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

public class DogMovement : MonoBehaviour
{
    [SerializeField]
    float forwardSpeed;

    //[SerializeField]
    //float backwardSpeed;

    //[SerializeField]
    //float strafeSpeed;

    [SerializeField]
    float sprintMultiplier = 2f;

    Rigidbody rigBod;

    bool isMoving = false;

    [SerializeField]
    float sprintTime = 5f;

    float sprintTimer;

    [SerializeField]
    float sprintCooldownTime = 15f;

    [SerializeField]
    float rayLength = 1f;

    float sprintCooldownTimer;

    Vector3 DownwardForce = new Vector3(0, -10, 0);

    bool isGrounded;


    // Start is called before the first frame update
    void Start()
    {
        rigBod = GetComponent<Rigidbody>();

        sprintTimer = sprintTime;
        sprintCooldownTimer = sprintCooldownTime;
    }

    // Update is called once per frame
    void Update()
    {
        followCursor();

        //Check to see if the Dog should be moving
        if (Input.GetKeyUp(KeyCode.W))
        {
            rigBod.velocity = Vector3.zero;
        }

        isGrounded = GroundCheck();
        //Debug.Log(isGrounded);

    }

    private void FixedUpdate()
    {

        //Move Forward
        if (Input.GetKey(KeyCode.W))
        {
            moveForward(forwardSpeed);
            isMoving = true;
        }
        //Sprint
        if (Input.GetKey(KeyCode.W) && Input.GetKey(KeyCode.LeftShift) && 
          sprintTimer > 0f)
        {
            sprint(forwardSpeed);
            sprintTimer -= Time.deltaTime;
        }

        if (!Input.GetKey(KeyCode.LeftShift) && sprintTimer < sprintTime)
        {
            sprintCooldownTimer -= Time.deltaTime;
        }

        if (sprintCooldownTimer <= 0f)
        {
            sprintTimer = sprintTime;
            sprintCooldownTimer = sprintCooldownTime;
        }

        ////Move Backward
        //if (Input.GetKey(KeyCode.S))
        //{
        //    moveBackward(backwardSpeed);
        //    isMoving = true;
        //}

        ////Strafe Left
        //if (Input.GetKey(KeyCode.A))
        //{
        //    strafeLeft(strafeSpeed);
        //    isMoving = true;
        //}

        ////Strafe Right
        //if (Input.GetKey(KeyCode.D))
        //{
        //    strafeRight(strafeSpeed);
        //    isMoving = true;
        //}

        // Stop the rigidbody from moving if no buttons are pressed
        if (!isMoving)
        {
            rigBod.velocity = Vector3.zero;
        }
        if (!isGrounded)
        {
            Vector3 newVelocity = new Vector3(rigBod.velocity.x,
              rigBod.velocity.y - 2f, rigBod.velocity.z);
            rigBod.velocity = newVelocity;
        }

    }

    /** Make Dag follow the player's cursor*/
    void followCursor()
    {
        //Get mouse position
        Vector3 mousePos = Input.mousePosition;

        //Convert mouse position to a point in the world
        Ray ray = Camera.main.ScreenPointToRay(mousePos);
        RaycastHit hit;

        if (Physics.Raycast(ray, out hit))
        {
            //Set targetPos to mouse position point in world
            Vector3 targetPos = hit.point;

            // Rotate Dag
            transform.LookAt(targetPos);

```

```
                //Block x and z rotations
                transform.eulerAngles = new Vector3(0, transform.eulerAngles.y, 0);
            }
        }

        void moveForward(float forwardSpeed)
        {
            rigBod.velocity = (transform.forward - (.1f * transform.up)) * forwardSpeed;
        }

        void moveBackward(float backwardSpeed)
        {
            rigBod.velocity = -transform.forward * backwardSpeed;
        }

        void strafeLeft(float strafeSpeed)
        {
            Vector3 left = new Vector3(-1, 0, 0);
            rigBod.velocity = left * strafeSpeed;
        }

        void strafeRight(float strafeSpeed)
        {
            Vector3 right = new Vector3(1, 0, 0);
            rigBod.velocity = right * strafeSpeed;
        }

        void sprint(float forwardSpeed)
        {
            rigBod.velocity = transform.forward * forwardSpeed * sprintMultiplier;
        }

        bool GroundCheck()
        {
            RaycastHit hit;

            if (Physics.Raycast(transform.position, Vector3.down, out hit, rayLength))
            {
                return true;
            }
            Debug.Log(transform.position.y);
            return false;
        }
    }

```