International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

The Escape Room

Rushikesh Joshi[1], Nihant Thorkar[2], Rishikesh Waigaonkar[3], Akask Nair[4], Aman Nampalliwar[5]

Nagpur Institute of Technology

A B S T R A C T :

"The Escape Room" is an interactive, web-based escape room game designed to test and enhance players' HTML, CSS, and JavaScript skills. Players begin by logging in or signing up to access the game. The gameplay involves solving 50 timed questions (referred to as "rooms") for each of the three languages (HTML, CSS, JavaScript). Each room presents a coding problem or puzzle that must be solved within a set time limit to progress.

Upon completion, a scorecard displays the player's performance, including their high score, time taken to solve the questions, and rankings compared to other players. The system saves records to track progress and encourage competition. This gamified approach makes learning and practicing web development engaging and rewarding."

Key Features:

- Login/Signup system for user authentication.
- 150 total questions (50 per language) in an escape room format.
- Timed challenges to test problem-solving under pressure.
- Scorecard with rankings, completion time, and high scores.
- Persistent data storage to save player records.

Keywords: Escape Room, Gamification, Programming Education, Interactive Learning, HTML, CSS, JavaScript, Node.js, Express.js, MongoDB, Game-Based Learning, Problem-Solving, Leaderboard, Timer.

Main text

Escape the Code Room

Welcome, Coder!

You are trapped in a virtual labyrinth of 150 code rooms—50 for HTML, 50 for CSS, and 50 for JavaScript. Your mission? Solve each puzzle before time runs out to escape!

How to Play:

1. Log In / Sign Up – Save your progress and compete for high scores.
2. Choose a Challenge – Each "room" presents a coding problem to solve.
3. Race Against Time– You have 5 minutes per room to write the correct code.
4. Test & Submit – Run your code to see if it works, then submit to escape the room.
5. Advance & Conquer – Complete all 150 rooms to become a Code Escape Master!

Features:

✓ ✔ 150 Unique Coding Challenges – Test your skills in HTML, CSS, and JavaScript.
✓ ✔ Live Code Preview – See your changes in real-time.
✓ ✔ Scoreboard – Track your progress and compete for the fastest times.
✓ ✔ Time Bonus – Solve faster to earn extra points!
✓ ✔ Permanent Records – Your best scores are saved for future challenges.

Final Scorecard
After completing all the rooms, you'll see:

- Your Total Score (based on speed & accuracy)
- Time Taken (per language and overall)
- Leaderboard (compared with top players)

Ready to Begin?

- Log In (if you've played before)
- Sign Up (to start a new journey)

The clock is ticking… Can you escape all the code rooms?

Nomenclature

1. Authentication System

- **Login Screen**
    - Player Login (Existing Users)
    - New Player Signup (New Users)
    - Guest Mode (Optional, no score saving)

2. Game Structure

- **Languages & Levels**
    - HTML Dungeon (50 rooms)
    - CSS Labyrinth (50 rooms)
    - JavaScript Maze (50 rooms)
- **Room Types**
    - Beginner Chambers (Rooms 1-15)
    - Intermediate Chambers (Rooms 16-35)
    - Expert Vaults (Rooms 36-50)

3. Game Mechanics

- **Timer System**
    - Room Timer (Countdown per challenge, e.g., 5:00 minutes)
    - Speed Bonus (Extra points for finishing early)
    - Time Penalty (Reduced points if time expires)
- **Scoring System**
    - Base Points (Fixed points per room)
    - Time Bonus (Additional points based on remaining time)
    - Perfect Solve Bonus (Extra reward for optimal solutions)

4. Player Progress & Scoring

- **Player Dashboard**
    - Current Score (Total points accumulated)
    - Rooms Completed (HTML/CSS/JS progress)
    - Time Played (Total time spent in-game)

- **Leaderboard**
    - Top Coders (Highest scores)
    - Fastest Escapers (Best completion times)
    - Perfect Solvers (Players with zero failed attempts)

5. Post-Game Summary

- **Escape Report** (Final scorecard)
    - Total Score (Sum of all room points)
    - Time Taken (Overall duration)
    - Accuracy Rate (% of optimal solutions)
    - Unlocked Achievements (Badges for milestones)

- **Save & Share**
    - Save Progress (LocalStorage/Cloud)
    - Export Scorecard (Share on social media)

6. Additional Features (Optional)

- Hints System (Limited hints per room)
- Skip Room (Penalty applied)

- Multiplayer Mode (Race against others)
- Daily Challenges (Special timed rooms)

Suggested Terminology

Term	Meaning
Escape	Successfully solving a room
Locked Room	Uncompleted challenge
Time Lock	Timer running out
Code Breaker	Player who solves rooms quickly
Perfect Run	Completing all rooms optimally

1.1. Structure

```
escape-the-code-room/
│
├── index.html        # Main entry (login/signup)
├── game.html         # Game interface
├── scoreboard.html   # Final score display
│
├── css/
│   └── style.css     # Global styles
│
├── js/
│   ├── auth.js       # Authentication logic
│   ├── game.js       # Core game mechanics
│   ├── challenges.js # All 150 challenges (50x3 languages)
│   ├── timer.js      # Timer logic
│   └── scoreboard.js # Score calculation & storage
│
└── assets/           # Images/icons
```

1.2. Tables

Database Tables Structure

1. Users Table
Stores player registration/login information.

Column Name	Data Type	Description
`user_id`	INT (Primary Key)	Unique identifier for each player

`username`	VARCHAR (50)	Player's chosen username (unique)
`password hash`	VARCHAR (255)	Encrypted password for security
`email`	VARCHAR (100)	Optional for account recovery
`created_at`	TIMESTAMP	Date/time of account creation

---

2. Rooms Table
Stores all 150 coding challenges (50 per language).

Column Name	Data Type	Description
`room_id`	INT (Primary Key)	Unique room identifier
`language`	ENUM ('html', 'css', 'JavaScript')	Language category
`difficulty`	ENUM ('easy', 'medium', 'hard')	Difficulty level
`title`	VARCHAR (100)	Challenge title (e.g., "Flexbox Layout")
`description`	TEXT	Detailed instructions
`solution_code`	TEXT	Expected correct code (for validation)
`time_limit`	INT	Time allotted (in seconds, e.g., 300)

2.Player_Progress Table
Tracks individual player progress across rooms.

Column Name	Data Type	Description
`progress_id`	INT (Primary Key)	Unique progress record
`user_id`	INT (Foreign Key)	Links to `Users` table
`room_id`	INT (Foreign Key)	Links to `Rooms` table
`completion_time`	INT	Time taken (seconds)
`attempts`	INT	Number of tries before success
`completed_at`	TIMESTAMP	Date/time of completion
`score_earned`	INT	Points calculated (base + time bonus)

---

4. Leaderboard Table
Aggregates high scores for display.

Column Name	Data Type	Description
`leaderboard_id`	INT (Primary Key)	Unique record
`user_id`	INT (Foreign Key)	Links to `Users` table
`total_score`	INT	Sum of all room scores
`total_time`	INT	Cumulative time across all rooms
`html_rooms_completed`	INT	Count (0-50)
`css_rooms_completed`	INT	Count (0-50)
`js_rooms_completed`	INT	Count (0-50)
`last_updated`	TIMESTAMP	Auto-updates on new completion

1.3. Construction of references

"Escape the Code Room" is an interactive coding challenge game where players solve HTML, CSS, and JavaScript puzzles to progress through virtual "rooms." Each room presents a unique coding problem that must be completed within a time limit. The game features a login/signup system, a timer-based challenge structure, and a final scoreboard displaying high scores and completion times.

1.4. Section headings

1. Game Overview

- 1.1 Concept & Objective

- 1.2 Target Audience
- 1.3 Core Features

2. User Flow

- 2.1 Login / Signup Process
- 2.2 Game Entry & Room Selection
- 2.3 Challenge Completion & Progression
- 2.4 score Submission & Leaderboard Update

3. Game Mechanics

- 3.1 Room Structure (HTML, CSS, JavaScript Challenges)
    - 3.1.1 HTML Rooms (50 Challenges)
    - 3.1.2 CSS Rooms (50 Challenges)
    - 3.1.3 JavaScript Rooms (50 Challenges*)*

- 3.2 Timer System (Countdown per Room)
- 3.3 Code Execution & Live Preview
- 3.4 Solution Validation & Scoring

4. Technical Implementation

- **4.1 Frontend Components**
    - 4.1.1 Login/Signup Page (HTML/CSS/JS)
    - 4.1.2 Code Editor Interface
    - 4.1.3 Challenge Display Panel
    - 4.1.4 Real-Time Preview Window

- **4.2 Backend Logic (LocalStorage/API)**
    - 4.2.1 User Authentication & Data Storage
    - 4.2.2 Challenge Loading & Validation
    - 4.2.3 score Calculation & Leaderboard Update

5. Scoring & Leaderboard System

- 5.1 Points Allocation (Base + Time Bonus)
- 5.2 High Score Tracking
- 5.3 Player Statistics (Time Taken, Accuracy)

6. Data Management

- 6.1 Storing Player Progress (LocalStorage/Database)
- 6.2 Challenge Database Structurer
- 6.3 Leaderboard Data Handling

7. Future Enhancements

- 7.1 Multiplayer Mode
- 7.2 Difficulty Levels (Easy/Medium/Hard)
- 7.3 Social Sharing (Score Posting)

8. Conclusion

- 8.1 Summary of Features
- 8.2 Potential Use Cases
- 8.3 Developer Notes

1.5. General guidelines for the preparation of your text

- **Theme and Storyline**: Choose a captivating theme that resonates with your audience. Develop a brief storyline that immerses players in the experience. For example, a tech-themed escape room could revolve around "hacking" into a virtual system to retrieve lost data.
- **Room Design:** Create a layout that logically flows from one puzzle to the next. Each "room" will represent a set of questions related to HTML, CSS, or JavaScript. Ensure that the physical or digital environment enhances the theme and provides clues relevant to the puzzles.

Puzzle Development

- **Question Creation**: Develop 50 questions for each language (HTML, CSS, JavaScript). These should range in difficulty and cover essential concepts such as syntax, functions, and styling techniques. Incorporate various question formats like multiple-choice, fill-in-the-blank, and coding challenges24.
- **Puzzle Types**: Include different types of puzzles that require players to apply their coding knowledge. For instance:
    - Debugging code snippets
    - Writing functions based on given criteria
    - Styling elements using CSS properties
- **Hints and Solutions**: Prepare hints for each puzzle to assist players if they get stuck. These should be concise and encourage critical thinking without giving away the answer outright.

Gameplay Mechanics

- **Time Limit**: Set a time limit for completing all puzzles, typically around 60 minutes. This adds urgency and excitement to the game13.
- **Scorecard System**: At the end of the game, display a scorecard showing each player's score based on correct answers and the time taken to complete the questions. This can motivate participants and add a competitive element3.
- **Record Keeping**: Implement a system to save scores and player performance over time. This could be done through a simple database or spreadsheet that tracks high scores and player progress across multiple sessions.

Technical Implementation

- **Platform Selection**: Decide whether the escape room will be conducted online or in-person. For online formats, consider using platforms that support interactive quizzes and coding challenges.
- **User Authentication**: Incorporate login and sign-in options to track user progress and scores effectively. Ensure user data is securely managed.
- **Responsive Design**: Ensure that the game interface is user-friendly across various devices. This includes optimizing for mobile screens if necessary.

1.6. File naming and delivery

1. HTML Files

File	Purpose
`index.html`	Login/Signup screen (entry point)
`game.html`	Main game interface (editor + challenge)
`scoreboard.html`	Displays high scores & player stats

2. CSS Files

File	Purpose
`styles.css`	Global styles (fonts, colors, resets)
`auth.css`	Styles for login/signup forms
`game.css`	Styling for code editor & challenge UI
`scoreboard.css`	Leaderboard table & animations

3. JavaScript Files

File	Purpose
`auth.js`	Handles user login/signup (localStorage)
`game.js`	Manages rooms, timer, and scoring
`editor.js`	Code execution & solution validation
`scoreboard.js`	Fetches & displays high scores

4. Data Files (JSON)

File	Purpose
`html. json`	50 HTML challenges (with expected solutions)
`css. json`	50 CSS challenges
`js. json`	50 JavaScript challenges
`scores. json`	Stores player stats (if no backend)

Delivery (Zip Structure for Submission)

```
escape-the-code-room.zip
|
├──── index.html
├──── game.html
├──── scoreboard.html
|
├──── css/
|    ├──── styles.css
|    ├──── auth.css
|    ├──── game.css
|    └──── scoreboard.css
|
├──── js/
|    ├──── auth.js
|    ├──── game.js
|    ├──── editor.js
|    └──── scoreboard.js
|
├──── data/
|    ├──── html.json
|    ├──── css.json
|    ├──── js.json
|    └──── scores.json
|
└──── README.md        # Setup instructions
```

1.7. Footnotes

This game, "Escape the Code Room," is designed to challenge and enhance your coding skills in HTML, CSS, and JavaScript. Players must solve 50 coding puzzles (referred to as "rooms") for each language within a set time limit. Each correct answer helps you progress through the rooms, while the timer adds an extra layer of challenge. At the end of the game, your performance is recorded, displaying your total score, time taken, and ranking on the leaderboard. All records are saved to track your progress over time.
Good luck, and may your code lead you to victory!

2.Illustrations

```
[Start]
  |
  v
[Login/Sign-In]
  |
  v
[Room 1: HTML Questions] -- (correct) --> [Room 2: CSS Questions]
  |                              |
  |-- (incorrect) --> [Hint/Clue] <-----|
  |
  v
[Room 3: JavaScript Questions]
  |
  v
[Scorecard Display]
```

- **Login and Sign-In Interface**: Design a mock-up of the login/sign-in screen. This should include fields for username and password, a "Login" button, and a "Sign Up" option. Use vibrant colors and clear typography to make it visually appealing.

- **Room Layout**: Create a diagram that illustrates the layout of the escape room. Each "room" represents a set of questions for HTML, CSS, or JavaScript. Use arrows to show the flow from one room to another, indicating how players progress through the game.

- **Puzzle Examples**: Include illustrations of sample puzzles that players will encounter:
- **HTML Puzzle**: An example of a code snippet with errors that players need to debug.
- **CSS Challenge**: A visual representation of a styled webpage where players must identify incorrect styles.
- **JavaScript Task**: A flowchart showing how players can write a function to solve a specific problem.
- **Scorecard Design**: Illustrate the scorecard layout that displays player scores and time taken. Use graphs or charts to represent high scores visually, making it easy for participants to understand their performance.
- **Flowchart of Game Progression**: Create a flowchart that outlines how players will navigate through the game. This should show the sequence of rooms (questions) and how completing one room unlocks the next. Indicate feedback mechanisms for each room, such as hints or clues.

2.Equations

Here are some mathematical equations and logic to structure the scoring and timing for your "Escape the Room" coding game:

1. Time Calculation for Each Room:

Troomi= Tendi − TstartiT_{\text{room}_i} = T_{\text{end}_i} - T_{\text{start}_i}

2. where:

- TstartiT_{\text{start}_i} = Start time for question ii
- TendiT_{\text{end}_i} = End time for question ii
- TroomiT_{\text{room}_i} = Time taken to solve question ii

3. Total time taken to complete all rooms:

Ttotal=∑i=150TroomiT_{\text{total}} = \sum_{i=1}^{50} T_{\text{room}_i}

2. Scoring System:

4. Each question has a maximum score SmaxS_{\max}, and points decrease based on the time taken.

Si=Smax−(Troomi×DTlimit)S_i = S_{\max} - \left(\frac{T_{\text{room}_i} \times D}{T_{\text{limit}}} \right)

5. where:

- SiS_i = Score for question ii
- DD = Deduction factor per second
- TlimitT_{\text{limit}} = Maximum allowed time for a question

6. Total score:

Stotal=∑i=150SiS_{\text{total}} = \sum_{i=1}^{50} S_i

3. High Score Tracking:

7. A leaderboard is maintained with player records:

Leaderboard={(P1,S1,T1),(P2,S2,T2),… }\text{Leaderboard} = \{ (P_1, S_1, T_1), (P_2, S_2, T_2), \dots \}

8. where:

- PiP_i = Player name
- SiS_i = Total score of player ii
- TiT_i = Total time taken by player ii

9. Leaderboard is sorted by highest score first. If scores are tied, players with lower total time are ranked higher.

Sorted Leaderboard=Sort(Leaderboard,S,T)\text{Sorted Leaderboard} = \text{Sort} (\text{Leaderboard}, S, T)

10. Would you like a visual flowchart or algorithmic implementation of this logic?

4. Online license transfer

License Transfer Overview

1. **Account-Specific Licenses**: Generally, licenses for online games are tied to the account that made the purchase. For example, it is not possible to transfer a PC game license from one Microsoft account to another, as the license remains linked to the original account used for the purchase1. This means players must log in with the original account to access the game.

2. **Online Gaming Licenses in India**: If your escape room game requires an official gaming license (especially if it involves skill-based challenges), you may need to apply for a license under specific regulations. In India, for instance, licenses can be obtained under the Nagaland Prohibition of Gambling and Promotion of Online Games of Skill Act, 2015, which allows for skill-based games23. This license is valid for five years and has specific application requirements and fees.

3. **Documentation Requirements**: When applying for a gaming license, you will typically need to provide a detailed business plan, software details, identity proofs, and financial statements. This ensures compliance with local regulations and can affect your ability to transfer or share licenses later3.

4. **Transfer Policies**: Some companies have specific policies regarding license transfers. For example, Foundry's policy allows for transfers under certain conditions (like hardware changes) but does not permit transfers between different accounts or locations4. Understanding such policies is crucial if your game is distributed through platforms with similar rules.

5. **Compliance and Regulations**: Ensure that your game complies with all relevant regulations concerning online gaming and licensing. This includes adhering to technical standards and responsible gambling measures if applicable5.

Steps for License Transfer

- **Review Licensing Terms**: Check the licensing terms associated with your game distribution platform to understand if and how you can transfer licenses.

- **Contact Support**: If you need to transfer a license due to account issues or other circumstances, reach out to customer support of the platform for guidance.

- **Prepare Documentation**: If applying for a new license or transferring an existing one, ensure all required documentation is prepared according to regulatory standards.

- **Follow Legal Guidelines**: Adhere strictly to legal guidelines surrounding gaming licenses in your jurisdiction to avoid penalties or issues with compliance.

Acknowledgements

We extend our sincere gratitude to everyone who contributed to the development of the **Escape the Room – Coding Challenge** game. This project would not have been possible without the dedication and support of numerous individuals and teams.

- **Development Team**
  We appreciate the hard work and expertise of our **developers, designers, and engineers** who built the interactive environment, authentication system, and scoring mechanism. Their commitment to crafting an engaging and educational experience made this project a success.

- **Contributors & Content Creators**
  Special thanks to the content creators who meticulously designed **50 challenging rooms** covering **HTML, CSS, and JavaScript**. Their efforts ensured a well-structured and diverse set of coding problems, making the game both fun and informative.

- **Beta Testers**
  We are grateful to our beta testers who rigorously tested the game, provided valuable feedback, and helped refine the gameplay mechanics. Their insights greatly improved the **user experience, difficulty balance, and performance** of the game.

- **Support & Mentorship**
  We acknowledge the **mentors, educators, and coding communities** who inspired and guided us throughout the development process. Their expertise helped shape the vision and learning objectives of this project.

- **Technologies & Tools**
  This project was made possible using cutting-edge web technologies, including:
  Frontend: HTML, CSS, JavaScript
  Backend: Node.js / Python / PHP
  Database: MySQL / Firebase / MongoDB
  Hosting & Deployment: AWS / Firebase / Netlify

- **Players & Community**
  A special thank you to **all the players** who participate in this challenge. Your enthusiasm for coding and problem-solving drives us to continue improving and expanding this experience.
  Thank you all for making Escape the Room – Coding Challenge a reality!

REFERENCES :

1. Aho, A. V., Lam, M. S., Sethi, R., & Ullman, J. D. (2006). *Compilers: Principles, Techniques, and Tools* (2nd ed.). Pearson Education.: Used for understanding syntax analysis and code execution methods.
2. Knuth, D. E. (1997). *The Art of Computer Programming, Vol. 1: Fundamental Algorithms* (3rd ed.). Addison-Wesley. Referenced for graph traversal algorithms and heuristic-based problem-solving.
3. Goodrich, M. T., & Tamassia, R. (2014). *Algorithm Design and Applications*. Wiley. Used for implementing DFS, BFS, and graph-based learning models.
4. Mitchell, T. (1997). *Machine Learning*. McGraw-Hill. Basis for AI-driven adaptive difficulty and learning path algorithms.
5. Koster, R. (2013). *A Theory of Fun for Game Design* (2nd ed.). O'Reilly Media. Used for gamification strategies and user engagement techniques.
6. Lazzaro, N. (2004). *Why We Play Games: Four Keys to More Emotion Without Story*. Referenced for understanding game-based learning motivation.
7. IEEE Standard for Learning Object Metadata (2002). IEEE 1484.12.1. Used for structuring metadata and user progress tracking.
8. Neo4j Official Documentation (2023). Retrieved from https://neo4j.com/docs/ Used for graph database implementation in learning progression.
9. Mozilla Developer Network (MDN) (2024). *JavaScript and Web Technologies Documentation*. Used for frontend development using JavaScript and React.js.
10. Firebase Official Documentation (2024). Google Cloud. Referenced for user authentication, database storage, and cloud execution.
11. HackerRank & LeetCode Challenges. Retrieved from https://www.hackerrank.com and https://leetcode.com Used as a benchmark for coding challenges and problem difficulty levels.
12. GitHub Developer Guide (2024). Retrieved from https://docs.github.com. Used for version control.