

# Documentación del Proyecto Integrador de la 1ª Evaluación

ROBERTO FERNÁNDEZ CORDERO

## Informe de Diseño y Desarrollo del Juego

**1. Introducción** El objetivo del proyecto ha sido desarrollar una aplicación web interactiva basada en un juego de tablero (10x10). El usuario controla a un héroe que debe alcanzar una meta (casilla 9,9) mediante tiradas de dados, implementando lógica de validación, persistencia de datos y manipulación del DOM.

## 2. Decisiones de Diseño

- **Arquitectura Modular:** Se ha separado la estructura (HTML), el estilo (CSS) y la lógica (JS) para facilitar el mantenimiento y la escalabilidad del código.
- **Generación Dinámica del DOM:** En lugar de escribir la tabla en HTML estático, se optó por generarla mediante JavaScript (`generarTablero`). Esto permite reiniciar la partida limpiamente sin recargar la página necesariamente y facilita futuras expansiones (tableros de distinto tamaño).
- **Validación Preventiva de Movimiento:** Para la experiencia de usuario, se decidió implementar una lógica que calcula *a priori* si un movimiento es válido. Si la suma de la posición actual más el dado excede los límites del tablero, el sistema no permite la interacción, evitando errores y frustración en el usuario (manejo de casos de borde).
- **Persistencia Local (LocalStorage):** Se eligió `localStorage` para guardar el récord, permitiendo que la puntuación máxima se mantenga incluso si el usuario cierra el navegador.

## 3. Desafíos Encontrados y Soluciones

- **Lógica de Movimiento Exacto:** Uno de los retos fue implementar que el usuario pudiera elegir casillas intermedias pero solo si el movimiento completo era válido.
  - *Solución:* Se implementó un algoritmo que primero verifica el destino final (`posicion + dado`). Solo si este destino está dentro de la matriz [0-9], se habilitan visualmente las casillas intermedias mediante la clase CSS `.activa`.
- **Sincronización de Eventos:** Al ganar, el `alert` del navegador detenía la renderización del último movimiento del héroe.
  - *Solución:* Se utilizó `setTimeout` con un pequeño retardo para asegurar que el navegador tuviera tiempo de actualizar el DOM (mover la imagen) antes de mostrar el mensaje de victoria y reiniciar el juego.

- **Gestión del Estado del DOM:** Al reiniciar partidas, se acumulaban elementos o eventos antiguos.
  - *Solución:* Se asegura la limpieza de contenedores (`innerHTML = ""`) y la eliminación de *event listeners* obsoletos al regenerar el tablero.

**4. Conclusión** La aplicación cumple con todos los requisitos funcionales especificados, ofreciendo una interfaz robusta, validación de errores de entrada y una experiencia de juego fluida y persistente.