Game Design Document

HTML5 Game Development

Submitted by: Udit Shrivastava – 20BCG10109
GitHub Repository Link: https://github.com/ZerNius/2D_Side_Scoller.git

Game Title: Super Dog Hero vs. Monster Mayhem

Game Concept: Super Dog Hero is a fearless canine superhero who must eliminate as many monsters as possible within a 30-second time frame. The game combines platforming, action, and time-based challenges to create an engaging and fast-paced experience.

Gameplay Mechanics:

1. Controls:
    - Arrow keys or WASD for movement (left/right and jump)
    - Enter to attack
2. Player Abilities:
    - Jumping: Super Dog Hero can jump to avoid obstacles and reach higher platforms. The longer the jump button is held, the higher the jump height.
    - Attack: Super Dog Hero can use his superpowers to attack monsters and eliminate them. A single attack will defeat regular monsters, while boss monsters may require multiple attacks.
3. Monster Behaviour:
    - Monsters will appear at random positions on the screen and move towards Super Dog Hero. Some monsters may have unique movement patterns or abilities.
    - Regular monsters: Move towards Super Dog Hero in a straight line.
    - Flying monsters: Fly in the air and require precise timing to attack.
    - Boss monsters: Have more health and special abilities. Defeating them rewards extra points.
4. Power-ups:
    - Occasionally, power-ups such as health boosts or temporary invincibility will appear for Super Dog Hero to collect. Power-ups can be obtained by touching them during gameplay.
    - Health boosts: Restore a portion of Super Dog Hero's health.
    - Temporary invincibility: Grants invulnerability to Super Dog Hero for a short duration, allowing easy monster elimination.

Level Design:

1. The game will have a continuous side-scrolling level with platforms and obstacles.
2. Platforms will vary in height and width, requiring precise jumping and navigation. Some platforms may move horizontally or vertically.
3. Obstacles may include spikes, moving platforms, or falling objects, adding challenge and complexity to the gameplay. Colliding with obstacles reduces Super Dog Hero's health.
4. The level will scroll automatically from left to right, creating a sense of progression and urgency. The level design may include multiple sections with increasing difficulty.

Art and Visual Style:

1. Super Dog Hero: Create a visually appealing dog superhero character with distinctive animations for movement and attacking. Consider different poses for walking, running, jumping, and attacking. Game is themed as a Halloween game so game style is created such.

   Screen shots of Main Character Sprite Sheet:

   This shows idle and running animation loops.

2. Monsters: Design various types of monsters with unique appearances and animations to differentiate them from each other. Consider incorporating different colour schemes or visual effects for boss monsters.

   There are three types of monsters:

   a. Flying Bug Monster: Travel in air in random unpredictable path.

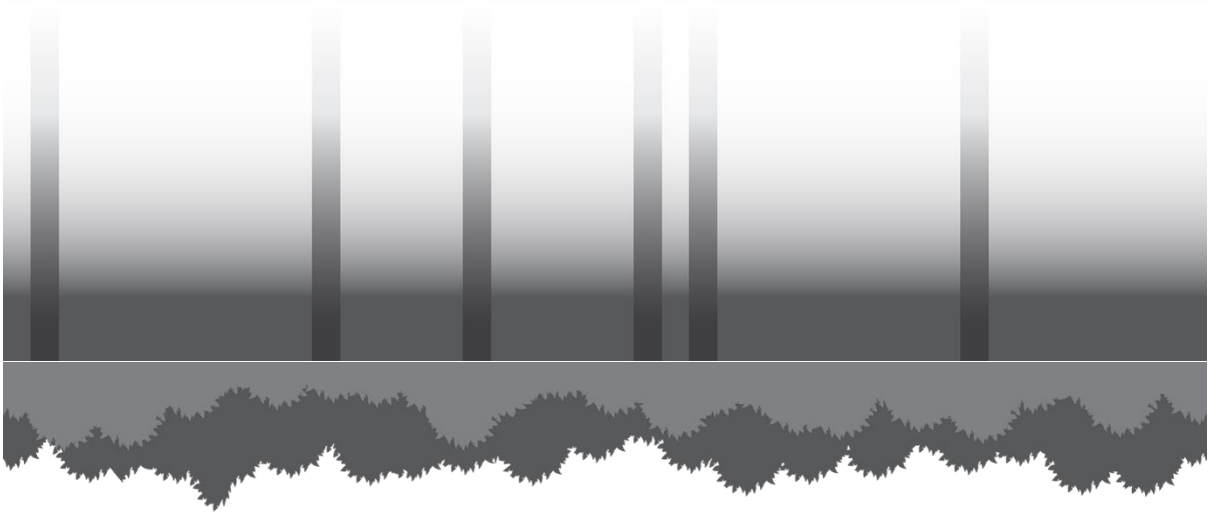
   

   b. Spider Monster which will drop from top of screen and will travel a linear vertical path.

   c. Plant monster which will only spawn on ground. Cannot move and is there to restrict player movement.

3. Environments: Create a Halloween themed and dynamic background with trees and bushes to give it appropriate jungle theme. Incorporate parallax scrolling for a sense of depth.
    a. Different layer used to create parallax effect:



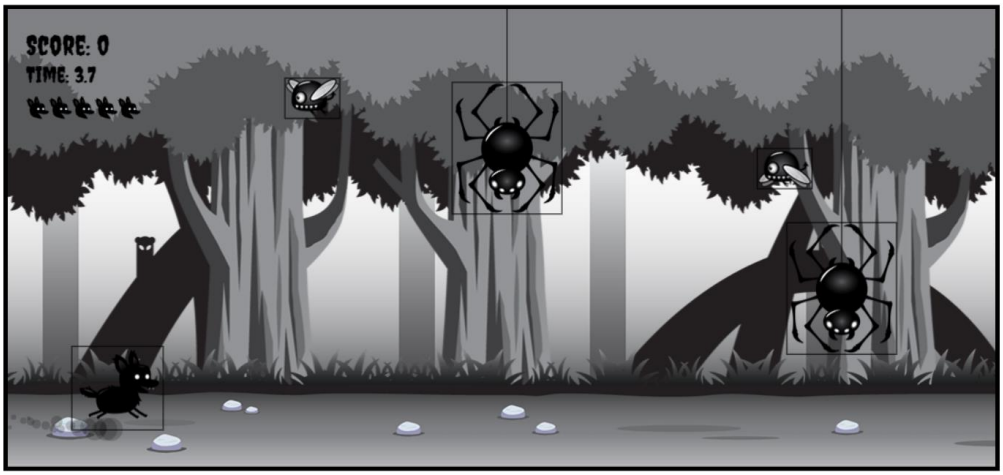
Audio and Music:

1. Background Music: Create an upbeat and energetic soundtrack that complements the fast-paced gameplay. Consider using different tracks for different sections or levels to add variety.
2. Sound Effects: Include sound effects for character movements, attacks, monster eliminations, and power-up pickups. Ensure the sound effects are distinct and satisfying to enhance the player's experience.

Game Systems:

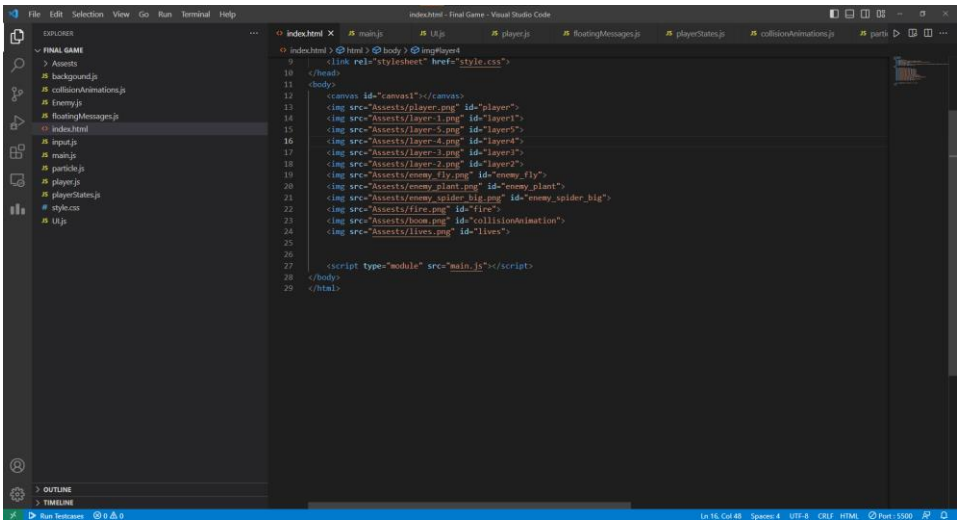
1. Collision Detection: Implement collision detection between Super Dog Hero, monsters, platforms, and obstacles to ensure proper interactions. Determine specific hitboxes for each entity to determine successful attacks and collisions.
2. Scoring System: Track the number of monsters defeated by Super Dog Hero and display the score on the screen. Assign different point values for regular monsters and boss monsters.
3. Timer: Display a countdown timer of 30 seconds and end the game when it reaches zero. Provide visual cues, such as a countdown displayed on-screen or an animated timer.
4. Health System: Assign health points to Super Dog Hero and reduce them upon colliding with monsters or hazards. End the game if health reaches zero. Display the current health as a visual indicator on the screen.
5. Game Over and Restart: Provide a game over screen with the final score and an option to restart the game. Include a prominent "Play Again" button to encourage players to try again.

Screen shots of Game with all the systems:



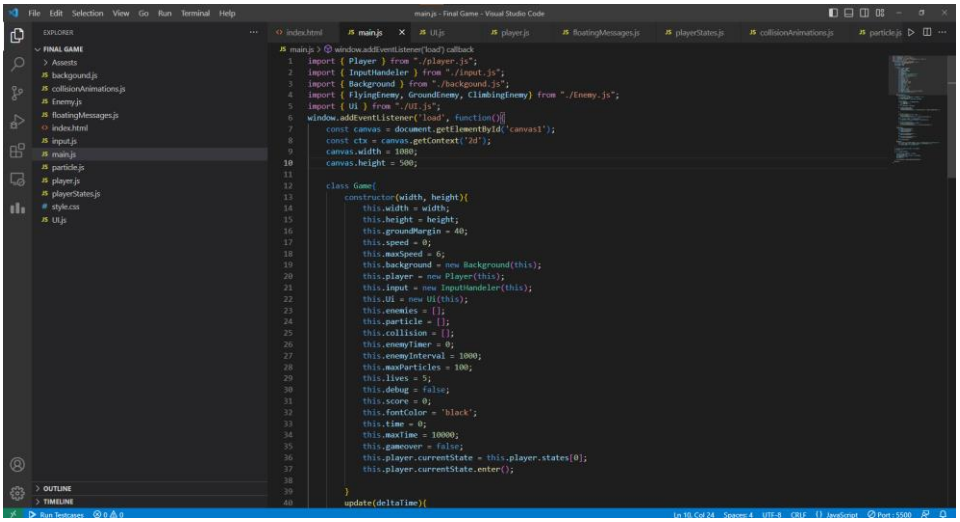
Screen capture showing squares used for collision detection.

Screenshots of Code:



```
<link rel="stylesheet" href="style.css">
</head>
<body>
    <canvas id="canvas1"></canvas>
    <img src="Assests/player.png" id="player">
    <img src="Assests/layer-1.png" id="layer1">
    <img src="Assests/layer-5.png" id="layer5">
    <img src="Assests/layer-4.png" id="layer4">
    <img src="Assests/layer-3.png" id="layer3">
    <img src="Assests/layer-2.png" id="layer2">
    <img src="Assests/enemy_fly.png" id="enemy_fly">
    <img src="Assests/enemy_plant.png" id="enemy_plant">
    <img src="Assests/enemy_spider_big.png" id="enemy_spider_big">
    <img src="Assests/fire.png" id="fire">
    <img src="Assests/boom.png" id="collisionAnimation">
    <img src="Assests/lives.png" id="lives">


    <script type="module" src="main.js"></script>
</body>
</html>
```



```
import { Player } from "./player.js";
import { InputHandeler } from "./input.js";
import { Background } from "./background.js";
import { FlyingEnemy, GroundEnemy, ClimbingEnemy} from "./Enemy.js";
import { Ui } from "./UI.js";
window.addEventListener('load', function(){
    const canvas = document.getElementById('canvas1');
    const ctx = canvas.getContext('2d');
    canvas.width = 1000;
    canvas.height = 500;

    class Game{
        constructor(width, height){
            this.width = width;
            this.height = height;
            this.groundMargin = 40;
            this.speed = 0;
            this.maxSpeed = 6;
            this.background = new Background(this);
            this.player = new Player(this);
            this.input = new InputHandeler(this);
            this.Ui = new Ui(this);
            this.enemies = [];
            this.particle = [];
            this.collision = [];
            this.enemyTimer = 0;
            this.enemyInterval = 1000;
            this.maxParticles = 100;
            this.lives = 5;
            this.debug = false;
            this.score = 0;
            this.fontColor = 'black';
            this.time = 0;
            this.maxTime = 10000;
            this.gameover = false;
            this.player.currentState = this.player.states[0];
            this.player.currentState.enter();

        }
        update(deltaTime){
```

Full code can be accessed in the git repository link given at top of document.