Project Code

1. BackGroundScroller.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//START OF OWN CODE//
public class BackgroundScroller : MonoBehaviour
{
    public Transform BG1, BG2;
    public float scrollSpeed;

    private float bgWidth;
    // Start is called before the first frame update
    void Start()
    {
        bgWidth =
BG1.GetComponent<SpriteRenderer>().sprite.bounds.size.x;
    }

    // Update is called once per frame
    void Update()
    {
        BG1.position = new Vector3(BG1.position.x - (scrollSpeed *
Time.deltaTime), BG1.position.y, BG2.position.z);
        BG2.position -= new Vector3(scrollSpeed * Time.deltaTime,
0f, 0f);

        if(BG1.position.x < -bgWidth - 1)
        {
            BG1.position += new Vector3(bgWidth * 2f, 0f, 0f);
        }

        if(BG2.position.x < -bgWidth - 1)
        {
            BG2.position += new Vector3(bgWidth * 2f, 0f, 0f);
        }
    }
```

```
}
//END OF OWN CODE
```

2. BossManager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//START OF OWN CODE
public class BossManager : MonoBehaviour
{
    public static BossManager instance;
    public string bossName;

    public int currentHealth = 100;



    public BattlePhase[] phases;
    public int currentPhase;
    public Animator bossAnim;


    public GameObject endExplosion;
    public bool battleEnding;
    public float timeToExplosionEnd;
    public float waitToEndLevel;

    public Transform theBoss;
    public int scoreValue = 5000;

    private void Awake()
    {
        instance = this;
    }
    // Start is called before the first frame update
    void Start()
    {
        UIManager.instance.bossName.text = bossName;
```

```
        UIManager.instance.bossHealthSlider.maxValue =
currentHealth;
        UIManager.instance.bossHealthSlider.value = currentHealth;

UIManager.instance.bossHealthSlider.gameObject.SetActive(true);

        MusicController.instance.PlayBoss();
    }
//END OF OWN CODE
    // Update is called once per frame
    void Update()
    {

        if(!battleEnding)
        {
            if(currentHealth <=
phases[currentPhase].healthToEndPhase)
            {

phases[currentPhase].removeAtPhaseEnd.SetActive(false);
                Instantiate(phases[currentPhase].addAtPhaseEnd,
phases[currentPhase].newSpawnPoint.position,
phases[currentPhase].newSpawnPoint.rotation);

                currentPhase++;

                bossAnim.SetInteger("Phase", currentPhase + 1);
            }
            else
            {
                for(int i = 0; i <
phases[currentPhase].phaseShots.Length; i++)
                {
                    phases[currentPhase].phaseShots[i].shotCounter
-= Time.deltaTime;


if(phases[currentPhase].phaseShots[i].shotCounter <= 0)
                    {
```

```
phases[currentPhase].phaseShots[i].shotCounter =
phases[currentPhase].phaseShots[i].timeBetweenShots;

Instantiate(phases[currentPhase].phaseShots[i].shot,
phases[currentPhase].phaseShots[i].firePoint.position,
phases[currentPhase].phaseShots[i].firePoint.rotation);
                    }
                }
            }
        }
    }
//START OF OWN CODE
    public void HurtBoss()
    {
        currentHealth--;
        UIManager.instance.bossHealthSlider.value = currentHealth;

        if(currentHealth <= 0 && !battleEnding)
        {

            battleEnding = true;
            StartCoroutine(EndBattleCo());
        }
    }

    public IEnumerator EndBattleCo()
    {

UIManager.instance.bossHealthSlider.gameObject.SetActive(false);
        Instantiate(endExplosion, theBoss.position,
theBoss.rotation);
        bossAnim.enabled = false;
        GameManager.instance.AddScore(scoreValue);

        yield return new WaitForSeconds(timeToExplosionEnd);

        theBoss.gameObject.SetActive(false);

        yield return new WaitForSeconds(waitToEndLevel);
```

```
        StartCoroutine(GameManager.instance.EndLevelCo());
    }

}

[System.Serializable]
public class BattleShot
{
    public GameObject shot;
    public float timeBetweenShots;
    public float shotCounter;
    public Transform firePoint;
}

[System.Serializable]
public class BattlePhase
{
    public BattleShot[] phaseShots;
    public int healthToEndPhase;
    public GameObject removeAtPhaseEnd;
    public GameObject addAtPhaseEnd;
    public Transform newSpawnPoint;
}
//END OF OWN CODE
```

3. DestroyOverTime.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//own code
public class DestroyOverTime : MonoBehaviour
{
    public float lifetime;
    // Start is called before the first frame update
    void Start()
    {

    }
```

```
    // Update is called once per frame
    void Update()
    {
        Destroy(gameObject, lifetime);
    }
}
//end of own code
```

4. EnemyController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//own code
public class EnemyController : MonoBehaviour
{
    public float moveSpeed;

    public Vector2 startDirection;

    public bool shouldChangeDirection;
    public float changeDirectionXPoint;
    public Vector2 changedDirection;

    public GameObject shotToFire;
    public Transform firePoint;
    public float timeBetweenShots;
    private float shotCounter;

    public bool canShoot;
    private bool allowShooting;

    public int currentHealth;
    public GameObject deathEffect;

    public int scoreValue = 100;

    public GameObject[] powerUps;
    public int dropSuccessRate = 20;
    // Start is called before the first frame update
```

```
    void Start()
    {
        shotCounter = timeBetweenShots;
    }

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

        if(!shouldChangeDirection)
        {
        transform.position += new Vector3(startDirection.x *
moveSpeed * Time.deltaTime, startDirection.y * moveSpeed *
Time.deltaTime, 0f);
        }
        else
        {
            if(transform.position.x > changeDirectionXPoint)
            {
                transform.position += new Vector3(startDirection.x *
moveSpeed * Time.deltaTime, startDirection.y * moveSpeed *
Time.deltaTime, 0f);
            }
            else
            {
                transform.position += new Vector3(changedDirection.x
* moveSpeed * Time.deltaTime, changedDirection.y * moveSpeed *
Time.deltaTime, 0f);
            }
        }
        if(allowShooting)
        {
            shotCounter -= Time.deltaTime;
            if(shotCounter <= 0)
            {
                shotCounter = timeBetweenShots;
                Instantiate(shotToFire, firePoint.position,
firePoint.rotation);
            }
        }
```

```
    }
//end of own code
    public void HurtEnemy()
    {
        currentHealth--;
        if(currentHealth <=0)
        {
            GameManager.instance.AddScore(scoreValue);

            int randomChance = Random.Range(0,100);
            if(randomChance < dropSuccessRate)
            {
                int randomPick = Random.Range(0, powerUps.Length);
                Instantiate(powerUps[randomPick],
transform.position, transform.rotation);
            }
            Destroy(gameObject);
            Instantiate(deathEffect, transform.position,
transform.rotation);
        }
    }
//own code
    private void OnBecameInvisible()
    {
        Destroy(gameObject);
    }

    private void OnBecameVisible()
    {
        if(canShoot)
        {
            allowShooting = true;
        }
    }
}
//end of own code
```

5. EnemyShots.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//own code
public class EnemyShot : MonoBehaviour
{
    public float shotSpeed = 7f;
    public GameObject impactEffect;

    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        transform.position -= new Vector3(shotSpeed *
Time.deltaTime, 0f, 0f);
    }

    private void OnTriggerEnter2D(Collider2D other)
    {
        Instantiate(impactEffect, transform.position,
transform.rotation);

        if(other.tag == "Player")
        {
            HealthManager.instance.HurtPlayer();
        }


        Destroy(this.gameObject);
    }

    private void OnBecameInvisible()
    {
        Destroy(this.gameObject);
```

```
    }
}
//end of own code
```

6. EnemyWave.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//own code
public class EnemyWave : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {
        transform.DetachChildren();
        Destroy(gameObject);
    }

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

    }
}
//end of own code
```

7. GameOverScreen.cs

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

public class GameCompleteScreen : MonoBehaviour
{
    public float timeBetweenTexts;
    public bool canExit;
```

```
    public string mainMenuName = "MainMenu";
    public Text score, pressKey;
    // Start is called before the first frame update
    void Start()
    {
        StartCoroutine(ShowTextCo());
    }

    // Update is called once per frame
    void Update()
    {
        if(canExit && Input.anyKeyDown)
        {
            SceneManager.LoadScene(mainMenuName);
        }
    }

    public IEnumerator ShowTextCo()
    {
        yield return new WaitForSeconds(timeBetweenTexts);
        score.text = "Final Score: " +
PlayerPrefs.GetInt("CurrentScore");
        score.gameObject.SetActive(true);
        yield return new WaitForSeconds(timeBetweenTexts);
        pressKey.gameObject.SetActive(true);
        canExit = true;

    }
}
```

8. GameManager.cs

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

public class GameManager : MonoBehaviour
{
```

```
    public static GameManager instance;

    public int currentLives = 3;

    public float respawnTime = 2f;

    public int currentScore;
    private int highScore;

    private int levelScore;

    public float waitForLevelEnd = 5f;

    public string nextLevel;

    private bool canPause;

    private void Awake()
    {
        instance = this;
    }

    void Start()
    {
        //first line
        currentLives = PlayerPrefs.GetInt("CurrentLives");
        UIManager.instance.livesText.text = "x" + currentLives;

        highScore = PlayerPrefs.GetInt("HighScore");
        UIManager.instance.highScoreText.text = "High Score: " +
highScore;

        //first line
        currentScore = PlayerPrefs.GetInt("CurrentScore");
        UIManager.instance.scoreText.text = "Score: " +
currentScore;

        canPause = true;
    }
```

```
    void Update()
    {
        if(Input.GetKeyDown(KeyCode.Escape) && canPause)
        {
            PauseUnpause();
        }
    }
//own code
    public void KillPlayer()
    {
        currentLives--;
        UIManager.instance.livesText.text = "x" + currentLives;

        if(currentLives > 0)
        {
            StartCoroutine(RespawnCo());
        }
        else
        {
            UIManager.instance.gameOverScreen.SetActive(true);
            WaveManager.instance.canSpawnWaves = false;

            MusicController.instance.PlayDefeat();
            PlayerPrefs.SetInt("HighScore", highScore);

            canPause = false;
        }
    }
//end own code
    public IEnumerator RespawnCo()
    {
        yield return new WaitForSeconds(respawnTime);
        HealthManager.instance.Respawn();

        WaveManager.instance.ContinueSpawning();
    }

    public void AddScore(int scoreToAdd)
    {
        currentScore += scoreToAdd;
```

```
        levelScore += scoreToAdd;
        UIManager.instance.scoreText.text = "Score: " +
currentScore;

        if(currentScore > highScore)
        {
            highScore = currentScore;
            UIManager.instance.highScoreText.text = "High Score: " +
highScore;
            //PlayerPrefs.SetInt("HighScore", highScore);
        }
    }


    public IEnumerator EndLevelCo()
    {
        UIManager.instance.levelEndScreen.SetActive(true);
        PlayerController.instance.stopMovement = true;
        MusicController.instance.PlayVictory();

        canPause = false;

        yield return new WaitForSeconds(.5f);

        UIManager.instance.endLevelScore.text = "Level Score: " +
levelScore;
        UIManager.instance.endLevelScore.gameObject.SetActive(true);

        yield return new WaitForSeconds(.5f);

        PlayerPrefs.SetInt("CurrentScore", currentScore);
        UIManager.instance.endCurrentScore.text = "Total Score: " +
currentScore;

UIManager.instance.endCurrentScore.gameObject.SetActive(true);

        if(currentScore == highScore)
        {
            yield return new WaitForSeconds(.5f);
            UIManager.instance.highScoreNotice.SetActive(true);
```

```
        }

        PlayerPrefs.SetInt("HighScore", highScore);
        PlayerPrefs.SetInt("CurrentLives", currentLives);

        yield return new WaitForSeconds(waitForLevelEnd);

        SceneManager.LoadScene(nextLevel);
    }

    public void PauseUnpause()
    {
        if(UIManager.instance.pauseScreen.activeInHierarchy)
        {
            UIManager.instance.pauseScreen.SetActive(false);
            Time.timeScale = 1f;
            PlayerController.instance.stopMovement = false;
        }
        else
        {
            UIManager.instance.pauseScreen.SetActive(true);
            Time.timeScale = 0f;
            PlayerController.instance.stopMovement = true;
        }
    }
}
```

9. HealthManager.cs

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

public class HealthManager : MonoBehaviour
{
    public static HealthManager instance;

    public int currentHealth;
    public int maxHealth;
```

```
    public GameObject deathEffect;

    public float invincibleLength = 2f;
    private float invincCounter;
    public SpriteRenderer sr;

    public int shieldPower;
    public int shieldMaxPower = 2;
    public GameObject theShield;

    private void Awake()
    {
        instance = this;
    }
    // Start is called before the first frame update
    void Start()
    {
        currentHealth = maxHealth;

        UIManager.instance.healthBar.maxValue = maxHealth;
        UIManager.instance.healthBar.value = currentHealth;

        UIManager.instance.shieldBar.maxValue = shieldMaxPower;
        UIManager.instance.shieldBar.value = shieldPower;
    }

    // Update is called once per frame
    void Update()
    {
        if(invincCounter >= 0)
        {
            invincCounter -= Time.deltaTime;

            if(invincCounter <= 0)
            {
                sr.color = new Color(sr.color.r, sr.color.g,
sr.color.b, 1f);
            }
        }
```

```
    }

    public void HurtPlayer()
    {
        if(invincCounter <= 0)
        {
            if(theShield.activeInHierarchy)
            {
                shieldPower--;
                if(shieldPower <= 0)
                {
                    theShield.SetActive(false);
                }
                UIManager.instance.shieldBar.value = shieldPower;
            }
            else
            {
            currentHealth--;
            UIManager.instance.healthBar.value = currentHealth;

                if(currentHealth <= 0)
                {
                    Instantiate(deathEffect, transform.position,
transform.rotation);
                    gameObject.SetActive(false);

                    GameManager.instance.KillPlayer();

                    WaveManager.instance.canSpawnWaves = false;
                }

                PlayerController.instance.doubleShotActive = false;
            }
        }
    }

    public void Respawn()
    {
        gameObject.SetActive(true);
        currentHealth = maxHealth;
```

```
        UIManager.instance.healthBar.value = currentHealth;

        invincCounter = invincibleLength;

        sr.color = new Color(sr.color.r, sr.color.g, sr.color.b,
.5f);
    }


    public void ActivateShield()
    {
        theShield.SetActive(true);
        shieldPower = shieldMaxPower;

        UIManager.instance.shieldBar.value = shieldPower;
    }
}
```

10. HurtPlayer.cs

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

public class HurtPlayer : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

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

    }
```

```
    private void OnCollisionEnter2D(Collision2D other)
    {
        if(other.gameObject.tag == "Player")
        {
            HealthManager.instance.HurtPlayer();
        }
    }
}
```

11. LevelEnd.cs

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

public class HurtPlayer : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

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

    }

    private void OnCollisionEnter2D(Collision2D other)
    {
        if(other.gameObject.tag == "Player")
        {
            HealthManager.instance.HurtPlayer();
        }
    }
}
```

12. MainMenu.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//own code
public class HurtPlayer : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

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

    }

    private void OnCollisionEnter2D(Collision2D other)
    {
        if(other.gameObject.tag == "Player")
        {
            HealthManager.instance.HurtPlayer();
        }
    }
}
//end of own code
```

13. MovingObjects.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//own code
public class MovingObjects : MonoBehaviour
{
    public float moveSpeed;
```

```
    // Update is called once per frame
    void Update()
    {
        transform.position -= new Vector3(moveSpeed *
Time.deltaTime, 0f, 0f);
    }

    private void OnBecameInvisible()
    {
        Destroy(gameObject);
    }
}
//end of own code
```

14. MusicController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//own code
public class MusicController : MonoBehaviour
{
    public static MusicController instance;
    public AudioSource levelMusic, victoryMusic, defeatMusic,
explosionMusic, bossMusic;

    private void Awake()
    {
        instance = this;
    }
    // Start is called before the first frame update
    void Start()
    {
        levelMusic.Play();
    }

    // Update is called once per frame
    void Update()
    {
```

```
    }

    void StopMusic()
    {
        levelMusic.Stop();
        victoryMusic.Stop();
        defeatMusic.Stop();
        explosionMusic.Stop();
        bossMusic.Stop();
    }

    public void PlayVictory()
    {
        StopMusic();
        victoryMusic.Play();
    }

    public void PlayDefeat()
    {
        StopMusic();
        defeatMusic.Play();
    }

    public void PlayExplosion()
    {
        StopMusic();
        explosionMusic.Play();
    }

    public void PlayBoss()
    {
        StopMusic();
        bossMusic.Play();
    }
}
//end own code
```

15. PlayerController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//own code
public class PlayerController : MonoBehaviour
{
    public static PlayerController instance;

    public float moveSpeed;
    public Rigidbody2D rb;

    public Transform bottomLeftLimit, topRightLimit;

    public Transform shotPoint;
    public GameObject shot;

    public float timeBetweenShots = .1f;
    private float shotCounter;


    public bool doubleShotActive;
    public float doubleShotOffset;


    public bool stopMovement;

    private void Awake()
    {
        instance = this;
    }

    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
```

```
    {
        if(!stopMovement)
        {
        rb.velocity = new Vector2(Input.GetAxisRaw("Horizontal"),
Input.GetAxisRaw("Vertical")) * moveSpeed;

        transform.position = new
Vector3(Mathf.Clamp(transform.position.x,
bottomLeftLimit.position.x, topRightLimit.position.x),
Mathf.Clamp(transform.position.y, bottomLeftLimit.position.y,
topRightLimit.position.y), transform.position.z);

        if(Input.GetButtonDown("Fire1"))
        {
            if(!doubleShotActive)
            {
            Instantiate(shot, shotPoint.position,
shotPoint.rotation);
            }
            else
            {
                Instantiate(shot, shotPoint.position + new
Vector3(0f, doubleShotOffset, 0f), shotPoint.rotation);
                Instantiate(shot, shotPoint.position - new
Vector3(0f, doubleShotOffset, 0f), shotPoint.rotation);
            }
            shotCounter = timeBetweenShots;
        }

        if(Input.GetButton("Fire1"))
        {
            shotCounter -= Time.deltaTime;
            if(shotCounter <= 0)
            {
                if(!doubleShotActive)
            {
            Instantiate(shot, shotPoint.position,
shotPoint.rotation);
            }
            else
```

```
            {
                Instantiate(shot, shotPoint.position + new
Vector3(0f, doubleShotOffset, 0f), shotPoint.rotation);
                Instantiate(shot, shotPoint.position - new
Vector3(0f, doubleShotOffset, 0f), shotPoint.rotation);
            }
                shotCounter = timeBetweenShots;
            }
        }
    }
    else
    {
        rb.velocity = Vector2.zero;
    }
    }
}
//end of own code
```

16. PlayerShot.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//own code
public class PlayerController : MonoBehaviour
{
    public static PlayerController instance;

    public float moveSpeed;
    public Rigidbody2D rb;

    public Transform bottomLeftLimit, topRightLimit;

    public Transform shotPoint;
    public GameObject shot;

    public float timeBetweenShots = .1f;
    private float shotCounter;
```

```
    public bool doubleShotActive;
    public float doubleShotOffset;


    public bool stopMovement;

    private void Awake()
    {
        instance = this;
    }

    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        if(!stopMovement)
        {
        rb.velocity = new Vector2(Input.GetAxisRaw("Horizontal"),
Input.GetAxisRaw("Vertical")) * moveSpeed;

        transform.position = new
Vector3(Mathf.Clamp(transform.position.x,
bottomLeftLimit.position.x, topRightLimit.position.x),
Mathf.Clamp(transform.position.y, bottomLeftLimit.position.y,
topRightLimit.position.y), transform.position.z);

        if(Input.GetButtonDown("Fire1"))
        {
            if(!doubleShotActive)
            {
            Instantiate(shot, shotPoint.position,
shotPoint.rotation);
            }
            else
            {
```

```
                Instantiate(shot, shotPoint.position + new
Vector3(0f, doubleShotOffset, 0f), shotPoint.rotation);
                Instantiate(shot, shotPoint.position - new
Vector3(0f, doubleShotOffset, 0f), shotPoint.rotation);
            }
            shotCounter = timeBetweenShots;
        }

        if(Input.GetButton("Fire1"))
        {
            shotCounter -= Time.deltaTime;
            if(shotCounter <= 0)
            {
                if(!doubleShotActive)
            {
            Instantiate(shot, shotPoint.position,
shotPoint.rotation);
            }
            else
            {
                Instantiate(shot, shotPoint.position + new
Vector3(0f, doubleShotOffset, 0f), shotPoint.rotation);
                Instantiate(shot, shotPoint.position - new
Vector3(0f, doubleShotOffset, 0f), shotPoint.rotation);
            }
                shotCounter = timeBetweenShots;
            }
        }
    }
    else
    {
        rb.velocity = Vector2.zero;
    }
    }
}
//end of own code
```

17. PowerUp.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
//own code
public class PowerUp : MonoBehaviour
{
    public bool isShield;

    public bool isDoubleShot;
    // Start is called before the first frame update
    void Start()
    {

    }

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

    }

    private void OnTriggerEnter2D(Collider2D other)
    {
        if(other.tag == "Player")
        {
            Destroy(gameObject);

            if(isShield)
            {
                HealthManager.instance.ActivateShield();
            }

            if(isDoubleShot)
            {
                PlayerController.instance.doubleShotActive = true;
            }
        }
    }
}
```

```
//end own code
```

18. UIManager.cs

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

public class UIManager : MonoBehaviour
{
    public static UIManager instance;

    public GameObject gameOverScreen;

    public Text livesText;

    public Slider healthBar, shieldBar;

    public Text scoreText, highScoreText;

    public GameObject levelEndScreen;

    public Text endLevelScore, endCurrentScore;
    public GameObject highScoreNotice;

    public GameObject pauseScreen;

    public string mainMenuName = "MainMenu";

    public Slider bossHealthSlider;
    public Text bossName;


    private void Awake()
    {
        instance = this;
    }
```

```
    // Start is called before the first frame update
    void Start()
    {

    }

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

    }

    public void Restart()
    {
        SceneManager.LoadScene(SceneManager.GetActiveScene().name);
        Time.timeScale = 1f;
    }

    public void QuitToMain()
    {
        SceneManager.LoadScene(mainMenuName);
        Time.timeScale = 1f;
    }

    public void Resume()
    {
        GameManager.instance.PauseUnpause();
    }
}
```

19. WaveManager.cs

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

public class WaveManager : MonoBehaviour
{
    public static WaveManager instance;
```

```
    public WaveObject[] waves;

    public int currentWave;
    public float timeToNextWave;

    public bool canSpawnWaves;

    void Awake()
    {
        instance = this;
    }
    // Start is called before the first frame update
    void Start()
    {
        timeToNextWave = waves[0].timeToSpawn;
    }

    // Update is called once per frame
    void Update()
    {
        if(canSpawnWaves)
        {
            timeToNextWave -= Time.deltaTime;
            if(timeToNextWave <= 0)
            {
                Instantiate(waves[currentWave].theWave,
transform.position, transform.rotation);

                    if(currentWave < waves.Length - 1)
                    {
                        currentWave++;

                        timeToNextWave =
waves[currentWave].timeToSpawn;
                    }
                    else
                    {
                        canSpawnWaves = false;
                    }
```

```
            }
        }
    }

    public void ContinueSpawning()
    {
        if(currentWave <= waves.Length -1 && timeToNextWave > 0)
        {
            canSpawnWaves = true;
        }
    }
}

[System.Serializable]
public class WaveObject
{
    public float timeToSpawn;
    public EnemyWave theWave;
}
```