punchSound;

public GameObject victoryPanel;

public Image Healthbar;

// Start is called before the first frame update

void Start()

{

punchSound = GetComponent<AudioSource>();

animator = GetComponent<Animator>();

walkHash = Animator.StringToHash("isWalking");

punchHash = Animator.StringToHash("Punch");

defeatHash = Animator.StringToHash("EnemyHealth");

Agent = gameObject.GetComponent<NavMeshAgent>();

Player = GameObject.FindGameObjectWithTag("Player");

Agent.updateRotation = true;

}

// Update is called once per frame

void Update()

{

if (Vector3.Distance(transform.position, Player.transform.position) < 5f)

{

Agent.SetDestination(Player.transform.position);

animator.SetBool(walkHash, true);

if(Vector3.Distance(transform.position, Player.transform.position) < 1.5f)

{

animator.SetTrigger(punchHash);

}

}

else

{

Agent.ResetPath();

animator.SetBool(walkHash, false);

}

}

public void PunchSound()

{

punchSound.Play();

}

private void OnTriggerEnter(Collider other)

{

if (other.gameObject.CompareTag("Damage"))

{

TakeDamage(20);

animator.SetFloat(defeatHash, enemyHealth);

}

}

void TakeDamage(float Damage)

{

enemyHealth -= Damage;

Healthbar.fillAmount -= Damage / 100;

}

public void EnemyDied()

{

victoryPanel.SetActive(true);

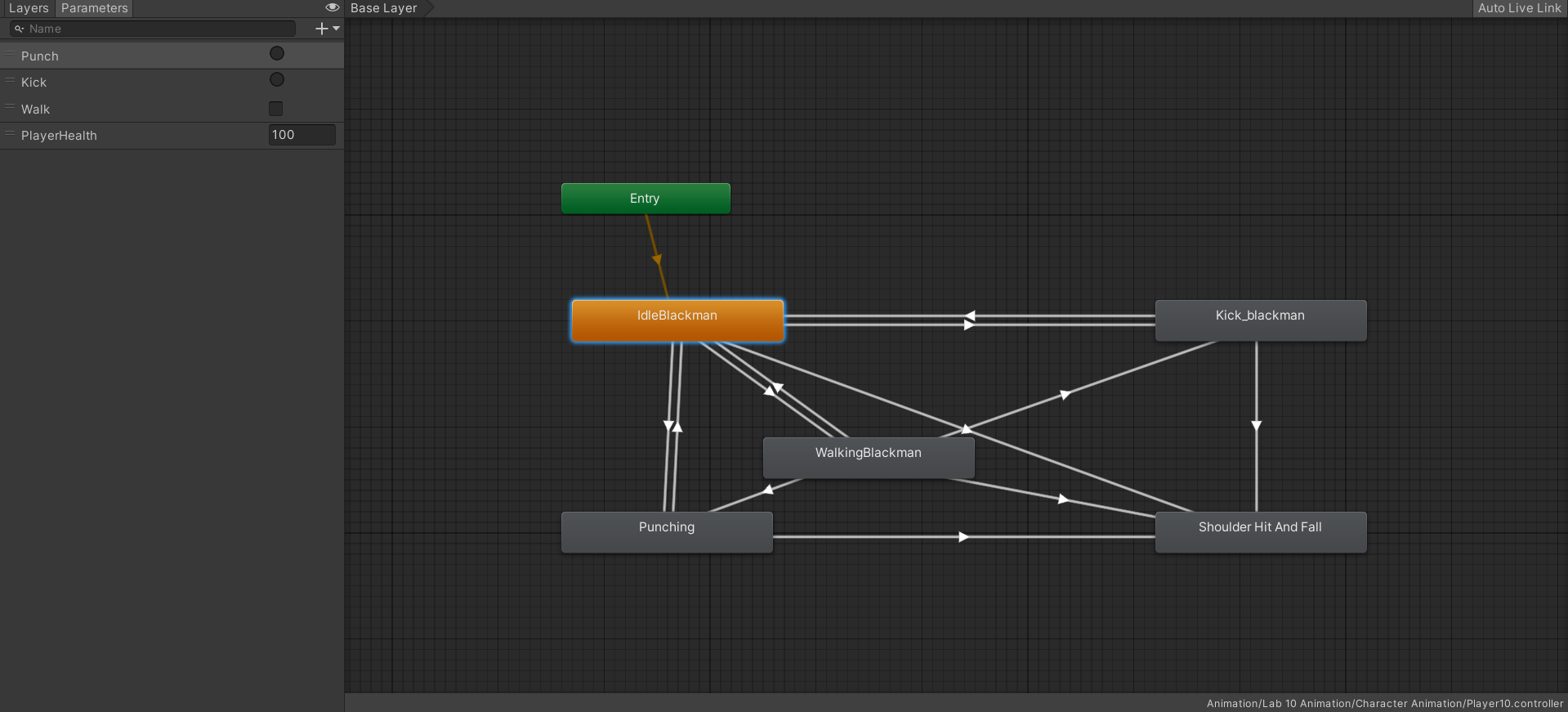
Time.timeScale = 0;

}

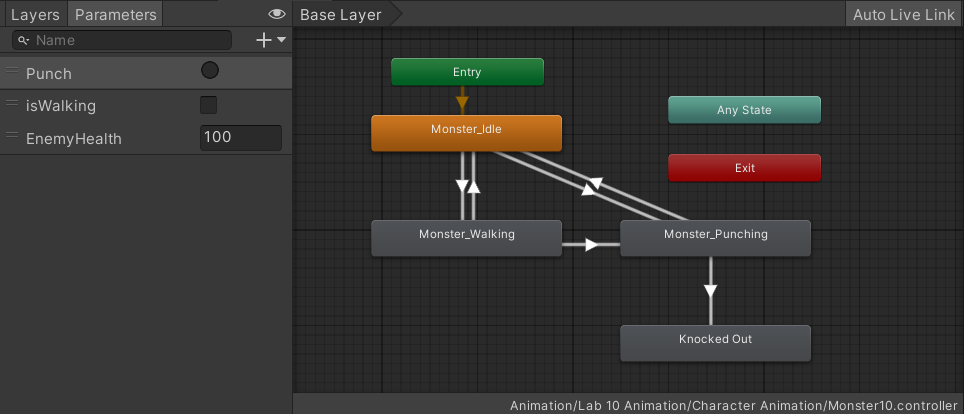
}

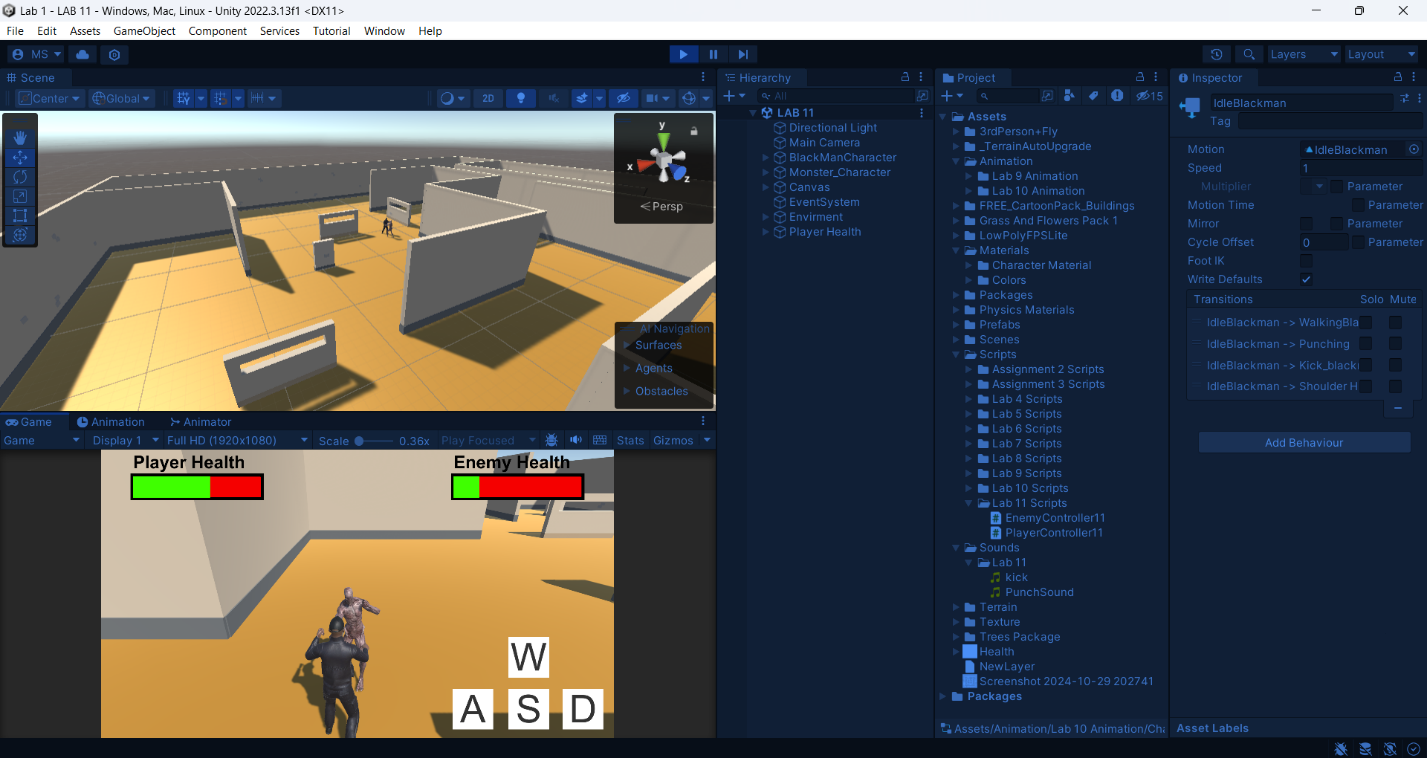
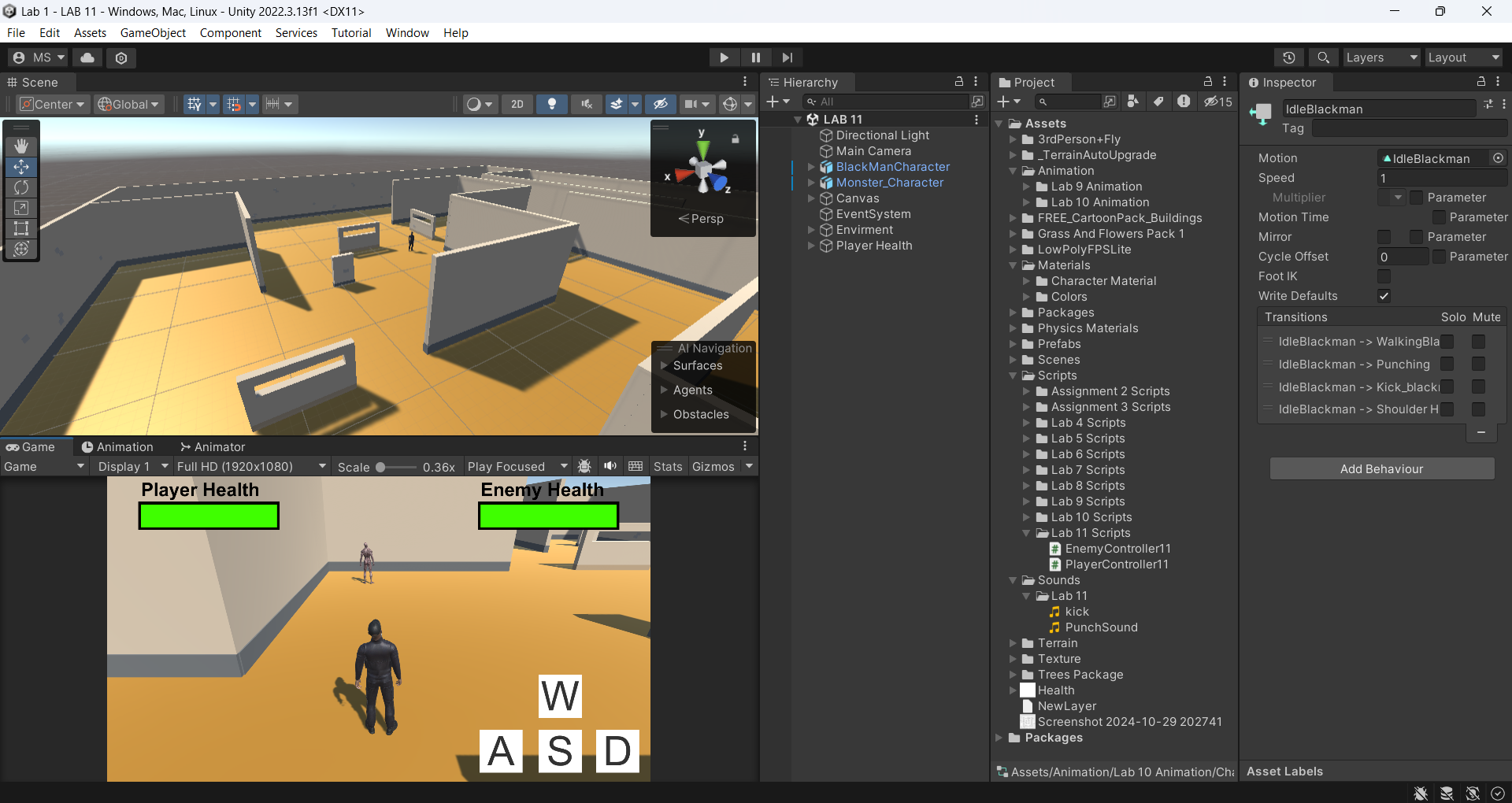
1. **Animation**

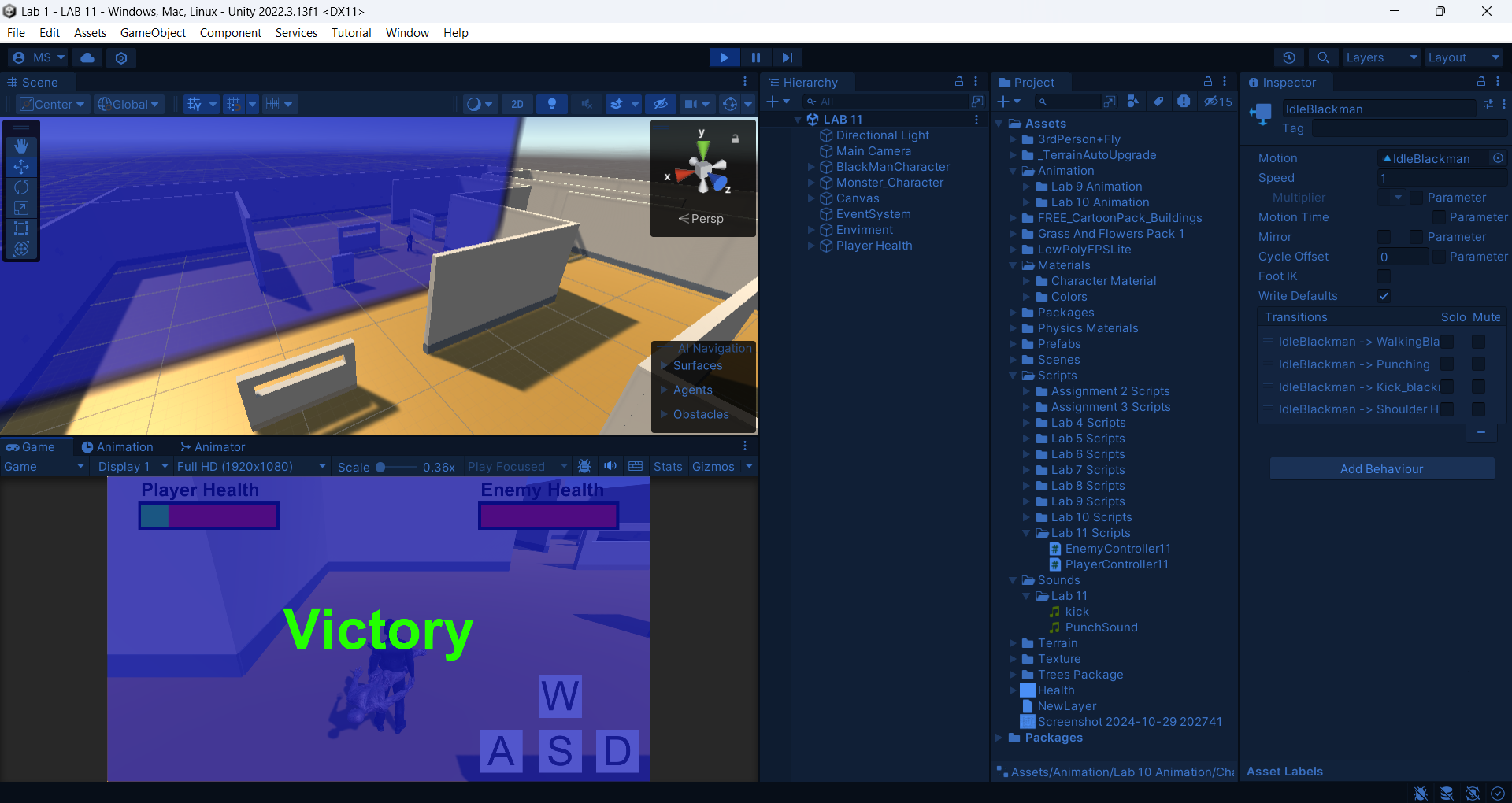
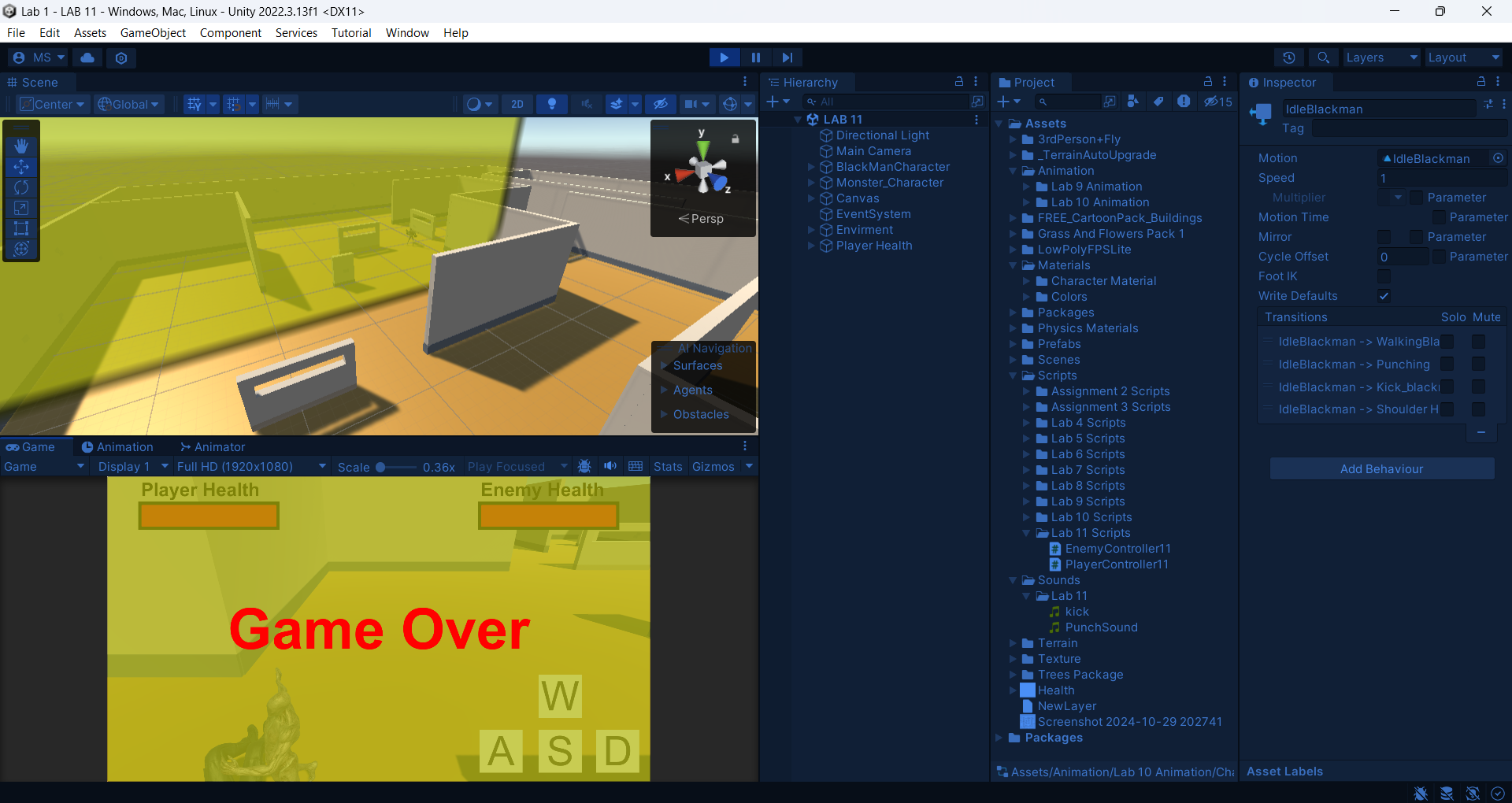
**Player Animator Controller:**

****

**Enemy Animator Controller:**

****

****



**Challenges:**

Debugging NavMesh baking issues for enemy pathfinding.

Syncing animation events with sound effects.

**Conclusion:**

All tasks were completed successfully. The modular delegate system and clean Animator setup allow for easy future expansions.

Attachments:

Screenshots of the dojo scene, UI, and health system.