**People’s Democratic Republic Of Algeria**

**Ministry Of Higher Education And Scientific Research**

**AHMED GHOUL**

**BY:**

**AHMED GHOUL AND ABDOU ABDESSAMED**

NWT2

**Project Report:**

**Interactive "Whack-a-Mole" Game**

**Table of Contents**

1. Project Overview
2. Key Features
   * Dynamic Grid Creation
   * Random Mole Appearance
   * Scoring System
   * Game Timer
   * Lives System
   * Pause and Restart Buttons
3. Technical Implementation
   * HTML Elements
   * JavaScript Logic
   * Styling and Animation
   * Audio Integration
4. Challenges and Solutions
5. Future Improvements
6. Conclusion

# Project Overview

This project is a browser-based interactive "Whack-a-Mole" game. It allows users to set game parameters such as grid dimensions, game duration, and mole appearance speed. The objective is to score points by clicking on randomly appearing moles within the allocated time while avoiding incorrect clicks. The game includes a scoring system, limited lives, and sound effects for user engagement.

# Key Features

## Dynamic Grid Creation:

Users specify the grid dimensions through input fields, and the grid dynamically adjusts to these inputs.

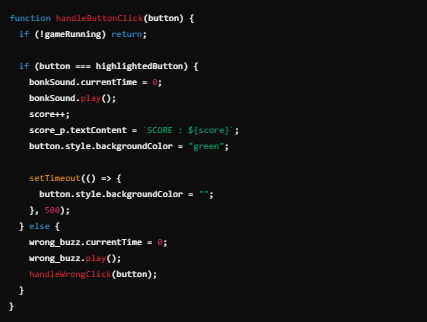
# Random Mole Appearance:

Moles appear at random buttons on the grid, ensuring no immediate repetition of the same location.



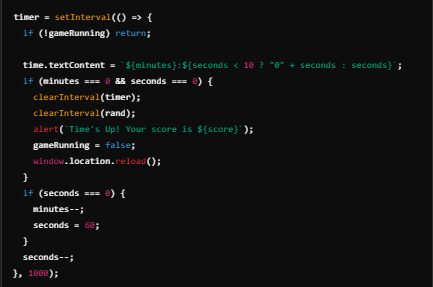
# Scoring System:

Players lose lives for incorrect clicks, with the game ending when lives reach zero.



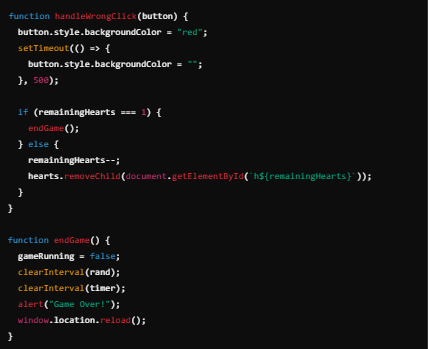
## Game Timer:

A countdown timer tracks the remaining time and ends the game when time runs out.



# Lives System:

Players lose lives for incorrect clicks, with the game ending when lives reach zero.



# Pause and Restart Buttons

Provides control over the game state with pause and restart functionality.



# Technical Implementation

## HTML Elements:

* + Dynamic creation of grid buttons (<button> elements) for the game interface.
  + Input fields for game parameters such as rows, columns, duration, and speed.

## JavaScript Logic:

* + **Game Initialization:**
    - Generates the grid layout and prepares the game environment.
  + **Mole Logic:**
    - Randomly selects grid buttons to display moles.
    - Clears moles after a specified timeout.
  + **Score Tracking:**
    - Updates the score on correct clicks.
  + **Lives Management:**
    - Reduces lives for incorrect clicks.
    - Ends the game when lives reach zero.
  + **Timer:**
    - Tracks the remaining time and ends the game when time runs out.

## Styling and Animation:

* + Utilizes CSS for mole animations and button state changes.
  + Custom cursors for interactive feedback.

## Audio Integration:

* + Implements audio feedback for user actions using the Audio API.
  + Handles audio playback errors gracefully.

# Challenges and Solutions

1. **Challenge:** Preventing duplicate mole appearances at the same location.
   * **Solution:** Ensured the random index generator avoids repeating the last used index.
2. **Challenge:** Managing the game state during pause and resume actions.
   * **Solution:** Used a gameRunning flag to control the game loop and input handling.
3. **Challenge:** Smoothly removing moles after timeout or incorrect interactions.
   * **Solution:** Added CSS animations for mole removal and ensured DOM updates synchronized with the game logic.

# Future Improvements

1. Add difficulty levels to vary mole appearance speed and game duration dynamically.
2. Implement a high-score leaderboard to track player performance across sessions.
3. Improve mobile responsiveness and touch input support.
4. Integrate themes or skins for visual customization.
5. Add accessibility features such as keyboard navigation and sound toggles.

# Conclusion

This project successfully demonstrates a fully functional "Whack-a-Mole" game with interactive features, sound effects, and dynamic gameplay. It is a fun, engaging, and scalable application that highlights key JavaScript and DOM manipulation concepts.