Assets\Scripts\PlayerRigidbody.cs

```
using UnityEngine;

[RequireComponent(typeof(Rigidbody))]
public class PlayerRigidbody : MonoBehaviour
{
    [SerializeField] private float _moveSpeed = 5f;
    [SerializeField] private float _jumpForce = 5f;
    [SerializeField] private float _doubleJumpForce = 5f;
    [SerializeField] private float _speedRotation = 10f;

    private Rigidbody _rb;
    private Vector2 _input;
    private Camera _followCam;
    private bool _isGrounded = true;
    private bool _canDoubleJump = false;

    void Start()
    {
        _rb = GetComponent<Rigidbody>();
        _followCam = Camera.main;
    }

    void Update()
    {
        Jump();
    }

    private void FixedUpdate()
    {
        _input = new Vector2(Input.GetAxis("Horizontal"), Input.GetAxis("Vertical"));

        Vector3 _moveDirection = new Vector3(-_input.x, 0f, -_input.y);
        _rb.MovePosition(_rb.position + (_moveDirection * _moveSpeed * Time.fixedDeltaTime));

        Vector3 movementInput = Quaternion.Euler(0, _followCam.transform.eulerAngles.y, 0) * new Vector3(_input.x, 0, _input.y);
        Vector3 movementDirection = movementInput.normalized;

        if (movementDirection != Vector3.zero)
        {
            Quaternion desiredRotation = Quaternion.LookRotation(movementDirection, Vector3.up);
            transform.rotation = Quaternion.Slerp(transform.rotation, desiredRotation, _speedRotation * Time.deltaTime);
        }

        _rb.AddForce(movementDirection * _moveSpeed * Time.deltaTime);
    }

    void Jump()
    {
        if (Input.GetButtonDown("Jump"))
        {
            if (_isGrounded)
            {
                _rb.AddForce(transform.up * _jumpForce, ForceMode.Impulse);
                _isGrounded = false;
                _canDoubleJump = true;
            }
            else if (_canDoubleJump)
            {
                _rb.velocity = new Vector3(_rb.velocity.x, 0f, _rb.velocity.z);
                _rb.AddForce(transform.up * _doubleJumpForce, ForceMode.Impulse);
                _canDoubleJump = false;
            }
        }
    }

    private void OnCollisionEnter(Collision collision)
    {
        if (collision.gameObject.CompareTag("Ground"))
        {
            _isGrounded = true;
        }
    }
}

```