An Assignment Work submitted for the subject

Web Game Programming - Practical

By

Athul K M

University Registration No.: *248378730004*

2024 - 2025

ALAGAPPA UNIVERSITY

(A State University Established by the Government of Tamil Nadu in 1985,
Accredited with A+ Grade by NAAC (CGPA 3.64) in the Third Cycle,
Graded as Category-I University and Granted Autonomy by MHRD-UGC,
MHRD-NIRF 2020 Rank : 36, QS 2020 India Rank : 24)
KARAIKUDI - 630 003, Tamil Nadu, India

TABLE OF CONTENTS

BRIEF

This project focuses on the development of a hyper-casual web game, designed to be lightweight, engaging, and accessible across desktop and mobile browsers. Hyper-casual games are known for their simple mechanics, short play sessions, and easy-to-learn controls, making them ideal for broad audiences.

The game will be developed using HTML5, JavaScript (with or without a framework like Phaser.js), and CSS for UI design. Core features include intuitive gameplay mechanics, responsive controls, minimalistic visuals, and basic sound effects to enhance user experience.

Key project stages involve concept ideation, mechanics implementation, UI/UX design, performance optimization, and deployment on web platforms such as GitHub Pages or itch.io. The game may also incorporate monetization strategies such as ads or sponsorships.

This project serves as a practical exercise in game development, web technologies, and user engagement, providing insights into creating and publishing hyper-casual games for the web.

LEARNING OUTCOMES

1. Game Development Fundamentals: This project will help you understand the core principles of hyper-casual game design, including how to create simple mechanics, engaging gameplay loops, and balanced difficulty scaling. You'll also learn how to manage game states, such as start screens, gameplay, and game-over screens.

2. Web Technologies & Game Frameworks: You will gain hands-on experience using HTML5 Canvas for rendering, JavaScript for game logic, and CSS for styling UI elements. Additionally, you can explore game development frameworks like Phaser.js, which simplify physics, animations, and interactions in web-based games.

3. User Experience & Interface Design: The project will teach you how to design intuitive and responsive controls that work on both desktop and mobile devices. You will also learn to create user-friendly interfaces, implement visual feedback mechanisms, and enhance player engagement with animations and sound effects.

4. Performance Optimization & Debugging: You will develop skills in optimizing game assets, writing efficient code, and using debugging techniques to improve performance. This includes working with browser developer tools, reducing memory leaks, and ensuring smooth gameplay across different devices.

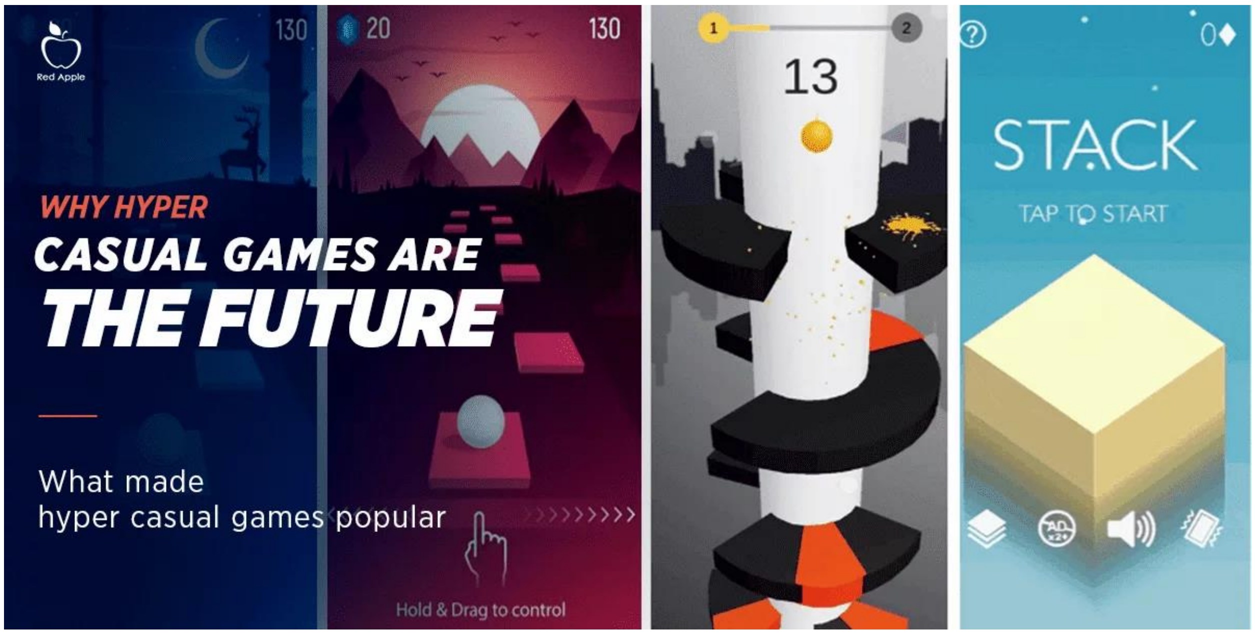
5. Audio & Visual Integration: This project will cover the use of sound effects, background music, and sprite animations to create an immersive experience. You will learn how to manage game assets effectively and apply CSS and JavaScript for smooth visual transitions and effects.

6. Game Deployment & Hosting: By the end of the project, you will understand how to optimize and package your game for the web. You will learn how to deploy it on platforms like GitHub Pages, Netlify, or itch.io and ensure cross-browser compatibility for a seamless user experience.

7. Monetization & Engagement Strategies : You will explore different monetization methods, including ads and in-game purchases, and learn how to increase player retention through features like leaderboards, daily challenges, and rewards. Understanding engagement strategies will help make your game more enjoyable and replayable.

8. Project Management & Problem-Solving: This project will help improve your planning, time management, and troubleshooting skills. You will learn how to prototype ideas, break down development into manageable tasks, debug efficiently, and iterate based on testing and feedback.

OVERVIEW OF HYPER-CASUAL GAMES



Hyper-casual web games are lightweight, browser-based games that feature simple mechanics, minimal UI, and instant gameplay. They require no downloads or installations, making them highly accessible to players on desktop and mobile devices. These games are designed for short, engaging play sessions and are often monetized through ads or microtransactions.

Key Characteristics:

- **Instant Playability:** Hyper-casual web games are built using HTML5, JavaScript, and lightweight frameworks, allowing them to be played directly in a web browser without requiring downloads or installations. This makes them highly accessible and ideal for mobile and desktop users alike.

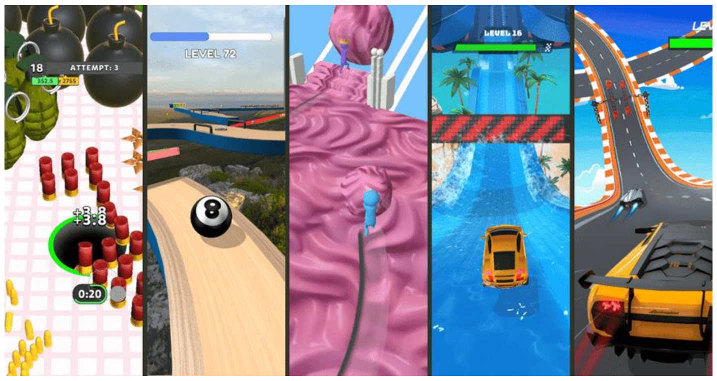
- **Minimal Controls:** These games rely on easy-to-learn mechanics, usually involving just one or two input methods, such as tapping, clicking, swiping, or pressing a few keyboard keys. The simplicity ensures that players can start playing without instructions.

- **Short Play Sessions:** Designed for quick bursts of gameplay, hyper-casual web games typically last anywhere from a few seconds to a couple of minutes per session. This makes them ideal for players looking for quick entertainment during breaks or in between tasks.

- **Lightweight & Optimized:** The design of hyper-casual web games focuses on clear, simple graphics and a clean user interface to avoid distractions. Bright colors, smooth animations, and clear feedback mechanisms (such as score pop-ups or subtle sound effects) enhance the experience without overwhelming the player.

- **Simple Visuals & UI**: Since these games run on web browsers, they need to be highly optimized for smooth performance across various devices. Developers often use efficient coding practices, compressed assets, and minimal animations to ensure fast loading times and seamless gameplay.

- **Monetization via Ads**: Often includes banner ads, rewarded ads, or in-game purchases.



Development Process:

- **Game Concept & Mechanics** – Define a simple, engaging idea with a fun loop.
- **Prototyping** – Develop a basic version to test core interactions.
- **Graphics & UI Design** – Create a minimal but appealing interface.
- **Performance Optimization** – Ensure smooth gameplay and quick load times.
- **Monetization & Deployment** – Integrate ads, host on platforms like GitHub Pages, itch.io, or Netlify.

Advantages of Hyper-Casual Web Games:

- **High Accessibility:** Since these games run on web browsers, they can be played instantly without requiring downloads. This lowers the entry barrier for casual players.

- **Fast Development Cycle:** Compared to complex video games, hyper-casual web games can be developed in a few weeks or even days, making them ideal for quick game releases.

- **Easy Monetization:** With ad integration and potential in-game purchases, developers can earn revenue without charging players upfront.

- **Cross-Platform Compatibility:** Most web games run smoothly on desktops, tablets, and mobile phones, expanding their audience reach.

Popular Examples of Hyper-Casual Web Games:

- **Flappy Bird (Web Version)** – Tap to keep the bird in the air while avoiding pipes.
- **2048** – A sliding tile puzzle game where numbers merge to reach 2048.

- **Slither.io** – A multiplayer snake game where you grow by eating other players.
- **Agar.io** – A multiplayer cell-eating game with a competitive leaderboard.
- **Doodle Jump (Web Version)** – Bounce upwards while avoiding obstacles.

- **Paper.io** – A territory-capturing game with smooth mechanics.

Technology Stack for Development:

Core Development Technologies

HTML5 – The backbone of the game's structure.

CSS – Styles UI elements and animations.

JavaScript – Handles interactivity, physics, and game logic.

Game Frameworks & Libraries (Optional):

Phaser.js – A popular framework for 2D browser games.

Three.js – Used for lightweight 3D games with WebGL.

PixiJS – Helps with rendering fast, smooth graphics.

Web Hosting & Deployment:

GitHub Pages – Free hosting for simple games.

Netlify – Easy deployment with continuous integration.

itch.io – A popular indie game platform with monetization options.

Hyper-casual web games continue to dominate the gaming landscape due to their simplicity, accessibility, and ease of development. They offer a great opportunity for developers to experiment with new game ideas, quickly deploy projects, and generate revenue through ads and microtransactions. However, retaining players and optimizing performance remain challenges that developers must address.

CODING LANGUAGE

HTML (HyperText Markup Language):

What is HTML?

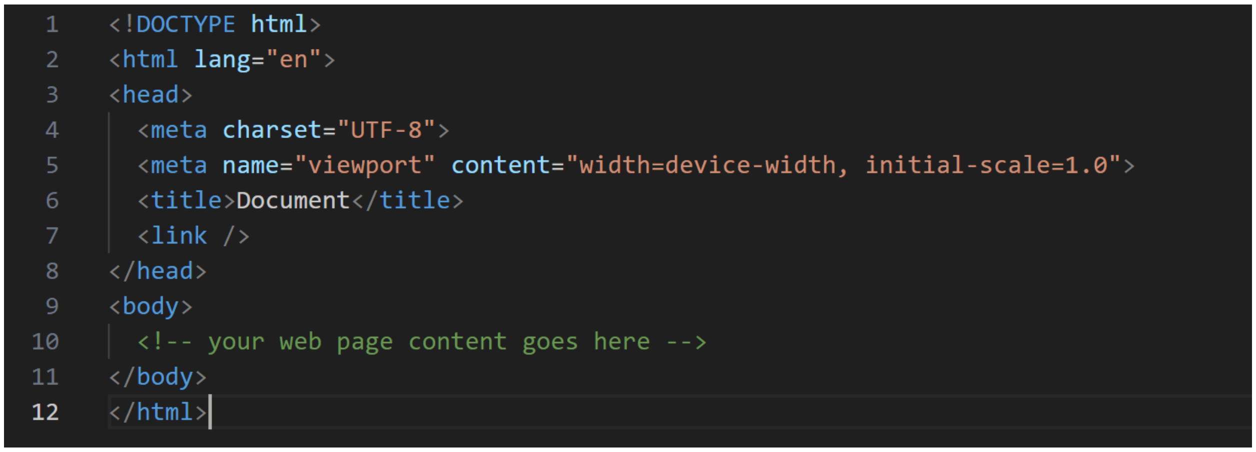
HTML (HyperText Markup Language) is the foundation of any web page or web-based game. It provides the **structure** for the game, defining elements like the canvas (where the game is drawn), buttons, scoreboards, and menus.

Key HTML Elements for Game Development

1. `<canvas>` - The primary element used to render the game graphics.
2. `<div>` - Used to organize UI elements like score counters.
3. `<button>` - Used for user interactions like start and restart buttons.
4. `<audio>` - Embed sounds and music in the game.

Basic Structure of an HTML Document:

Every HTML document follows a specific structure. Below is a simple example:



```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <link />
</head>
<body>
  <!-- your web page content goes here -->
</body>
</html>
```

Explanation of HTML Elements:

1. <!DOCTYPE html> – Declares the document as an HTML5 document.
2. <html> – The root element that contains the entire HTML page.
3. <head> – Contains metadata about the document (like title and character set).
4. <title> – Sets the title of the web page (visible on the browser tab).
5. <body> – Contains all the **visible** content of the webpage.
6. <h1> – A heading element (h1 to h6 are available for different heading levels).
7. <p> – Defines a paragraph.
8. <a href="URL"> – Creates a hyperlink.
9. <img src="image.jpg" alt="description"> – Displays an image.
10. <div> – A block-level container for grouping elements.

CSS (Cascading Style Sheets):

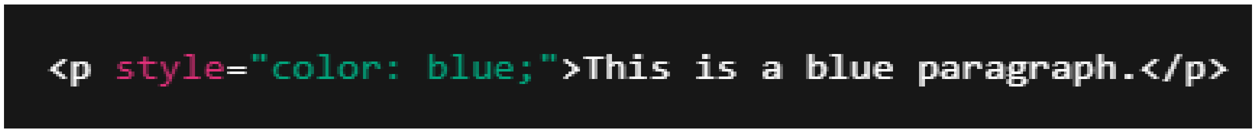
What is CSS?:

CSS is used to **style** HTML elements. It controls colors, fonts, layouts, spacing, and responsiveness.

Types of CSS:

There are three ways to apply CSS:

Inline CSS (applied directly within an HTML tag)



```
<p style="color: blue;">This is a blue paragraph.</p>
```

Internal CSS (inside a <style> tag in the <head>)

```
<style>
    p {
        color: blue;
    }
</style>
```

External CSS (in a separate .css file)

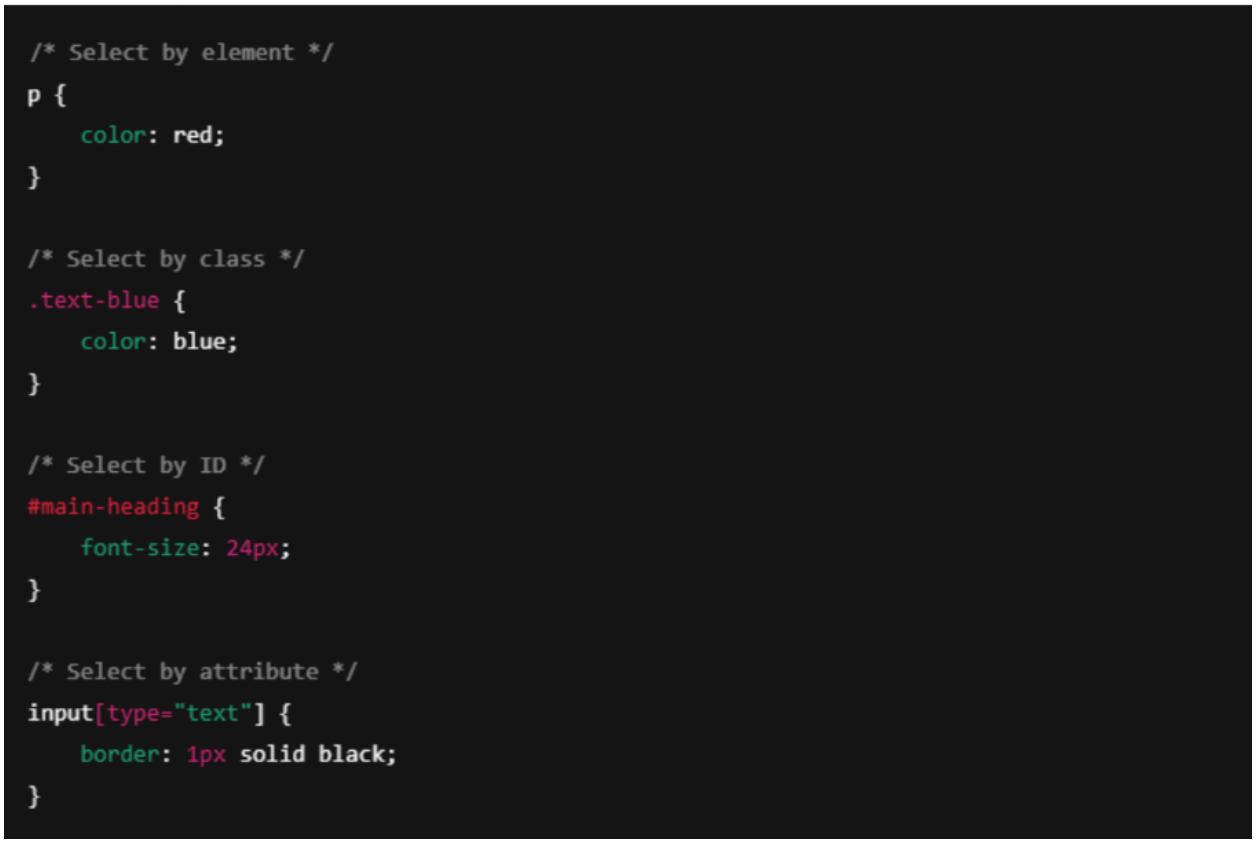
```
p {
    color: blue;
}
```

Then link the CSS file in HTML :

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

CSS Selectors:

CSS uses selectors to target elements.



```
/* Select by element */
p {
    color: red;
}

/* Select by class */
.text-blue {
    color: blue;
}

/* Select by ID */
#main-heading {
    font-size: 24px;
}

/* Select by attribute */
input[type="text"] {
    border: 1px solid black;
}
```

CSS Box Model:

Every HTML element is considered a box with:

1. Content – The actual text or image.
2. Padding – Space inside the border.
3. Border – The outer boundary.
4. Margin – Space outside the border.

```
div {
    width: 300px;
    padding: 10px;
    border: 5px solid black;
    margin: 20px;
}
```

JavaScript (JS):

What is JavaScript?

JavaScript is a programming language that makes web pages interactive. It handles user input, animations, calculations, and dynamic content.

Adding JavaScript to HTML

JavaScript can be added by:

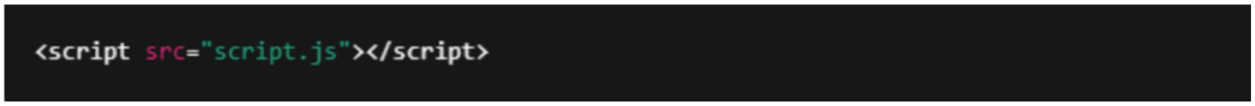
Inline (Not Recommended)

```
<button onclick="alert('Hello!')">Click Me</button>
```

Internal (Inside a <script> tag)

```
<script>
    alert("Welcome to my website!");
</script>
```

External (Best Practice)



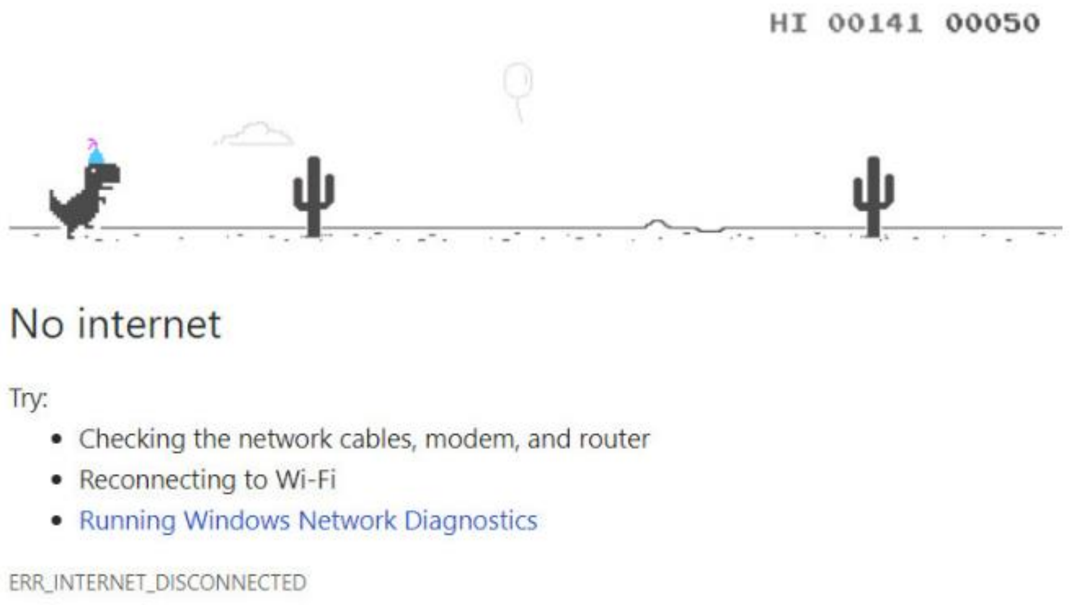
```
<script src="script.js"></script>
```

- HTML structures the content.
- CSS styles and formats the content.
- JavaScript makes the content interactive.

REFERENCE AND RESEARCH

Hyper-casual games are lightweight, easy-to-play games with minimal mechanics and instant gameplay. They focus on simple yet addictive interactions, allowing players to start playing with little to no tutorial. These games typically have short session lengths and are designed for quick entertainment. Here are some of the games I have played and researched to make my game:

1. Google Dino Game:



No internet

Try:

- Checking the network cables, modem, and router
- Reconnecting to Wi-Fi
- Running Windows Network Diagnostics

ERR_INTERNET_DISCONNECTED

Game Mechanics:

- The player controls a **T-Rex** that automatically runs from left to right.
- The goal is to **avoid obstacles** (such as cacti and flying pterodactyls).
- The player can make the **dino jump** using the **spacebar** (on PC) or by **tapping the screen** (on mobile).
- As the game progresses, **speed increases**, making it more challenging.
- It has a **night mode** (screen turns dark) after reaching a certain score.

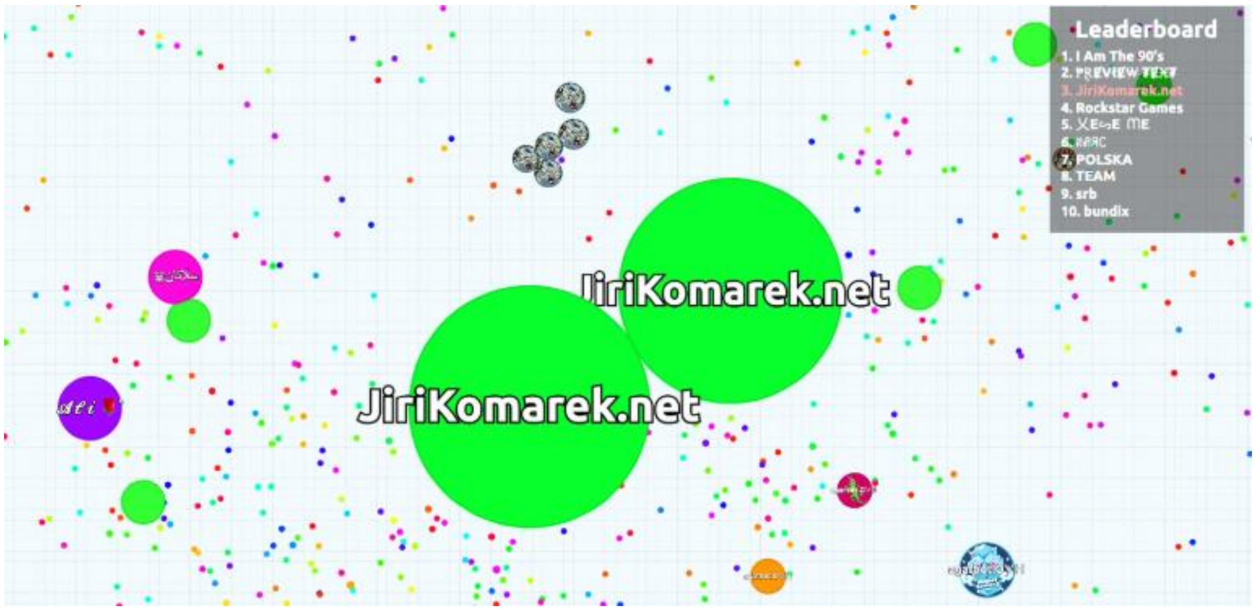
Development & Technology:

- The game is built using **JavaScript**, **HTML5**, and **CSS**.
- The graphics are simple **pixel art**, keeping it lightweight and fast.
- It uses **collision detection** to determine when the dino hits an obstacle.
- The game has an **infinite loop**, meaning it continues until the player loses.

Fun Fact:

- It was created by **Sebastien Gabriel**, a Chrome UX designer, in **2014**.
- The T-Rex theme was chosen as a joke about **going back to the prehistoric era when there was no internet**.

2. Agar.io:



Agar.io is a **multiplayer online action game** where players control a **cell** in a petri dish-like environment. The goal is to **grow larger** by consuming smaller cells while avoiding being eaten by bigger ones.

Gameplay:

- Players start as a **small cell** and must consume **pellets (food) and smaller players** to grow.
- Larger cells move **slower**, while smaller cells are **faster** and more agile.
- Players can **split** their cell into two to launch part of themselves forward, a useful strategy for attacking or escaping.

- Cells can also **eject mass** to feed smaller allies or lighten up for quick escapes.
- The game features **team modes**, **FFA (Free-for-All)**, and **battle royale-style gameplay**.

Technology & Development:

- Developed by **Matheus Valadares**, a Brazilian developer, and released in **2015**.
- Uses **HTML5 and JavaScript (for frontend) and C++ (for backend)**.
- Runs on **WebSockets**, enabling real-time multiplayer gameplay.

Popularity & Impact:

- Became a viral hit due to its **simple yet competitive gameplay**.
- Inspired similar games like **Slither.io** and **Diep.io**.

3. Krunker.io:

Krunker.io is a **fast-paced, multiplayer first-person shooter (FPS)** that can be played directly in a web browser. It features a variety of **game modes, weapons, and maps**, making it one of the most popular browser-based FPS games.

Gameplay:

- Players choose a **class** (such as Triggerman, Hunter, Detective) with different weapons.
- The objective is to **eliminate opponents** in modes like **Free-for-All, Team Deathmatch, Capture the Flag**, and more.
- The game has **fluid movement mechanics**, including **bunny hopping and sliding** for increased speed.
- Players can **customize skins, mods, and maps**, allowing for a high level of personalization.

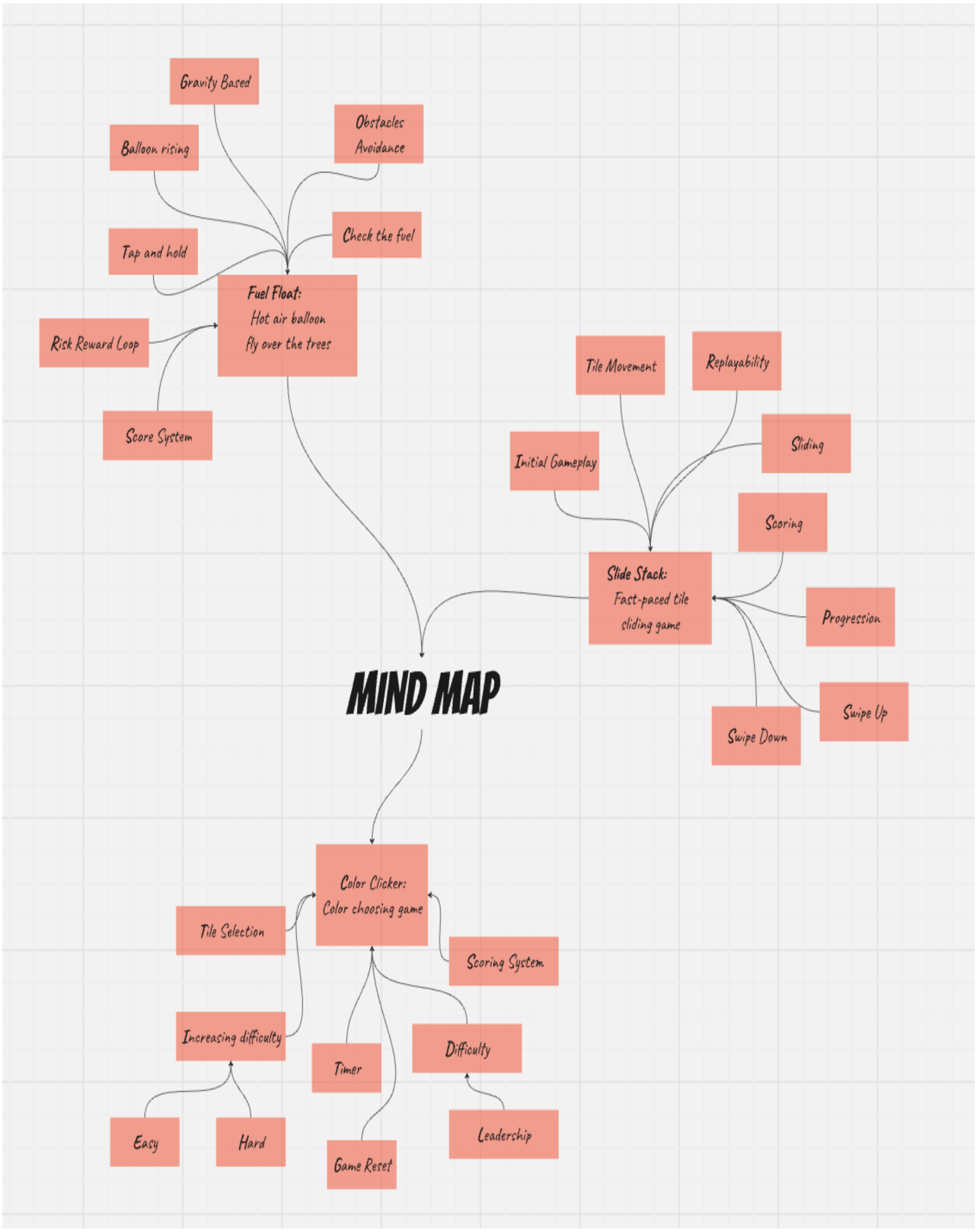
Technology & Development:

- Developed by **Sidney de Vries** and released in **2018**.
- Built using **HTML5, JavaScript, and Three.js (for 3D graphics)**.
- Uses **WebSockets** for real-time multiplayer interaction.
- Runs efficiently in browsers without needing downloads, but also has a standalone client for better performance.

Popularity & Impact:

- One of the most successful browser-based FPS games, with **millions of active players**.
- Features a **marketplace for trading skins and items**.

MIND MAP

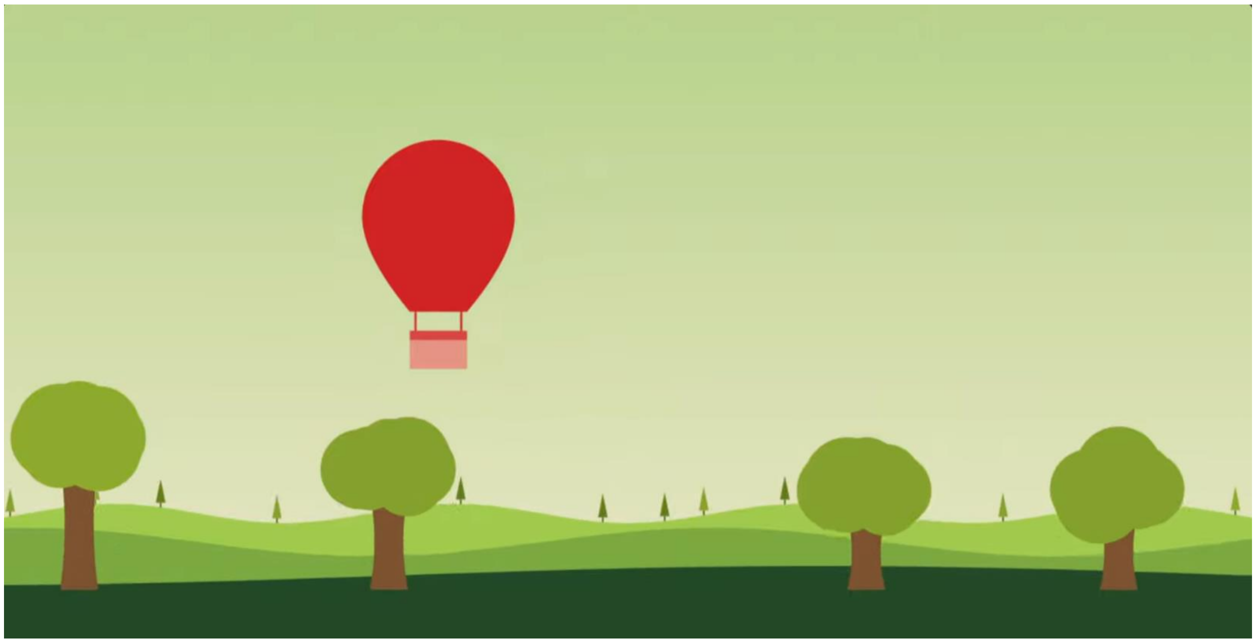

Gravity Based
Balloon rising
Obstacles Avoidance
Check the fuel
Tap and hold
Fuel Float: Hot air balloon fly over the trees
Risk Reward Loop
Score System
Tile Movement
Replayability
Initial Gameplay
Sliding
Scoring
Slide Stack: Fast-paced tile sliding game
Progression
MIND MAP
Swipe Up
Swipe Down
Color Clicker: Color choosing game
Tile Selection
Scoring System
Increasing difficulty
Timer
Difficulty
Easy
Hard
Game Reset
Leadership

CONCEPT GENERATION

Concept 1: FUEL FLOAT

Concept Overview:

FUEL FLOAT is a hyper-casual web game where you control a vibrant hot air balloon soaring through a dense forest. The goal is simple: dodge as many trees as possible while staying airborne. The player holds down the screen to keep the balloon rising, but there's a catch — the balloon's fuel drains the longer you hold, adding a layer of tension to every second of flight. Releasing the screen lets the balloon descend, helping avoid obstacles but also bringing it dangerously close to the forest floor.



Inspired from **Flappy Bird, Jetpack Joyride (but smoother and more gradual).**

Game Mechanics:

1. Tap-and-Hold / Touch Control: Holding down the screen makes the balloon rise; releasing makes it fall.

2. Fuel Consumption Mechanic: The balloon consumes fuel while rising. Holding too long drains fuel faster.

3. Obstacle Avoidance: Trees and possibly branches act as obstacles to dodge.

4. Gravity-Based Movement: Releasing touch lets gravity naturally pull the balloon down.

5. Risk-Reward Loop: Rising high helps avoid ground obstacles but drains fuel; staying low conserves fuel but is risky.

6. Score System: Score increases the longer the player survives or based on distance.

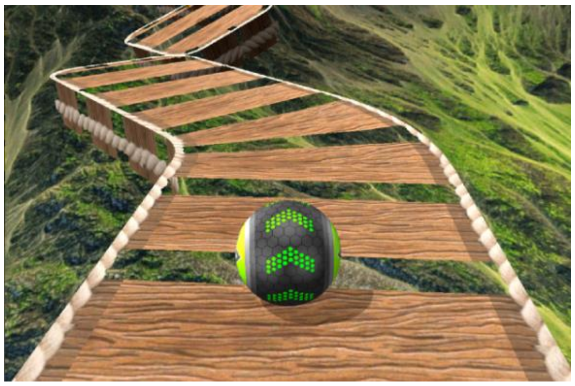
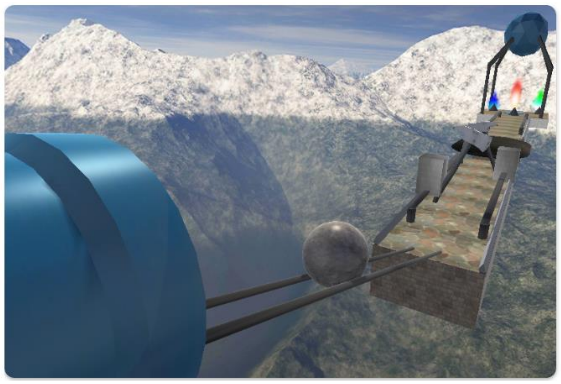
Implementation:

- HTML: Provides the structure with a button and score display.
- CSS: Styles the button and text for a visually appealing experience.
- JavaScript: Handles click events and updates the score dynamically.

Concept 2: SLIDE STACK

Concept Overview:

SLIDE STACK is a fast-paced, one-thumb hyper-casual game where players control a sliding tile that automatically moves forward on a never-ending path made of stacked platforms. Some stacks are too high to pass — so players must **swipe down** to break the tile and shrink it or **swipe up** to grow it temporarily to bridge small gaps. The goal is to survive as long as possible by perfectly managing your tile's size while dodging and adapting to the increasingly chaotic terrain.

Inspired from ball rolling games from playstore

Game Mechanics:

1. Initial Gameplay : The player sees a simple tile sliding forward on a stack path.

2. Core Loop Begins: The tile moves forward automatically. Stacks of varying heights begin appearing.

3. Scoring & Progression: Score increases the longer you survive.

4. Game Over: Player crashes into a stack or falls through a gap.

5. Replayability: Unlock new tile skins (block, jelly, metal, etc.) based on score milestones. Background themes unlock with progression (city, cyber, desert, clouds).

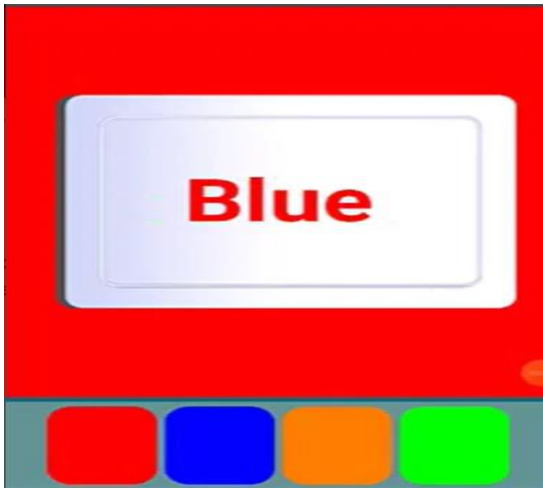
Implementation:

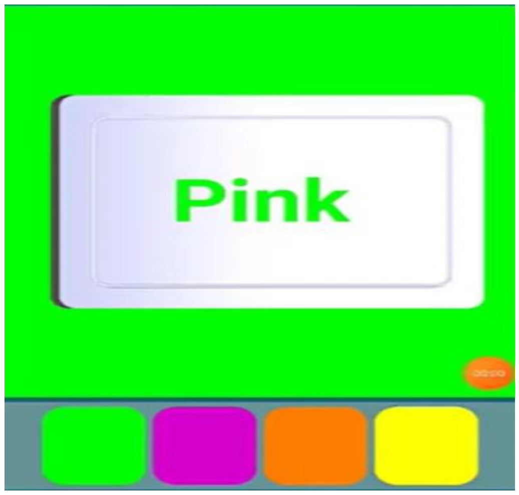
- HTML: Creates a game container and player element.
- CSS: Defines the game area, player, and obstacles.
- JavaScript:
    - Listens for keyboard input to move the player.
    - Generates obstacles at random positions.
    - Checks for collisions and ends the game when necessary.

Concept 3: COLOR CLICKER

Concept Overview:

COLOR CLICKER is a lightning-fast reflex game where a color name appears at the top of the screen and below it is multiple-colored tiles. You must tap the tile that matches the actual color that appear before the time runs out.





Game Mechanics:

1. Color Display Mechanic: A color name (like "Red", "Blue", "Green") is displayed at the top.

2. Tile Selection Mechanic: A set of color tiles/buttons is shown on the screen.Player must tap the tile that matches the correct color based on the rule .

3. Timer / Countdown Mechanic: Players are given a short time limit (like 2–3 seconds) to make a decision. If time runs out, it’s a miss/fail.

4. Scoring System: Points are awarded for each correct tap.

5. Increasing Difficulty: Timer gets shorter as the player can choose the difficulty such as easy or hard.

Implementation:

- **HTML:** Creates the game board and card elements.
- **CSS:** Styles the cards, game board, and animations for flipping.
- **JavaScript:**
    - Click on the color shapes.
    - Leadership board can collect the players stats.
    - Game winning celebration animation.
    - Resetting of the leaderboard.
    - Set the difficulty of the game with easy and hard mode.

FINALNALIZED CONCEPT

Game Overview: Fast Reflex Game

Title: COLOR CLICKER

Genre: Hyper Casual Web Game

Platform: Web

Concept Overview:

COLOR CLICKER is a lightning-fast reflex game where a color name appears at the top of the screen and below it is multiple-colored tiles. You must tap the tile that matches the actual color that appear before the time runs out.

Game Mechanics:

1. Color Display Mechanic: A color name (like "Red", "Blue", "Green") is displayed at the top.

2. Tile Selection Mechanic: A set of color tiles/buttons is shown on the screen.Player must tap the tile that matches the correct color based on the rule .

3. Timer / Countdown Mechanic: Players are given a short time limit (like 2–3 seconds) to make a decision. If time runs out, it’s a miss/fail.

4. Scoring System: Points are awarded for each correct tap.

5. Increasing Difficulty: Timer gets shorter as the player can choose the difficulty such as easy or hard.

Design Considerations:

1. User-Centered Design:

- **Simple and Intuitive UI:** The interface is clean and self-explanatory, making it easy for players of all ages to understand the gameplay.
- **Immediate Feedback:** Users get instant feedback through color changes, score updates, and game-over messages.
- **Minimal Text Input:** Only a name and difficulty level are required, reducing cognitive load.

2. Game Flow and Challenge Balancing:

- **Difficulty Selection:** The game offers *Easy* and *Hard* modes to accommodate both beginners and challenge-seeking players.
- **Gradual Pacing:** Countdown timers are tuned per difficulty level, ensuring a sense of urgency without being overwhelming.
- **Randomization:** Color selections are randomized to ensure replayability and prevent predictability.

3. Visual Design:

- **Bright, High-Contrast Colors:** Designed for quick visual recognition and engagement.
- **Rounded Buttons & Containers:** Soft shapes make the interface friendly and reduce visual sharpness.
- **Responsive Layout:** Uses flexbox to adapt to different screen sizes and orientations.

4. Cognitive Load and Accessibility:

- **Visual Matching Task:** Simple decision-making (match color) avoids cognitive overload.
- **Readable Fonts and Button Sizes:** Text and clickable areas are large and legible, aiding accessibility.

- **Timer Display:** Countdown shown clearly to support reaction-based gameplay.

5. Motivation and Retention:

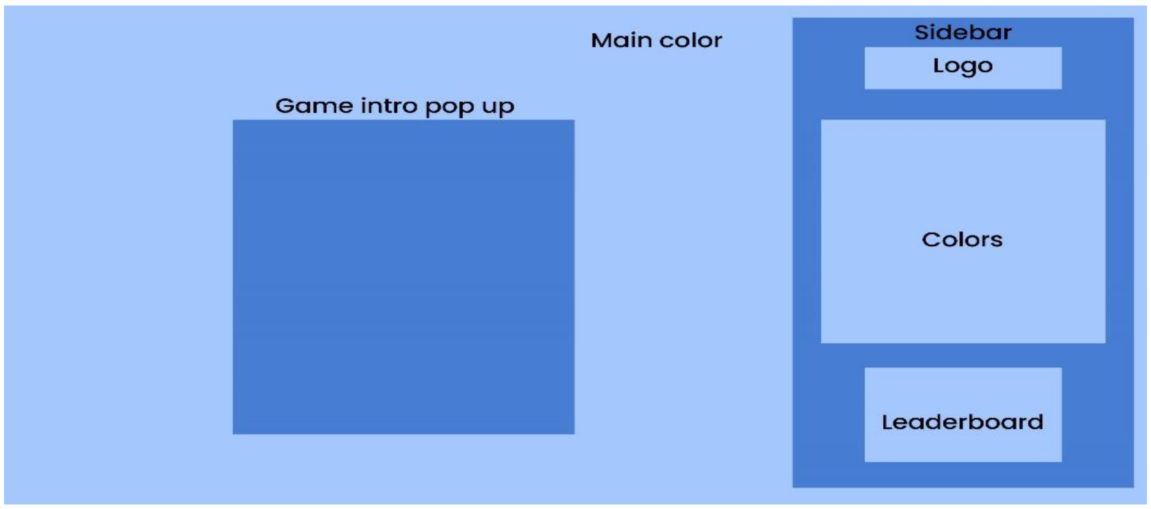
- **Leaderboard:** Promotes competitiveness and replayability by showing high scores and player names.
- **Confetti Celebration:** Adds a psychological reward for achieving high scores.
- **Personalization:** Inputting a name increases player engagement and identity within the game.

6. Persistence and Local Storage:

- **Leaderboard Stored in LocalStorage:** Game data persists across sessions without requiring a database.
- **Reset Functionality:** Lets users clear progress and restart fresh whenever they want.

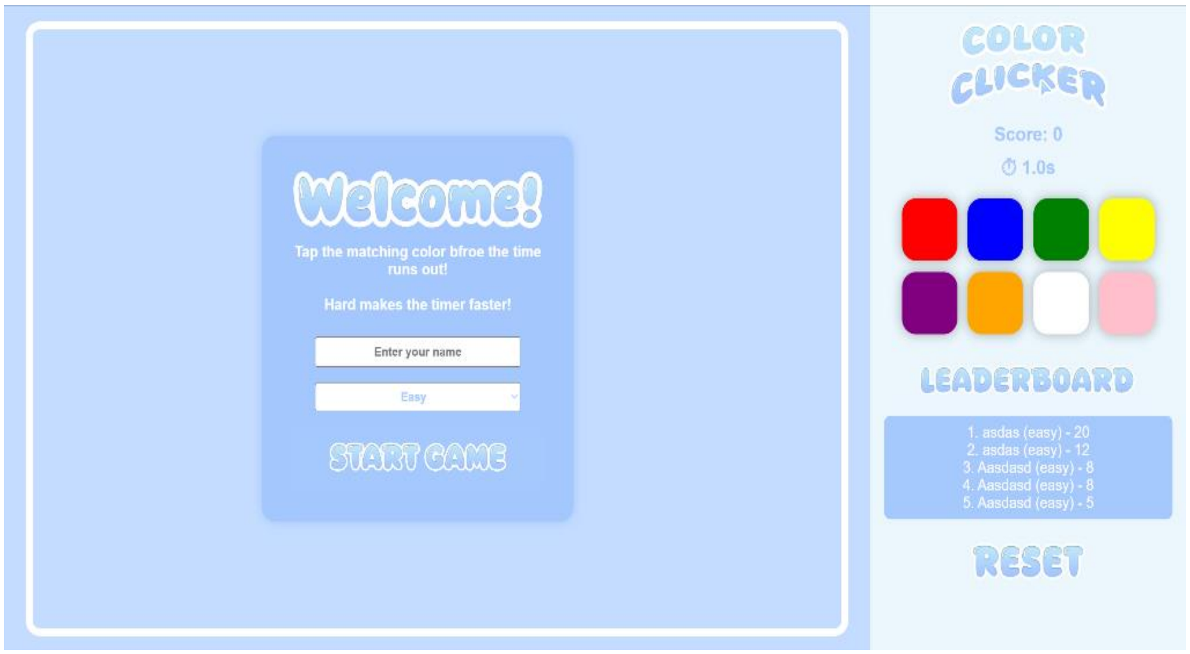
7. Technical Simplicity:

- **No External Libraries:** Game is built with HTML, CSS, and JavaScript only—easy to maintain and deploy.
- **Optimized for Performance:** Lightweight logic ensures smooth performance even on lower-end devices.



Implementation:

- **HTML:** Creates the game board and card elements.
- **CSS:** Styles the cards, game board, and animations for flipping.
- **JavaScript:**
    - Click on the color shapes.
    - Leadership board can collect the players stats.
    - Game winning celebration animation.
    - Resetting of the leaderboard.
    - Set the difficulty of the game with easy and hard mode.

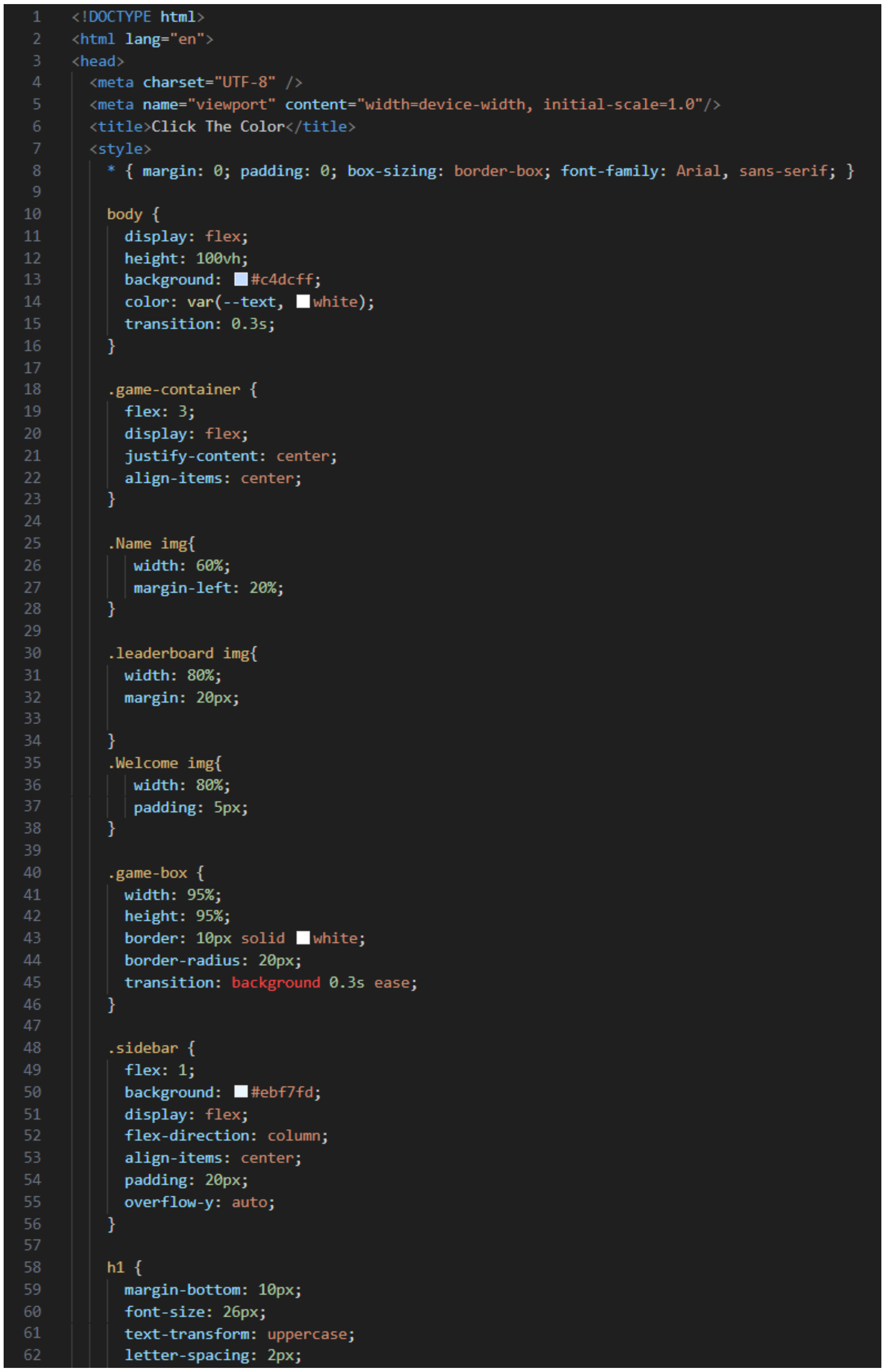


Tap the matching color bfroe the time runs out!

Hard makes the timer faster!

1. asdas (easy) - 20
2. asdas (easy) - 12
3. Aasdasd (easy) - 8
4. Aasdasd (easy) - 8
5. Aasdasd (easy) - 5

CODE EXPLANATION

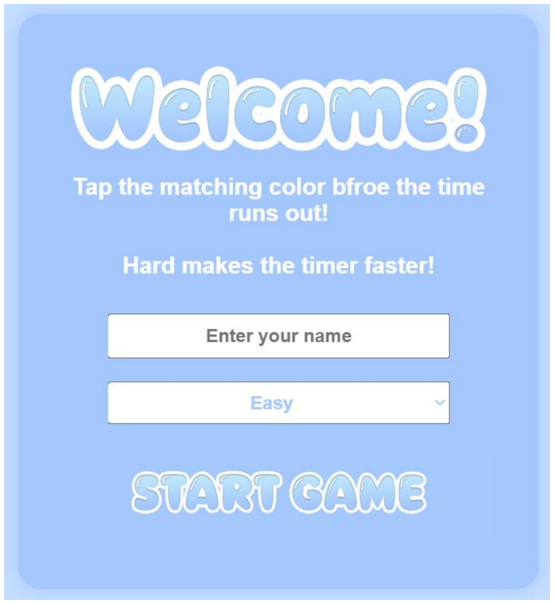


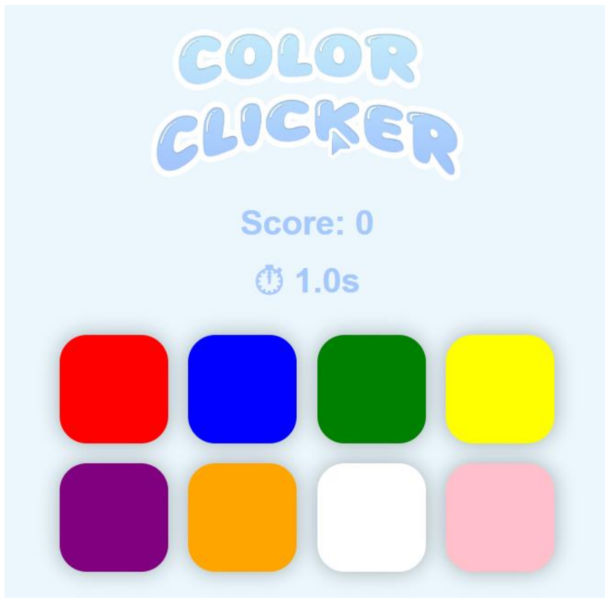
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Click The Color</title>
  <style>
    * { margin: 0; padding: 0; box-sizing: border-box; font-family: Arial, sans-serif; }

    body {
      display: flex;
      height: 100vh;
      background: #c4dcff;
      color: var(--text, white);
      transition: 0.3s;
    }

    .game-container {
      flex: 3;
      display: flex;
      justify-content: center;
      align-items: center;
    }

    .Name img{
       width: 60%;
       margin-left: 20%;
    }

    .leaderboard img{
      width: 80%;
      margin: 20px;

    }
    .Welcome img{
       width: 80%;
       padding: 5px;
    }

    .game-box {
      width: 95%;
      height: 95%;
      border: 10px solid white;
      border-radius: 20px;
      transition: background 0.3s ease;
    }

    .sidebar {
      flex: 1;
      background: #ebf7fd;
      display: flex;
      flex-direction: column;
      align-items: center;
      padding: 20px;
      overflow-y: auto;
    }

    h1 {
      margin-bottom: 10px;
      font-size: 26px;
      text-transform: uppercase;
      letter-spacing: 2px;
```

Key Features:

1. Gameplay Features:

- Fast-paced Reaction Game: Players must tap the button that matches the background color before the timer runs out.
- Two Difficulty Modes:
    - Easy: More time to react, slower countdown.
    - Hard: Less time, faster countdown.
- Randomized Color Challenges: Game background randomly switches between 8 colors, requiring quick matching.

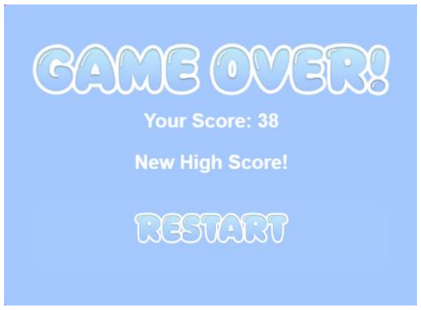

Welcome!
Tap the matching color bfroe the time runs out!
Hard makes the timer faster!
Enter your name
Easy
START GAME



2. User Interface & UX:

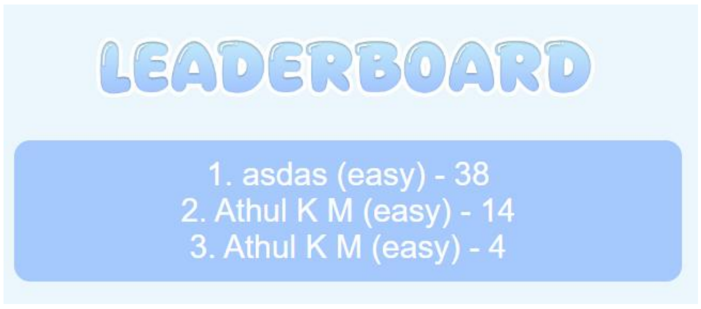
- Intro Popup: Stylish introduction with game rules, name input, and difficulty selection.

- Game Over Popup: Displays final score and “New High Score” message with confetti animation.
- Live Scoreboard & Timer: Real-time score and countdown display.
- Clickable Colored Buttons: Buttons visually match the possible background colors.



Your Score: 38

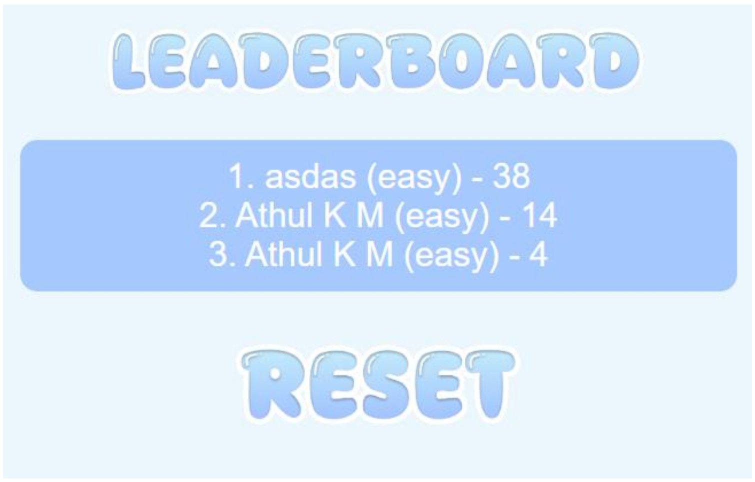
New High Score!



1. asdas (easy) - 38
2. Athul K M (easy) - 14
3. Athul K M (easy) - 4

3. Leaderboard System:

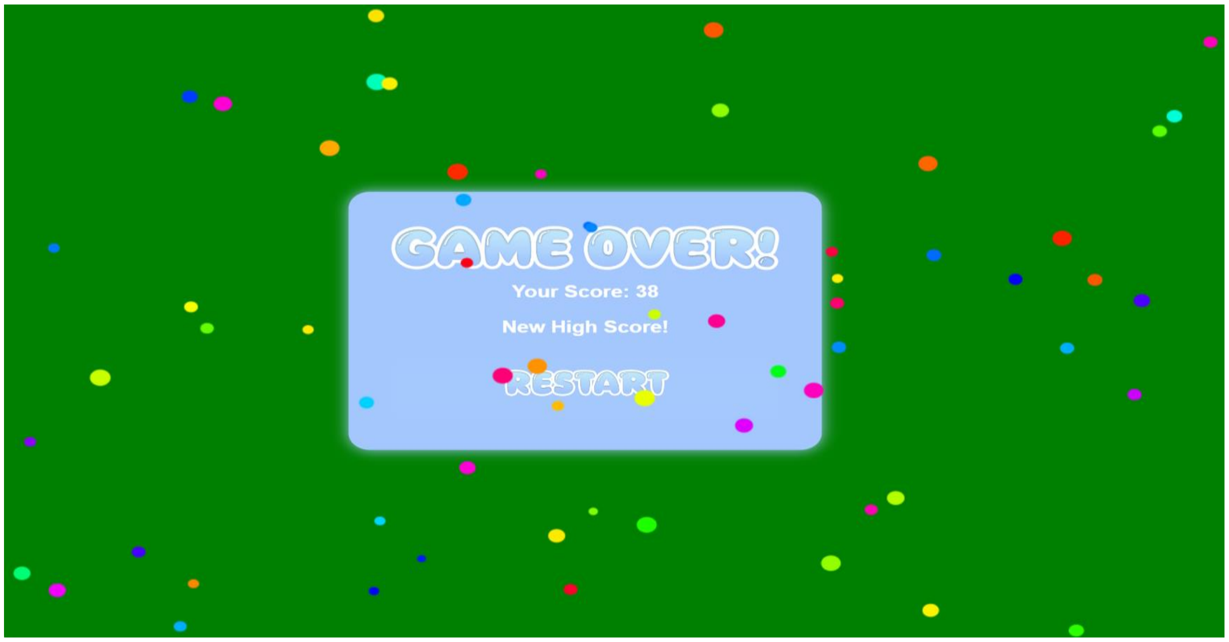
- Player Name & Difficulty Tracking: Stores top 5 players with name, score, and difficulty.
- LocalStorage Integration: Leaderboard data is saved even after refreshing the page.
- Reset Leaderboard Option: Users can reset the leaderboard to start fresh.



1. asdas (easy) - 38
2. Athul K M (easy) - 14
3. Athul K M (easy) - 4

4. Visual Effects & Animations:

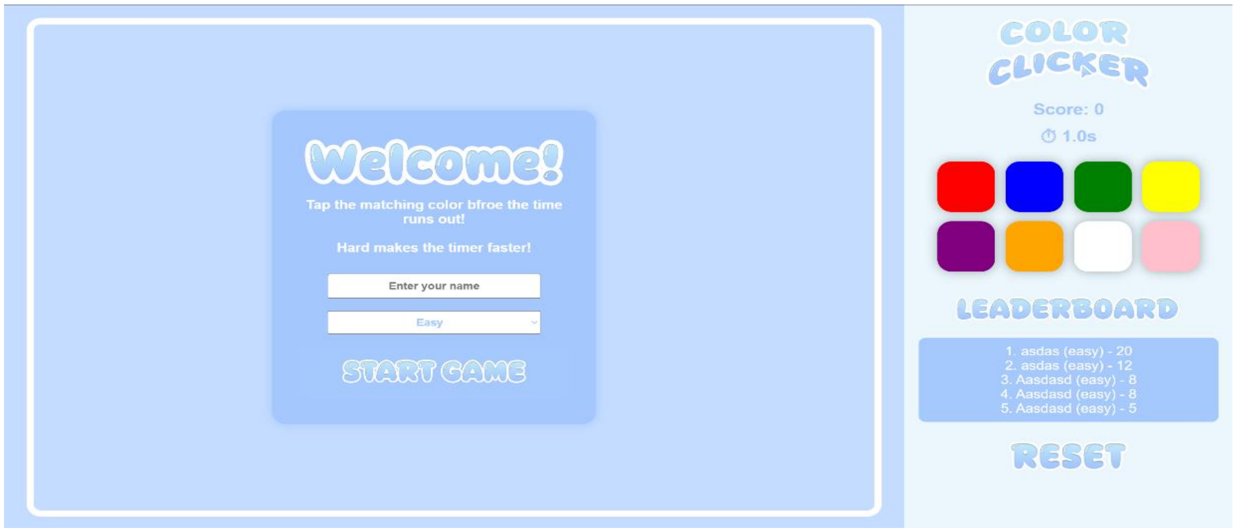
- Confetti Celebration: Triggered on setting a new high score.
- Animated Popups: Slight hover/animation effects to add polish to the UI.
- Colorful Theme: Bright, kid-friendly design with visually engaging graphics.



5. Other Features:

- Responsive Design: Layout adjusts to screen size with flexible flexbox layout.
- Lightweight Performance: No external libraries or frameworks—just plain HTML, CSS, and JavaScript.
- Image-Based Buttons & Titles: Custom PNG images enhance visual appeal (e.g., title, reset, start).

OUTPUT



Tap the matching color bfroe the time runs out!

Hard makes the timer faster!

Enter your name

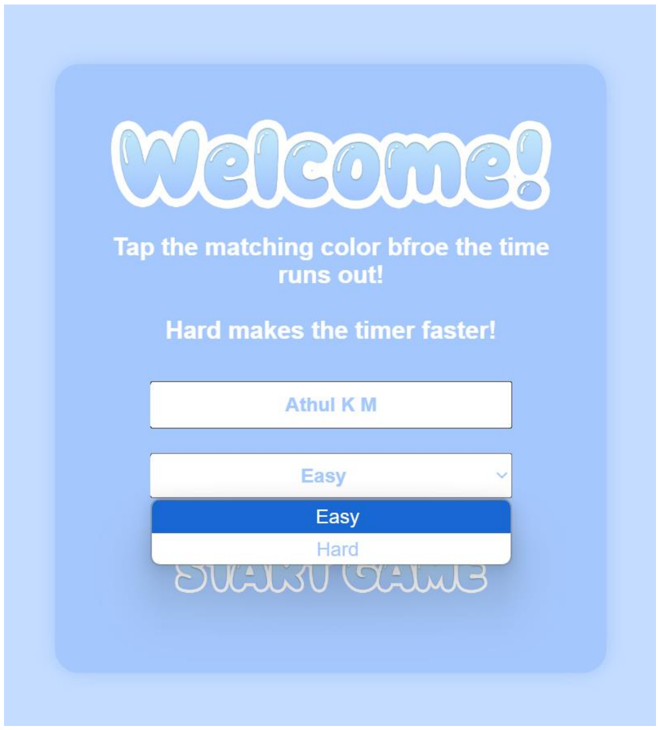
Easy

Score: 0

⏱ 1.0s

1. asdas (easy) - 20
2. asdas (easy) - 12
3. Aasdasd (easy) - 8
4. Aasdasd (easy) - 8
5. Aasdasd (easy) - 5

Game Interface of **COLOR CLICKER**

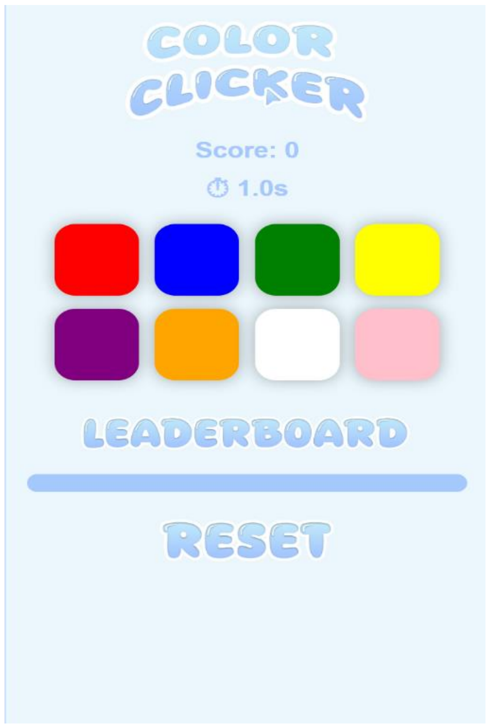


Tap the matching color bfroe the time runs out!

Hard makes the timer faster!

Athul K M

Easy

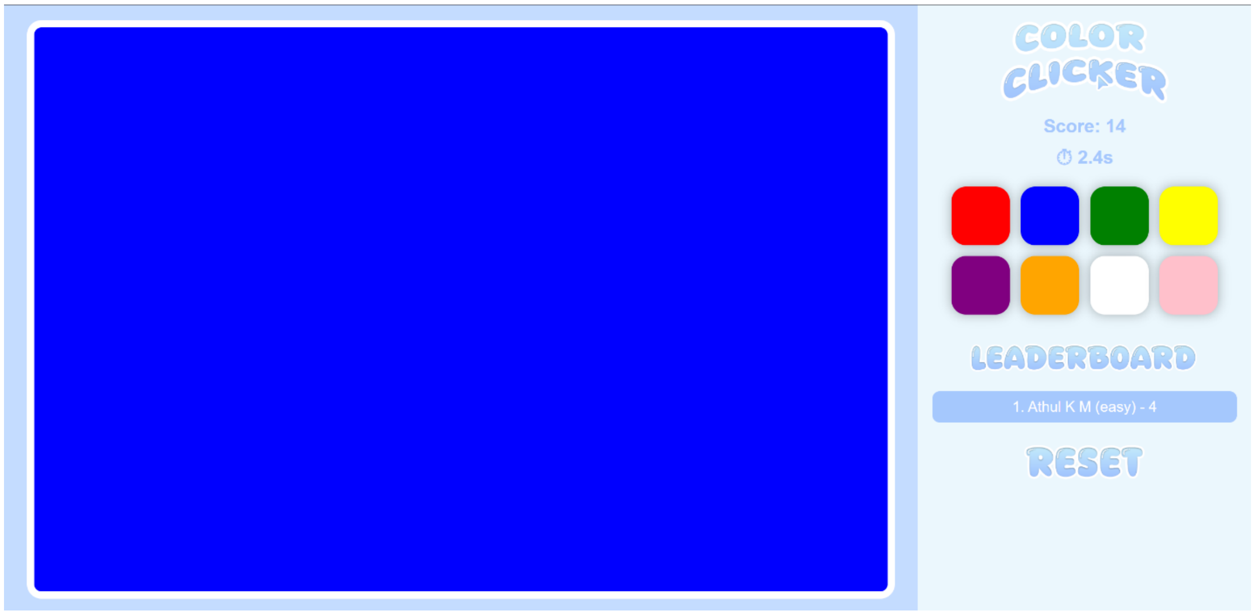


Score: 0

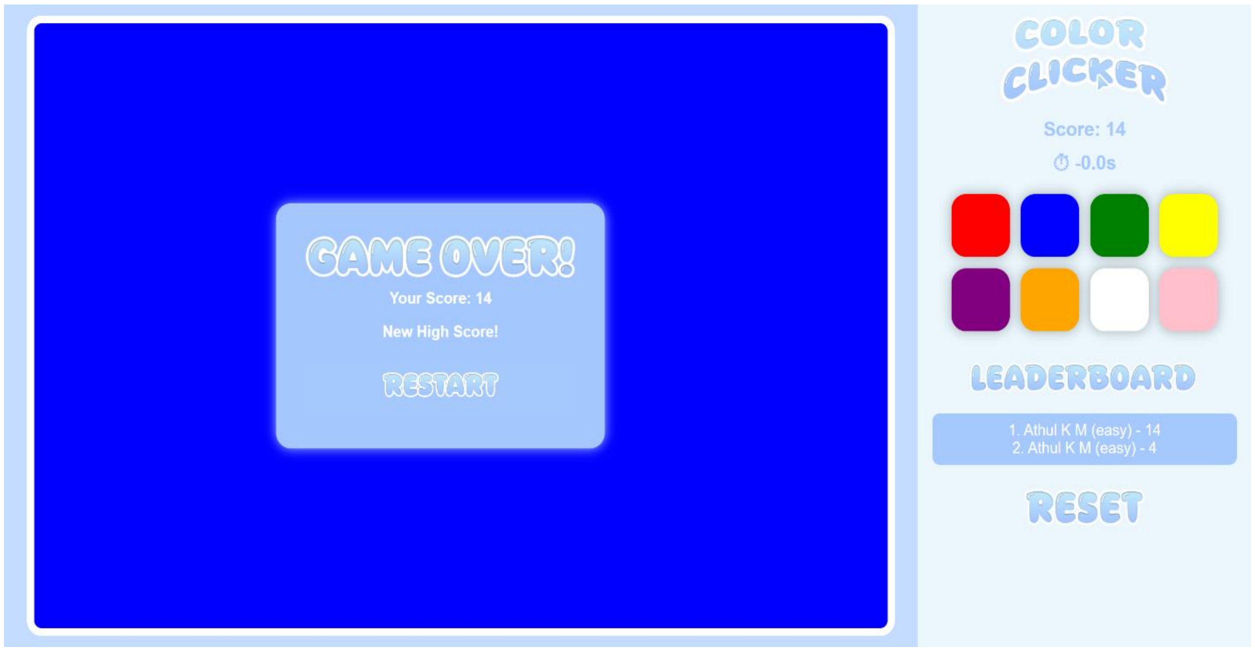
⏱ 1.0s

Introduction of the game

Sidebar of the game



Gameplay



Game Over