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

Факультет інформатики та обчислювальної
техніки Кафедра обчислювальної техніки
(ОТ)

Звіт до лабораторної роботи №1

з дисципліни:

«Розробка ігрових застосувань. Unity рішення»

на тему «Дослідження базового патерну ігрового рушія Unity на прикладі двовимірної технології»

Перевірив:
доцент каф. ІСТ
Катін П. Ю.

Виконав:
студент 4 курсу, гр. ІТ-91
Левак О.О. (варіант 1)

2022 р.

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.

Також лабораторна робота дає основні навички розробки з використанням IDE ігрового рушія. Дається можливість роботи з іншим типом IDE за вибором студента та поузгодженню з викладачем.

Для підготовки ЛР № 1 може бути використаний будь-який сучасний ПК на основі мікропроцесора AMD64 (Intel® 64) або ARM. Операційна система Windows,

Linux абоMac OS (Macintosh Operating System).

Вхідні дані ЛР 1.

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 обовязковим. Дозволяється тримати проект локально. У проекті реалізовані всі вимоги відповідно до завдання і варіантів. Проект має запускатися на машині студента і викладача. У разі наявностіпомилок проект не зараховується.

Завдання

Репозиторій у системі контролю версій. Створити проект 2D. Загальні вимоги.

Акаунт на GitHab, на даному етапі за бажанням. Репозиторій на GitHab з проектом.

Назва GameProgLab1Group**Num**, де зафарбовано номер групи.

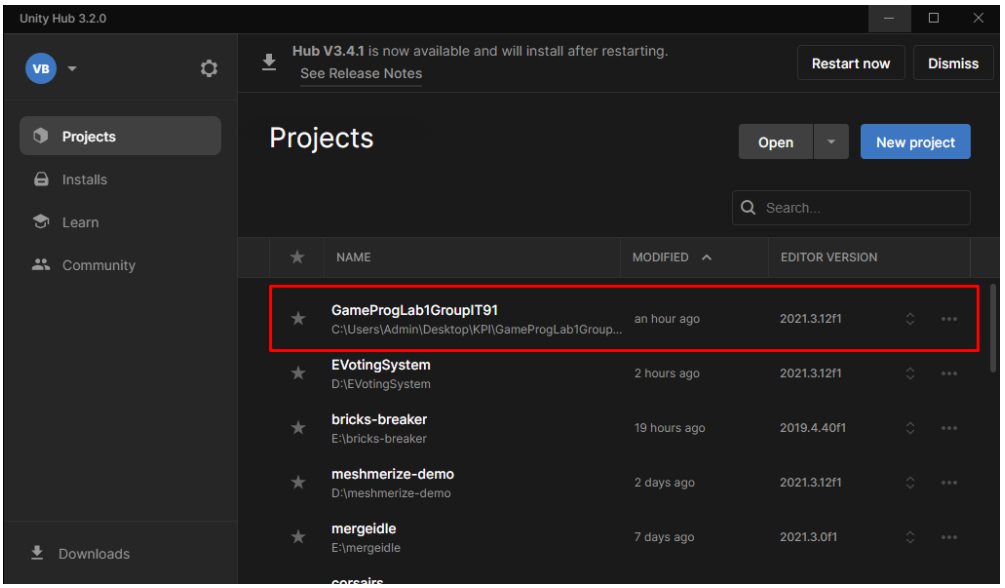
Установка ігрового рушія. Створений проект IDE (2D) на основі рушія, що містить

1 сцену, ігровий персонаж. Можуть бути включені інші елементи. Розроблений і налагоджений скрипт для управління ігровим персонажем. Достатньо продемонструвати рух ліворуч, праворуч, стрибки, коректну фізику, зупинку перед перешкодою. ***Проект розташовано у репозиторій на GitHab***, основна мета полягає у дослідженні і підтвердженні володіння обраною IDE (2D) і технологією розподіленої системи контролю версій.

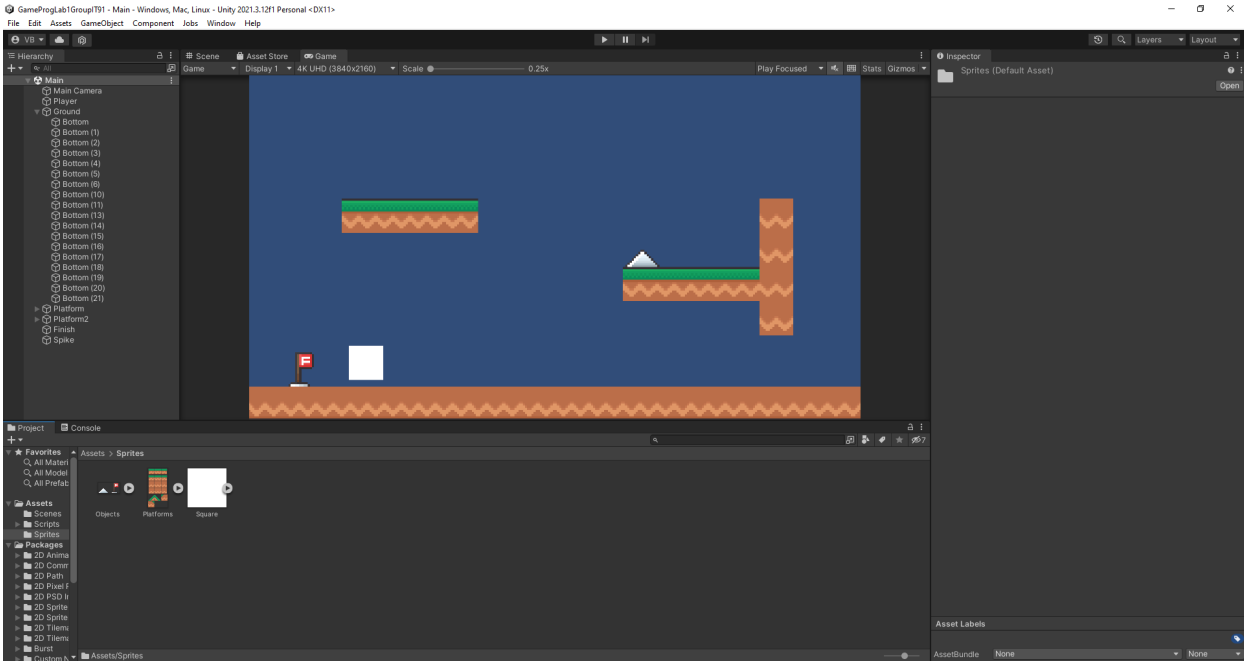
1.2 Хід виконання

Варіант 1: примітив – квадрат, асет - https://assetstore.unity.com/packages/2d/characters/simple-2d-platformer-assets-pack-188518

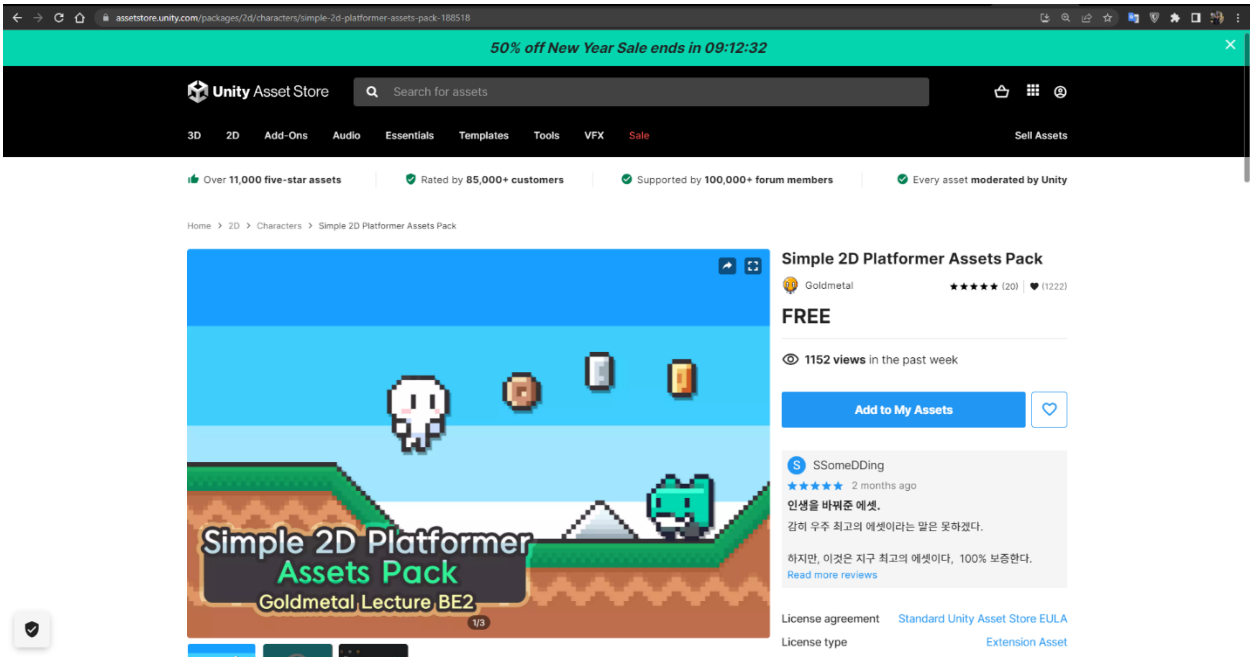
- Було створено 2D проект за домогою UnityHub з назвою GameProgLab1GroupIT91, котрий поміщено у репозиторій GameProgLab1Group**IT91:**



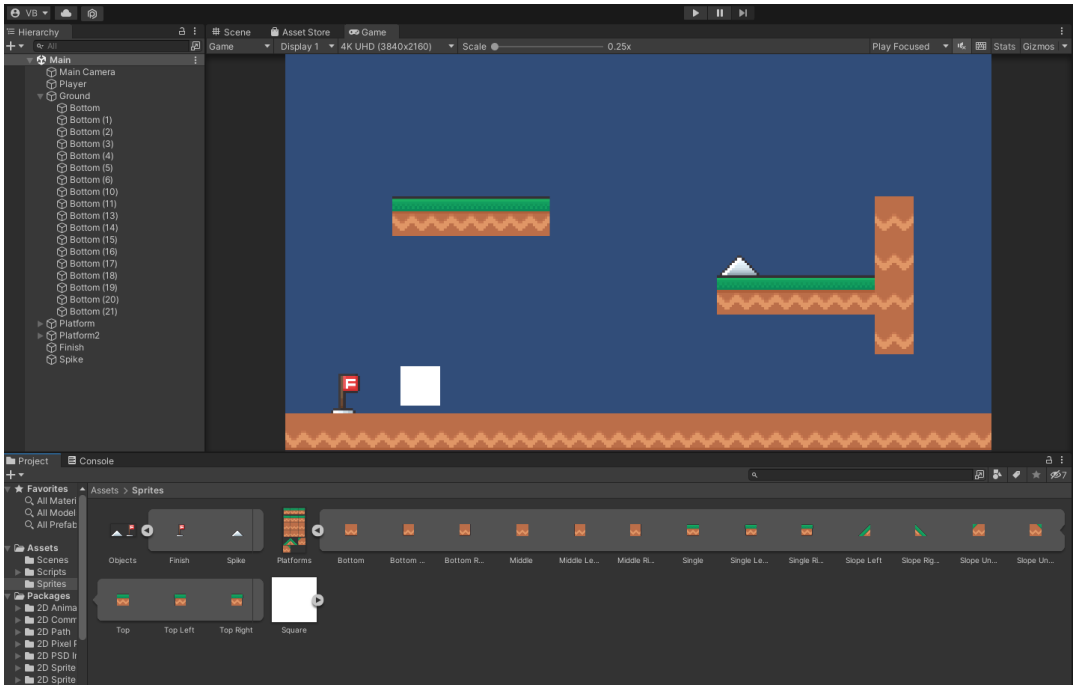
- Далі було створено каталог для спрайтів, і в ньому відповідно створено спрайт з примітивів згідно з варіантом завдання:



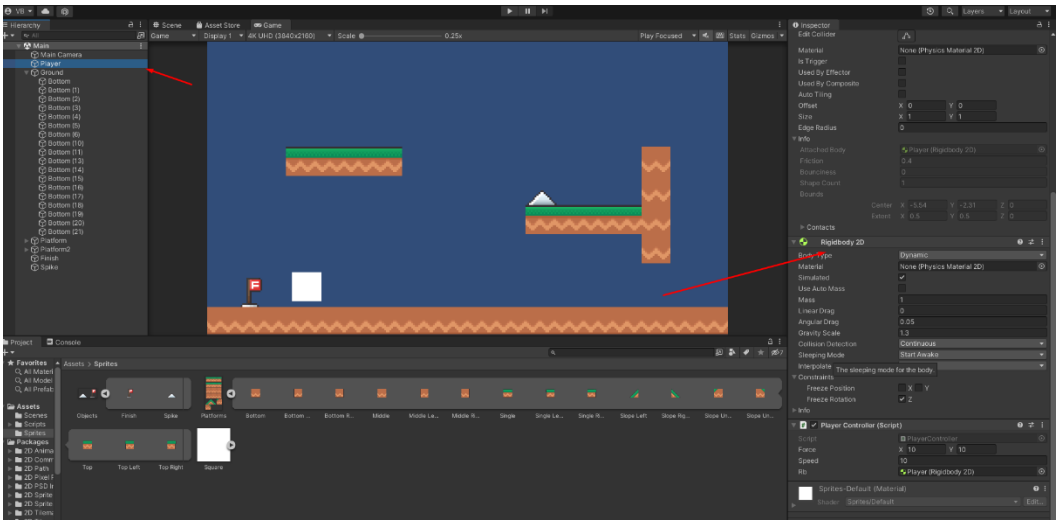
- Наступним кроком було обрано та завантажено відповідний до варіанту набір асетів:



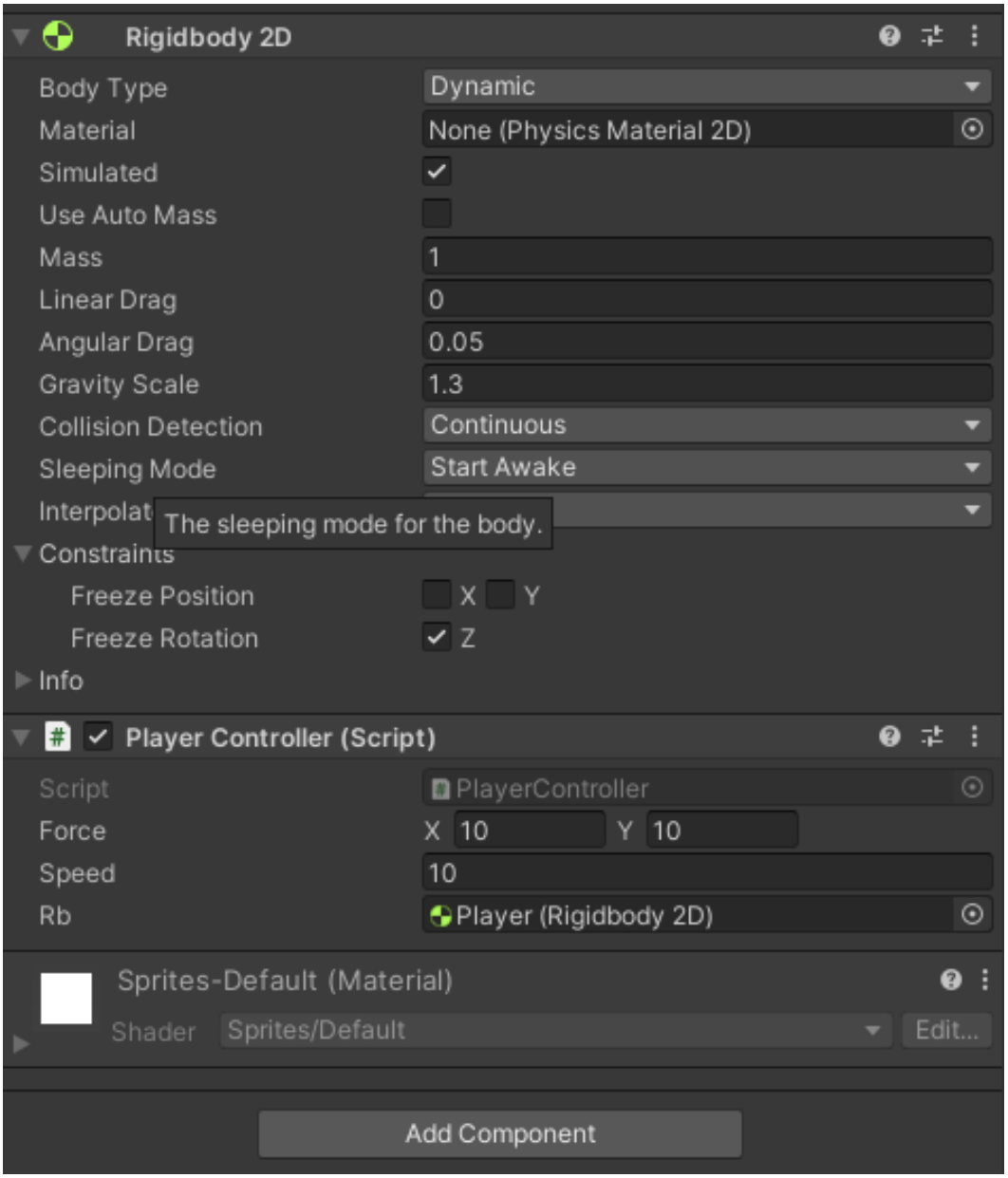
- Після здійснених вищенаведених підготовчих дій, було імпортовано набір асетів, та обрано з них спрайти для конструювання платформи гри, також було додано примітив, результат наведено нижче:



- До всіх об’єктів платформи було додано Box колайдер, а до ігрового об’єкту у результаті проведених порівнянь було прийнято рішення додати Box колайдер, до декорацій (прапор) колайдерів додавати не потрібно, так як вони не взаємодіють з ігровим об’єктом:

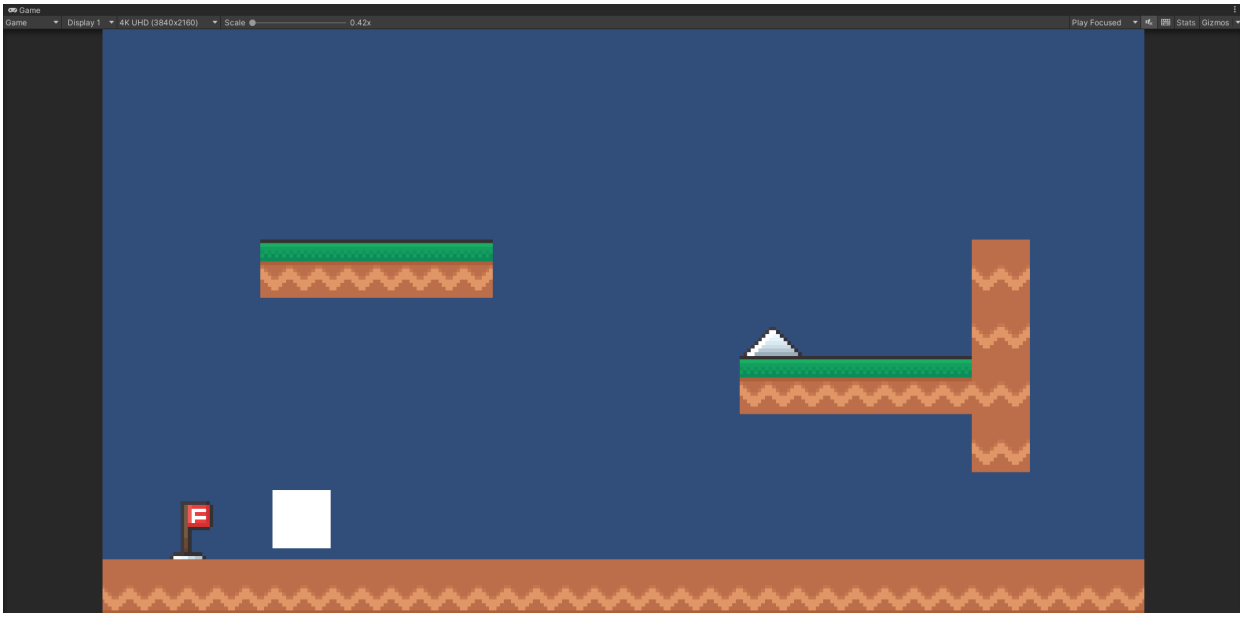


- Наступним кроком було розроблено базовий скрипт, який відповідає за рухи, стрибки, та умову коли ігровий об’єкт покидає ігрову зону, вихідних код скрипту наведено нижче:

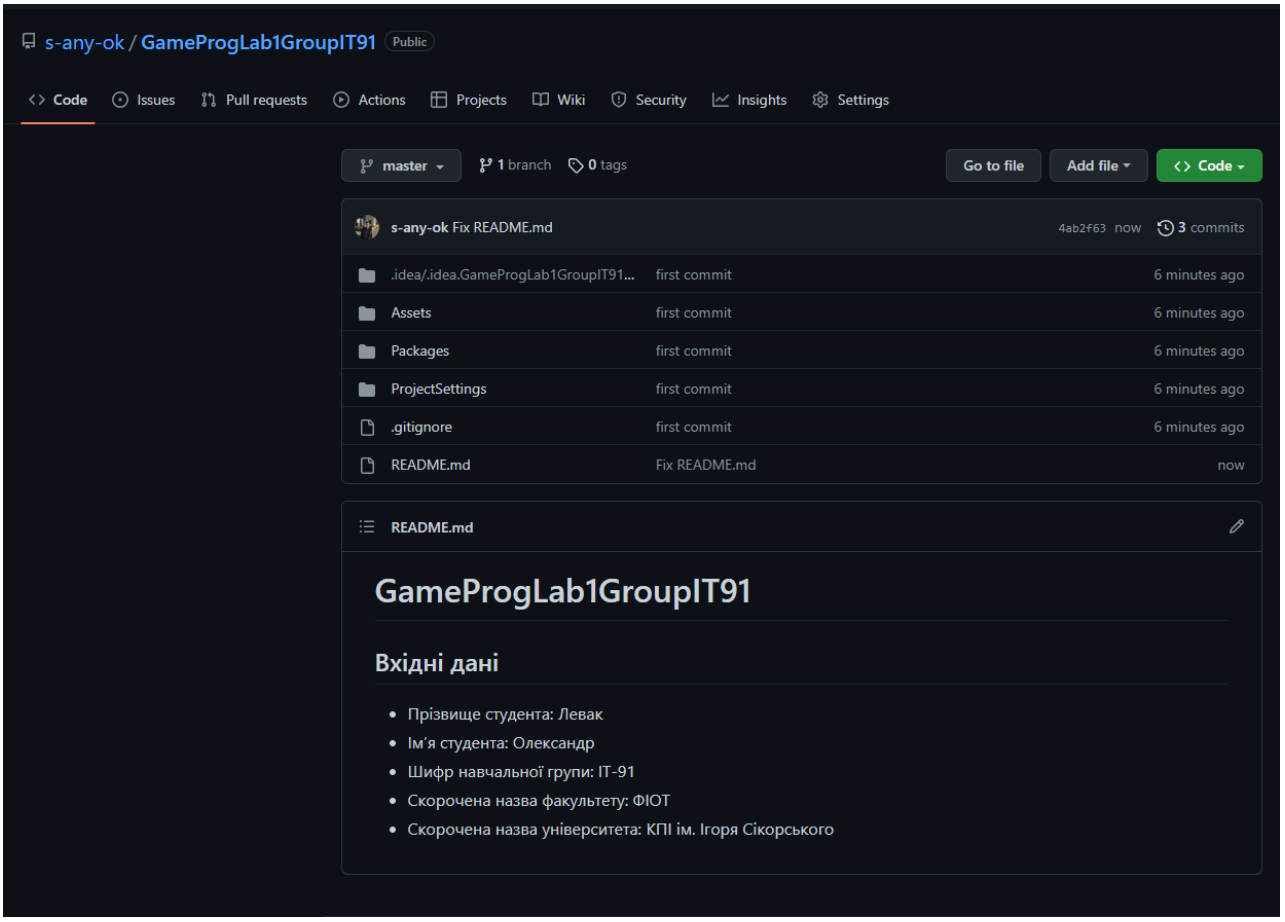


```
using System;
using UnityEngine;

namespace Player.Controller
{
  public class PlayerController : MonoBehaviour
  {
    [SerializeField] private Vector2 _force = new(10, 10);
    [SerializeField] private float _speed = 10f;
    [SerializeField] private Rigidbody2D _rb;

    private bool _isInAir;
    private Vector3 _initialPosition;

    private void Start()
    {
      _initialPosition = transform.position;
    }
    private void Update()
    {
      if (Input.GetButton("Horizontal"))
      {
        MoveHorizontal();
      }

      if (Input.GetKey(KeyCode.Space) && !_isInAir)
      {
        Jump();
      }
      CheckOutOfMap();
    }
    private void OnCollisionEnter2D(Collision2D collision)
    {
      if (collision.gameObject.layer == LayerMask.NameToLayer("Ground"))
      {
        _isInAir = false;
      }
    }
    private void MoveHorizontal()
    {
      Vector3 horizontalMovement = transform.right * Input.GetAxis("Horizontal");
      transform.position =
        Vector3.MoveTowards(transform.position, transform.position + horizontalMovement, _speed *
Time.deltaTime);
    }

    private void Jump()
    {
      _isInAir = true;
      _rb.AddForce(transform.up * _force, ForceMode2D.Impulse);
    }

    private void CheckOutOfMap()
    {
      if (Math.Abs(transform.position.x) > 9 || transform.position.y < -4)
      {
        transform.position = _initialPosition;
      }
    }

  }
}
```

- Тепер все готово для запуску, результат запуску наведено нижче:



- 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 (2D) на основі рушія, що містить 1 сцену 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 познайомивсь з такими базовими поняттями як спрайти та як їх додавати до сцени, класами BoxCollaider2d, CircleCollaider2d, Rigidbody2d та їх основними призначеннями і налаштуваннями. Окремим кроком було розроблено скрипт, що відповідає за основні рухи, стрибки та додано до ігрового об’єкта. Під час розробки використовувалась система контролю версій git та і основні зміни фіксувались відповідними комітами, також проект було опубліковано в віддалений репозиторій Github.