// حالة اللعبة

Const gameState = {

"player": {

"name": "الرئيس الجولاني",

"age": 42,

"reputation": 75,

"army\_strength": 100,

"nuclear\_weapons": 2,

"economy": 60,

"public\_support": 80,

"alliances": ["روسيا", "الصين", "كوريا الشمالية"],

"decisions": [],

},

"world": {

"is\_peaceful": false,

"tension": 30,

"countries": {

"سوريا": {"status": "محررة", "support": 95, "strength": 85},

"لبنان": {"status": "قيد التحرير", "support": 60, "strength": 40},

"إيران": {"status": "صديقة", "support": 70, "strength": 90},

"تركيا": {"status": "حيادية", "support": 50, "strength": 75},

"إسرائيل": {"status": "عدوة", "support": 0, "strength": 95},

},

},

"turn": 1,

"max\_turns": 10,

"history": [],

};

// عناصر DOM

Const gameTextEl = document.getElementById('game-text');

Const choicesContainerEl = document.getElementById('choices-container');

Const inputContainerEl = document.getElementById('input-container');

Const userInputEl = document.getElementById('user-input');

Const submitBtnEl = document.getElementById('submit-btn');

Const statusDisplayEl = document.getElementById('status-display');

Const saveBtnEl = document.getElementById('save-btn');

Const loadBtnEl = document.getElementById('load-btn');

// عرض النص في اللعبة

Function displayText(text, colorClass = '') {

Const messageEl = document.createElement('div');

messageEl.classList.add('message');

if (colorClass) messageEl.classList.add(colorClass);

messageEl.innerHTML = text;

gameTextEl.appendChild(messageEl);

gameTextEl.scrollTop = gameTextEl.scrollHeight;

}

// إنشاء شريط الحالة

Function getStatusBar(value) {

Const maxBars = 10;

Const filledBars = Math.min(maxBars, Math.max(0, Math.floor(value / 10)));

Const emptyBars = maxBars – filledBars;

Let color;

If (value < 30) {

Color = '#f95959'; // أحمر

} else if (value < 60) {

Color = '#f8961e'; // أصفر

} else {

Color = '#43aa8b'; // أخضر

}

Return `<span style="color: ${color}">${'■'.repeat(filledBars)}${'□'.repeat(emptyBars)}</span>`;

}

// عرض حالة اللعبة

Function displayStatus() {

Const player = gameState.player;

Const world = gameState.world;

Let html = `

<div class="status-container">

<div class="status-item">

<div>👑 ${player.name} (${player.age} سنة)</div>

<div>🌟 السمعة: ${getStatusBar(player.reputation)} ${player.reputation}</div>

<div>💪 قوة الجيش: ${getStatusBar(player.army\_strength)} ${player.army\_strength}</div>

<div>☢️ الأسلحة النووية: ${player.nuclear\_weapons}</div>

</div>

<div class="status-item">

<div>💰 الاقتصاد: ${getStatusBar(player.economy)} ${player.economy}</div>

<div>❤️ دعم الشعب: ${getStatusBar(player.public\_support)} ${player.public\_support}</div>

<div>🌍 التوتر العالمي: ${getStatusBar(world.tension)} ${world.tension}</div>

<div>🔄 الدور: ${gameState.turn}/${gameState.max\_turns}</div>

</div>

</div>

`;

If (player.alliances.length > 0) {

Html += `<div>🤝 التحالفات: ${player.alliances.join(', ')}</div>`;

}

statusDisplayEl.innerHTML = html;

}

// إضافة حدث للتاريخ

Function addToHistory(event) {

gameState.history.push({

"turn": gameState.turn,

"event": event,

"timestamp": new Date().toLocaleString()

});

}

// التحقق من شروط النهاية

Function checkGameEnd() {

Const player = gameState.player;

If (player.reputation <= 0) {

Return {"ended": true, "reason": "فقدت كل سمعتك وسقط نظامك!"};

}

If (player.army\_strength <= 0) {

Return {"ended": true, "reason": "تم تدمير جيشك بالكامل!"};

}

If (player.public\_support <= 0) {

Return {"ended": true, "reason": "فقدت دعم الشعب وحدثت ثورة ضدك!"};

}

If (gameState.turn >= gameState.max\_turns) {

Return {"ended": true, "reason": "انتهت المهلة الزمنية المحددة للعبة."};

}

Return {"ended": false};

}

// عرض الخيارات

Function displayChoices(choices) {

choicesContainerEl.innerHTML = '';

choices.forEach((choice, index) => {

const button = document.createElement('button');

button.classList.add('choice-btn');

button.innerHTML = `

<strong>${index + 1}. ${choice.text}</strong>

${choice.effects ? `<div class="text-info">التأثيرات: ${choice.effects.join(', ')}</div>` : ''}

`;

Button.addEventListener('click', () => handleChoice(index + 1));

choicesContainerEl.appendChild(button);

});

}

// معالجة الاختيار

Function handleChoice(choice) {

currentDecisionHandler(choice);

}

// بدء اللعبة

Function startGame() {

gameTextEl.innerHTML = '';

displayText('<h2>🎮 مرحباً بك في لعبة تحرير الأمة - النسخة الإلكترونية</h2>', 'text-highlight');

displayText('<hr>', 'text-info');

displayText('الوضع الحالي: أنت الرئيس الجولاني، تقود أمة تنهض من رماد الحرب.', 'text-info');

displayText('هدفك: قيادة الأمة نحو التحرير الكامل مع الحفاظ على الاستقرار.', 'text-info');

displayStatus();

setTimeout(firstDecision, 1500);

}

// القرار الأول

Function firstDecision() {

displayText('<hr>', 'text-info');

displayText('<strong>🚨 موقف حرج: بعد تحرير سوريا، تواجه خياراً صعباً في لبنان:</strong>', 'text-danger');

const choices = [

{

Text: "مهاجمة جنوب لبنان مباشرة",

Effects: ["-15 قوة جيش", "+10 دعم لبناني", "+10 توتر عالمي"]

},

{

Text: "إرسال وفد دبلوماسي للتفاوض",

Effects: ["+10 سمعة", "+5 دعم شعبي", "-5 توتر عالمي"]

},

{

Text: "طلب دعم روسي لمنطقة عسكرية",

Effects: ["تحالف جديد", "-5 سمعة", "+15 توتر عالمي"]

}

];

displayChoices(choices);

currentDecisionHandler = handleFirstDecision;

}

Function handleFirstDecision(choice) {

addToHistory(`القرار الأول: ${choice}`);

gameState.turn += 1;

const player = gameState.player;

const world = gameState.world;

if (choice == 1) {

player.decisions.push("هجوم مباشر على جنوب لبنان");

player.army\_strength -= 15;

world.countries["لبنان"].support += 10;

world.tension += 10;

displayText("<strong>⚔️ أطلقت عملية عسكرية قوية!</strong>", "text-danger");

displayText("الجيش تقدم ولكن خسر بعض القوات. الشعب اللبناني منقسم.", "text-warning");

} else if (choice == 2) {

Player.decisions.push("مفاوضات مع المقاومة");

Player.reputation += 10;

Player.public\_support += 5;

World.tension -= 5;

displayText("<strong>🕊️ أرسلت وفداً دبلوماسياً.</strong>", "text-success");

displayText("بعض الفصائل تجاوبت والبعض رفض. تحسنت صورتك دولياً.", "text-info");

} else if (choice == 3) {

Player.decisions.push("طلب دعم روسي لمنطقة عسكرية");

Player.alliances.push("ميليشيا روسية خاصة");

Player.reputation -= 5;

World.tension += 15;

displayText("<strong>🤝 حصلت على الدعم الروسي!</strong>", "text-warning");

displayText("روسيا أبدت استعداداً ولكن بشروط اقتصادية صعبة.", "text-info");

}

displayStatus();

setTimeout(secondDecision, 2000);

}

// القرار الثاني

Function secondDecision() {

Const gameEnd = checkGameEnd();

If (gameEnd.ended) {

endGame(gameEnd.reason);

return;

}

displayText('<hr>', 'text-info');

displayText('<strong>🔥 تصاعد الأوضاع: حان الوقت لاتخاذ قرار استراتيجي:</strong>', 'text-danger');

const choices = [

{

Text: "تصنيع الرمح الأسود (صاروخ متطور)",

Effects: ["+1 سلاح نووي", "-20 اقتصاد", "+20 توتر عالمي"]

},

{

Text: "إعلان حالة الطوارئ والتعبئة العامة",

Effects: ["+10 قوة جيش", "+5 سمعة", "-10 اقتصاد"]

},

{

Text: "حملة إعلامية لكسب الدعم الإسلامي",

Effects: ["+15 سمعة", "+10 دعم شعبي", "+5 اقتصاد"]

}

];

displayChoices(choices);

currentDecisionHandler = handleSecondDecision;

}

Function handleSecondDecision(choice) {

addToHistory(`القرار الثاني: ${choice}`);

gameState.turn += 1;

const player = gameState.player;

const world = gameState.world;

if (choice == 1) {

player.nuclear\_weapons += 1;

player.economy -= 20;

world.tension += 20;

displayText("<strong>☢️ تم تطوير الرمح الأسود!</strong>", "text-danger");

displayText("سلاح جديد يرعب الأعداء لكن استنزف الاقتصاد.", "text-warning");

} else if (choice == 2) {

Player.reputation += 5;

Player.army\_strength += 10;

Player.economy -= 10;

displayText("<strong>🚨 تم إعلان حالة الطوارئ!</strong>", "text-warning");

displayText("الشعب استجاب للنداء لكن الضغوط الدولية تزايدت.", "text-info");

} else if (choice == 3) {

Player.reputation += 15;

Player.public\_support += 10;

Player.economy += 5;

displayText("<strong>📢 انطلقت الحملة الإعلامية!</strong>", "text-success");

displayText("تعاطف الملايين معك وتدفقت التبرعات.", "text-info");

}

displayStatus();

setTimeout(thirdDecision, 2000);

}

// القرار الثالث

Function thirdDecision() {

Const gameEnd = checkGameEnd();

If (gameEnd.ended) {

endGame(gameEnd.reason);

return;

}

displayText('<hr>', 'text-info');

displayText('<strong>⚠️ أزمة داخلية: انقلاب عسكري محتمل من أحد قادة الجيش!</strong>', 'text-danger');

displayText("كيف تتصرف؟", "text-info");

const choices = [

{

Text: "إقالته فوراً ومحاكمته علناً",

Effects: ["-10 قوة جيش", "+5 سمعة", "استقرار مؤقت"]

},

{

Text: "محاولة استمالته بالتفاوض",

Effects: ["+5 دعم شعبي", "-5 سمعة", "حل مؤقت"]

},

{

Text: "إهمال الأمر والتركيز على الخارج",

Effects: ["-15 دعم شعبي", "مخاطر مستقبلية"]

}

];

displayChoices(choices);

currentDecisionHandler = handleThirdDecision;

}

Function handleThirdDecision(choice) {

addToHistory(`القرار الثالث: ${choice}`);

gameState.turn += 1;

const player = gameState.player;

if (choice == 1) {

player.army\_strength -= 10;

player.reputation += 5;

displayText("<strong>⚖️ تمت الإقالة والمحاكمة!</strong>", "text-warning");

displayText("البعض رأى الأمر عدالة والبعض اعتبره قسوة. الوضع مستقر مؤقتاً.", "text-info");

} else if (choice == 2) {

Player.public\_support += 5;

Player.reputation -= 5;

displayText("<strong>🤝 تمت تسوية الأزمة بالحوار.</strong>", "text-success");

displayText("حصل القائد على منصب أعلى. استقر الوضع مؤقتاً.", "text-info");

} else if (choice == 3) {

Player.public\_support -= 15;

displayText("<strong>😰 تجاهلت التحذيرات...</strong>", "text-danger");

displayText("انفجرت الأزمة بعد أسابيع. خسرت السيطرة جزئياً على الداخل.", "text-warning");

}

displayStatus();

setTimeout(finalDecision, 2000);

}

// القرار النهائي

Function finalDecision() {

Const gameEnd = checkGameEnd();

If (gameEnd.ended) {

endGame(gameEnd.reason);

return;

}

displayText('<hr>', 'text-info');

displayText('<strong>🏁 القرار الأخير: وصلت لنقطة تحول تاريخية!</strong>', 'text-highlight');

const choices = [

{

Text: "إعلان الحرب الشاملة على إسرائيل",

Effects: ["مخاطر عالية", "مجد أو دمار"]

},

{

Text: "توقيع معاهدة سلام مشروطة",

Effects: ["استقرار مؤقت", "انتقادات داخلية"]

},

{

Text: "الاستمرار في المقاومة المحدودة",

Effects: ["استراتيجية طويلة المدى"]

}

];

displayChoices(choices);

currentDecisionHandler = handleFinalDecision;

}

Function handleFinalDecision(choice) {

addToHistory(`القرار النهائي: ${choice}`);

gameState.turn += 1;

const endingMessages = {

1: "أعلنت الحرب الشاملة. التاريخ سيحكم على قرارك...",

2: "اخترت السلام المشروط. حققت الاستقرار لكن على حساب بعض الأهداف.",

3: "اخترت المقاومة المستمرة. طريق طويل لكنه مدروس."

};

endGame(endingMessages[choice]);

}

// نهاية اللعبة

Function endGame(reason) {

gameTextEl.innerHTML = '';

choicesContainerEl.innerHTML = '';

displayText('<h2>🎯 نهاية اللعبة</h2>', 'text-highlight');

displayText(`<strong>📄 السبب:</strong> ${reason}`, 'text-warning');

displayText('<hr>', 'text-info');

displayStatus();

displayText('<strong>📚 ملخص القرارات:</strong>', 'text-highlight');

gameState.player.decisions.forEach((decision, Index) => {

displayText(`${index + 1}. ${decision}`, 'text-info');

});

Const player = gameState.player;

Const totalScore = (player.reputation + player.army\_strength +

Player.public\_support + player.economy –

gameState.world.tension);

displayText('<hr>', 'text-info');

displayText('<strong>🏆 تقييم الأداء:</strong>', 'text-highlight');

displayText(`النتيجة الإجمالية: ${totalScore}`, 'text-info');

if (totalScore >= 300) {

displayText("🌟 أداء ممتاز! قائد عظيم", "text-success");

} else if (totalScore >= 200) {

displayText("👍 أداء جيد! قائد ناجح", "text-warning");

} else {

displayText("🤔 يمكن تحسين الأداء في المرة القادمة", "text-danger");

}

displayText('<hr>', 'text-info');

displayText('💾 سيتم إضافة المزيد من المحتوى في النسخة القادمة...', 'text-info');

const restartBtn = document.createElement('button');

restartBtn.classList.add('choice-btn');

restartBtn.textContent = 'إعادة اللعبة';

restartBtn.addEventListener('click', startGame);

choicesContainerEl.appendChild(restartBtn);

}

// حفظ اللعبة

Function saveGame() {

localStorage.setItem('liberationGameSave', JSON.stringify(gameState));

displayText('تم حفظ اللعبة بنجاح!', 'text-success');

}

// تحميل اللعبة

Function loadGame() {

Const savedGame = localStorage.getItem('liberationGameSave');

If (savedGame) {

Object.assign(gameState, JSON.parse(savedGame));

displayText('تم تحميل اللعبة بنجاح!', 'text-success');

displayStatus();

// يمكنك هنا استئناف اللعبة من حيث توقفت

} else {

displayText('لا يوجد حفظ سابق للعبة!', 'text-danger');

}

}

// أحداث الأزرار

saveBtnEl.addEventListener('click', saveGame);

loadBtnEl.addEventListener('click', loadGame);

// متغير للتعامل مع القرار الحالي

Let currentDecisionHandler = null;

// بدء اللعبة عند تحميل الصفحة

Window.onload = startGame;