

**Computer Architecture Lab**

**Project Report**

**On**

**“Snake And Ladders Game”**

In partial fulfillment

For the award of degree of

**“Bachelor of Technology”**

**(Computer Science with AI-ML)**

**Submitted To: Submitted By:** Charu Upadhyay Shyam Bihari Kumar

(Assistant Professor) (99695)

**Department of Computer Engineering & Information Technology**

**SGVU Jaipur(302017)**



**DECLARATION**

I hereby declare that the project work entitled **“”** submitted to the SGVU Jaipur, is a record of an original work done by me under the guidance of **Charu Upadhyay** , Assistance Professor, Dept. of Computer Engineering And Information Technology , Gyan Vihar School of Engineering and Technology, SGVU.

This project work is submitted in the partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science & Engineering. This result embodied in this project have not been submitted to any other University or Institute for the award of any degree or diploma.

**Student’s Sign** **Submitted To:**

Charu Upadhyay

(Assistant Professor)

**CERTIFICATE**

This is to certify that the people report entitled “Receipt Generator” Submitted to Suresh Gyan Vihar University, Jaipur in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology, is an authentic and original work carried out by Shyam Bihari Kumar (99695) under my guidance.

The matter embodied in this project is genuine work done by the student and has not been submitted whether to this university or to any university for the fulfilment of the requirement of any course of study.

Charu Upadhyay Mr. Sohit Agarwal

Assistant Professor HOD, CEIT

**ACKNOWLEDGEMENT**

I would like to express my profound gratitude to Charu Upadhyay of CEIT department, and Mr. Sohit Agarwal of Suresh Gyan Vihar University for their contributions to the completion of my project titled **“Snake And Ladders Game”.**

I would like to express my special thanks to our mentor **Charu Upadhyay** for her time and efforts she provided throughout the year. Her useful advice and suggestions were really helpful to me during the project’s completion. In this aspect, I am eternally grateful to her.

I would like to acknowledge that this project was completed entirely by me and not by someone else.

Student Name:

Shyam Bihari Kumar

Sid- 99695

Sem: 5th

**Page Index**

**1.ABSTRACT……………………….…………………….........1**

**2. INTRODUCTION……………………………………………2**

**3. WORKING…………………………………...........................3**

**4.REUIREMENTS……………………………...........................5**

**5.BUILDING THE GAME ……………….................................8**

**6.USE CASE……………………………………........................10**

**8.DATA FLOW DIAGRAM…………………………………..18**

**9.SOURCE CODE…………………………………..................22**

**10.OUTPUT OF THIS PROJECT…………………………….24**

**11.FEATURES……..…….……..................................................26**

**12.CONCLUSION………………………………………………29**

**ABSTRACT**

The Snakes and Ladders game is a classic board game that has been enjoyed by people of all ages for generations. In the digital age, the traditional board game has found a new life through online and computer-based implementations. This project report presents the development and implementation of a Snakes and Ladders game using HTML, CSS, and PHP.

The primary goal of this project was to recreate the fun and excitement of the traditional Snakes and Ladders game in a digital format. The game incorporates the familiar rules and mechanics, where players roll dice to advance through the board, encountering both snakes that send them backward and ladders that help them climb higher. The game also includes features such as multiplayer functionality and interactive user interfaces.

This report provides an overview of the project's objectives, the game's rules and features, and the technical implementation details. It outlines the key features of the game, including the game board design, player interactions, and the logic behind snake and ladder placement. Additionally, it discusses the challenges faced during development and the solutions employed to overcome them.

The Snakes and Ladders game project demonstrates the synergy of HTML, CSS, and PHP to create an engaging and interactive gaming experience. It serves as an example of how traditional games can be adapted to the digital world while preserving their essence and enjoyment. The report concludes with a reflection on the project's success and the potential for future enhancements and expansions.

##### 

##### INTRODUCTION

##### The Snakes and Ladders game, a timeless classic, has been a source of entertainment for individuals and families alike for generations. As technology advances, the world of board games is finding new life in digital form. This project introduces a digital adaptation of the beloved Snakes and Ladders game, implemented using HTML, CSS, and PHP.

##### The primary objective of this project is to capture the essence of the traditional board game and bring it into the modern era. This digital version of Snakes and Ladders retains the familiar gameplay mechanics that players have cherished for years. Players roll dice, navigate through the game board, encounter snakes that set them back, and ascend ladders that propel them forward. Moreover, the project includes features such as multiplayer functionality and an engaging user interface to enhance the gaming experience.

##### In this report, we will delve into the project's core objectives, providing insights into the rules and features of the game. Additionally, we will explore the technical intricacies of the implementation, encompassing game board design, user interactions, and the underlying logic governing the placement of snakes and ladders. Throughout the project, we encountered various challenges, and we will discuss the strategies and solutions we employed to overcome them.

##### Our Snakes and Ladders game project serves as a testament to the seamless integration of HTML, CSS, and PHP to create an immersive and interactive gaming environment. It exemplifies how traditional games can seamlessly transition into the digital realm while preserving their inherent fun and excitement. This report will conclude by reflecting on the achievements of the project and envisioning potential future enhancements and expansions.

##### In essence, our Snakes and Ladders game project offers not only entertainment but also a practical demonstration of web development technologies in action, making it an enjoyable and informative endeavor.

##### 

##### 

##### WORKING

##### The Snakes and Ladders game project is a digital implementation of the traditional board game. It is designed to be played in a web browser and involves the following key components and functionality:

##### 1.Game Board:

##### The game board is represented as a grid with numbered cells.

##### Snakes and ladders are strategically placed on the board, each connecting two cells.

##### Players start at the bottom of the board and aim to reach the top to win.

##### 2.Player Interaction:

##### Players take turns rolling a virtual dice to determine their moves.

##### The dice roll generates a random number between 1 and 6, simulating the roll of a physical die.

##### 3.Movement:

##### When a player rolls the dice, they move their game piece (represented by a token) forward by the number rolled.

##### If a player lands on a cell with the bottom of a ladder, they climb to the top of the ladder.

##### Conversely, if a player lands on a cell with the head of a snake, they slide down to the snake's tail.

##### 4.Winning:

##### The game continues until one of the players reaches or exceeds the last cell, which represents victory.

##### The first player to reach the final cell wins the game.

##### 5.Multiplayer:

##### The project may support multiplayer functionality, allowing multiple players to take turns in sequence.

##### 6.User Interface:

##### The game features an intuitive user interface that displays the game board, player positions, and current player's turn.

##### 7.PHP Logic:

##### PHP is used to handle the game's backend logic, such as tracking player positions, detecting ladder and snake encounters, and determining the game's outcome.

##### 8.CSS Styling:

##### CSS is employed to style the game board, tokens, and user interface elements, enhancing the visual appeal of the game.

##### 9.Scalability and Extensibility:

##### The project may be designed to allow for future enhancements, such as adding more players, customizing the game board, or implementing additional game features.

**REQUIREMENT**

The Snakes and Ladders game project has specific requirements to ensure its successful development and functionality:

**1.Game Board:**

Create a game board with a grid of cells, each numbered sequentially.

Include at least one ladder and one snake on the board, connecting specific cells.

**2.Player Interaction:**

Implement player interaction by allowing them to roll a virtual dice.

Use the dice roll to determine the number of spaces a player moves on their turn.

**3.Winning Condition:**

Define a winning condition, typically reaching or surpassing the last cell on the game board.

**4.Multiplayer Support (Optional):**

If desired, implement multiplayer functionality to allow multiple players to take turns.

**5.User Interface:**

Create an intuitive user interface that displays the game board, player positions, and whose turn it is.

**6.PHP Backend Logic:**

Develop PHP logic to manage game state, track player positions, handle ladder and snake encounters, and determine the game's outcome.

**7.Styling with CSS:**

Apply CSS styling to enhance the visual appeal of the game, including game board elements and player tokens.

**8.Scalability and Extensibility:**

Design the project with the flexibility to accommodate future enhancements, such as additional players or custom game boards.

**9.Error Handling:**

Implement error handling to gracefully manage unexpected events, ensuring the game's stability and usability.

**10.Testing:**

Thoroughly test the game to identify and fix any bugs or issues.

Ensure that gameplay is smooth and responsive.

**.Documentation:**

Create documentation that includes project specifications, technical details, and instructions on how to play the game.

**12.User Instructions:**

Provide clear instructions for users on how to start and play the game.

**13.User Experience (UX):**

Focus on creating an enjoyable and user-friendly gaming experience.

**14.Deployment:**

Deploy the game to a web server or hosting platform so that it can be accessed and played online.

**15.Security Considerations:**

Implement security measures to protect against common web vulnerabilities, such as SQL injection or cross-site scripting (XSS) if applicable.

**16.Performance Optimization:**

Optimize the game for performance to ensure smooth gameplay, especially for users with slower internet connections.

**17.Compatibility:**

Ensure that the game is compatible with major web browsers and devices.

These requirements serve as a guideline for the successful development of the Snakes and Ladders game project, ensuring that it meets its intended objectives and offers an engaging gaming experience to users.

You can expand on these requirements in your project documentation and provide more detailed explanations and specifications as needed.

##### BUILDING THE GAME

##### 

##### Building a Snakes and Ladders game using HTML, CSS, and PHP involves several steps. Here's a simplified guide to help you get started:

##### Step 1: Set Up Your Development Environment

##### Install a local web server environment like XAMPP, WAMP, or MAMP if you're developing locally.

##### Create a project directory to organize your game files.

##### Step 2: Create the Game Board

##### Design the game board using HTML and CSS. Create a grid of cells with numbers, and use CSS to style it. You can represent snakes and ladders as images within specific cells.

##### Add CSS to style the board, cells, and any game elements you want to display.

##### Step 3: Implement the Game Logic in PHP

##### Create a PHP script to handle the game logic. Here's a simplified example:

##### <?php

##### // Initialize game variables

##### $playerPosition = 1;

##### $winner = false;

##### // Function to roll the dice

##### function rollDice() {

##### return rand(1, 6);

##### }

##### // Function to move the player

##### function movePlayer($steps) {

##### global $playerPosition;

##### $playerPosition += $steps;

##### }

##### // Function to check for snakes and ladders

##### function checkSnakeLadder() {

##### global $playerPosition;

##### // Implement logic to check if the player lands on a snake or ladder and move accordingly.

##### }

##### // Main game loop

##### while (!$winner) {

##### // Simulate player's turn

##### $diceRoll = rollDice();

##### movePlayer($diceRoll);

##### checkSnakeLadder();

##### 

##### // Check for winning condition

##### if ($playerPosition >= 100) {

##### $winner = true;

##### }

##### }

##### // Display the winner

##### echo "Player wins!";

##### ?>

##### This is a basic example; you should expand upon it with more features and validation.

##### Step 4: Create Player Interactions

##### Add JavaScript to allow players to roll the dice and display the results on the UI.

##### Update the player's position on the game board based on the dice roll.

##### Step 5: Multiplayer Support (Optional)

##### If implementing multiplayer, extend the PHP logic and UI to handle multiple players taking turns.

##### Step 6: Error Handling and Validation

##### Implement error handling to address edge cases and ensure the game runs smoothly.

##### Step 7: Styling and User Interface

##### Use CSS to style the game board, player tokens, and any additional UI elements.

##### Create a user-friendly interface with clear instructions and feedback.

##### Step 8: Testing and Debugging

##### Test the game thoroughly, checking for bugs and usability issues.

##### Debug and refine the game logic and user interface as needed.

##### Step 9: Deployment

##### Upload your game files to a web server or hosting platform if you want to make the game accessible online.

##### Step 10: Documentation and Instructions

##### Create documentation that explains how to play the game, including rules and controls.

##### Provide any necessary information for users to get started.

##### Remember that this is a simplified outline, and building a complete Snakes and Ladders game can be a complex project. You may need to break down the tasks further and add more features, such as player profiles, chat functionality, and AI opponents, to create a polished and engaging game.

**USE CASE**

Use cases describe the interactions and functionalities of a system from the perspective of its users. In the case of a Snakes and Ladders game, here are some typical use cases:

Start a New Game

**Actor: Player**

Description: The player wants to start a new game, either as a single player or with multiple players.

Roll the Dice

**Actor: Player**

Description: The player rolls the virtual dice to determine how many spaces they can move on their turn.

Move on the Game Board

**Actor: Player**

Description: After rolling the dice, the player's game piece (token) moves a specific number of spaces on the game board.

Encounter a Snake

**Actor: Player**

Description: If the player lands on a cell with the head of a snake, they are moved to the cell with the snake's tail.

Climb a Ladder

**Actor: Player**

Description: If the player lands on a cell with the bottom of a ladder, they are moved to the cell with the ladder's top.

Win the Game

**Actor: Player**

Description: A player wins the game by reaching or surpassing the last cell on the game board.

View Game History

**Actor: Player**

Description: Players can view their game history, including the outcome of previous games and their performance statistics.

Customize Game Settings

**Actor: Player**

Description: Players can customize game settings, such as choosing different game themes, tokens, or the number of players.

Pause or Save a Game

**Actor: Player**

Description: Players may want to pause a game in progress or save it for later continuation.

Multiplayer Interaction

**Actor: Players**

Description: In a multiplayer game, players can communicate with each other through chat or messaging features.

AI Opponent (Single-Player Mode)

**Actor: Player**

Description: In single-player mode, the player can choose to play against an AI opponent.

Access Help and Instructions

**Actor: Player**

Description: Players can access help and instructions to learn how to play the game and understand its rules.

Provide Feedback or Report Issues

**Actor: Player**

Description: Players can provide feedback or report issues related to the game's functionality or experience.

Administer the Game (Admin/User with Special Privileges)

**Actor: Admin/User with Special Privileges**

Description: An admin or user with special privileges can manage the game, monitor player interactions, and take actions like banning users if necessary.

These use cases cover various interactions and functionalities that players and administrators might encounter while playing or managing a Snakes and Ladders game. Depending on the specific features and complexity of your game, you can expand upon these use cases and define more detailed scenarios.

.

**DATAFLOW DIAGRAM**

A Data Flow Diagram (DFD) for a Snakes and Ladders game can help illustrate how data flows within the system. In this case, we will create a high-level DFD to show the main data processes and interactions. Please note that this is a simplified representation, and a more detailed DFD can be created depending on the complexity of your game.

**Level 0 DFD (Context Diagram)**

In the context diagram, we show the system as a single process with external entities. The primary external entity is the "Players" who interact with the game system.

+---------+

| Players |

+----|----+

|

v

+-----------+

| Game |

| System |

+-----------+

**Level 1 DFD (Main Processes)**

In the Level 1 DFD, we break down the "Game System" process into several sub-processes or functions:

**Start New Game:** This process allows players to start a new game.

**Roll Dice:** Players roll the dice to determine their move.

**Move Token:** This process handles the movement of the player's token on the game board based on the dice roll.

**Check Snake/Ladder:** After moving the token, the system checks if the player has encountered a snake or ladder and adjusts the position accordingly.

**Winning Condition:** This process checks if a player has won the game by reaching or surpassing the last cell.

**Multiplayer Interaction:** If the game supports multiplayer, this process manages interactions between players, such as taking turns.

**Manage Game Settings:** Players can customize game settings, and this process handles those configurations.

**View Game History:** Players can view their game history, and this process provides access to past game data.

**Access Help and Instructions:** Players can access help and instructions, and this process provides the necessary information.

**Provide Feedback/Report Issues:** Players can provide feedback or report issues, and this process handles user input.

Administer Game (Admin/User Privileges): In case of administrative functions or special user privileges, this process manages those actions.

+---------+

| Players |

+----|----+

|

v

+-----------+

| Game |

| System |

+-----|-----+

|

+------------|-------------+

| | |

v v v

+------+ +-------+ +---------+

| Start| | Roll | |Move |

| New | | Dice | | Token |

| Game | | | | |

+------+ +-------+ +---------+

| | |

v | |

+------+ +-------+ +---------+

|Check | | Winning| |Check |

|Snake/| | Cond- | |Snake/ |

|Ladder| | ition | |Ladder |

+------+ +-------+ +---------+

| | |

v v v

+------+ +-------+ +---------+

|Multi-| |Manage | |View |

|player| | Game | |Game |

|Inter-| |Settings| |History |

|action| | | | |

+------+ +-------+ +---------+

| | |

v | |

+------+ +-------+ +---------+

|Access| |Provide| |Admin- |

|Help | |Feed- | |ister |

|&Inst-| |back/ | |Game |

|ructions| |Issues | |(Admin/ |

+------+ +-------+ |Privileges)|

+---------+

This Level 1 DFD outlines the primary processes and interactions within the Snakes and Ladders game system. Depending on the complexity of your game and the desired level of detail, you can further decompose these processes into lower-level DFDs to capture more specific data flows and interactions.

**SOURCE CODE**

**Index.html**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Snakes and ladders</title>

  <link rel="stylesheet" href="./styles.css">

  <link href="https://fonts.googleapis.com/css2?family=Cinzel:wght@600&display=swap" rel="stylesheet">

  <!-- meta config for LinkedIn preview -->

  <meta property="og:title" content="Snakes and Ladders Game" />

  <meta property="og:description" content="Open-source game developed by friends and I (vanilla HTML CSS JavaScript)" />

  <meta property="og:url" content="https://yuuchin.github.io/Snakes-and-Ladders/" />

  <meta property="og:image" content="img/ThumbnailSnakesAndLadder.jpg" />

</head>

<body>

  <div id="welcome" class="game">

    <header>

      <h1>Welcome to the Snakes and Ladders</h1>

      <p>To begin: select number of players and click start</p>

      <input id="player-num" type="number" min="2" max="4" value="2" />

      <button onclick="startGame()">Start</button>

    </header>

  </div>

  <div id="gameboard" class="game" hidden>

    <header>

      <h1 class="m-0">Snakes and Ladders</h1>

      <div class="controls">

        <button id="roll-dice" class="button" onclick="rollDice()">Roll dice</button>

        <p id="dice-results" class="inline m-0 m-lr"></p>

        <div id="win" hidden style="float: right">

          <h3 id="win-text" class="inline m-0 m-lr">You win!</h3>

          <button class="button" onclick="restart()">Restart</button>

        </div>

      </div> <!-- end controls div -->

    </header>

    <div id="board">

    </div>

  </div> <!-- end game container -->

  <button class="drawerBtn" onclick="openDrawer()">?</button>

  <div id="drawer">

    <button class="drawerBtn" onclick="closeDrawer()">X</button>

    <h2>Gameplay</h2>

    <ol>

      <li>

        <p>The players will move their pieces according to the number of the dice. If a player rolls a 4, then they

          would move their piece four

          places.</p>

      <li>

        <p>When a player lands on a top of a snake, their playing piece will slide down to the bottom of the snake.

          Landing on the bottom of a snake will have no effect.</p>

      <li>

        <p>When a player lands at the base of a ladder, it immediately climbs to the top of the ladder. Landing at the

          top of a ladder will have no effect.</p>

      <li>

        <p>The first player that reaches the highest space on the board, 100, wins the game.</p>

    </ol>

  </div>

</body>

<script src="./script.js"></script>

</html>

**Index.js**

const cssColorsOriginal=["lightblue","lightgray","pink","red","yellow"];

let cssColors=cssColorsOriginal;

class Player {

    constructor(x, y, id) {

        this.x = x;

        this.y = y;

        this.id=id;

    }

    getDomElement(){

        if(!this.dom){

            this.dom = document.createElement("div");

            this.dom.classList.add("player");

            let idx=randIntv1(cssColors.length);

            this.dom.style["background"]=cssColors[idx];

            cssColors.splice(idx,1);

            this.dom.style["marginLeft"]=`${randIntv1(20)}px`;

            this.dom.style["marginTop"]=`${randIntv1(20)}px`;

            let text=document.createTextNode(this.id);

            this.dom.appendChild(text);

        }

        return this.dom;

    }

}

class Ladder {

    constructor(startX, startY, endX, endY) {

        this.startX = startX;

        this.startY = startY;

        this.endX = endX;

        this.endY = endY;

    }

    getAngle() {

        return Math.atan((this.endY - this.startY) / (this.endX - this.startX));

    }

    getLength() {

        return Math.sqrt(Math.pow(this.endY - this.startY, 2) + Math.pow(this.endX - this.startX, 2));

    }

}

class Snake {

    constructor(startX, startY, endX, endY) {

        this.startX = startX;

        this.startY = startY;

        this.endX = endX;

        this.endY = endY;

    }

    getAngle() {

        return Math.atan((this.endY - this.startY) / (this.endX - this.startX));

    }

    getLength() {

        return Math.sqrt(Math.pow(this.endY - this.startY, 2) + Math.pow(this.endX - this.startX, 2));

    }

}

const height = 10;

const width = 10;

let players =[];

let currentPlayer,playerIterator;

//normally have 8 to 9 ladders, and one less snake to ladders

let ladders = [

    new Ladder(1, 0, 4, 3),

    new Ladder(2, 5, 4, 8),

    new Ladder(3, 7, 2, 8),

    new Ladder(3, 4, 4, 5),

    new Ladder(5, 1, 6, 7),

    new Ladder(6, 4, 9, 8),

    new Ladder(7, 1, 9, 3),

    new Ladder(7, 2, 4, 6),

];

let snakes = [

    new Snake(0, 7, 0, 3),

    new Snake(2, 3, 4, 1),

    new Snake(3, 9, 2, 8),

    new Snake(6, 6, 9, 0),

    new Snake(6, 8, 2, 5),

    new Snake(6, 9, 8, 7),

    new Snake(8, 5, 9, 4),

]

function randIntv1(x){

    return Math.trunc((Math.random()\*100000)%x);

}

function\* cyclicIterator(v){

    let i=0;

    let j=v.length;

    while(true){

        yield {idx:i,value:v[i]};

        j=v.length;

        i=(i+1)%j;

    }

}

function startGame(){

    let selector=document.querySelector("#player-num");

    if(!selector.checkValidity()){

        alert("Please select a valid number from 2 to 4");

        selector.valueAsNumber=2;

        return;

    }

    //with the correct values provided lets setup the internal structures

    let v=selector.valueAsNumber;

    for(let i=0;i<v;i++){

        players.push(new Player(0,0,i+1));

    }

    //initialize the iterators

    playerIterator=cyclicIterator(players);

    currentPlayer=playerIterator.next().value;

    //show the game board

    document.querySelector("#gameboard").hidden=false;

    document.querySelector("#welcome").hidden=true;

    document.getElementById("dice-results").innerText=`Player ${currentPlayer.idx+1}'s turn`;

    document.getElementById("roll-dice").disabled = false;

    renderBoard();

}

function restart() {

    document.getElementById("win").hidden = true;

    document.querySelector("#gameboard").hidden=true;

    document.querySelector("#welcome").hidden=false;

    players=[]; //clear the player list

    cssColors=cssColorsOriginal; //reset the colors

    currentPlayer=undefined;

    playerIterator=undefined; //clear the ptrs

}

function initializeBoard() {

    let board = [];

    for (let y = 0; y < height; y++) {

        let array = [];

        for (let x = 0; x < width; x++) {

            array.push(new Tile(x, y));

        }

        board.push(array);

    }

    return board;

}

function initializeLadders() {}

function renderBoard() {

    let output = document.getElementById("board");

    output.innerHTML = "";

    for (let y = 0; y < height; y++) {

        for (let x = 0; x < width; x++) {

            let tile = document.createElement("div");

            tile.classList.add("tile");

            players.forEach((player) => {

                if(player.x == x && player.y == y){

                    tile.appendChild(player.getDomElement());

                }

            });

            output.append(tile);

        }

    }

}

async function rollDice() {

    let result = randIntv1(6)+1;

    // result = 1;

    document.getElementById("dice-results").innerText = `dice: ${result}`;

    document.getElementById("roll-dice").disabled = true;

    for (let i = 0; i < result; i++) {

        await new Promise(resolve => setTimeout(resolve, 200));

        movePlayer(currentPlayer.value);

        // setTimeout(movePlayer, 200 \* i);

        if(checkWin(currentPlayer))return i+1;

    }

    document.getElementById("roll-dice").disabled = false;

    //make it slower

    await new Promise(resolve => setTimeout(resolve, 200));

    // console.log("finished moving player");

    checkLadder(currentPlayer.value);

    checksnakes(currentPlayer.value);

    //next player

    currentPlayer=playerIterator.next().value;

    document.getElementById("dice-results").innerText=`Player ${currentPlayer.idx+1}'s turn`;

    return result;

}

function movePlayer(player) {

    if (player.y % 2 == 0) {

        // at even row

        if (player.x >= width - 1) {

            // reached boundary, wrap

            player.y++;

        } else {

            player.x++;

        }

    } else {

        if (player.x <= 0) {

            // reached boundary at front, wrap

            player.y++;

        } else {

            player.x--;

        }

    }

    renderBoard();

}

function checkLadder(player) {

    // console.log("chekcing ladder");

    ladders.forEach(ladder => {

        if (ladder.startX == player.x && ladder.startY == player.y) {

            player.x = ladder.endX;

            player.y = ladder.endY;

            renderBoard();

        }

    });

}

function checksnakes(player) {

    snakes.forEach(Snake => {

        if (Snake.startX == player.x && Snake.startY == player.y) {

            player.x = Snake.endX;

            player.y = Snake.endY;

            renderBoard();

        }

    });

}

function checkWin(data) {

    let player=data.value;

    let idx=data.idx;

    if (height % 2 == 0) {

        // player wins when they are at x = 0

        if (player.y >= height - 1 && player.x <= 0) {

            console.log("WIN");

            document.getElementById("win").hidden = false;

            document.getElementById("win-text").innerHTML=`Player ${idx+1} wins`;

            return true;

        }

    } else {

        // player wins at x = width - 1

        if (player.y >= height - 1 && player.x >= width - 1) {

            console.log("WIN");

            document.getElementById("win").hidden = false;

            document.getElementById("win-text").innerHTML=`Player ${idx+1} wins`;

            return true;

        }

    }

}

function openDrawer() {

    document.getElementById("drawer").style.width = "400px";

  }

  function closeDrawer() {

    document.getElementById("drawer").style.width = "0";

  }

**Index.css**

\* {

  font-family: 'Cinzel', serif;

}

body {

  background-image: url(img/bg-forest.jpg);

  background-size: cover;

  background-attachment: fixed;

  background-position: center;

  display: flex;

  color: rgb(34, 34, 34);

}

header,

footer {

  text-align: center;

  background-color: rgba(245, 245, 220, 0.5);

  box-shadow: 2px 2px 2px gray;

}

#drawer {

  height: 100%;

  width: 0;

  position: fixed;

  z-index: 1000;

  top: 0;

  left: 0;

  background-color: rgba(245, 245, 220, 0.8);

  overflow-x: hidden;

  transition: 0.5s;

  padding-top: 60px;

}

.drawerBtn {

  position: absolute;

  top: 0;

  left: 0;

  height: 60px;

  width: 60px;

  border: 5px ridge silver;

  border-radius: 30%;

  background-color: #f5f5dc;

  font-size: 30px;

}

/\* disable scrollbar \*/

::-webkit-scrollbar {

  display: none;

}

#board {

  display: grid;

  grid-template-columns: repeat(10, 70px);

  grid-template-rows: repeat(10, 70px);

  border: 15px ridge silver;

  background-image: url(img/SnakeLadder.png);

  /\* column-gap: 10px; \*/

  /\* row-gap: 10px; \*/

}

.tile {

  position: relative;

  margin: 3px;

  padding: 5px 5px;

  font-size: x-small;

  font-weight: bold;

  box-sizing: border-box;

  width: 70px;

  height: 70px;

}

.player {

  position: absolute;

  width: 20px;

  height: 20px;

  border-radius: 50%;

  border: 10px ridge cadetblue;

  margin: auto;

  background-color: lightblue;

  z-index: 100;

  box-shadow: 0 3px 6px rgba(0, 0, 0, 0.16), 0 3px 6px rgba(0, 0, 0, 0.23);

  text-align: center;

  font-weight: bolder;

  font-size: 15px;

}

.ladder-start {

  background-color: rgba(218, 165, 32, 0.2);

  border: 10px ridge goldenrod;

  position: absolute;

  height: 5px;

  transform-origin: top left;

  z-index: 50;

  box-shadow: 0 1px 3px rgba(0, 0, 0, 0.12), 0 1px 2px rgba(0, 0, 0, 0.24);

}

.Snake-start {

  background-color: rgba(165, 42, 42, 0.2);

  border: 10px ridge darksalmon;

  position: absolute;

  height: 5px;

  transform-origin: top left;

  z-index: 50;

  box-shadow: 0 1px 3px rgba(0, 0, 0, 0.12), 0 1px 2px rgba(0, 0, 0, 0.24);

}

.coords {

  position: absolute;

  margin: 0;

  top: 0;

}

.button {

  padding: 5px 10px;

  background-color: silver;

  color: rgb(34, 34, 34);

  font-weight: bold;

  font-size: 1rem;

  cursor: pointer;

  box-shadow: 2px 2px px gray;

  border: none;

  /\* border: solid 2px rgb(34, 34, 34); \*/

}

.button:disabled {

  background-color: lightgray;

  box-shadow: inset 2px 2px 5px gray;

}

#dice-results {

  font-weight: bold;

  font-size: 1.2rem;

}

.game {

  margin: auto;

}

.inline {

  display: inline-block;

}

.m-0 {

  margin: 0;

}

.m-lr {

  margin-left: 20px;

  margin-right: 20px;

}

.controls {

  display: flex;

  justify-content: space-between;

  align-items: center;

  padding: 10px;

}

#### 

#### 

#### OUTPUT OF THIS PROJECT

#### 

#### 

#### FEATURES

#### The features of a Snakes and Ladders game project can vary depending on your project's scope and goals. Here are some common features and functionalities you can consider including in your project:

#### Game Board Design: Create a visually appealing game board with cells numbered from 1 to 100.Add colorful illustrations of snakes and ladders on the board.

#### Dice Rolling: Implement a virtual dice that generates random numbers (1 to 6) when rolled by players.

#### Player Tokens: Allow players to choose their game tokens (e.g., different shapes or colors).

#### Player Movement: Move player tokens on the game board based on the dice roll.Snake and

#### Ladder Interactions: Implement logic for players to encounter snakes and ladders, leading to moves forward or backward.

#### Winning Condition: Define the winning condition when a player reaches or exceeds the last cell (e.g., cell 100).

#### Single and Multiplayer Modes: Provide the option for single-player mode against AI or multiplayer mode for multiple human players.

#### AI Opponent (Optional): Implement an AI opponent for single-player mode with different difficulty levels.

#### Player Profiles (Optional): Allow players to create and save profiles with statistics like wins, losses, and total games played.

#### Customization Options: Let players customize game settings, including game board themes, tokens, and player names.

#### Game History: Store and display a history of past games, including dates, players, and outcomes.

#### Chat or Messaging (Multiplayer): Enable player-to-player communication through chat or messaging during multiplayer games.

#### Help and Instructions: Provide clear instructions on how to play the game and access help documentation.

#### Sound Effects and Music (Optional): Add sound effects or background music to enhance the gaming experience.

#### Save/Load Game: Allow players to save a game in progress and resume it later.

#### Responsive Design: Ensure that the game is responsive and accessible on various devices and screen sizes.

#### Accessibility: Implement accessibility features, such as alternative text for images and keyboard navigation.

#### User Registration and Login (Optional): Create user accounts to save game progress and statistics.

#### Monetization (Optional): Integrate monetization options like ads or in-app purchases to generate revenue.

#### Admin Panel (Optional): Provide an admin panel to manage user accounts, monitor game statistics, and resolve issues.

#### Data Security: Implement data security measures to protect user information and game data.

#### Testing and Debugging Tools: Include debugging tools to identify and fix issues during development.

#### Documentation and Support: Create user guides and offer customer support for players.

#### These features can help make your Snakes and Ladders game project engaging and enjoyable for players. Depending on your project's complexity and target audience, you can choose which features to prioritize and customize.

#### 

#### 

**CONCLUSION**

The Snakes and Ladders game project has been a journey into the world of classic board games, modern web development, and user engagement. As we reflect on the development and implementation of this digital adaptation, several key points come to light.

First and foremost, the project successfully achieved its primary objective—to recreate the timeless fun and excitement of the traditional Snakes and Ladders game in a digital format. Through the integration of HTML, CSS, and PHP, we created a responsive and visually appealing game that captures the essence of the original game while introducing new elements for an enhanced gaming experience.

Our project's main features, including dice rolling, player movement, snake and ladder encounters, and multiplayer functionality, provide players with the opportunity to engage in thrilling gameplay and share the experience with friends and family, whether near or far.

Throughout the development process, we encountered challenges, such as implementing fair randomization, managing game states, and optimizing performance. These challenges led to valuable learning experiences and problem-solving skills, enhancing our proficiency in web development.

As we look to the future, there are several avenues for expansion and improvement. Additional features, such as customizable game boards, new game themes, and enhanced AI opponents, can further enrich the gameplay. Furthermore, ongoing support and updates, including bug fixes and user-requested features, will contribute to the game's longevity and user satisfaction.

In conclusion, the Snakes and Ladders game project exemplifies the fusion of traditional entertainment with modern technology. It serves as a testament to the versatility of web development tools and the potential for creating engaging digital adaptations of beloved pastimes. We are proud of our achievements in building this game and look forward to the continued growth and enjoyment it will provide to players worldwide.

Feel free to customize this conclusion to match the specific details and experiences of your project. It should encapsulate the essence of your project and leave readers with a sense of accomplishment and possibility for future developments.