Chapter - 1

INTRODUCTION

1. INTRODUCTION

The digital landscape has transformed the way we entertain ourselves, with gaming emerging as a ubiquitous pastime enjoyed by millions worldwide. In this era of interconnectedness, the internet serves as a gateway to immersive gaming experiences, transcending the limitations of traditional gaming platforms. It is within this context that "GameZ" emerges - an innovative mini-game website crafted using the powerful trifecta of HTML, CSS, and JavaScript. GameZ represents a convergence of creativity, technology, and user-centric design principles, aiming to redefine the gaming experience for enthusiasts of all ages and backgrounds. By harnessing the capabilities of web technologies, GameZ offers an accessible and engaging platform where players can dive into a myriad of captivating mini-games without the need for specialized hardware or software installations.

At its core, GameZ embodies the ethos of inclusivity, catering to a diverse audience of casual gamers, enthusiasts, and competitive players alike. Whether it's a quick puzzle challenge, an adrenaline-pumping arcade showdown, or a strategic brain teaser, GameZ boasts a curated selection of mini-games designed to captivate and challenge players of varying skill levels.

1.1 Current System

Traditional gaming platforms predominantly rely on standalone applications or console-based gaming systems. These systems offer immersive gaming experiences but often require specific hardware and software configurations. Furthermore, they may lack accessibility for users who do not possess the required hardware or are unable to afford dedicated gaming consoles.

1.2 Need of Proposed System

The proposed system, GameZ, addresses the need for a web-based gaming platform that offers accessibility, convenience, and a diverse range of mini-games for users of all demographics. By leveraging web technologies such as HTML, CSS, and JavaScript, GameZ aims to provide an inclusive gaming experience accessible through any standard web browser on various devices, including desktops, laptops, tablets, and smartphones.

1.3 Problem Definition

The challenge lies in developing a mini-game website that seamlessly integrates HTML, CSS, and JavaScript to provide an engaging and interactive gaming experience while ensuring compatibility across various browsers and devices. Additionally, the system must be designed with scalability, security, and user experience in mind to cater to a wide audience of gamers with varying preferences and technical proficiency levels.

Chapter - 2

SYSTEM DEVELOPMENT LIFE CYCLE

2. SYSTEM DEVELOPMENT LIFE CYCLE (SDLC)

The Software Development Life Cycle (SDLC) is a systematic process for planning, creating, testing, deploying, and maintaining information systems. Here's a high-level overview of how the SDLC could be applied to the development of "A Mini Game Website: GameZ".

i. Planning:

- **Objective:**
  - Define the purpose and goals of the project.
- **Activities:**
  - Identify system requirements.
  - Define the scope and limitations.
  - Determine hardware and software components (Browser compatible devices, IDE, Technologies and tools to be used etc.).

ii. Analysis:

- **Objective:**
  - Understand and document user needs and system requirements.
- **Activities:**
  - Conduct a feasibility study.
  - Define user stories and use cases.
  - Specify functional and non-functional requirements.

iii. Design:

- **Objective:**
  - Create a blueprint for the system based on requirements.
- **Activities:**
  - Architectural design of the system components.
  - Design data structures and algorithms.
  - Create schematics for hardware connections.
  - Plan for data storage and communication protocols.

iv. Implementation:

- **Objective:**
  - Build the system according to the design specifications.
- **Activities:**
  - Write HTML, CSS, and JS code for mini games.
  - Arrange code in a proper manner to get the desired output (Code Stiching).
  - Implement server-side scripts for data processing and storage.

v. Testing:

- **Objective:**
  - Verify that the system meets requirements and functions as intended.

- **Activities:**
  - Unit testing of code modules.
  - System testing to validate end-to-end functionality.
  - Address and fix any identified issues.

vi. Deployment:

- **Objective:**
  - Introduce the system into the operational environment.
- **Activities:**
  - Deploy the code with complete configuration to the browser environment.
  - Configure the server for production use.

vii. Maintenance and Support:

- **Objective:**
  - Ensure ongoing system performance and address any issues.
- **Activities:**
  - Monitor system performance.
  - Provide user support.
  - Implement updates or improvements based on feedback or changing requirements.

Considerations for Web Projects:

- Browser compatibility with the technology used.
- Documentation for code and software-hardware setup.
- Testing of the final product in a controlled environment before actual deployment.

The exact details and steps may vary based on the specific requirements and complexity of the 'A Mini Game Website: GameZ'. This SDLC model provides a structured approach to guide the development process from initial planning to ongoing maintenance.

2.1 SDLC Models

n SDLC Model also termed as process model, it is a pictorial and diagrammatic representation of the software life cycle. A life cycle model represents all the methods required to make a software product transit through its life cycle stages. It also captures the structure in which these methods are to be undertaken.

There are different types of System Development Life Cycle Models are available for software/system development, some of them are:

- Linear Waterfall Model
- Iterative Model
- Rapid Application Development Model
- Agile Model
- Spiral Model

These are some popular SDLC Models which are used for software/system development nowadays. Each model offers advantages and drawbacks depending on the project requirements, timelines, and team dynamics.

2.2 SDLC Model Used

For the development of GameZ, an **Agile** SDLC model was chosen. Agile methodologies, such as Scrum or Kanban, prioritize iterative development, collaboration, and flexibility. Agile allows for rapid prototyping, continuous feedback, and the ability to adapt to changing requirements, making it well-suited for dynamic projects like GameZ.
The meaning of Agile is swift or versatile. "Agile process model" refers to a software development approach based on iterative development. Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.

Phases of Agile Model

1. Requirement Analysis: In this step, the development team must gather the requirements, by interaction with the customer. development team should plan the time and effort needed to build the project. Based on this information you can evaluate technical and economic feasibility.

2. Design: In this step, the development team will use user-flow-diagram or high-level UML diagrams to show the working of the new features and show how they will apply to the existing software. Wireframing and designing user interfaces are done in this phase.

3. Development/ Iteration: In this step, development team members start working on their project, which aims to deploy a working product.

4. Quality Assurance/ Testing: Testing involves Unit Testing, Integration Testing, and System Testing. A brief introduction of these three tests is as follows:

- Unit Testing: Unit testing is the process of checking small pieces of code to ensure that the individual parts of a program work properly on their own. Unit testing is used to test individual blocks (units) of code.
- Integration Testing: Integration testing is used to identify and resolve any issues that may arise when different units of the software are combined.
- System Testing: Goal is to ensure that the software meets the requirements of the users and that it works correctly in all possible scenarios.

5. Deployment: In this step, the development team will deploy the working project to end users.

The time required to complete an iteration is known as a Time Box. Time-box refers to the maximum amount of time needed to deliver an iteration to customers. So, the end date for an iteration does not change. However, the development team can decide to reduce the delivered functionality during a Time-box if necessary to deliver it on time. The Agile model's central principle is delivering an increment to the customer after each Time-box.

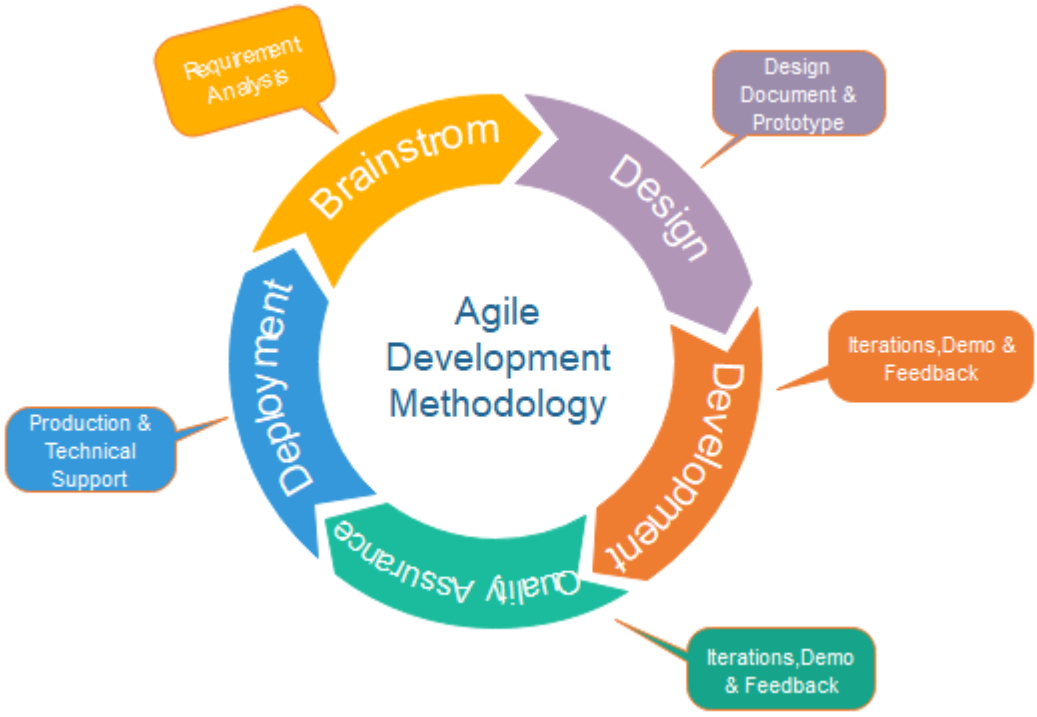


Figure 1: Agile Model

Overall, the Agile model prioritizes delivering value to customers quickly, adapting to change efficiently, and fostering collaboration and transparency throughout the development lifecycle. It has become increasingly popular in the software industry for its ability to support innovation, reduce risk, and deliver better outcomes for stakeholders.

Chapter - 3

ANALYSIS

3. ANALYSIS

The analysis phase of "GameZ" involves a comprehensive examination of various aspects related to the development and implementation of the mini-game website. This phase encompasses requirement analysis, specification, and use case analysis to establish a clear understanding of the project's objectives, scope, and user expectations.

3.1 Requirement Analysis

Requirement analysis involves understanding and documenting the needs, expectations, and constraints of stakeholders regarding the features, functionality, and user experience of GameZ. This phase includes gathering requirements through stakeholder interviews, surveys, market research, and competitor analysis.

1. Gaming Preferences: Understanding the diverse preferences of potential users, including preferred game genres, difficulty levels, and desired features such as multiplayer functionality or social sharing options.

2. Technical Requirements: Identifying the technical specifications and constraints of the project, such as browser compatibility, device responsiveness, performance benchmarks, and scalability considerations.

3. User Experience: Evaluating user experience factors such as ease of navigation, visual aesthetics, responsiveness, and accessibility to ensure an engaging and enjoyable gaming experience for all users.

3.2 Requirement Specification

Based on the analysis conducted in the previous phase, detailed requirement specifications were documented for GameZ. These specifications outline the specific features, game types, user interactions, technical specifications, performance metrics, and acceptance criteria for the project.

- **Functional Requirements:**

Clearly defining the specific features, functionalities, and interactions expected within GameZ, such as game selection menus, gameplay mechanics, scoring systems, and user authentication mechanisms.

1. Game Selection: Users should be able to browse and select from a variety of mini-games available on GameZ.

2. Gameplay Mechanics: Each mini-game should have clear gameplay mechanics, including controls, objectives, and win/lose conditions.

3. User Authentication: Registered users should have the ability to log in to their accounts to access personalized features such as saving game progress, tracking high scores, and interacting with social features.

4. Accessibility: GameZ should adhere to accessibility standards to ensure that all users, including those with disabilities, can access and enjoy the mini-games with ease.

5. Responsive Design: The website layout and interface should be responsive and adapt seamlessly to different screen sizes and devices, providing a consistent user experience across desktops, laptops, tablets, and smartphones.

6. Multiplayer Functionality (Optional): GameZ may include multiplayer functionality for select mini-games, allowing users to compete or collaborate with friends and other players in real-time.

- **Non-functional Requirements:**

Outlining the non-functional aspects of the system, including performance benchmarks, security requirements, browser compatibility, and compliance with web standards and regulations.

1. Performance: GameZ should load quickly and operate smoothly, with minimal latency and optimal performance even under high user traffic.

2. Security: User data, including login credentials and personal information, should be securely stored and protected from unauthorized access or data breaches.

3. Browser Compatibility: GameZ should be compatible with a wide range of web browsers, including popular options such as Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge, ensuring accessibility for users on different platforms.

4. Scalability: The website architecture should be designed to accommodate growth and scalability, allowing for the addition of new mini-games, features, and users without compromising performance or stability.

5. User Experience: GameZ should prioritize user experience, with intuitive navigation, visually appealing interfaces, and engaging gameplay mechanics that keep users coming back for more.

6. Maintainability: The codebase and architecture of GameZ should be well-organized and documented, facilitating easy maintenance, updates, and future enhancements by the development team.

3.3 Use Case Analysis

Use case analysis involves defining the various scenarios and interactions between users and the system. This phase helps identify the primary actors, system functionalities, and the flow of activities within GameZ. Use cases are typically represented using diagrams such as Unified Modeling Language (UML) use case diagrams, sequence diagrams, and activity diagrams. This phase helps identify primary actors, system functionalities, and potential edge cases. Key components of use case analysis for GameZ include:

3.3.1 Use Case Diagram

Actors:

- **User:** The person interacting with GameZ.
- **GameZ:** The microcontroller responsible for measuring garbage level.
- **System/Server:** A localhost/server that is responsible for data rendered to the server (GameZ).

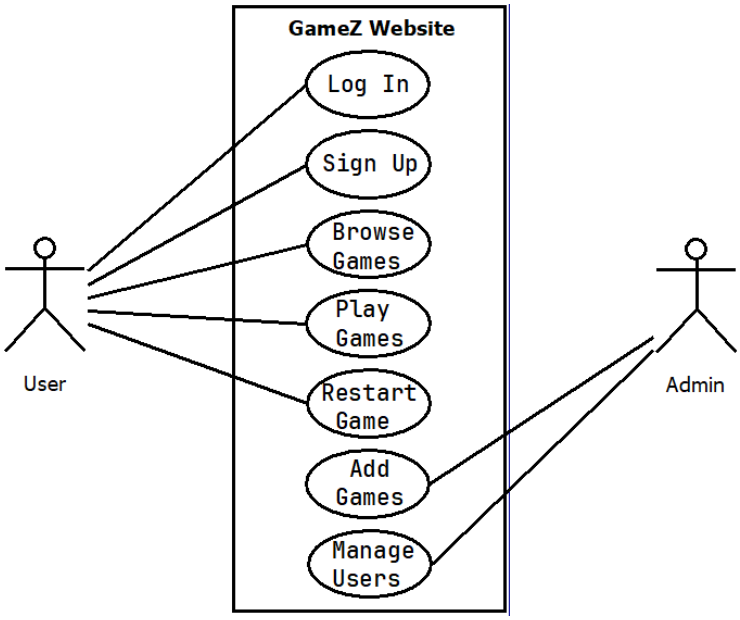


Figure 2: Use Case Diagram

3.3.2 Use Case Description

1. Browse Mini-Games:

- **Actors:** Guest Users, Registered Users
- **Description:** Users can browse the available mini-games on GameZ, view descriptions, screenshots, and ratings, and select a game to play.
- **Flow:**
  - User navigates to the GameZ website.
  - User browses the list of available mini-games.
  - User clicks on a mini-game to view more details.

- User decides whether to play the selected game or continue browsing.

2. Login to Account:

- **Actors:** Registered Users
- **Description:** Registered users can log in to their accounts to access personalized features such as saving game progress, tracking high scores, and interacting with social features.
- **Flow:**
  - User clicks on the "Login" button on the GameZ homepage.
  - User enters their username and password.
  - System validates the credentials and logs the user into their account.
  - User gains access to personalized features and content.

3. Play Mini-Game:

- **Actors:** Guest Users, Registered Users
- **Description:** Users can play mini-games hosted on GameZ, following the gameplay mechanics and objectives of each game.
- **Flow:**
  - User selects a mini-game from the available options.
  - User starts the game and follows the on-screen instructions.
  - User interacts with the game using keyboard controls, mouse clicks, or touch gestures.
  - User completes the game objectives and receives a score or outcome based on their performance.

This use case analysis provides an overview of the key interactions and scenarios within ‘GameZ’.

Chapter - 4

DESIGN

4. DESIGN

The design phase encompasses the conceptualization and visualization of GameZ's architecture, user interface, and game mechanics. This phase translates the requirements and use cases into tangible design elements, ensuring coherence, usability, and aesthetic appeal.

4.1 System Flow Diagram

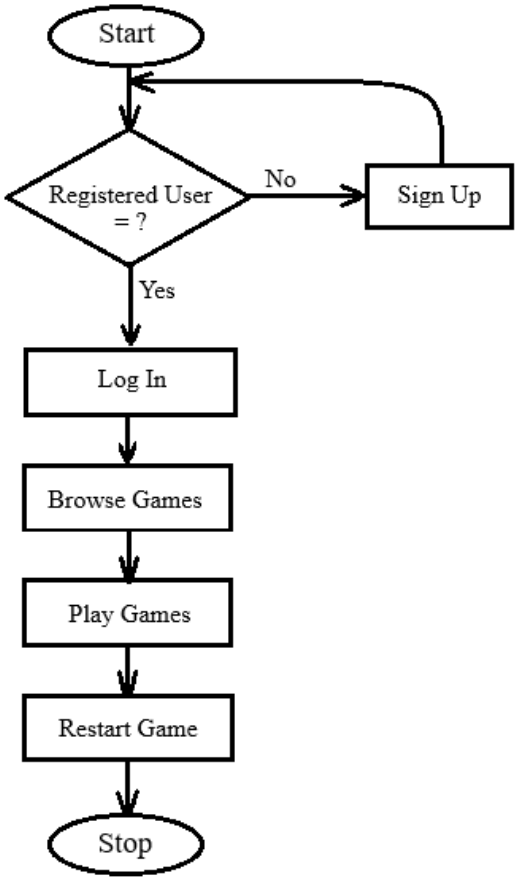


Figure 3: System Flow Diagram

The System Flow Diagram illustrates the high-level flow of activities and interactions within the GameZ website. It provides an overview of the main processes and user interactions, including game selection, user authentication, gameplay, and social sharing. The flow diagram visually represents the sequence of steps users take when interacting with the website, from accessing the homepage to playing mini-games and sharing achievements on social media platforms.

4.2 Modules Identified

• **User Authentication Module:** This module handles user authentication and account management functionalities, including user registration, login, logout, and password recovery.

• **Game Selection Module:** The Game Selection module allows users to browse and select from a variety of mini-games available on GameZ. It includes features such as game categories, search functionality, and game descriptions.

• **Gameplay Module:** The Gameplay module encompasses the core gameplay mechanics and features of individual mini-games hosted on GameZ. It includes game logic, controls, scoring systems, and win/lose conditions.

• **Leaderboard Module:** The Leaderboard module displays the highest scores achieved by players for each mini-game, promoting competition and motivation among users. It includes features such as score tracking, ranking, and user profiles.

4.3 E-R Diagram

The Entity-Relationship (E-R) Diagram depicts the relationships between different entities (or tables) in the GameZ database. It illustrates the structure of the database schema, including tables for users, games, scores, and social interactions. The E-R diagram visually represents the relationships between these entities, such as one-to-many relationships between users and scores, and many-to-many relationships between users and games.

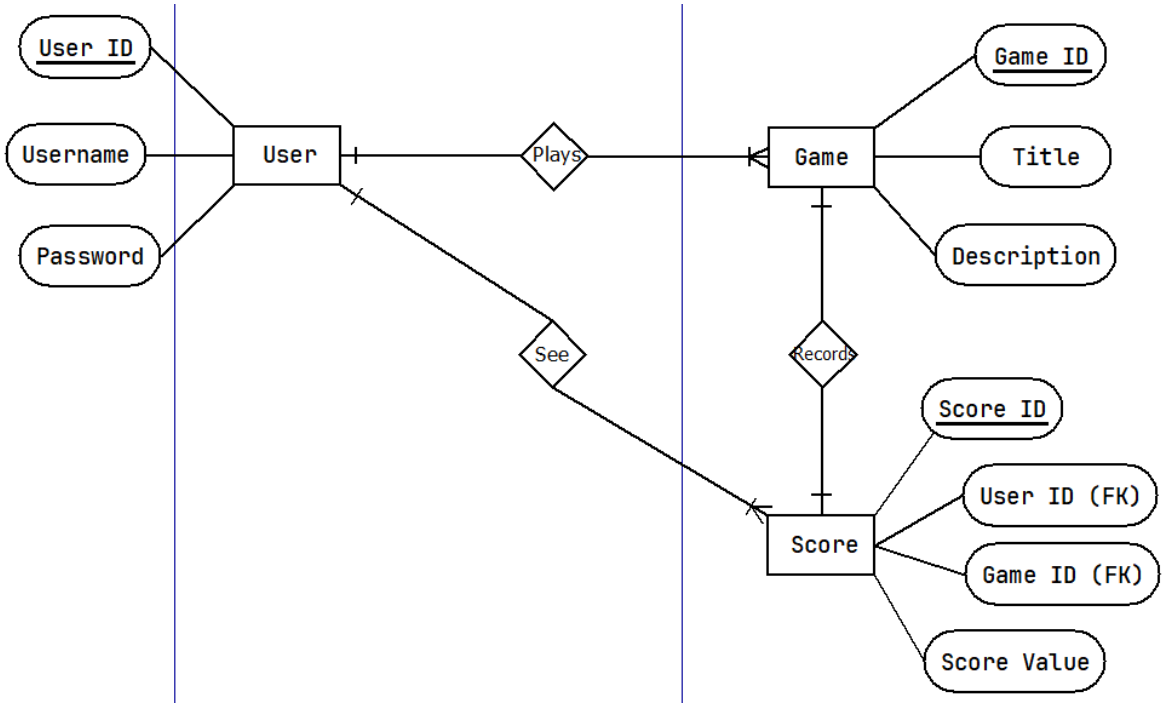
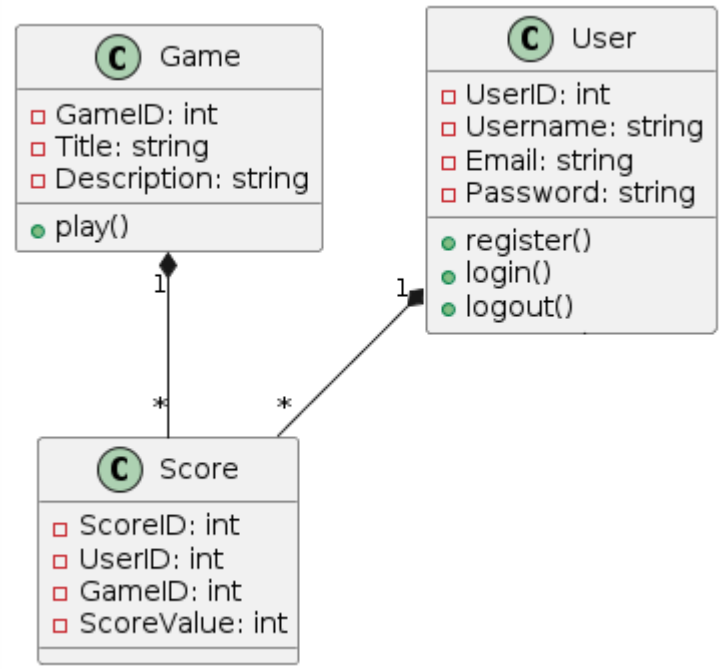


Figure 4: E-R Diagram

4.4 Class Diagram

The Class Diagram represents the structure and relationships of classes (or objects) in the GameZ system. It outlines the classes, attributes, methods, and associations between different components of the system, including user accounts, mini-games, scoring systems, and social sharing functionalities. The Class Diagram provides a blueprint for the object-oriented design of



the system, facilitating communication between developers and guiding the implementation process.

Figure 5: Class Diagram

4.5 Sequence Diagram

The Sequence Diagram illustrates the sequence of interactions between different components of the GameZ system during a specific user scenario, such as playing a mini-game and sharing the high score on social media. It depicts the flow of messages and method calls between objects or modules, showing the order of execution and the exchange of data between components. The Sequence Diagram provides a detailed view of the system's behavior and interactions, helping developers understand the underlying logic and dependencies of the system.

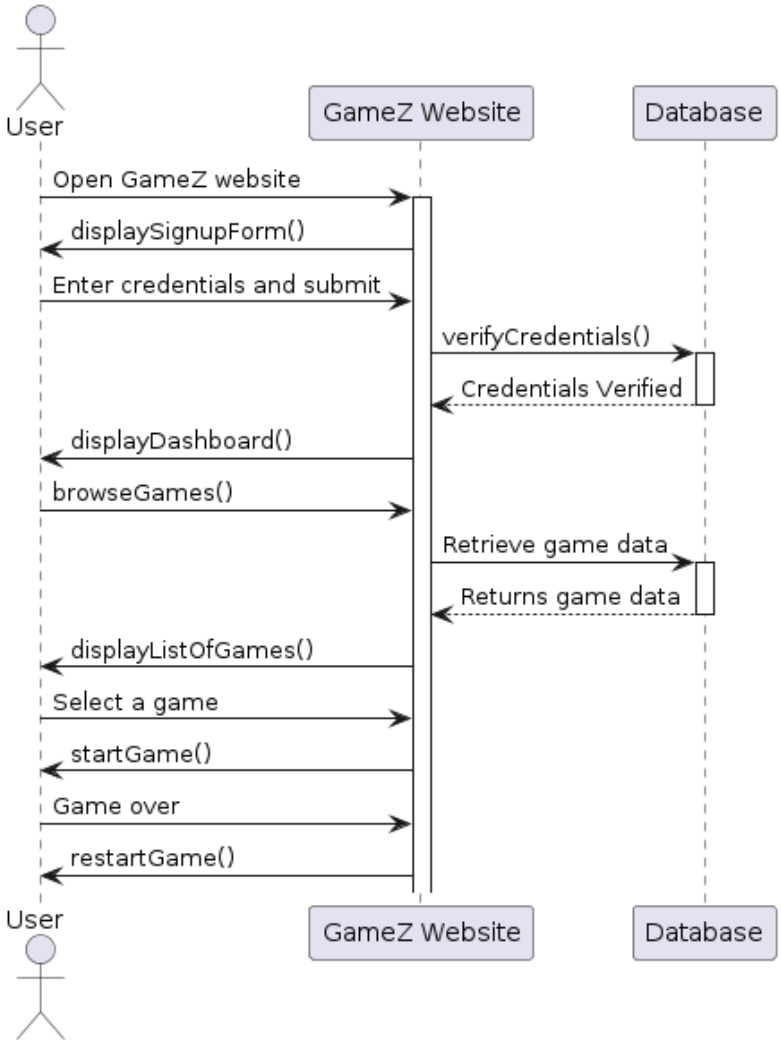


Figure 6: Sequence Diagram

Here all the diagrams e.g. System flow diagram, E-R diagram, Class Diagram, Sequence diagram and different modules used in the implementation are identified. This overview explains the design of the system well.

Chapter - 5

IMPLEMENTATION

5. IMPLEMENTATION

5.1 Platform Used

5.1.1 Hardware Platform

GameZ is designed to run on any standard computing device with internet connectivity. This includes desktop computers, laptops, tablets, smartphones, and gaming consoles that support modern web browsers.

5.1.2 Software Platform

Software platforms like IDE (e.g. Visual Studio Code, Vim, Sublime etc.) for design and construction of the project and Browsers (e.g. Chrome, Safari, Edge etc.) to run and test the final product/project.

Here are the descriptions of the game components implemented using hardware and software platforms with their snapshots.

1. Flappy Bird:

- **Description:** Flappy Bird is a side-scrolling game where the player controls a bird, guiding it through a series of obstacles by tapping the screen or pressing a key to make the bird flap its wings and ascend. The objective is to navigate the bird through narrow gaps between pipes without colliding with them. The game ends if the bird collides with a pipe or falls to the ground.

- **Implementation:** HTML canvas element is used to render the game graphics and animations. JavaScript is utilized to handle game logic, including player input, obstacle generation, collision detection, and scoring.

2. Guess the Number:

- **Description:** Guess the Number is a classic guessing game where the player attempts to guess a randomly generated number within a specified range. The game provides feedback to the player after each guess, indicating whether the guessed number is higher or lower than the target number. The player continues guessing until they correctly identify the target number.

- **Implementation:** HTML provides the user interface elements for displaying game instructions, input fields for guessing numbers, and feedback messages. JavaScript generates a random number, compares the player's guesses with the target number, and updates the interface accordingly.

3. Quiz Game:

- **Description:** The Quiz Game presents players with a series of multiple-choice questions on various topics, such as general knowledge, trivia, or educational subjects. Players select their answers from a list of options, and the game provides immediate feedback on the correctness of each answer. At the end of the quiz, players receive a score based on the number of correct answers.

- **Implementation:** HTML displays the quiz questions, answer options, and buttons for player interaction. JavaScript handles the logic for presenting questions, validating answers, calculating scores, and advancing to the next question.

4. Rock Paper Scissors:

- **Description:** Rock Paper Scissors is a hand game typically played between two players, where each player simultaneously forms one of three shapes with their hand: rock, paper, or scissors. The winner is determined based on the rules: rock crushes scissors, scissors cuts paper, and paper covers rock. The game can be played against the computer or another player.

- **Implementation:** HTML provides the game interface, including buttons for selecting rock, paper, or scissors. JavaScript generates the computer's choice, compares it with the player's choice, and determines the winner based on the game rules.

5. Slide Puzzle:

- **Description:** Slide Puzzle is a classic puzzle game where players rearrange shuffled tiles to form a complete picture or pattern. The puzzle consists of a grid of square tiles, with one tile missing to allow sliding of adjacent tiles into the empty space. The objective is to reassemble the tiles into the original configuration within the shortest number of moves.

- **Implementation:** HTML displays the puzzle grid and tiles, CSS provides styling for the tiles and background image, and JavaScript handles the logic for shuffling tiles, detecting moves, and checking for puzzle completion.

6. Snake Game:

- **Description:** Snake Game is a classic arcade game where players control a snake that grows in length as it consumes food items placed on the game board. The snake moves continuously in a specific direction, and players must navigate it to avoid colliding with walls, obstacles, or its own body. The game ends if the snake collides with a boundary or itself.

- **Implementation:** HTML canvas element is used to render the game board and snake graphics. JavaScript handles the logic for snake movement, collision detection, food generation, and score calculation.

7. Tic Tac Toe:

- **Description:** Tic Tac Toe is a two-player game played on a 3x3 grid, where players take turns marking spaces with their respective symbols (usually X and O). The objective is to form a horizontal, vertical, or diagonal line of three matching symbols. The game ends when one player achieves this goal or the grid is filled with no winner.

- **Implementation:** HTML provides the game grid and buttons for player interaction. JavaScript manages the game state, checks for winning conditions, updates the grid with player moves, and determines the outcome of each game.

8. Tetris:

- **Description:** Tetris is a tile-matching puzzle game where players manipulate falling tetramine's (geometric shapes made up of four square blocks) to create complete horizontal lines without gaps. When a line is completed, it disappears, and any blocks above it collapse downwards. The game ends if the stack of blocks reaches the top of the playing field.

- **Implementation:** HTML canvas element is used to render the game board and tetrominoes. JavaScript controls the falling tetrominoes, detects collisions, clears completed lines, updates the game board, and manages player input for rotating and moving tetrominoes.

5.2 Implementation Level Details

Detailed implementation strategies were employed to integrate HTML, CSS, and JavaScript effectively. This involved adhering to web standards, best practices, and design patterns to ensure maintainability, scalability, and cross-browser compatibility.

- **Module:** This code module is of home page structure and design of GameZ.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>GameZ: A Mini Game Website</title>
    <link href="https://cdnjs.cloudflare.com/ajax/libs/tailwindcss/v3.4.3/tailwind.min.css"
rel="stylesheet">
    <style>
        @import
url('https://fonts.googleapis.com/css2?family=Nunito:ital,wght@0,200..1000;1,200..1000&dis
play=swap');
:root {
    --primary-color: rgb(11, 11, 59);
    --secondary-color: rgb(160, 0, 160);
    --main-bg-color: rgb(11, 11, 80);
    --font-family-1: 'Nunito', 'sans-serif';
    --font-family-2: 'Gilroy', 'sans-serif';
}
* {
    margin: 0px;
    padding: 0px;
    box-sizing: border-box;
}
```

```
body {
    max-width: 100vw;
    height: 100vh;
    font-family: var(--font-family-1);
}
header {
    padding: 3px 0px;
    color: white;
    background-color: var(--primary-color);
}
main {
    padding: 2rem 2rem;
    background: var(--main-bg-color);
    color: white;
}
footer {
    padding: 2vh 0px;
    background-color: var(--main-bg-color);
    color: white;
    text-align: center;
}
#first-half-nav {
    display: flex;
}
.logo:active {
    scale: 0.8;
    transition: all ease 0.5s;
}
.link:active {
    text-decoration: none;
    color: white;
}
#nav-bar ul li {
    padding: 10px 20px;
}
.link:hover {
    font-weight: 600;
    border-bottom: 3px solid var(--secondary-color);
}
#search-bar {
```

```
    width: 20vw;
    padding: 2px;
    margin: auto;
    color: black;
    border-radius: 5px;
}
#search-btn {
    background-color: var(--secondary-color);
    padding: 2px 5px;
    margin: auto 5px;
    border-radius: 5px
}
#log-in {
    background-color: transparent;
    padding: 1px 5px;
    margin: auto;
    margin-left: 15px;
    color: var(--secondary-color);
    border: 2px solid var(--secondary-color);
    border-radius: 5px;
    float: right;
}
#sign-up {
    background-color: var(--secondary-color);
    padding: 2px 5px;
    margin: auto 5px;
    color: white;
    border-radius: 5px;
}
.btn {
    &:hover {
        filter: grayscale(0.2);
    }
    &:active{
        scale: 0.8;
        transition: all ease 0.5s;
    }
}
.line {
    border-top: 1px solid grey;
```

```
}
.carousel {
    display: flex;
    align-items: center;
}
.slider {
    width: 50%;
    height: 45vh;
    margin: 5vh auto;
    box-shadow: 0px 0px 3px grey;
    position: relative;
    overflow: hidden;
}
.slide {
    width: 100%;
    transition: all ease 1s;
    position: absolute;
}
.slider-nav {
    text-align: center;
}
.slider-nav button {
    background-color: transparent;
    color: white;
    padding: 3px 5px;
    margin: 3vh 3px 0px;
    border: 2px solid white;
    border-radius: 5px;
    &:hover {
        box-shadow: 0 0 10px aqua;
    }
}
.nav-items-link {
    padding-right: 1em;
    font-size: xx-large;
    font-weight: bold;
    border-right: 3px solid var(--secondary-color);
}
.main-container {
    padding: 3rem;
```

```
    background-color: var(--main-bg-color);
    color: white;
    display: grid;
    row-gap: 4vw;
    column-gap: 2vw;
    grid-template-columns: repeat(4, 20vw);
    grid-template-rows: repeat(2, 20vw);
}
.container {
    height: fit-content;
    text-align: center;
    border: 2px solid grey;
    border-radius: 10px;
    overflow: hidden;
    &:hover {
        box-shadow: 0 0 10px aqua;
        filter: grayscale(0.6);
    }
}
.game-name {
    background: var(--secondary-color);
    font-size: larger;
    font-weight: bold;
}
    </style>
</head>
<body>
    <header class="">
        <nav id="nav-bar" class="flex">
            <div id="first-half-nav" class="w-fit">
                <span class="logo text-lg font-bold px-6 py-2"><a href="#">GameZ</a></span>
                <ul class="nav-items flex pl-32">
                    <li class="link px-1 py-0"><a href="#dashboard">Dashboard</a></li>
                    <li class="link px-1 py-0"><a href="#contact">Contact</a></li>
                    <li class="link px-1 py-0"><a href="#about">About</a></li>
                    <li class="link px-1 py-0"><a href="#help-&-support">Help &
Support</a></li>
                </ul>
            </div>
            <div id="second-half-nav" class="search-div flex pl-36">
```

```
            <input id="search-bar" class="" type="text" placeholder="Search">
            <button id="search-btn" class="btn">Search</button>
            <button id="log-in" class="btn">Log In</button>
            <button id="sign-up" class="btn">Sign Up</button>
          </div>
        </nav>
        <div class="line"></div>
        <div class="carousel">
          <div class="slider-nav"><button class="btn"
onclick="goPrev()">Prev</button></div>
          <div class="slider">
            <img src="../img/Background-main.jpg" alt="Image" class="slide">
            <img src="../img/Background-main.jpg" alt="Image" class="slide">
            <img src="../img/Background-main.jpg" alt="Image" class="slide">
            <img src="../img/Background-main.jpg" alt="Image" class="slide">
          </div>
          <div class="slider-nav">
            <button class="btn" onclick="goNext()">Next</button>
          </div>
        </div>
    </header>
    <main>
        <div id="dashboard" class="inline-block nav-items-link">
          <h1 class="">Dashboard</h1>
        </div>
        <div class="main-container">
          <div class="container">
            <a href="./Tic-Tac-Toe/tic-tac-toe.html" target="_blank">
              <h3 class="game-name">Tic-Tac-Toe</h3>
              <img src="./Tic-Tac-Toe/img/tic-tac-toe-logo.png" alt="">
            </a>
          </div>
          <div class="container">
            <a href="./Rock-Paper-Scissor/rock-paper-scissor.html" target="_blank">
              <h3 class="game-name">Rock Paper Scissor</h3>
              <img src="./Rock-Paper-Scissor/img/rock-paper-scissor-logo.jpg" alt="">
            </a>
          </div>
          <div class="container">
            <a href="./Snake-Game/snake-game.html" target="_blank">
```

```
            <h3 class="game-name">Snake Game</h3>
            <img src="./Snake-Game/img/snake-game-logo.jpg" alt="">
          </a>
        </div>
        <div class="container">
          <a href="./Guess-The-Number/guess-the-number.html" target="_blank">
            <h3 class="game-name">Guess The Number</h3>
            <img src="./Guess-The-Number/img/guess-the-number-logo.png" alt="">
          </a>
        </div>
        <div class="container">
          <a href="./Flappy-Bird/flappy-bird.html" target="_blank">
            <h3 class="game-name">Flappy Bird</h3>
            <img src="./Flappy-Bird/img/flappy-bird-logo.jpg" alt="">
          </a>
        </div>
        <div class="container">
          <a href="./Quiz-Game/quiz-game.html" target="_blank">
            <h3 class="game-name">Quiz Game</h3>
            <img src="./Quiz-Game/img/quiz-game-logo.jpg" alt="">
          </a>
        </div>
        <div class="container">
          <a href="./Slide-Puzzle/slide-puzzle.html" target="_blank">
            <h3 class="game-name">Slide Puzzle</h3>
            <img src="./Slide-Puzzle/img/slide-puzzle-logo.jpg" alt="">
          </a>
        </div>
      </div>
      <div id="about" class="inline-block nav-items-link">
        <h1 class="">About</h1>
      </div>
      <div id="contact" class="inline-block nav-items-link">
        <h1 class="">Contact</h1>
      </div>
      <div id="help-&-support" class="inline-block nav-items-link">
        <h1 class="">Help & Support</h1>
      </div>
    </main>
    <footer>
```

```
        <div class="line"></div>
        <p>Copyright &copy;MITM Boys | All Rights Reserved</p>
    </footer>
    <script >
        // Image Slider
const slides = document.querySelectorAll(".slide");
var lenOfSlides = slides.length
var counter = 0;
slides.forEach(
    (slide, idx) => {
        slide.style.left = `${idx * 100}%`;
    }
)
// go to the next image
const goNext = () => {
    if(counter >= lenOfSlides - 1){
        counter = 0;
    } else{
        counter++;
    }
    slideImage();
}
// go to the previous image
const goPrev = () => {
    if(counter <= 0){
        counter = lenOfSlides - 1;
    } else{
        counter--;
    }
    slideImage();
}
// logic to slide image
const slideImage = () => {
    slides.forEach(
        (slide) => {
            slide.style.transform = `translateX(-${counter*100}%)`;
        }
    )
}
    </script>
```

</body>
</html>

- **Snapshots:** Homepage: GameZ

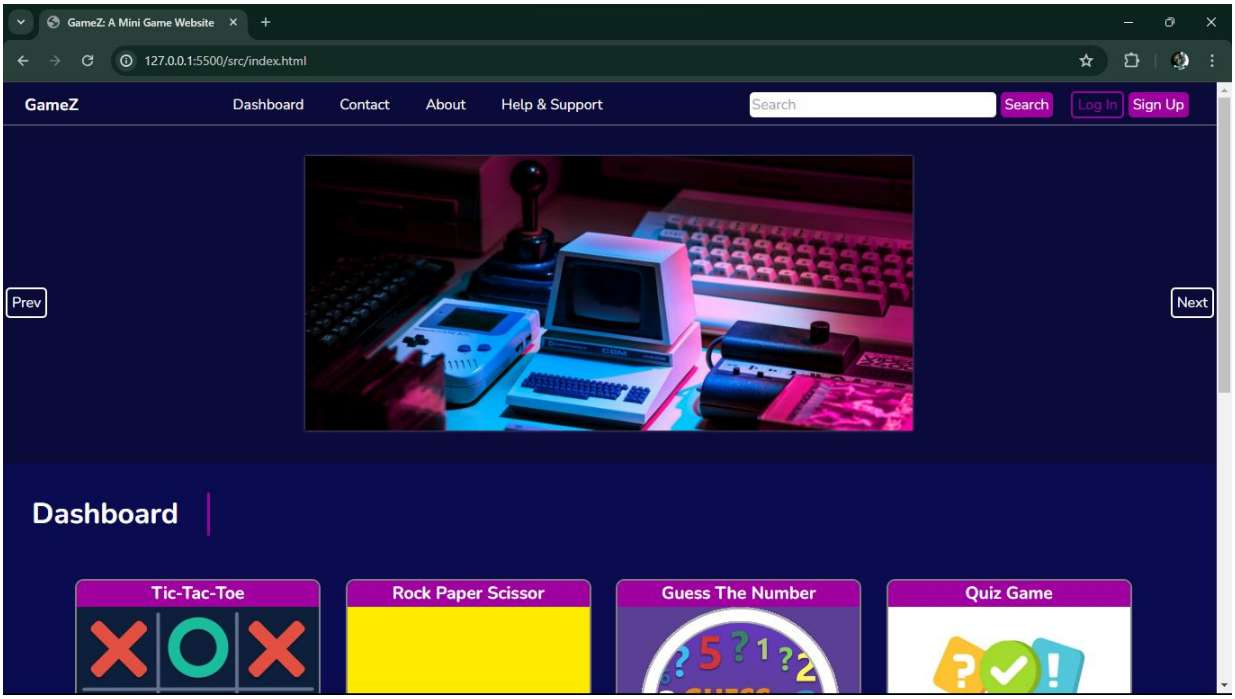


Image: Homepage-1(GameZ)

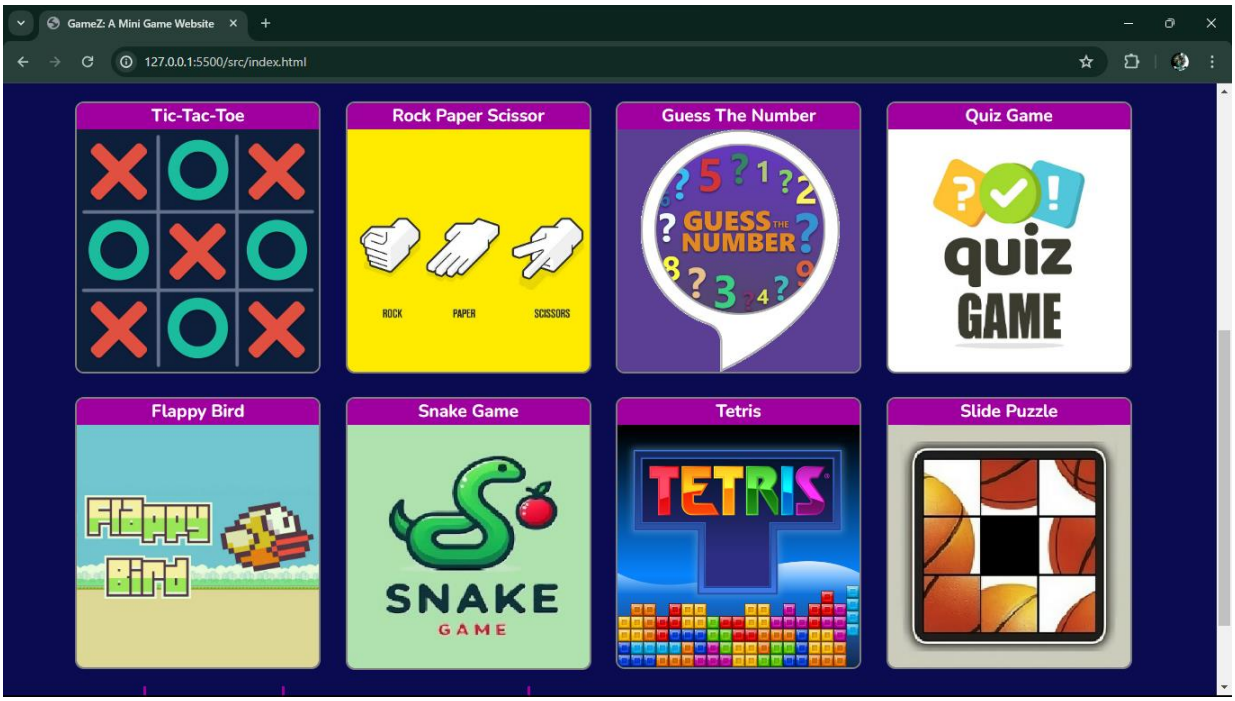


Image: Homepage-2(GameZ)

5.3 TESTING

5.3.1 Test Approach

The testing phase of "GameZ" involves thorough validation and verification of the website's functionality, usability, compatibility, and performance. Testing ensures that the mini-game website meets the specified requirements and delivers a seamless gaming experience for users across various devices and platforms.

1. Functional Testing:

- Gameplay Mechanics: Test each mini-game to ensure that the gameplay mechanics, controls, scoring systems, and win/lose conditions function as expected.
- User Authentication: Verify the functionality of user authentication features, including registration, login, logout, password recovery, and session management.
- Leaderboard: Test the leaderboard feature to ensure that it accurately displays high scores for each mini-game and updates in real-time.
- Social Sharing: Validate the social sharing functionality to ensure that users can easily share their gaming achievements on social media platforms.

2. Compatibility Testing:

- Cross-Browser Compatibility: Test the website on popular web browsers such as Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge to ensure consistent functionality and appearance.
- Device Compatibility: Test the website on various devices, including desktops, laptops, tablets, and smartphones, to ensure responsiveness and usability across different screen sizes and resolutions.

3. Usability Testing:

- Navigation and User Interface: Evaluate the website's navigation menus, layout, and user interface elements to ensure intuitiveness, ease of use, and accessibility for users.
- Game Selection: Test the process of browsing and selecting mini-games to ensure clarity, organization, and effectiveness in presenting game options to users.

4. Performance Testing:

- Page Loading Speed: Measure the loading speed of different pages and game components to ensure optimal performance and responsiveness.
- Game Performance: Test the performance of each mini-game, including graphics rendering, animation smoothness, and responsiveness to user input, to identify any performance bottlenecks or lag.

5. Security Testing:

- Data Encryption: Verify that sensitive user data, such as login credentials and personal information, is encrypted during transmission and storage to prevent unauthorized access.

- User Authentication: Test the strength and reliability of user authentication mechanisms to prevent security vulnerabilities such as brute force attacks or session hijacking.

Testing Tools & Environment

Testing tools & environments includes hardware and software platforms similar as used in development of the project. Generally, Chrome/Safari/Edge preferred as a hardware platforms and VS Code, Vim, Sublime are used as an IDE.

5.3.2 Test Cases

A set of test cases were developed to evaluate different aspects of GameZ, including gameplay mechanics, user interface responsiveness, accessibility, security, and error handling. Test cases were designed to cover both positive and negative scenarios to ensure robustness and reliability.

• Inputs

1. User interactions (keyboard, mouse, touch gestures).
2. Game configuration settings.
3. External data sources (APIs, quiz questions).
4. System configurations and software dependencies.

• Expected Output

The expected output of this project should be every component of the project must function as desired with a good user interface and user experience.

1. Responsive website interface.
2. Engaging mini-games with interactive gameplay.
3. User features like registration, login, and social sharing.
4. Database updates for user profiles and game data.
5. Fast performance and stable operation.
6. Secure handling of user data and prevention of vulnerabilities.

5.3.3 Testing Procedure

Testing procedures involved executing the test cases systematically, recording observations, identifying issues, and implementing necessary fixes.

1) Functional Testing
2) Compatibility Testing
3) Usability Testing
4) Performance Testing
5) Security Testing

5.4 Important Features

Key features of GameZ include:

- A diverse collection of mini-games catering to different genres and player preferences.
- Intuitive user interface design with seamless navigation and interactive elements.
- Leaderboard functionality to track and display high scores and achievements.
- Social sharing options to allow users to share their gaming experiences with friends and followers.
- Responsive design for optimal viewing and interaction across a wide range of devices and screen sizes.

5.5 Limitations

Despite its strengths, GameZ may have some limitations, including:
- Performance issues on low-end devices or older web browsers due to resource-intensive game mechanics or graphical effects.
- Compatibility challenges with certain web browsers or devices, requiring additional testing and optimization efforts.
- The need for continuous updates and maintenance to address evolving technologies, security vulnerabilities, and user feedback.

5.6 Future Work

Future enhancements for GameZ may include:
- Integration of multiplayer functionality to allow players to compete or collaborate with others in real-time.
- Addition of new game genres, levels, and challenges to expand the variety and depth of gaming experiences.
- Enhancement of social features such as player profiles, friend lists, and messaging systems to foster community engagement.
- Optimization for emerging web technologies such as WebAssembly, WebGL, or WebRTC to unlock new possibilities for immersive gaming experiences.

Chapter - 6

CONCLUSION

6. CONCLUSION

In conclusion, the development of GameZ demonstrates the potential of HTML, CSS, and JavaScript in creating engaging and accessible web-based gaming experiences. Through an iterative development process, comprehensive testing, and user feedback, GameZ offers a platform that caters to diverse gaming preferences while paving the way for future innovations in web-based gaming. By leveraging web technologies and following best practices in software development, GameZ exemplifies the possibilities of online gaming in the modern digital era.

The integration of various mini-games, including Flappy Bird, Guess the Number, Quiz Game, Rock Paper Scissors, Slide Puzzle, Snake Game, Tic Tac Toe, and Tetris, has added diversity and excitement to the gaming experience offered by GameZ. Each game was meticulously crafted to provide interactive gameplay mechanics, captivating visuals, and challenging puzzles, catering to a wide range of player preferences and skill levels.

As we reflect on the journey of developing "GameZ," we acknowledge the dedication, creativity, and collaboration of the development team, whose collective efforts have brought this project to fruition. We also extend our gratitude to the users and testers who provided valuable feedback and insights throughout the development process, contributing to the continuous improvement and refinement of the website.

Looking ahead, we remain committed to maintaining and enhancing the GameZ platform, incorporating new features, mini-games, and improvements based on user feedback and emerging technologies. Our vision is to continue providing an immersive and enjoyable gaming experience for users worldwide, fostering a vibrant community of gamers and enthusiasts on the GameZ platform.

In conclusion, GameZ represents not only a mini-game website but a testament to the power of creativity, innovation, and collaboration in the realm of web development. We are proud of the journey we have embarked on and look forward to the exciting adventures that lie ahead in the world of gaming with GameZ.

Chapter – 7

REFERENCES

7. REFERENCES

- *www.w3schools.com/web-technology*
- *www.w3schools.com/game-development-in-web-technology*
- *https://www.codingnepalweb.com/create-game-html-css-javascript*
- *https://www.youtube.com/mini-game-development-using-html-css-and-js*
- *https://github.com/mini-games-using-javascript*