```
 1  using System.Collections;
 2  using System.Collections.Generic;
 3  using UnityEngine;
 4
 5  public class Turret : MonoBehaviour {
 6
 7      private Transform target;
 8      private Enemy targetEnemy;
 9
10      [Header("General")]
11      public float range = 15f;
12
13      [Header("Use Bullets (default)")]
14      public GameObject bulletPrefab;
15      public float fireRate = 1f;
16      private float fireCountdown = 0f;
17
18      [Header("Use Laser")]
19      public bool useLaser = false;
20
21      public int damageOverTime = 30;
22      public float slowAmount = .5f;
23
24      public LineRenderer lineRenderer;
25      public ParticleSystem impactEffect;
26      public Light impactLight;
27
28      [Header("Unity Setup Fields")]
29      public string enemyTag = "Enemy";
30      public Transform partToRotate;
31      public float turnSpeed = 10f;
32
33      public Transform firePoint;
34
35      // Use this for initialization
36      void Start () {
37          InvokeRepeating("UpdateTarget", 0f, 0.5f);
38      }
39
40      void UpdateTarget () {
41          GameObject[] enemies = GameObject.FindGameObjectsWithTag(enemyTag);
42          float shortestDistance = Mathf.Infinity;
43          GameObject nearestEnemy = null;
44
45          foreach (GameObject enemy in enemies) {
46              float distanceToEnemy = Vector3.Distance(transform.position,
                    enemy.transform.position);
47
48              if (distanceToEnemy < shortestDistance) {
49                  shortestDistance = distanceToEnemy;
50                  nearestEnemy = enemy;
51              }
52          }
```

```
53
54          if (nearestEnemy != null && shortestDistance <= range) {
55              target = nearestEnemy.transform;
56              targetEnemy = nearestEnemy.GetComponent<Enemy>();
57          }
58          else {
59              target = null;
60          }
61      }
62
63      // Update is called once per frame
64      void Update () {
65          if (target == null) {
66              if (useLaser) {
67                  if (lineRenderer.enabled) {
68                      lineRenderer.enabled = false;
69                      impactEffect.Stop();
70                      impactLight.enabled = false;
71                  }
72              }
73              return;
74          }
75
76          LockOnTarget();
77
78          if (useLaser) {
79              Laser();
80          }
81          else {
82              if (fireCountdown <= 0f)
83              {
84                  Shoot();
85                  fireCountdown = 1f / fireRate;
86              }
87
88              fireCountdown -= Time.deltaTime;
89          }
90      }
91
92      void LockOnTarget () {
93          // Target lock on
94          Vector3 dir = target.position - transform.position;
95          Quaternion lookRotation = Quaternion.LookRotation(dir);
96          Vector3 rotation = Quaternion.Lerp(partToRotate.rotation,
              lookRotation, Time.deltaTime * turnSpeed).eulerAngles;
97          partToRotate.rotation = Quaternion.Euler(0f, rotation.y, 0f);
98      }
99
100     void Laser () {
101         FindObjectOfType<AudioManager>().Play("LaserGun");
102
103         targetEnemy.TakeDamage(damageOverTime * Time.deltaTime);
104         targetEnemy.Slow(slowAmount);
```

```
105
106         if (!lineRenderer.enabled) {
107             lineRenderer.enabled = true;
108             impactEffect.Play();
109             impactLight.enabled = true;
110         }
111
112         // Laser starting point and ending point
113         lineRenderer.SetPosition(0, firePoint.position);
114         lineRenderer.SetPosition(1, target.position);
115
116         Vector3 dir = firePoint.position - target.position;
117
118         impactEffect.transform.position = target.position + dir.normalized;
119
120         impactEffect.transform.rotation = Quaternion.LookRotation(dir);
121     }
122
123     void Shoot () {
124         FindObjectOfType<AudioManager>().Play("Dukenukem");
125
126         GameObject bulletGO = (GameObject)Instantiate(bulletPrefab,
              firePoint.position, firePoint.rotation);
127         Bullet bullet = bulletGO.GetComponent<Bullet>();
128
129         if (bullet != null) {
130             bullet.Seek(target);
131         }
132     }
133
134     // Draws a red wired sphere around the object
135     void OnDrawGizmosSelected () {
136         Gizmos.color = Color.red;
137         Gizmos.DrawWireSphere(transform.position, range);
138     }
139 }
140
```