НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки (ОТ)

Звіт до лабораторної роботи №2

з дисципліни:

«Розробка ігрових застосувань. Unity рішення»

на тему «Дослідження базового патерну ігрового рушія Unity на прикладі тривимірного ігрового застосунку»

Перевірив:
доцент каф. ІСТ
Катін П. Ю.

Виконав:
студент 4 курсу, гр. ІП-93
Говоруха М.А. (варіант 2)

2022р.

1.1 Завдання

Мета роботи: полягає у набутті знань, умінь та навичок з технології розробленняоснов проекту з використанням обраної мови програмування у обраній парадигмі.

Надається досвід створення репозиторію у системі контролю версій і знання елементівсередовища розробки і основи вихідного коду для управління грою .

Також лабораторна робота дає основні навички розробки з використанням IDEігрового рушія.

Вхідні дані ЛР 2.

Прізвище студента; ім’я студента; шифр навчальної групи; скорочена назва факультету; скорочена назва університета. Порядковий номер у списку, що визначає варіант.

Вхідні дані ЛР 1.

Репозиторій на GitHab з проектом. У окремому файлі вказана вся первинна інформація, що обговорена у вхідних даних. На даному етапі репозиторій не є обов'язковим. Дозволяється тримати проект локально. У проекті реалізовані всі вимоги відповідно до завдання і варіантів. Проект має запускатися на машині студента і викладача. У разі наявності помилок проект не зараховується.

Завдання

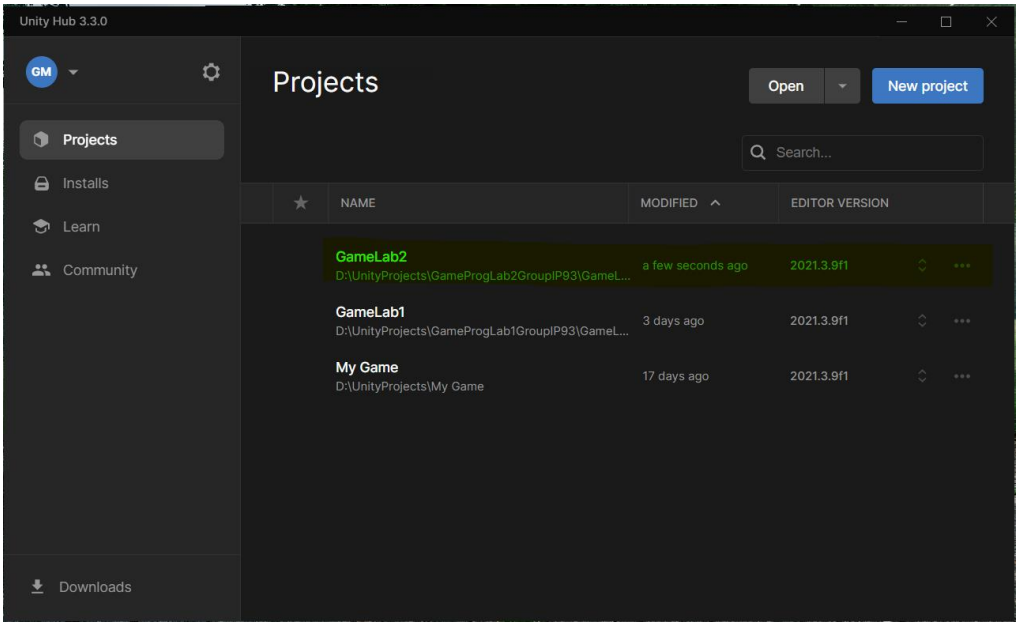
Створений проект IDE (3D) на основі рушія, що містить 2 сцени, ігровий персонаж. Можуть бути включені інші елементи. Розроблений і налагоджений скрипт для управління ігровим персонажем. Додані умови переходу між сценами. В якості ігрового персонажа використовується звичайний примітив. Достатньо продемонструвати рух ліворуч, праворуч, стрибки, коректну фізику, зупинку перед перешкодою, набір балів і перехід з одного рівня наінший. Можливо, за бажанням реалізувати інтерфейс гравця.

Проект розташовано у репозиторій на GitHab, основна мета полягає у дослідженні і підтвердженні володіння обраною IDE (3D) і технологією розподіленої системи контролю версій. У разі виконання всіх умов і відмінного захисту надається 12,5 балів. Разом 25 балів.

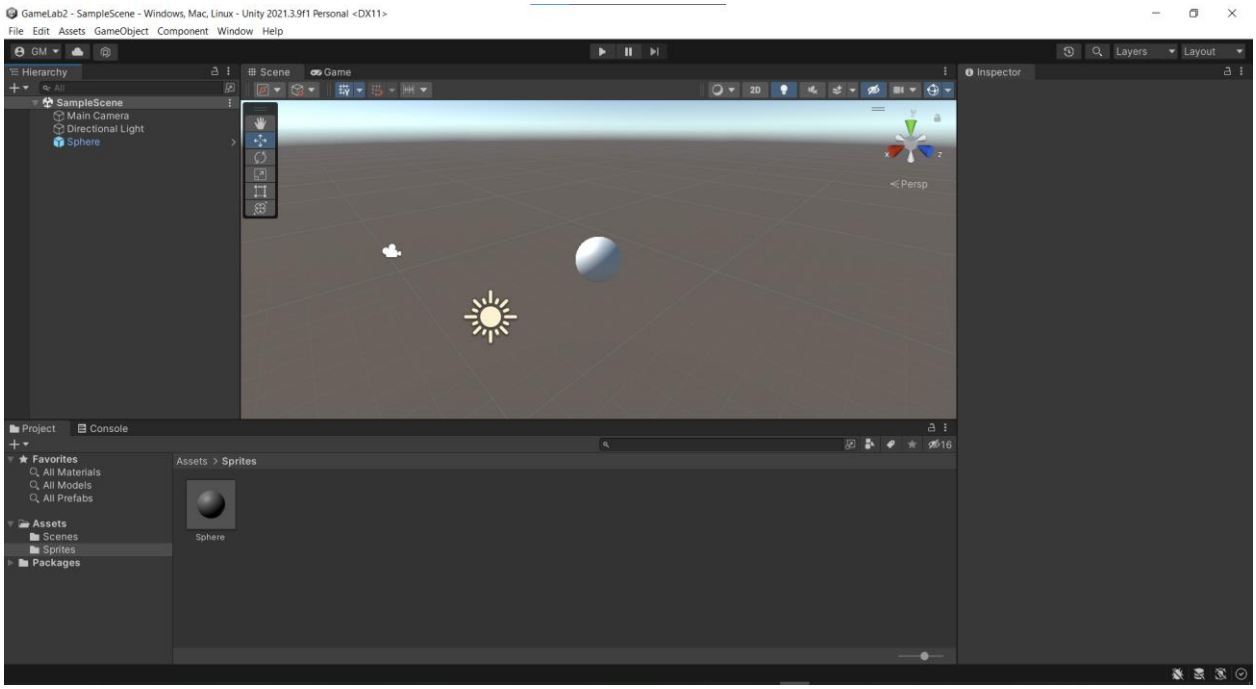
1.2 Хід виконання

Варіант 2: примітив – сфера, асет - [Basic Bedroom Starterpack | 3D Props | Unity Asset Store]

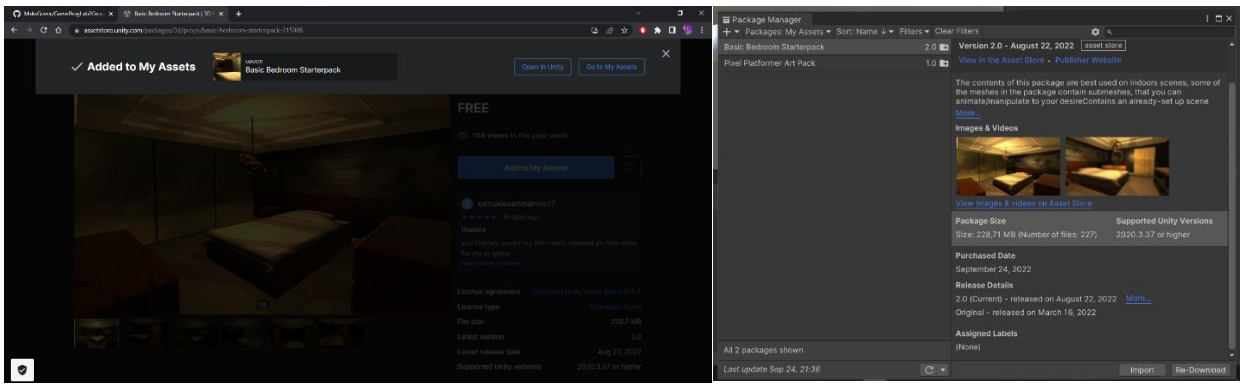
- Було створено 3D проект за домогою UnityHub з назвою GameLab2, котрий поміщено у репозиторій GameProgLab2GroupIP93:



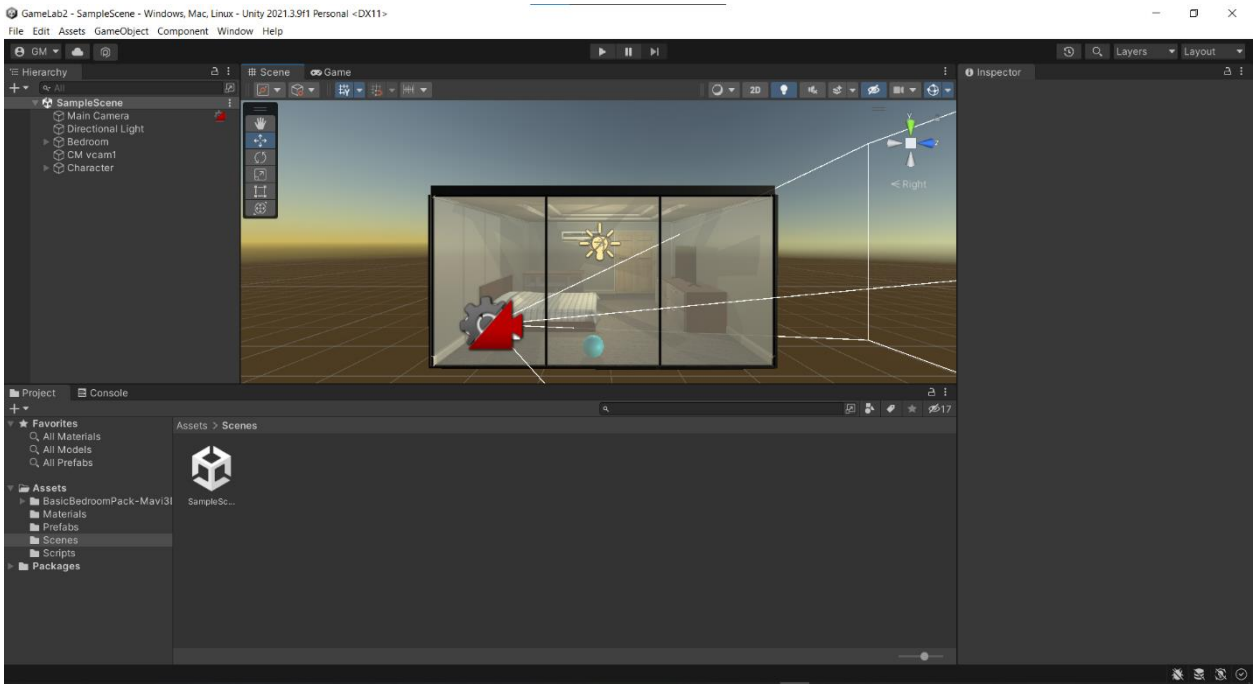
- Наступним кроком було створено примітив відповідно варіанту:

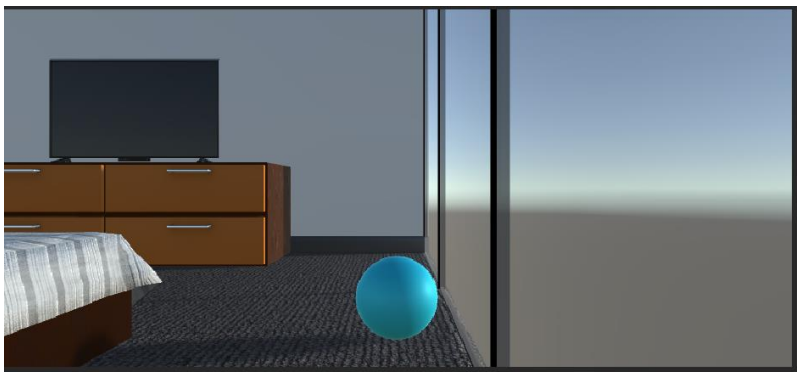


- Далі було обрано та завантажено відповідний до варіанту набір асетів:

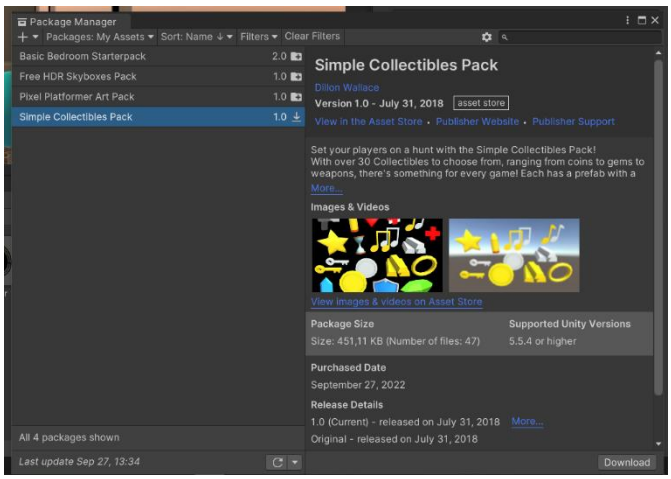


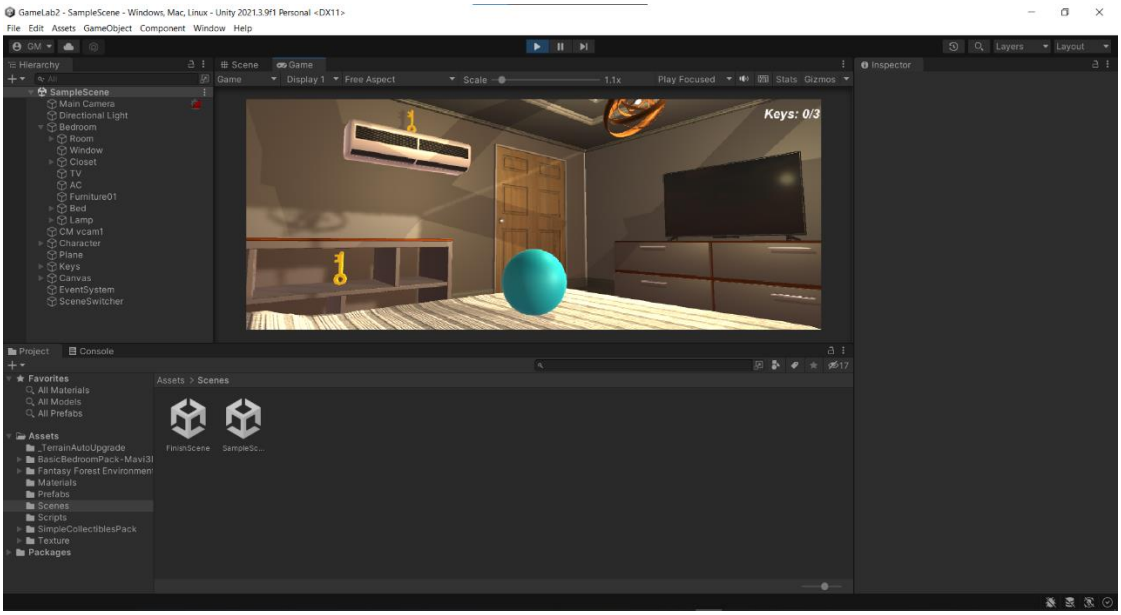
- Після здійснених вищенаведених підготовчих дій, було імпортовано набір асетів, та обрано з них префаби для конструювання мапи гри, також змінено його колір ігрового об’єкту та додано камеру від першої особи за допомогою Cinemachine, результат наведено нижче:



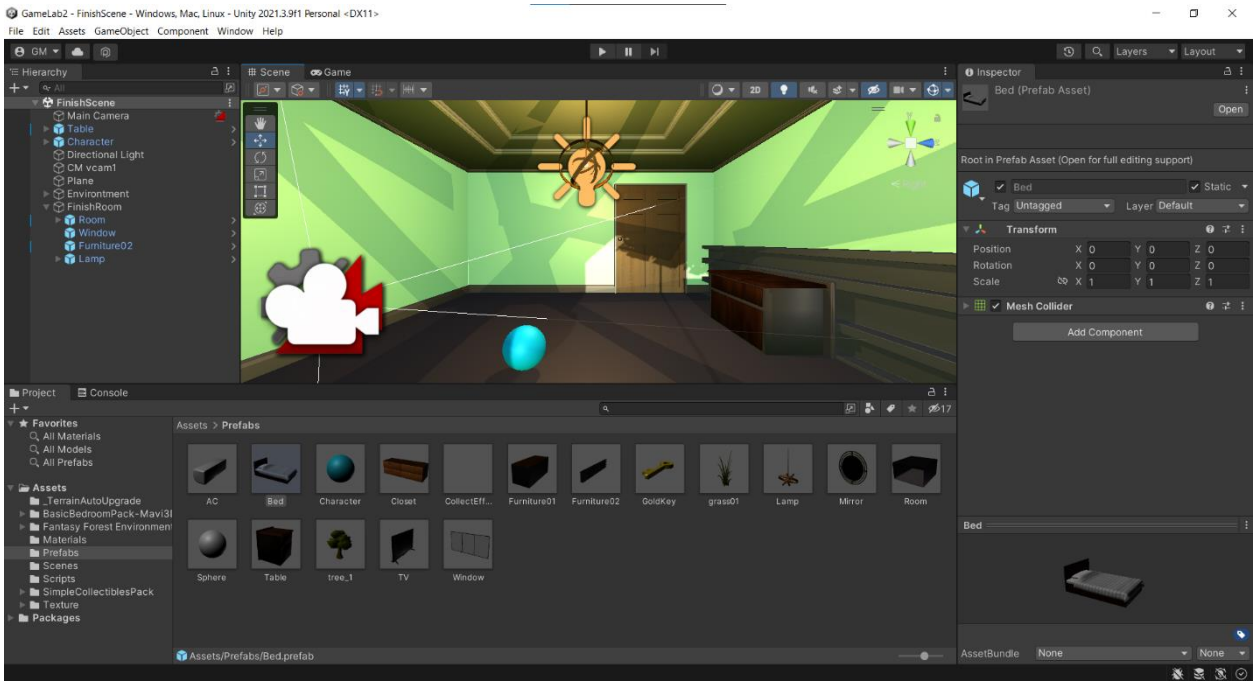


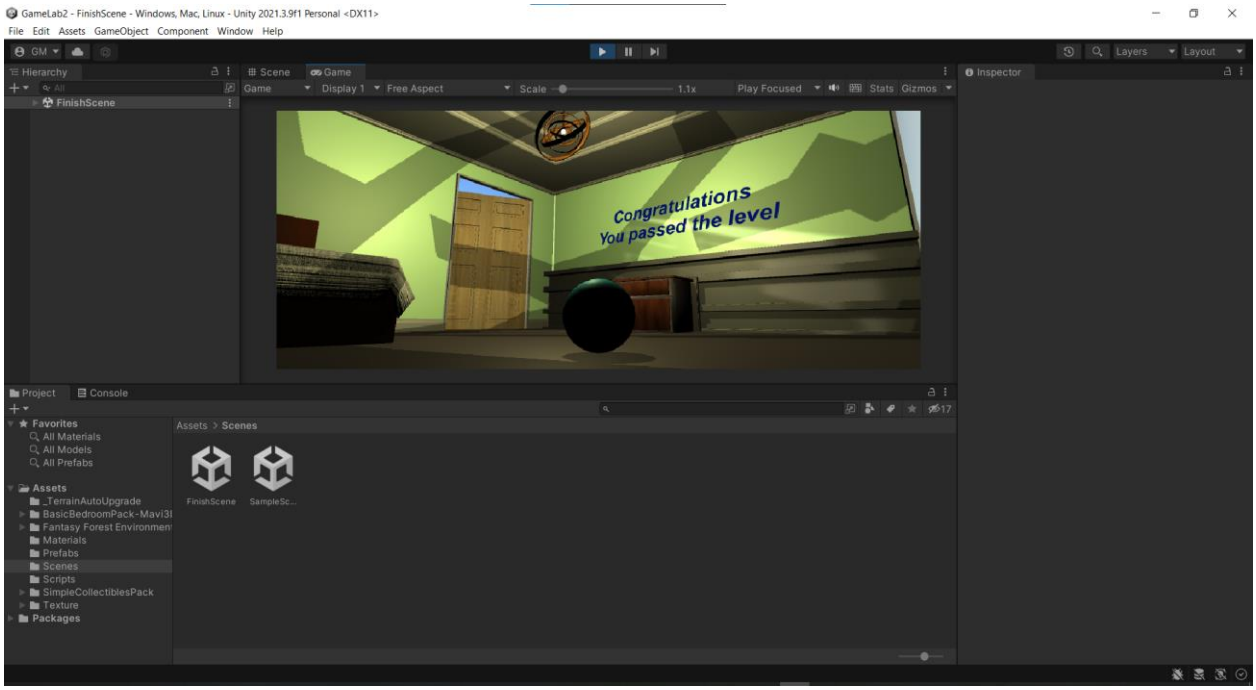
- Для набору балів і переходу на новий рівень було додано новий асет з «ключем», та додана логіка збору ключів і набору балів:





- Далі було створено нову сцену, і додано скрипт, що має метод переходу на нову сцену:





Скрипти проеку:

- CameraMove.cs:

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

public class CameraMove : MonoBehaviour
{
    public Transform cameraTarget;
    public Transform model;
    public float minAngle;
    public float maxAngle;
    public float mouseSensitivity;
    public bool staticCamera;

    void Start()
    {
        Cursor.lockState = CursorLockMode.Locked;
        Cursor.visible = false;
    }

    void Update()
    {
        cameraTarget.rotation *= Quaternion.AngleAxis(Input.GetAxis("Mouse X") *
mouseSensitivity, Vector3.up);
        cameraTarget.rotation *= Quaternion.AngleAxis(-Input.GetAxis("Mouse Y") *
mouseSensitivity, Vector3.right);

        float angleX = cameraTarget.localEulerAngles.x;
        if (angleX > 180 && angleX < maxAngle)
        {
            angleX = maxAngle;
        }
        else if (angleX < 180 && angleX > minAngle)
        {
            angleX = minAngle;
        }

        cameraTarget.localEulerAngles = new Vector3(angleX, cameraTarget.localEulerAngles.y,
0);
        if (staticCamera)
```

```
        {
            model.rotation = Quaternion.Euler(0, cameraTarget.rotation.eulerAngles.y, 0);
        }
    }
}
```

- CharacterMove.cs

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

public class CharacterMove : MonoBehaviour
{
    public Transform model;
    private Transform mainCamera;
    public Rigidbody rb;
    public float speed = 3f;
    private bool isGrounded;

    void Start()
    {
        rb = GetComponent<Rigidbody>();
        mainCamera = Camera.main.transform;
    }

    private void OnCollisionStay(Collision collision)
    {
        isGrounded = true;
    }

    void FixedUpdate()
    {
        Vector3 forwardCam = new Vector3(mainCamera.forward.x, 0f, mainCamera.forward.z);
        Vector3 rightCam = new Vector3(mainCamera.right.x, 0f, mainCamera.right.z);
        Vector3 movingVector = Input.GetAxis("Horizontal") * rightCam.normalized +
Input.GetAxis("Vertical") * forwardCam.normalized;

        if (!mainCamera.GetComponent<CameraMove>().staticCamera && movingVector.magnitude >=
0.2f)
        {
            model.rotation = Quaternion.LookRotation(forwardCam, Vector3.up);
        }

        if (Input.GetButton("Horizontal") || Input.GetButton("Vertical"))
        {
            rb.AddForce(movingVector * speed, ForceMode.VelocityChange);
        }

        if (Input.GetKey(KeyCode.Space) && isGrounded)
        {
            rb.AddForce(new Vector3(0, 100, 0));
            isGrounded = false;
        }
    }
}
```

- KeyCollect.cs

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

public class KeysCollect : MonoBehaviour
{
    public bool rotate = true; // do you want it to rotate?
    public float rotationSpeed = 50;
    public GameObject collectEffect;
    private ScoreCounter scoreCounter;
```

```
    void Start()
    {
        scoreCounter = FindObjectOfType<ScoreCounter>();
    }

    // Update is called once per frame
    void Update()
    {
        if (rotate)
            transform.Rotate(Vector3.up * rotationSpeed * Time.deltaTime, Space.World);
    }

    void OnTriggerEnter(Collider other)
    {
        if (other.tag == "Player")
        {
            CollectKey();
        }
    }

    public void CollectKey()
    {
        if (collectEffect)
        {
            Instantiate(collectEffect, transform.position, Quaternion.identity);
        }

        scoreCounter.IncScore();
        Destroy(gameObject);
    }
}
```

- ScoreCounter.cs

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

public class ScoreCounter : MonoBehaviour
{

    private int score;
    public int keysCount = 3;
    public Text scoreView;
    private SceneSwitcher sceneSwitcher;

    void Start()
    {
        sceneSwitcher = FindObjectOfType<SceneSwitcher>();
    }

    // Update is called once per frame
    void Update()
    {
        scoreView.text = $"Keys: {score}/{keysCount}";
    }

    public void IncScore()
    {
        score++;
        if (score >= keysCount)
        {
            sceneSwitcher.Switch();
        }
    }
}
```

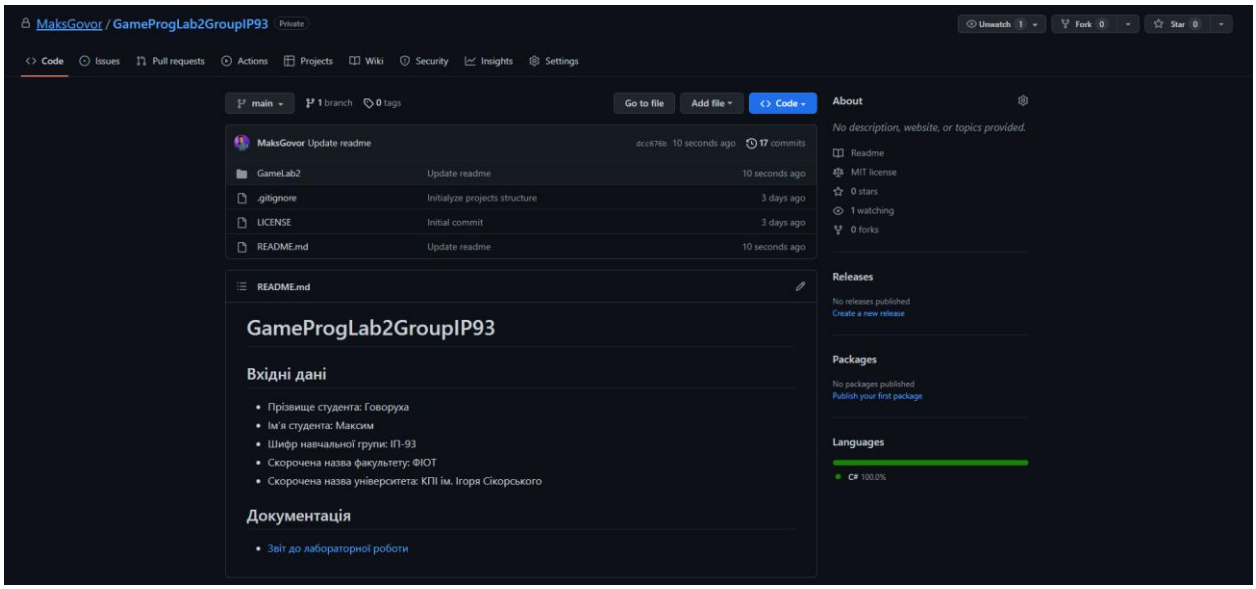
- SceneSwitcher.cs

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

public class SceneSwitcher : MonoBehaviour
{
    public string sceneName;

    public void Switch()
    {
        SceneManager.LoadScene(sceneName);
    }
}
```

- Під проект відповідно створено віддалений репозиторій GitHub і відповідно до вимог оформлено основну інформацію в Readme.md:



1.3 Висновки

За період виконання даної роботи я набув базових навичок розробки з використанням IDE ігрового рушія Unity, шляхом створення проекту (3D) на основі рушія, що містить 2 сцени та ігрового персонажа. Ігровий персонаж згідно завдання було створено як примітив, а інші об’єкти потрібно було взяти з assetstore, тому для цього було попередньо створено обліковий запис юніті за допомогою якого здійснено завантаження обраного асету. Також в ході виконання завдання я познайомився з такими базовими поняттями як додавання скрипта до ігрового об’єкту, управління камерою в 3D середовищі, класами BoxCollaider, MeshCollaider, Rigidbody та їх основними призначеннями і налаштуваннями. Окремим кроком було розроблено скрипти, що відповідають за основні рухи, стрибки ігрового об’єкта, переміщення камери, набір балів, переходу на нову сцену. Під час розробки використовувалась система контролю версій git та і основні зміни фіксувались відповідними комітами, також проект було опубліковано в віддалений репозиторій GitHub.