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

public class PlayerAttack : PlayerAnimation
{
    //itens
    public GameObject[] weapon;

    //particles
    [Header ("Particulas")]
    public GameObject[] particle;


    [Header("SkillsPrefabs")]
    public GameObject powerPrefab;
    public GameObject earthShatterPrefab;

    [HideInInspector]
    public bool canAttack;

    private bool nextAttack = true;
    private bool rangeAttack = true;

    public AvatarMask[] attackMask;

    private void Awake()
    {
        canAttack = true;
    }

    void Update ()
    {
        Debug.Log(canAttack);

        if (canAttack)
        {
            InputAttacks();
            ParticleSystem();
            ActivateWeapon();
        }

    }


    private void InputAttacks()
    {
        Attack(1, Input.GetButton("Fire1") && nextAttack);
        Attack(2, Input.GetButton("Fire2") && nextAttack);

        Skill(1, Input.GetKey(KeyCode.Q) && nextAttack);
        Skill(2, Input.GetKey(KeyCode.E) && nextAttack);
        Skill(3, Input.GetKey(KeyCode.R) && nextAttack);

        if (Input.GetButton("Fire1") && nextAttack)
        {

            StartCoroutine(OnCompleteAttackAnimation(0.5f));
        }
        else if(Input.GetButton("Fire2") && nextAttack)
        {
            StartCoroutine(OnCompleteAttackAnimation(0.5f));
        }
        else if (Input.GetKey(KeyCode.Q) && nextAttack)
        {
            rangeAttack = true;
            StartCoroutine(OnCompleteAttackAnimation(0.7f));
        }
        else if (Input.GetKey(KeyCode.E) && nextAttack)
        {
            rangeAttack = true;
            StartCoroutine(OnCompleteAttackAnimation(0.8f));
        }
        else if (Input.GetKey(KeyCode.R) && nextAttack)
        {
            StartCoroutine(OnCompleteAttackAnimation(0.5f));
        }

    }

    private void ParticleSystem()
    {
        particle[0].SetActive(isAttacking(1));
        particle[1].SetActive(isAttacking(2));
        particle[2].SetActive(isAttacking(5) && rangeAttack);
        particle[4].SetActive(isAttacking(7));

    }
    private void RangeAttack()
    {
        //Create the power from powerprefab
        var power = (GameObject)Instantiate(
            powerPrefab,
            particle[2].transform.position,
            particle[2].transform.rotation);

        //add velocity to the power
        power.GetComponent<Rigidbody>().velocity = this.transform.forward *
          10;

        //destroy the bullet after
          Destroy(power, 1f);
    }
    private void MeleeRangeAttack()
    {
        //Create the power from powerprefab
        var power = (GameObject)Instantiate(
            earthShatterPrefab,
            particle[3].transform.position,
            particle[3].transform.rotation);

        //add velocity to the power
        //power.GetComponent<Rigidbody>().velocity = this.transform.forward * 10;

        //destroy the bullet after
        Destroy(power, 1.5f);
    }

    private void ActivateWeapon()
    {
        weapon[0].SetActive(!isAttacking(0) && !isAttacking(5));
        weapon[1].SetActive(isAttacking(0) || isAttacking(5));

    }
    IEnumerator Rotate(float duration, bool isRotate)
    {
        if(isRotate)
        {
            Quaternion startRot = transform.rotation;
            float t = 0.0f;
            while (t < duration)
            {
                t += Time.deltaTime;
                transform.rotation = startRot * Quaternion.AngleAxis(t / duration * 360f, transform.up); //or transform.right if you want it to be locally based
                yield return null;
            }
            transform.rotation = startRot;
        }
    }

    IEnumerator OnCompleteAttackAnimation(float coldown)
    {
        nextAttack = false;
        yield return new WaitForSeconds(coldown);
        if(isAttacking(5) && rangeAttack)
        {
            RangeAttack();
            rangeAttack = false;
        }
        if (isAttacking(6) && rangeAttack)
        {
            MeleeRangeAttack();
            rangeAttack = false;
        }

        nextAttack = true;
    }
}

```