Hand Calculator Project Code

# File: index.html

<!DOCTYPE html>  
<html lang="ar" dir="rtl">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>حاسبة الهند</title>  
   
 <!-- أيقونات التطبيق -->  
 <link rel="icon" type="image/svg+xml" href="app-icon.svg">  
 <link rel="apple-touch-icon" href="app-icon.svg">  
   
 <!-- إعدادات iOS -->  
 <meta name="apple-mobile-web-app-capable" content="yes">  
 <meta name="apple-mobile-web-app-status-bar-style" content="black-translucent">  
 <meta name="apple-mobile-web-app-title" content="حاسبة الهند">  
 <meta name="theme-color" content="#1A237E">  
   
 <!-- إعدادات التطبيق على الهاتف -->  
 <meta name="apple-mobile-web-app-capable" content="yes">  
 <meta name="apple-mobile-web-app-status-bar-style" content="black-translucent">  
 <meta name="apple-mobile-web-app-title" content="حاسبة الهند">  
 <link rel="stylesheet" href="style.css">  
 <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>  
</head>  
<body>  
 <div class="controls-top">  
 <button onclick="toggleTheme()" id="themeToggle" class="control-btn theme-btn">  
 <span class="light-icon">☀️</span>  
 <span class="dark-icon">🌙</span>  
 </button>  
 <button onclick="showCreator()" id="infoButton" class="control-btn info-btn" title="معلومات عن المطور">  
 ℹ️  
 </button>  
 </div>  
 <div class="container">  
 <h1>حاسبة الهند</h1>  
   
 <div class="players">  
 <div class="player">  
 <h2>الفريق 1</h2>  
 <input type="text" id="player1Name" placeholder="اسم الفريق 1" class="player-name">  
 <div class="special-moves">  
 <button onclick="addSpecialMove(1, 'khales')" class="special-btn khales">خالص</button>  
 <button onclick="addSpecialMove(1, 'hand')" class="special-btn hand">هند</button>  
 <button onclick="addSpecialMove(1, 'ajkari')" class="special-btn ajkari">أجكري</button>  
 <button onclick="addSpecialMove(1, 'amkari')" class="special-btn amkari">أمكري</button>  
 <button onclick="addSpecialMove(1, 'amkariWalon')" class="special-btn amkari-walon">أمكري ولون</button>  
 </div>  
 <div class="score-input">  
 <div class="score-row">  
 <input type="number" id="player1Score" placeholder="النقاط" min="0">  
 <input type="number" id="player1Down" placeholder="نزول" min="0" class="down-input">  
 <input type="number" id="player1Minus" placeholder="خصم" min="0" class="minus-input">  
 </div>  
 <button onclick="addScore(1)">إضافة</button>  
 </div>  
  
 <div class="total-score">المجموع: <span id="player1Total">0</span></div>  
 </div>  
  
 <div class="player">  
 <h2>الفريق 2</h2>  
 <input type="text" id="player2Name" placeholder="اسم الفريق 2" class="player-name">  
 <div class="special-moves">  
 <button onclick="addSpecialMove(2, 'khales')" class="special-btn khales">خالص</button>  
 <button onclick="addSpecialMove(2, 'hand')" class="special-btn hand">هند</button>  
 <button onclick="addSpecialMove(2, 'ajkari')" class="special-btn ajkari">أجكري</button>  
 <button onclick="addSpecialMove(2, 'amkari')" class="special-btn amkari">أمكري</button>  
 <button onclick="addSpecialMove(2, 'amkariWalon')" class="special-btn amkari-walon">أمكري ولون</button>  
 </div>  
 <div class="score-input">  
 <div class="score-row">  
 <input type="number" id="player2Score" placeholder="النقاط" min="0">  
 <input type="number" id="player2Down" placeholder="نزول" min="0" class="down-input">  
 <input type="number" id="player2Minus" placeholder="خصم" min="0" class="minus-input">  
 </div>  
 <button onclick="addScore(2)">إضافة</button>  
 </div>  
  
 <div class="total-score">المجموع: <span id="player2Total">0</span></div>  
 </div>  
 </div>  
  
 <div class="rounds">  
 <h3>سجل الجولات</h3>  
 <div id="roundHistory"></div>  
 <div class="chart-container">  
 <canvas id="scoreChart"></canvas>  
 </div>  
 </div>  
  
 <div class="game-info">  
 <div class="game-settings">  
 <div class="setting-item">  
 <label for="maxRoundsInput">عدد الجولات:</label>  
 <input type="number" id="maxRoundsInput" min="1" max="20" value="7" onchange="updateMaxRounds()">  
 </div>  
 </div>  
 <div class="game-rules">  
 <p class="rules-text">الفائز هو من يجمع أقل نقاط 🌟</p>  
 </div>  
 <div class="round-counter">  
 <span>الجولة الحالية: </span>  
 <span id="currentRound">1</span>  
 <span> / </span>  
 <span id="maxRounds">7</span>  
 </div>  
 </div>  
  
 <div class="stats-container">  
 <div class="stats player1-stats">  
 <h3>إحصائيات الفريق 1</h3>  
 <div>أعلى نقاط: <span id="player1HighScore">0</span></div>  
 <div>متوسط النقاط: <span id="player1Average">0</span></div>  
 <div>عدد الانتصارات: <span id="player1Wins">0</span></div>  
 </div>  
 <div class="stats player2-stats">  
 <h3>إحصائيات الفريق 2</h3>  
 <div>أعلى نقاط: <span id="player2HighScore">0</span></div>  
 <div>متوسط النقاط: <span id="player2Average">0</span></div>  
 <div>عدد الانتصارات: <span id="player2Wins">0</span></div>  
 </div>  
 </div>  
 <div class="game-history">  
 <h3>سجل الألعاب السابقة</h3>  
 <div id="gameHistory" class="history-container"></div>  
 </div>  
  
 <div class="controls">  
 <button onclick="newGame()" class="new-game">🎮 لعبة جديدة</button>  
 <button onclick="undoLastRound()" class="undo">↩️ تراجع</button>  
 <button onclick="resetStats()" class="reset-stats">🔄 إعادة تعيين الإحصائيات</button>  
 </div>  
  
 <div class="winner" id="winnerDisplay"></div>  
 </div>  
 <script src="script.js"></script>  
 <!-- النافذة المنبثقة -->  
 <div class="modal-overlay" id="modalOverlay"></div>  
 <div class="creator-modal" id="creatorModal">  
 <h3>معلومات عن المطور</h3>  
 <p>تم تطوير هذا التطبيق بواسطة</p>  
 <h2>محمد عالي الجش</h2>  
 <p>للتواصل: <a href="mailto:buyer997@hotmail.com">buyer997@hotmail.com</a></p>  
 </div>  
</body>  
</html>

--------------------------------------------------

# File: manifest.json

{  
 "name": "حاسبة الهند",  
 "short\_name": "حاسبة الهند",  
 "icons": [  
 {  
 "src": "playing-card-icon.svg",  
 "sizes": "any",  
 "type": "image/svg+xml",  
 "purpose": "any"  
 },  
 {  
 "src": "icon-192.png",  
 "sizes": "192x192",  
 "type": "image/png"  
 },  
 {  
 "src": "icon-512.png",  
 "sizes": "512x512",  
 "type": "image/png"  
 }  
 ],  
 "theme\_color": "#1A237E",  
 "background\_color": "#ffffff",  
 "display": "standalone",  
 "orientation": "portrait",  
 "scope": "/hand-calculator/",  
 "start\_url": "/hand-calculator/"  
}

--------------------------------------------------

# File: script.js

const TOTAL\_ROUNDS = 7; // عدد الجولات الثابت  
  
  
const specialMoves = {  
 khales: {  
 name: 'خالص',  
 penalty: 200,  
 bonus: -30  
 },  
 hand: {  
 name: 'هند',  
 penalty: 400,  
 bonus: -60  
 },  
 ajkari: {  
 name: 'أجكري',  
 penalty: 800,  
 bonus: -120  
 },  
 amkari: {  
 name: 'أمكري',  
 penalty: 1600,  
 bonus: -240  
 },  
 amkariWalon: {  
 name: 'أمكري ولون',  
 penalty: 2400,  
 bonus: -320  
 }  
};  
  
// تهيئة وضع السمة  
function initTheme() {  
 const savedTheme = localStorage.getItem('theme') || 'light';  
 document.documentElement.setAttribute('data-theme', savedTheme);  
}  
  
// تبديل وضع السمة  
function toggleTheme() {  
 const currentTheme = document.documentElement.getAttribute('data-theme');  
 const newTheme = currentTheme === 'light' ? 'dark' : 'light';  
   
 document.documentElement.setAttribute('data-theme', newTheme);  
 localStorage.setItem('theme', newTheme);  
}  
  
// إظهار معلومات المطور  
function showCreator() {  
 document.getElementById('modalOverlay').classList.add('show');  
 document.getElementById('creatorModal').classList.add('show');  
  
 // إغلاق النافذة عند الضغط على الخلفية  
 document.getElementById('modalOverlay').onclick = function() {  
 document.getElementById('modalOverlay').classList.remove('show');  
 document.getElementById('creatorModal').classList.remove('show');  
 };  
}  
  
// تهيئة الوضع عند تحميل الصفحة  
document.addEventListener('DOMContentLoaded', initTheme);  
  
// استرجاع سجل الألعاب من الذاكرة المحلية  
document.addEventListener('DOMContentLoaded', () => {  
 updateGameHistory();  
});  
  
let gameState = {  
 player1: {  
 name: localStorage.getItem('player1Name') || '',  
 total: 0,  
 scores: [],  
 highScore: parseInt(localStorage.getItem('player1HighScore')) || 0,  
 totalGames: parseInt(localStorage.getItem('player1TotalGames')) || 0,  
 wins: parseInt(localStorage.getItem('player1Wins')) || 0  
 },  
 player2: {  
 name: localStorage.getItem('player2Name') || '',  
 total: 0,  
 scores: [],  
 highScore: parseInt(localStorage.getItem('player2HighScore')) || 0,  
 totalGames: parseInt(localStorage.getItem('player2TotalGames')) || 0,  
 wins: parseInt(localStorage.getItem('player2Wins')) || 0  
 },  
 rounds: [],  
 maxRounds: parseInt(localStorage.getItem('maxRounds')) || 7  
};  
  
// تحديث أسماء اللاعبين عند الكتابة  
document.getElementById('player1Name').addEventListener('input', function(e) {  
 gameState.player1.name = e.target.value;  
});  
  
document.getElementById('player2Name').addEventListener('input', function(e) {  
 gameState.player2.name = e.target.value;  
});  
  
// إضافة نقاط للاعب  
function addSpecialMove(playerNum, moveType) {  
 const move = specialMoves[moveType];  
  
 // تحديث نقاط اللاعبين مباشرة  
 if (playerNum === 1) {  
 // اللاعب 1 حقق الحركة  
 gameState.player1.total += move.bonus; // يحصل على المكافأة (سالبة)  
 gameState.player2.total += move.penalty; // يحصل على الغرامة (موجبة)  
 } else {  
 // اللاعب 2 حقق الحركة  
 gameState.player2.total += move.bonus; // يحصل على المكافأة (سالبة)  
 gameState.player1.total += move.penalty; // يحصل على الغرامة (موجبة)  
 }  
  
 // إضافة الحركة إلى سجل الجولات  
 gameState.rounds.push({  
 player1Score: playerNum === 1 ? move.bonus : move.penalty,  
 player2Score: playerNum === 2 ? move.bonus : move.penalty,  
 specialMove: {  
 type: moveType,  
 winner: playerNum  
 }  
 });  
  
 // تحديث المجموع  
 document.getElementById(`player1Total`).textContent = gameState.player1.total;  
 document.getElementById(`player2Total`).textContent = gameState.player2.total;  
  
 // تحديث عداد الجولات مع الحركات الخاصة  
 updateRoundCounter(true);  
  
 // تحديث سجل الجولات  
 updateRoundHistory(true);  
  
 // التحقق من الفائز  
 checkWinner();  
}  
  
  
  
function updateMaxRounds() {  
 const newMaxRounds = parseInt(document.getElementById('maxRoundsInput').value);  
 if (newMaxRounds >= 1 && newMaxRounds <= 20) {  
 gameState.maxRounds = newMaxRounds;  
 document.getElementById('maxRounds').textContent = newMaxRounds;  
 updateRoundCounter();  
 }  
}  
  
function resetStats() {  
 if (confirm('هل أنت متأكد من إعادة تعيين جميع الإحصائيات؟')) {  
 gameState.player1.highScore = 0;  
 gameState.player1.totalGames = 0;  
 gameState.player1.wins = 0;  
 gameState.player2.highScore = 0;  
 gameState.player2.totalGames = 0;  
 gameState.player2.wins = 0;  
 updateStats();  
 localStorage.removeItem('gameStats');  
 }  
}  
  
function updateStats() {  
 // تحديث إحصائيات اللاعب 1  
 document.getElementById('player1HighScore').textContent = gameState.player1.highScore;  
 document.getElementById('player1Wins').textContent = gameState.player1.wins;  
 document.getElementById('player1Average').textContent =   
 gameState.player1.totalGames > 0 ?   
 Math.round(gameState.player1.highScore / gameState.player1.totalGames) : 0;  
  
 // تحديث إحصائيات اللاعب 2  
 document.getElementById('player2HighScore').textContent = gameState.player2.highScore;  
 document.getElementById('player2Wins').textContent = gameState.player2.wins;  
 document.getElementById('player2Average').textContent =   
 gameState.player2.totalGames > 0 ?   
 Math.round(gameState.player2.highScore / gameState.player2.totalGames) : 0;  
}  
  
function loadStats() {  
 const savedStats = localStorage.getItem('gameStats');  
 if (savedStats) {  
 const stats = JSON.parse(savedStats);  
 gameState.player1.highScore = stats.player1.highScore || 0;  
 gameState.player1.totalGames = stats.player1.totalGames || 0;  
 gameState.player1.wins = stats.player1.wins || 0;  
 gameState.player2.highScore = stats.player2.highScore || 0;  
 gameState.player2.totalGames = stats.player2.totalGames || 0;  
 gameState.player2.wins = stats.player2.wins || 0;  
 updateStats();  
 }  
}  
  
function saveStats() {  
 const stats = {  
 player1: {  
 highScore: gameState.player1.highScore,  
 totalGames: gameState.player1.totalGames,  
 wins: gameState.player1.wins  
 },  
 player2: {  
 highScore: gameState.player2.highScore,  
 totalGames: gameState.player2.totalGames,  
 wins: gameState.player2.wins  
 }  
 };  
 localStorage.setItem('gameStats', JSON.stringify(stats));  
}  
  
function addScore(playerNum) {  
 const scoreInput = document.getElementById(`player${playerNum}Score`);  
 const downInput = document.getElementById(`player${playerNum}Down`);  
 const minusInput = document.getElementById(`player${playerNum}Minus`);  
 let score = parseInt(scoreInput.value) || 0;  
 const down = parseInt(downInput.value) || 0;  
 const minus = parseInt(minusInput.value) || 0;  
   
 if (isNaN(score) || score < 0) {  
 alert('الرجاء إدخال رقم صحيح موجب');  
 return;  
 }  
  
 const player = gameState[`player${playerNum}`];  
 const otherPlayerNum = playerNum === 1 ? 2 : 1;  
 const otherPlayer = gameState[`player${otherPlayerNum}`];  
  
 // إذا كان هناك نزول، نطبق الحسابات على النزول فقط  
 if (down > 0) {  
 score = down;  
 }  
  
 player.scores.push(score);  
 player.total += score;  
  
 // تطبيق الخصم من المجموع مباشرة  
 if (minus > 0) {  
 player.total = Math.max(0, player.total - minus);  
 }  
  
 // تحديث المجموع المعروض  
 document.getElementById(`player${playerNum}Total`).textContent = player.total;  
   
 // الحصول على نقاط اللاعب الآخر  
 const otherScoreInput = document.getElementById(`player${otherPlayerNum}Score`);  
 const otherDownInput = document.getElementById(`player${otherPlayerNum}Down`);  
 const otherMinusInput = document.getElementById(`player${otherPlayerNum}Minus`);  
 let otherScore = parseInt(otherScoreInput.value) || 0;  
 const otherDown = parseInt(otherDownInput.value) || 0;  
 const otherMinus = parseInt(otherMinusInput.value) || 0;  
  
 if (isNaN(otherScore) || otherScore < 0) {  
 alert('الرجاء إدخال رقم صحيح موجب للاعب الآخر');  
 return;  
 }  
  
 // إذا كان هناك نزول للاعب الآخر، نطبق الحسابات على النزول فقط  
 if (otherDown > 0) {  
 otherScore = otherDown;  
 }  
  
 // تحديث نقاط اللاعب الآخر  
 otherPlayer.scores.push(otherScore);  
 otherPlayer.total += otherScore;  
  
 // تطبيق الخصم من المجموع مباشرة  
 if (otherMinus > 0) {  
 otherPlayer.total = Math.max(0, otherPlayer.total - otherMinus);  
 }  
 document.getElementById(`player${otherPlayerNum}Total`).textContent = otherPlayer.total;  
  
 // إضافة الجولة إلى السجل  
 gameState.rounds.push({  
 player1Score: playerNum === 1 ? score : otherScore,  
 player2Score: playerNum === 2 ? score : otherScore,  
 player1Down: playerNum === 1 ? down : otherDown,  
 player2Down: playerNum === 2 ? down : otherDown,  
 player1Minus: playerNum === 1 ? minus : otherMinus,  
 player2Minus: playerNum === 2 ? minus : otherMinus  
 });  
  
 // تفريغ حقول الإدخال  
 scoreInput.value = '';  
 downInput.value = '';  
 minusInput.value = '';  
 otherScoreInput.value = '';  
 otherDownInput.value = '';  
 otherMinusInput.value = '';  
  
 // تحديث سجل الجولات فقط  
 updateRoundHistory(false);  
  
 // التحقق من الفائز  
 checkWinner();  
}  
  
// تحديث سجل الجولات  
function updateRoundCounter(isSpecialMove = false) {  
 let currentRound;  
 // حساب عدد الجولات العادية فقط (بدون النزول والخصم)  
 const normalRounds = gameState.rounds.filter(round =>   
 !(round.player1Down || round.player2Down || round.player1Minus || round.player2Minus)  
 ).length;  
  
 if (isSpecialMove) {  
 currentRound = normalRounds + 1;  
 } else {  
 currentRound = normalRounds;  
 }  
   
 document.getElementById('currentRound').textContent = currentRound;  
 document.getElementById('maxRounds').textContent = gameState.maxRounds;  
 document.getElementById('maxRoundsInput').value = gameState.maxRounds;  
  
 // تحديث المخزون المحلي  
 localStorage.setItem('maxRounds', gameState.maxRounds);  
}  
  
// تحديث عدد الجولات  
function updateMaxRounds() {  
 const newMaxRounds = parseInt(document.getElementById('maxRoundsInput').value);  
 if (newMaxRounds >= 1 && newMaxRounds <= 20) {  
 gameState.maxRounds = newMaxRounds;  
 localStorage.setItem('maxRounds', newMaxRounds);  
 updateRoundCounter();  
 playSound('button');  
 }  
}  
  
// تحديث نقاط الفوز  
  
  
function updateRoundHistory(isSpecialMove = false) {  
 const history = document.getElementById('roundHistory');  
 history.innerHTML = '';  
  
 // تحديث عداد الجولات فقط للحركات الخاصة  
 if (isSpecialMove) {  
 document.getElementById('currentRound').textContent = gameState.rounds.length + 1;  
 }  
  
 gameState.rounds.forEach((round, index) => {  
 const roundDiv = document.createElement('div');  
 roundDiv.className = 'round-entry';  
   
 const player1Name = gameState.player1.name || 'الفريق 1';  
 const player2Name = gameState.player2.name || 'الفريق 2';  
  
 if (round.specialMove) {  
 const move = specialMoves[round.specialMove.type];  
 const winner = round.specialMove.winner === 1 ? player1Name : player2Name;  
 const loser = round.specialMove.winner === 1 ? player2Name : player1Name;  
 roundDiv.innerHTML = `  
 <span>الجولة ${index + 1}:</span>  
 <span>${winner} فاز بـ ${move.name}${round.specialMove.divided ? ' (مقسوم على 2)' : ''}</span>  
 <span>(للفريق: ${round.player1Score} نقطة، للخصم: ${round.player2Score} نقطة)</span>  
 `;  
 } else {  
 let player1Info = `${round.player1Score} نقطة`;  
 let player2Info = `${round.player2Score} نقطة`;  
  
 if (round.player1Down > 0) {  
 player1Info = `⬇️ ${round.player1Down} نزول`;  
 }  
 if (round.player2Down > 0) {  
 player2Info = `⬇️ ${round.player2Down} نزول`;  
 }  
 if (round.player1Minus > 0) {  
 player1Info = `${player1Info} \n➖ ${round.player1Minus} خصم من المجموع`;  
 }  
 if (round.player2Minus > 0) {  
 player2Info = `${player2Info} \n➖ ${round.player2Minus} خصم من المجموع`;  
 }  
  
 roundDiv.innerHTML = `  
 <span>الجولة ${index + 1}:</span>  
 <span>${player1Name}: ${player1Info}</span>  
 <span>${player2Name}: ${player2Info}</span>  
 `;  
 }  
   
 history.appendChild(roundDiv);  
 });  
  
 // تمرير إلى آخر جولة  
 history.scrollTop = history.scrollHeight;  
}  
  
  
// التحقق من الفائز  
function checkWinner() {  
 const winnerDisplay = document.getElementById('winnerDisplay');  
 const player1Name = gameState.player1.name || 'الفريق 1';  
 const player2Name = gameState.player2.name || 'الفريق 2';  
  
 winnerDisplay.textContent = ''; // مسح الرسالة السابقة  
  
 // التحقق من انتهاء الجولات  
 if (gameState.rounds.length >= TOTAL\_ROUNDS) {  
 let winner, winnerScore;  
   
 if (gameState.player1.total < gameState.player2.total) {  
 winner = player1Name;  
 winnerScore = gameState.player1.total;  
 } else {  
 winner = player2Name;  
 winnerScore = gameState.player2.total;  
 }  
  
 winnerDisplay.textContent = `🎉 مبروك! ${winner} هو الفائز بمجموع ${winnerScore} نقطة! 🎉`;  
 disableGameButtons();  
 sounds.win.play();  
  
 // حفظ اللعبة في السجل  
 saveGameToHistory(winner === player1Name ? gameState.player1 : gameState.player2);  
 } else {  
 // عرض عدد الجولات المتبقية  
 const remainingRounds = TOTAL\_ROUNDS - gameState.rounds.length;  
 winnerDisplay.textContent = `باقي ${remainingRounds} جولات`;  
 }  
  
}  
  
// بدء لعبة جديدة  
function disableGameButtons() {  
 // تعطيل أزرار الحركات الخاصة  
 document.querySelectorAll('.special-btn').forEach(btn => btn.disabled = true);  
   
 // تعطيل حقول إدخال النقاط  
 document.querySelectorAll('input[type="number"]').forEach(input => input.disabled = true);  
   
 // تعطيل أزرار إضافة النقاط  
 document.querySelectorAll('button[onclick^="addScore"]').forEach(btn => btn.disabled = true);  
}  
  
function enableGameButtons() {  
 // تفعيل أزرار الحركات الخاصة  
 document.querySelectorAll('.special-btn').forEach(btn => btn.disabled = false);  
   
 // تفعيل حقول إدخال النقاط  
 document.querySelectorAll('input[type="number"]').forEach(input => input.disabled = false);  
   
 // تفعيل أزرار إضافة النقاط  
 document.querySelectorAll('button[onclick^="addScore"]').forEach(btn => btn.disabled = false);  
}  
  
function resetDowns() {  
 // إعادة تعيين حقول النزول إلى الصفر  
 document.getElementById('player1Down1').value = '0';  
 document.getElementById('player1Down2').value = '0';  
 document.getElementById('player2Down1').value = '0';  
 document.getElementById('player2Down2').value = '0';  
}  
  
function saveGameToHistory(winner) {  
 const gameHistory = JSON.parse(localStorage.getItem('gameHistory') || '[]');  
 const gameData = {  
 date: new Date().toLocaleString('ar'),  
 winner: winner.name || `اللاعب ${winner === gameState.player1 ? '1' : '2'}`,  
 player1: {  
 name: gameState.player1.name || 'اللاعب 1',  
 score: gameState.player1.total  
 },  
 player2: {  
 name: gameState.player2.name || 'اللاعب 2',  
 score: gameState.player2.total  
 }  
 };  
 gameHistory.unshift(gameData);  
 if (gameHistory.length > 10) gameHistory.pop(); // الاحتفاظ بآخر 10 ألعاب فقط  
 localStorage.setItem('gameHistory', JSON.stringify(gameHistory));  
 updateGameHistory();  
}  
  
function updateGameHistory() {  
 const gameHistory = JSON.parse(localStorage.getItem('gameHistory') || '[]');  
 const historyContainer = document.getElementById('gameHistory');  
 historyContainer.innerHTML = gameHistory.map(game => `  
 <div class="history-item">  
 <span>${game.date}</span>  
 <span>الفائز: ${game.winner}</span>  
 <span>${game.player1.name}: ${game.player1.score} | ${game.player2.name}: ${game.player2.score}</span>  
 </div>  
 `).join('');  
}  
  
function newGame() {  
 if (!confirm('هل أنت متأكد من بدء لعبة جديدة؟')) {  
 return;  
 }  
  
 const oldState = gameState;  
 gameState = {  
 player1: {  
 name: gameState.player1.name,  
 total: 0,  
 scores: [],  
 highScore: oldState.player1.highScore || 0,  
 totalGames: oldState.player1.totalGames || 0,  
 wins: oldState.player1.wins || 0  
 },  
 player2: {  
 name: gameState.player2.name,  
 total: 0,  
 scores: [],  
 highScore: oldState.player2.highScore || 0,  
 totalGames: oldState.player2.totalGames || 0,  
 wins: oldState.player2.wins || 0  
 },  
 rounds: [],  
 maxRounds: oldState.maxRounds || 7  
 };  
  
 // إعادة تعيين العرض  
 document.getElementById('player1Total').textContent = '0';  
 document.getElementById('player2Total').textContent = '0';  
 document.getElementById('roundHistory').innerHTML = '';  
 document.getElementById('winnerDisplay').textContent = '';  
 document.getElementById('currentRound').textContent = '0';  
 enableGameButtons();  
 playSound('newGame');  
}  
  
// التراجع عن آخر جولة  
function undoLastRound() {  
 playSound('undo');  
 if (gameState.rounds.length === gameState.maxRounds) {  
 // إعادة تفعيل الأزرار عند التراجع عن الجولة الأخيرة  
 enableGameButtons();  
 }  
  
  
 if (gameState.rounds.length === 0) {  
 alert('لا توجد جولات للتراجع عنها');  
 return;  
 }  
  
 const lastRound = gameState.rounds.pop();  
   
 // تحديث مجاميع اللاعبين  
 gameState.player1.total -= lastRound.player1Score;  
 gameState.player2.total -= lastRound.player2Score;  
  
 // تحديث العرض  
 document.getElementById('player1Total').textContent = gameState.player1.total;  
 document.getElementById('player2Total').textContent = gameState.player2.total;  
 document.getElementById('winnerDisplay').textContent = '';  
   
 updateRoundHistory();  
}

--------------------------------------------------

# File: style.css

:root {  
 --bg-color: #f0f2f5;  
 --container-bg: white;  
 --text-color: #1a1a1a;  
 --border-color: #ddd;  
 --shadow-color: rgba(0, 0, 0, 0.1);  
 --primary-color: #3498db;  
 --primary-hover: #2980b9;  
 --success-color: #2ecc71;  
 --success-hover: #27ae60;  
 --danger-color: #e74c3c;  
 --danger-hover: #c0392b;  
}  
  
[data-theme="dark"] {  
 --bg-color: #1a1a1a;  
 --container-bg: #2c2c2c;  
 --text-color: #f0f2f5;  
 --border-color: #404040;  
 --shadow-color: rgba(0, 0, 0, 0.3);  
 --primary-color: #2980b9;  
 --primary-hover: #3498db;  
 --success-color: #27ae60;  
 --success-hover: #2ecc71;  
 --danger-color: #c0392b;  
 --danger-hover: #e74c3c;  
}  
  
\* {  
 margin: 0;  
 padding: 0;  
 box-sizing: border-box;  
 font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;  
}  
  
.controls-top {  
 position: fixed;  
 top: 1rem;  
 left: 1rem;  
 z-index: 1000;  
 display: flex;  
 gap: 0.5rem;  
}  
  
.control-btn {  
 background: var(--container-bg);  
 border: 2px solid var(--border-color);  
 border-radius: 50%;  
 width: 40px;  
 height: 40px;  
 cursor: pointer;  
 display: flex;  
 align-items: center;  
 justify-content: center;  
 font-size: 1.2rem;  
 padding: 0;  
 transition: all 0.3s ease;  
}  
  
.control-btn:hover {  
 transform: scale(1.1);  
 box-shadow: 0 0 10px var(--shadow-color);  
}  
  
.control-btn:active {  
 transform: scale(0.9);  
}  
  
.theme-btn {  
 background: var(--container-bg);  
 border: 2px solid var(--border-color);  
 border-radius: 50%;  
 width: 40px;  
 height: 40px;  
 cursor: pointer;  
 display: flex;  
 align-items: center;  
 justify-content: center;  
 font-size: 1.2rem;  
 padding: 0;  
 transition: transform 0.3s ease;  
}  
  
.theme-btn:hover {  
 transform: rotate(360deg);  
}  
  
[data-theme="dark"] .light-icon,  
[data-theme="light"] .dark-icon {  
 display: none;  
}  
  
body {  
 background-color: var(--bg-color);  
 color: var(--text-color);  
 line-height: 1.6;  
}  
  
.container {  
 max-width: 1200px;  
 margin: 1rem auto;  
}  
  
.info-btn {  
 opacity: 0.7;  
}  
  
.info-btn:hover {  
 opacity: 1;  
 transform: scale(1.1);  
}  
  
.creator-modal {  
 display: none;  
 position: fixed;  
 top: 50%;  
 left: 50%;  
 transform: translate(-50%, -50%);  
 background-color: var(--container-bg);  
 border: 2px solid var(--border-color);  
 border-radius: 10px;  
 padding: 20px;  
 z-index: 1000;  
 text-align: center;  
 box-shadow: 0 4px 6px var(--shadow-color);  
 min-width: 300px;  
}  
  
.creator-modal.show {  
 display: block;  
 animation: fadeIn 0.3s;  
}  
  
.modal-overlay {  
 display: none;  
 position: fixed;  
 top: 0;  
 left: 0;  
 right: 0;  
 bottom: 0;  
 background-color: rgba(0, 0, 0, 0.5);  
 z-index: 999;  
}  
  
.modal-overlay.show {  
 display: block;  
}  
  
@keyframes fadeIn {  
 from { opacity: 0; transform: translate(-50%, -60%); }  
 to { opacity: 1; transform: translate(-50%, -50%); }  
}  
  
.container {  
 padding: 1rem;  
 background-color: var(--container-bg);  
 border-radius: 12px;  
 box-shadow: 0 2px 10px var(--shadow-color);  
}  
  
h1 {  
 text-align: center;  
 color: #2c3e50;  
 margin-bottom: 2rem;  
}  
  
.players {  
 display: grid;  
 grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));  
 gap: 2rem;  
 margin-bottom: 2rem;  
}  
  
.player {  
 padding: 1.5rem;  
 background-color: #f8f9fa;  
 border-radius: 8px;  
 text-align: center;  
}  
  
.special-moves {  
 display: grid;  
 grid-template-columns: repeat(3, 1fr);  
 gap: 0.5rem;  
 margin-bottom: 1rem;  
}  
  
  
  
.special-btn {  
 padding: 0.5rem;  
 border: none;  
 border-radius: 4px;  
 color: white;  
 font-weight: bold;  
 cursor: pointer;  
 transition: all 0.3s ease;  
 position: relative;  
 overflow: hidden;  
}  
  
.special-btn::before {  
 content: '';  
 position: absolute;  
 top: 50%;  
 left: 50%;  
 width: 0;  
 height: 0;  
 background-color: rgba(255, 255, 255, 0.2);  
 border-radius: 50%;  
 transform: translate(-50%, -50%);  
 transition: width 0.6s ease-out, height 0.6s ease-out;  
}  
  
.special-btn:active::before {  
 width: 200%;  
 height: 200%;  
}  
  
.special-btn:hover {  
 transform: translateY(-2px);  
 box-shadow: 0 4px 8px var(--shadow-color);  
}  
  
.special-btn:active {  
 transform: translateY(0);  
 box-shadow: 0 2px 4px var(--shadow-color);  
}  
  
.special-btn:hover {  
 transform: translateY(-2px);  
 box-shadow: 0 2px 5px rgba(0,0,0,0.2);  
}  
  
.khales {  
 background-color: #4CAF50;  
}  
  
.hand {  
 background-color: #2196F3;  
}  
  
.ajkari {  
 background-color: #9C27B0;  
}  
  
.amkari {  
 background-color: #FF5722;  
}  
  
.amkari-walon {  
 background-color: #E91E63;  
 grid-column: span 3;  
}  
  
.player h2 {  
 color: #34495e;  
 margin-bottom: 1rem;  
}  
  
.player-name {  
 width: 100%;  
 padding: 0.5rem;  
 margin-bottom: 1rem;  
 border: 1px solid #ddd;  
 border-radius: 4px;  
 text-align: center;  
}  
  
  
  
.score-input {  
 display: flex;  
 flex-direction: column;  
 gap: 0.5rem;  
 margin-bottom: 1rem;  
}  
  
.score-row {  
 display: flex;  
 gap: 0.5rem;  
}  
  
.down-input {  
 width: 80px;  
 padding: 0.5rem;  
 border: 1px solid var(--border-color);  
 border-radius: 4px;  
 text-align: center;  
 background-color: #fff3f3;  
 color: #e74c3c;  
 font-weight: bold;  
}  
  
.down-input:focus {  
 outline: none;  
 border-color: #e74c3c;  
 box-shadow: 0 0 0 2px rgba(231, 76, 60, 0.2);  
}  
  
.minus-input {  
 width: 80px;  
 padding: 0.5rem;  
 border: 1px solid var(--border-color);  
 border-radius: 4px;  
 text-align: center;  
 background-color: #fff0f8;  
 color: #e91e63;  
 font-weight: bold;  
}  
  
.minus-input:focus {  
 outline: none;  
 border-color: #e91e63;  
 box-shadow: 0 0 0 2px rgba(233, 30, 99, 0.2);  
}  
  
.score-input input {  
 flex: 1;  
 padding: 0.5rem;  
 border: 1px solid #ddd;  
 border-radius: 4px;  
 text-align: center;  
}  
  
button {  
 padding: 0.5rem 1rem;  
 background-color: var(--primary-color);  
 color: white;  
 border: none;  
 border-radius: 4px;  
 cursor: pointer;  
 transition: background-color 0.3s;  
}  
  
button:hover {  
 background-color: var(--primary-hover);  
}  
  
.total-score {  
 font-size: 1.2rem;  
 font-weight: bold;  
 color: #2c3e50;  
}  
  
.rounds {  
 margin-top: 2rem;  
 padding: 1rem;  
 background-color: #f8f9fa;  
 border-radius: 8px;  
}  
  
.rounds h3 {  
 color: #34495e;  
 margin-bottom: 1rem;  
 text-align: center;  
}  
  
#roundHistory {  
 max-height: 200px;  
 overflow-y: auto;  
 padding: 1rem;  
}  
  
.round-entry {  
 display: flex;  
 justify-content: space-between;  
 padding: 0.5rem;  
 border-bottom: 1px solid var(--border-color);  
 white-space: pre-line;  
 line-height: 1.5;  
}  
  
.round-entry span {  
 flex: 1;  
 text-align: center;  
}  
  
.game-info {  
 text-align: center;  
 margin: 2rem 0;  
 padding: 1rem;  
 background-color: var(--container-bg);  
 border-radius: 8px;  
 box-shadow: 0 2px 4px var(--shadow-color);  
}  
  
.game-settings {  
 display: flex;  
 justify-content: center;  
 gap: 2rem;  
 margin-bottom: 1rem;  
}  
  
.setting-item {  
 display: flex;  
 align-items: center;  
 gap: 0.5rem;  
}  
  
.setting-item input {  
 width: 60px;  
 padding: 0.25rem;  
 border: 1px solid var(--border-color);  
 border-radius: 4px;  
 text-align: center;  
 font-size: 1rem;  
 background-color: var(--bg-color);  
 color: var(--text-color);  
}  
  
.setting-item input:focus {  
 outline: none;  
 border-color: var(--primary-color);  
 box-shadow: 0 0 0 2px rgba(52, 152, 219, 0.2);  
}  
  
.rounds-info {  
 margin: 0;  
 font-weight: bold;  
 color: var(--text-color);  
}  
  
/\* تنسيق الأزرار المعطلة \*/  
button:disabled {  
 opacity: 0.5;  
 cursor: not-allowed;  
}  
  
input:disabled {  
 opacity: 0.5;  
 cursor: not-allowed;  
 background-color: var(--border-color);  
}  
  
.round-counter {  
 display: flex;  
 align-items: center;  
 justify-content: center;  
 gap: 0.5rem;  
 margin-bottom: 1rem;  
 font-size: 1.2rem;  
 font-weight: bold;  
 color: var(--text-color);  
 background-color: var(--container-bg);  
 padding: 0.5rem 1rem;  
 border-radius: 8px;  
 border: 2px solid var(--border-color);  
}  
  
#currentRound, #maxRounds {  
 color: var(--primary-color);  
 font-size: 1.4rem;  
}  
  
.win-points-container input {  
 width: 100px;  
 padding: 0.5rem;  
 border: 1px solid var(--border-color);  
 border-radius: 4px;  
 text-align: center;  
 background-color: var(--container-bg);  
 color: var(--text-color);  
}  
  
.game-history {  
 margin-top: 2rem;  
 padding: 1rem;  
 background-color: var(--container-bg);  
 border-radius: 8px;  
 border: 1px solid var(--border-color);  
}  
  
.history-container {  
 max-height: 200px;  
 overflow-y: auto;  
 padding: 1rem;  
 background-color: var(--bg-color);  
 border-radius: 4px;  
}  
  
.history-item {  
 padding: 0.5rem;  
 border-bottom: 1px solid var(--border-color);  
 display: flex;  
 justify-content: space-between;  
 align-items: center;  
}  
  
.history-item:last-child {  
 border-bottom: none;  
}  
  
.controls {  
 display: flex;  
 justify-content: center;  
 gap: 1rem;  
 margin-top: 2rem;  
 flex-wrap: wrap;  
}  
  
.controls button {  
 min-width: 150px;  
 padding: 0.75rem 1.5rem;  
 font-size: 1.1rem;  
 border: none;  
 border-radius: 8px;  
 cursor: pointer;  
 transition: all 0.3s ease;  
 display: flex;  
 align-items: center;  
 justify-content: center;  
 gap: 0.5rem;  
}  
  
.new-game {  
 background: var(--success-color);  
 color: white;  
}  
  
.new-game:hover {  
 background: var(--success-hover);  
 transform: translateY(-2px);  
}  
  
.undo {  
 background: var(--primary-color);  
 color: white;  
}  
  
.undo:hover {  
 background: var(--primary-hover);  
 transform: translateY(-2px);  
}  
  
.reset-stats {  
 background: var(--danger-color);  
 color: white;  
}  
  
.reset-stats:hover {  
 background: var(--danger-hover);  
 transform: translateY(-2px);  
}  
  
.new-game {  
 background-color: var(--success-color);  
}  
  
.new-game:hover {  
 background-color: var(--success-hover);  
}  
  
.undo {  
 background-color: var(--danger-color);  
}  
  
.undo:hover {  
 background-color: var(--danger-hover);  
}  
  
.winner {  
 text-align: center;  
 margin-top: 2rem;  
 padding: 1rem;  
 font-size: 1.5rem;  
 font-weight: bold;  
 color: #27ae60;  
}  
  
@media (max-width: 600px) {  
 .players {  
 grid-template-columns: 1fr;  
 }  
  
 .container {  
 margin: 1rem;  
 padding: 1rem;  
 }  
}

--------------------------------------------------