## Front End Engineering-II

Project Report

Semester-IV (Batch-2022)

Title of the Project

A red and white sign

Description automatically generated with low confidence

**Supervised By: Submitted By:**

Faculty Name : Raveesh samkaria Name : Suhavi Jindal

Roll No. : 2210990867

Group: G-13

**Department of Computer Science and Engineering**

## Chitkara University Institute of Engineering & Technology,

## Chitkara University, Punjab

**Table of content:**

|  |  |  |
| --- | --- | --- |
| **1** | **Introduction** | **1** |
| **2** | **Objectives** | **1** |
| **3** | **Technologies Used** | **1** |
| **4** | **Implementation** | **2** |
| **5** | **Challenges Faced** | **2-3** |
| **6** | **Folder structure** | **3** |
| **7** | **Code** | **3-9** |
| **8** | **Future Enhancements** | **10** |
| **9** | **Result** | **10-11** |
| **10** | **Conclusion** | **12** |
| **11** | **References** | **12** |

**Introduction:**

The Snake Game is a classic arcade game where the player controls a snake that moves around the screen, eating food pellets to grow longer. The game ends when the snake collides with itself or the boundaries of the game screen. In this project, we implemented the Snake Game using Bootstrap for layout and styling and JavaScript for game logic.

**Objectives**:

- Develop a classic Snake Game using web technologies.

- Utilize Bootstrap framework for responsive and visually appealing design.

- Implement game logic using JavaScript for user interaction and game mechanics.

- Ensure cross-browser compatibility and responsiveness for seamless gameplay experience across devices.

**Technologies Used:**

- HTML5: For structure and layout of the web page.

- CSS3: For styling and visual enhancements.

- Bootstrap: Front-end framework for responsive design and UI components.

- JavaScript: For implementing game logic and user interaction.

**Implementation:**

HTML Structure: Used HTML to create the basic structure of the game interface, including the game canvas and score display.

Bootstrap Integration: Leveraged Bootstrap classes and components for responsive design, grid layout, buttons, and modal dialogs.

CSS Styling: Customized styles using CSS to enhance the visual appeal of the game, including colors , fonts, and animations.

JavaScript Logic: Implemented the core game logic using JavaScript, including snake movement, collision detection, food generation, score tracking, and game over conditions.

-Event Handling: Used JavaScript event listeners to capture user input for controlling the snake's direction.

Canvas Drawing: Utilized HTML5 Canvas API to draw the snake, food pellets, and game boundaries dynamically.

Game Loop: Implemented a game loop to update the game state and render the changes on the canvas at regular intervals.

Responsive Design: Ensured that the game interface adapts gracefully to different screen sizes and orientations using Bootstrap's responsive grid system and CSS media queries.

**Challenges Faced:**

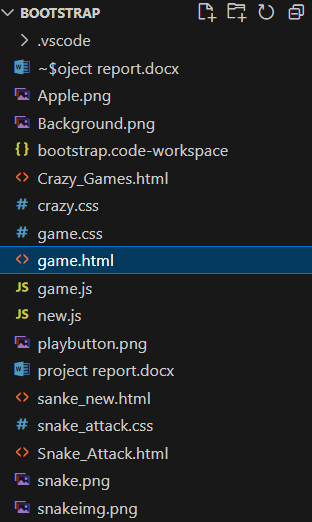
- Managing the snake's movement and collision detection algorithmically.

- Ensuring smooth and responsive user interaction across devices.

- Designing an intuitive and visually appealing user interface using Bootstrap components.

- Optimizing performance for efficient rendering and gameplay experience.

**Folder Structure:**

****

**Bootstrap code:**

<!doctype html>

<html lang="en">

<head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <title>Snake\_Attack</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet"

        integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">

    <link rel="stylesheet" href="game.css">

</head>

<body>

    <div class="container1 text-center  ">

        <div class="row ">

            <div class=" div\_1 col-md-8 w-100 fs-1 fst-italic fw-bolder fw-bold text-light">Snake\_Attack\_Mania</div>

        </div>

        <div class="container2 d-flex justify-content-around">

            <div id="board"></div>

            <div class="scores ms-5 text-start">

                <div id="scoreBox">Score: 0</div>

                <div id="hiscoreBox">HighScore: 0</div>

            </div>

        </div>

    </div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"

        integrity="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"

        crossorigin="anonymous"></script>

    <script src="game.js"></script>

</body>

</html>

**Css code:**

body{

    background-image: url('https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSlzONcgxW0U-iOQ5BYgt4RRa-9qh0f6Rmkng&s');

    background-repeat: no-repeat;

    background-size: cover;

    display: flex;

    justify-content: center;

    align-items: center;

}

#board{

    background-image: url("https://cdn.tutsplus.com/cdn-cgi/image/width=480/mobile/uploads/legacy/Corona-SDK\_Build-A-Snake-Game/1/6.png");

    background-repeat: no-repeat;

    background-size: cover;

    width: 90vmin;

    height: 92vmin;

    border: 2px solid black;

    display: grid;

    grid-template-rows: repeat(18, 1fr);

    grid-template-columns: repeat(18, 1fr);

}

#scoreBox{

    font-size: 39px;

    font-weight: bold;

    font-family: 'New Tegomin', serif;

    color: antiquewhite;

}

#hiscoreBox{

    font-size: 39px;

    font-weight: bold;

    font-family: 'New Tegomin', serif;

    color: antiquewhite;

}

.head{

    background: linear-gradient(rgb(240, 124, 124), rgb(228, 228, 129));

    background-color: #88b858;

    border: 2px solid rgb(34, 4, 34);

    transform: scale(1.02);

    border-radius: 9px;

}

.snake{

    background-color: purple;

    background-color: #b4cd3e;

    border: .25vmin solid white;

    border-radius: 12px;

}

.food{

    background: linear-gradient(red, purple);

    background-color: #ff6347;

    border: .25vmin solid black;

    border-radius: 8px;

    box-shadow: 2px 2px 5px rgba(0, 0, 0, 0.5);

}

**Javascript**

// Game Constants & Variables

let inputDir = {x: 0, y: 0};

const foodSound = new Audio('music/food.mp3');

const gameOverSound = new Audio('music/gameover.mp3');

const moveSound = new Audio('music/move.mp3');

const musicSound = new Audio('music/music.mp3');

let speed = 10;

let score = 0;

let lastPaintTime = 0;

let snakeArr = [

    {x: 13, y: 15}

];

food = {x: 6, y: 7};

// Game Functions

function main(ctime) {

    window.requestAnimationFrame(main);

    // console.log(ctime)

    if((ctime - lastPaintTime)/1000 < 1/speed){

        return;

    }

    lastPaintTime = ctime;

    gameEngine();

}

function isCollide(snake) {

    // If you bump into yourself

    for (let i = 1; i < snakeArr.length; i++) {

        if(snake[i].x === snake[0].x && snake[i].y === snake[0].y){

            return true;

        }

    }

    // If you bump into the wall

    if(snake[0].x >= 18 || snake[0].x <=0 || snake[0].y >= 18 || snake[0].y <=0){

        return true;

    }

    return false;

}

function gameEngine(){

    // Part 1: Updating the snake array & Food

    if(isCollide(snakeArr)){

        gameOverSound.play();

        musicSound.pause();

        inputDir =  {x: 0, y: 0};

        alert("Game Over. Press any key to play again!");

        snakeArr = [{x: 13, y: 15}];

        musicSound.play();

        score = 0;

    }

    // If you have eaten the food, increment the score and regenerate the food

    if(snakeArr[0].y === food.y && snakeArr[0].x ===food.x){

        foodSound.play();

        score += 1;

        if(score>hiscoreval){

            hiscoreval = score;

            localStorage.setItem("highscore", JSON.stringify(hiscoreval));

            hiscoreBox.innerHTML = "HighScore: " + hiscoreval;

        }

        scoreBox.innerHTML = "Score: " + score;

        snakeArr.unshift({x: snakeArr[0].x + inputDir.x, y: snakeArr[0].y + inputDir.y});

        let a = 2;

        let b = 16;

        food = {x: Math.round(a + (b-a)\* Math.random()), y: Math.round(a + (b-a)\* Math.random())}

    }

    // Moving the snake

    for (let i = snakeArr.length - 2; i>=0; i--) {

        snakeArr[i+1] = {...snakeArr[i]};

    }

    snakeArr[0].x += inputDir.x;

    snakeArr[0].y += inputDir.y;

    // Part 2: Display the snake and Food

    // Display the snake

    board.innerHTML = "";

    snakeArr.forEach((e, index)=>{

        snakeElement = document.createElement('div');

        snakeElement.style.gridRowStart = e.y;

        snakeElement.style.gridColumnStart = e.x;

        if(index === 0){

            snakeElement.classList.add('head');

        }

        else{

            snakeElement.classList.add('snake');

        }

        board.appendChild(snakeElement);

    });

    // Display the food

    foodElement = document.createElement('div');

    foodElement.style.gridRowStart = food.y;

    foodElement.style.gridColumnStart = food.x;

    foodElement.classList.add('food')

    board.appendChild(foodElement);

}

// Main logic starts here

musicSound.play();

let highscore = localStorage.getItem("highscore");

if(highscore === null){

    hiscoreval = 0;

    localStorage.setItem("highscore", JSON.stringify(hiscoreval))

}

else{

    hiscoreval = JSON.parse(highscore);

    hiscoreBox.innerHTML = "HighScore: " + highscore;

}

window.requestAnimationFrame(main);

window.addEventListener('keydown', e => {

    // Check if the pressed key is an arrow key or Enter key

    if (['ArrowUp', 'ArrowDown', 'ArrowLeft', 'ArrowRight', 'Enter'].includes(e.key)) {

        inputDir = { x: 0, y: 1 }; // Start the game

        moveSound.play();

        switch (e.key) {

            case "ArrowUp":

                console.log("ArrowUp");

                inputDir.x = 0;

                inputDir.y = -1;

                break;

            case "ArrowDown":

                console.log("ArrowDown");

                inputDir.x = 0;

                inputDir.y = 1;

                break;

            case "ArrowLeft":

                console.log("ArrowLeft");

                inputDir.x = -1;

                inputDir.y = 0;

                break;

            case "ArrowRight":

                console.log("ArrowRight");

                inputDir.x = 1;

                inputDir.y = 0;

                break;

            default:

                break;

        }

    }

});

**Future Enhancements:**

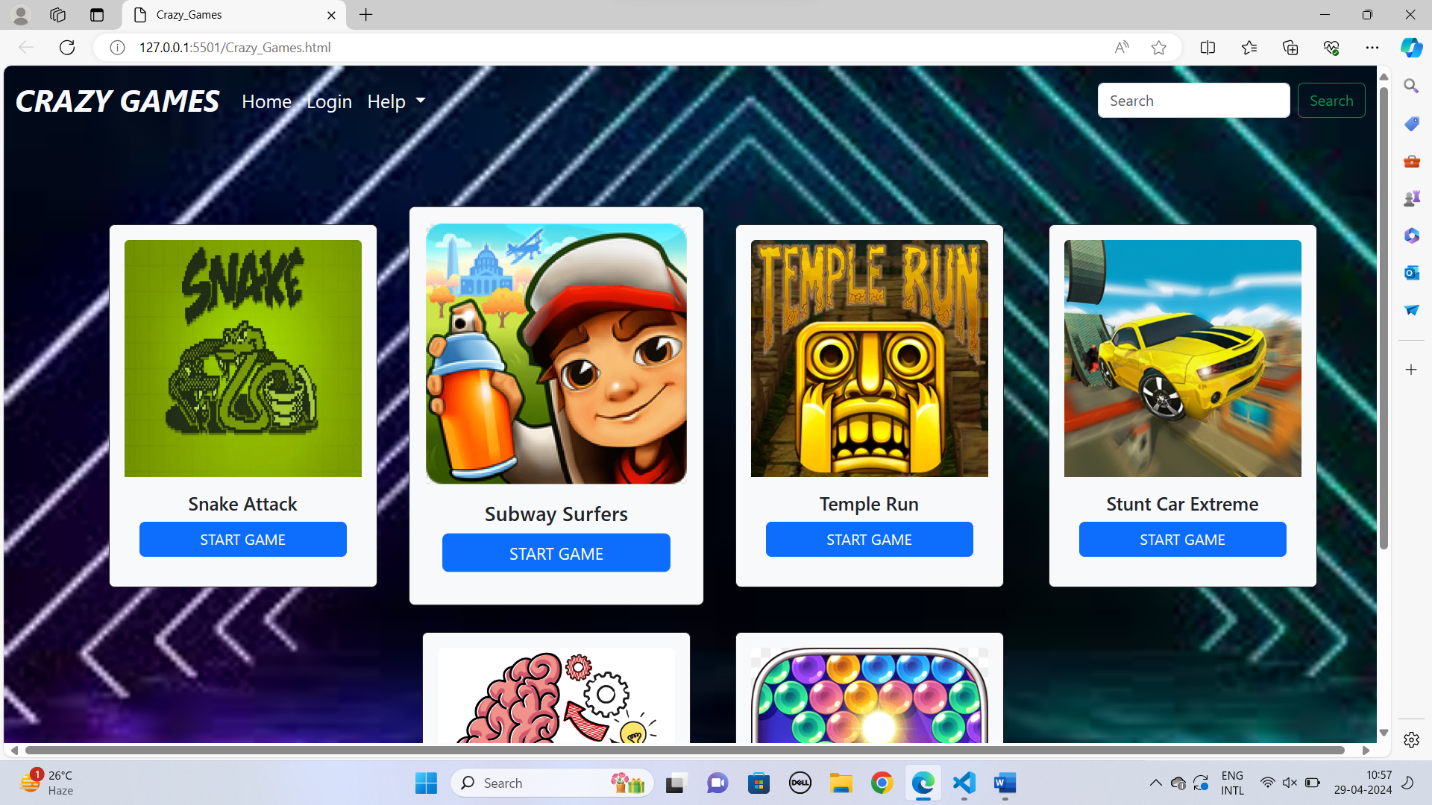
- Adding sound effects and background music for immersive gameplay.

- Implementing additional game features such as power-ups, obstacles, and multiplayer mode.

- Enhancing visual effects and animations to make the game more engaging.

- Integrating social sharing and leaderboard functionality for competitive gaming experience.

**Result:**

****

****

****

**Conclusion:**

In conclusion, the Snake Game project successfully demonstrates the integration of Bootstrap and JavaScript to create a classic arcade game with modern web technologies. By leveraging the power of Bootstrap for responsive design and UI components, combined with JavaScript for game logic and interactivity, we have developed an enjoyable and accessible gaming experience that can be enjoyed across different devices and screen sizes. With further enhancements and optimizations, the Snake Game has the potential to become even more engaging and captivating for players of all ages.

**References:**

**Bootstrap Documentation:** <https://getbootstrap.com/>

Javascript: <https://www.w3schools.com/js/>

Youtube Inspiration: <https://www.youtube.com/watch?v=2ZDnw6ifdSI&t=1447s&pp=ygUWc25ha2UgZ2FtZSBodG1sIGNzcyBqcw%3D%3D>