--- game.js ---

import { GameScene } from '/gamescene.js';  
import { languageData } from './language.js';  
  
let game;  
let speedDown = 50;  
let basketSpeed = 200;  
  
function updateTexts() {  
 const lang = localStorage.getItem("language") || "vi";  
 const texts = languageData[lang];  
  
 document.getElementById("gameTitle").innerText = texts.title;  
 document.getElementById("startGameBtn").innerText = texts.startGame;  
 document.getElementById("settingsBtn").innerText = texts.settings;  
 document.getElementById("fallSpeedLabel").childNodes[0].textContent = texts.fallSpeed + " ";  
 document.getElementById("basketSpeedLabel").childNodes[0].textContent = texts.basketSpeed + " ";  
 document.getElementById("controlTypeTitle").innerText = texts.controlType;  
 document.getElementById("keyboardLabel").innerText = texts.keyboard;  
 document.getElementById("handtrackingLabel").innerText = texts.handtracking;  
 document.getElementById("languageSelectTitle").innerText = texts.languageSelect;  
 document.getElementById("saveSettingsBtn").innerText = texts.save;  
 document.getElementById("backToMenuBtn").innerText = texts.back;  
 document.getElementById("returnToMenuBtn").innerText = texts.returnMenu;  
}  
  
document.getElementById("settingsBtn").addEventListener("click", () => {  
 document.getElementById("mainMenu").style.display = "none";  
 document.getElementById("settings").style.display = "block";  
});  
  
document.getElementById("saveSettingsBtn").addEventListener("click", () => {  
 speedDown = parseInt(document.getElementById("speedSlider").value);  
 basketSpeed = parseInt(document.getElementById("basketSpeedSlider").value);  
  
 const selectedControl = document.querySelector('input[name="controlType"]:checked').value;  
 const selectedLanguage = document.getElementById("languageSelect").value;  
  
 const selectedGameMode = document.querySelector('input[name="gameMode"]:checked').value;  
  
 localStorage.setItem("gameMode", selectedGameMode);  
 localStorage.setItem("controlType", selectedControl);  
 localStorage.setItem("fallSpeed", speedDown);  
 localStorage.setItem("basketSpeed", basketSpeed);  
 localStorage.setItem("language", selectedLanguage);  
  
 updateTexts();  
 alert("Cài đặt đã được lưu!");  
});  
  
document.getElementById("backToMenuBtn").addEventListener("click", () => {  
 document.getElementById("settings").style.display = "none";  
 document.getElementById("mainMenu").style.display = "block";  
});  
  
document.getElementById("startGameBtn").addEventListener("click", () => {  
 document.getElementById("mainMenu").style.display = "none";  
 document.getElementById("gameScreen").style.display = "block";  
  
 if (game) game.destroy(true);  
 game = new Phaser.Game({  
 type: Phaser.WEBGL,  
 width: 500,  
 height: 500,  
 canvas: document.querySelector("#gameCanvas"),  
 physics: { default: "arcade", arcade: { gravity: { y: 0 } } },  
 scene: [GameScene],  
 });  
});  
  
document.getElementById("returnToMenuBtn").addEventListener("click", () => {  
 document.getElementById("gameScreen").style.display = "none";  
 document.getElementById("mainMenu").style.display = "block";  
 if (game) game.destroy(true);  
});  
  
// Khởi động  
updateTexts();

--- gamescene.js ---

import { languageData } from './language.js';  
  
export class GameScene extends Phaser.Scene {  
 constructor() {  
 super("scene-game");  
 this.player;  
 this.target;  
 this.cursor;  
 this.score = 0;  
 this.timeElapsed = 0;  
 this.scoreText;  
 this.timeText;  
 this.controlText;  
 this.mode = localStorage.getItem("gameMode") || "free";  
 this.remainingTime = 60; // 60s nếu cần  
 this.timerText; // sẽ tạo 1 text hiện thời gian đếm ngược  
 this.gameEnded = false; // kiểm soát trạng thái kết thúc  
 }  
  
 preload() {  
 this.load.image("bg", "public/assets/caytao.png");  
 this.load.image("basket", "public/assets/basket.png");  
 this.load.image("apple", "public/assets/apple.png");  
 }  
  
 create() {  
 if (this.mode === "60s") {  
 this.timerText = this.add.text(300, 10, "60", { fontSize: "22px", fill: "#ff3333", stroke: "#000", strokeThickness: 2 });  
 }  
  
 const lang = localStorage.getItem("language") || "vi";  
 const texts = languageData[lang];  
  
 this.add.image(0, 0, "bg").setOrigin(0, 0);  
 this.player = this.physics.add.image(0, 500, "basket").setCollideWorldBounds(true);  
 this.target = this.physics.add.image(Math.random() \* 500, 0, "apple").setVelocityY(50);  
 this.cursor = this.input.keyboard.createCursorKeys();  
  
 this.scoreText = this.add.text(10, 10, `${texts.score}: 0`, { fontSize: "22px", fill: "#ffcc00", stroke: "#000", strokeThickness: 2 });  
 this.timeText = this.add.text(10, 40, `${texts.time}: 0s`, { fontSize: "20px", fill: "#66ff99", stroke: "#000", strokeThickness: 2 });  
 this.controlText = this.add.text(10, 70, "", { fontSize: "18px", fill: "#ffffff", strokeThickness: 1 });  
 this.modeText = this.add.text(10, 100, "", { fontSize: "18px", fill: "#ffffff", strokeThickness: 1 });  
  
 this.physics.add.overlap(this.player, this.target, this.catchApple, null, this);  
 }  
  
 update() {  
 const lang = localStorage.getItem("language") || "vi";  
 const texts = languageData[lang];  
   
 const savedControlType = localStorage.getItem("controlType") || "keyboard";  
 this.controlText.setText(`${texts.control}: ${savedControlType === "keyboard" ? texts.keyboard : texts.handtracking}`);  
 this.modeText.setText(`Chế độ: ${this.mode === "60s" ? "60 Giây" : "Tự Do"}`);  
   
 if (this.mode === "60s" && !this.gameEnded) {  
 this.remainingTime -= this.game.loop.delta / 1000;  
 if (this.remainingTime <= 0) {  
 this.remainingTime = 0;  
 this.endGame();  
 }  
 this.timeText.setText(`${texts.time}: ${Math.floor(this.remainingTime)}s`);  
 } else {  
 this.timeElapsed += this.game.loop.delta / 1000;  
 this.timeText.setText(`${texts.time}: ${Math.floor(this.timeElapsed)}s`);  
 }  
   
 if (this.cursor.left.isDown) {  
 this.player.setVelocityX(-200);  
 } else if (this.cursor.right.isDown) {  
 this.player.setVelocityX(200);  
 } else {  
 this.player.setVelocityX(0);  
 }  
   
 if (this.target.y >= 500) {  
 this.target.setY(0);  
 this.target.setX(Math.random() \* 500);  
 }  
 }   
  
 catchApple() {  
 const lang = localStorage.getItem("language") || "vi";  
 const texts = languageData[lang];  
  
 this.score += 10;  
 this.scoreText.setText(`${texts.score}: ${this.score}`);  
 this.target.setY(0);  
 this.target.setX(Math.random() \* 500);  
 }  
  
 endGame() {  
 this.gameEnded = true;  
 this.physics.pause();  
 const lang = localStorage.getItem("language") || "vi";  
 const texts = languageData[lang];  
  
 this.add.text(150, 220, `Kết thúc!\n${texts.score}: ${this.score}`, {  
 fontSize: "28px",  
 fill: "#ffffff",  
 align: "center",  
 stroke: "#000",  
 strokeThickness: 3  
 }).setOrigin(0.5);  
 }  
}

--- index.html ---

<!DOCTYPE html>  
<html lang="vi">  
  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Game Hứng Táo</title>  
 <script src="https://cdn.jsdelivr.net/npm/phaser@3.60.0/dist/phaser.js"></script>  
 <link rel="stylesheet" href="style.css">  
</head>  
  
<body>  
 <h1 id="gameTitle">Game Hứng Táo</h1>  
  
 <div id="mainMenu" style="display: block;">  
 <button id="startGameBtn">Chơi ngay</button>  
 <button id="settingsBtn">Cài đặt</button>  
 </div>  
  
 <div id="settings">  
 <label id="fallSpeedLabel">Tốc độ rơi: <input id="speedSlider" type="range" min="10" max="100" step="10"  
 value="50"></label>  
 <span id="speedValue">50</span>  
 <br><br>  
 <label id="basketSpeedLabel">Tốc độ di chuyển giỏ: <input id="basketSpeedSlider" type="range" min="50" max="500"  
 step="50" value="200"></label>  
 <span id="basketSpeedValue">200</span>  
  
 <h3 id="controlTypeTitle">Chọn kiểu điều khiển:</h3>  
 <label><input type="radio" name="controlType" value="keyboard" checked> <span id="keyboardLabel">Bàn  
 phím</span></label>  
 <label><input type="radio" name="controlType" value="handtracking"> <span id="handtrackingLabel">Hand  
 Tracking</span></label>  
  
 <h3 id="languageSelectTitle">Chọn ngôn ngữ:</h3>  
 <select id="languageSelect">  
 <option value="vi" selected>Tiếng Việt</option>  
 <option value="en">English</option>  
 <option value="zh">中文</option>  
 </select>  
  
 <h3>Chọn chế độ chơi:</h3>  
 <label><input type="radio" name="gameMode" value="free" checked> Chế độ Tự Do</label>  
 <label><input type="radio" name="gameMode" value="60s"> Chế độ 60 Giây</label>  
  
 <button id="saveSettingsBtn">Lưu</button>  
 <button id="backToMenuBtn">Quay lại</button>  
 </div>  
  
 <div id="gameScreen">  
 <canvas id="gameCanvas"></canvas>  
 <button id="returnToMenuBtn">Về Menu</button>  
 </div>  
  
 <script type="module" src="game.js"></script>  
</body>  
  
</html>

--- language.js ---

export const languageData = {  
 vi: {  
 title: "Game Hứng Táo",  
 startGame: "Chơi ngay",  
 settings: "Cài đặt",  
 fallSpeed: "Tốc độ rơi:",  
 basketSpeed: "Tốc độ di chuyển giỏ:",  
 controlType: "Chọn kiểu điều khiển:",  
 keyboard: "Bàn phím",  
 handtracking: "Hand Tracking",  
 languageSelect: "Chọn ngôn ngữ:",  
 save: "Lưu",  
 back: "Quay lại",  
 returnMenu: "Về Menu",  
 score: "Điểm",  
 time: "Thời gian",  
 control: "Điều khiển"  
 },  
 en: {  
 title: "Apple Catch Game",  
 startGame: "Play Now",  
 settings: "Settings",  
 fallSpeed: "Fall Speed:",  
 basketSpeed: "Basket Move Speed:",  
 controlType: "Select Control Type:",  
 keyboard: "Keyboard",  
 handtracking: "Hand Tracking",  
 languageSelect: "Select Language:",  
 save: "Save",  
 back: "Back",  
 returnMenu: "Return to Menu",  
 score: "Score",  
 time: "Time",  
 control: "Control"  
 },  
 zh: {  
 title: "接苹果游戏",  
 startGame: "开始游戏",  
 settings: "设置",  
 fallSpeed: "掉落速度:",  
 basketSpeed: "篮子移动速度:",  
 controlType: "选择控制方式:",  
 keyboard: "键盘",  
 handtracking: "手部追踪",  
 languageSelect: "选择语言:",  
 save: "保存",  
 back: "返回",  
 returnMenu: "返回菜单",  
 score: "得分",  
 time: "时间",  
 control: "控制方式"  
 }  
 };

--- style.css ---

@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;700&display=swap');  
  
#settings {  
 display: flex;  
 flex-direction: column;  
 align-items: center;  
 gap: 15px; /\* Tạo khoảng cách giữa các phần \*/  
 padding: 20px;  
 background-color: rgba(30, 30, 30, 0.9);  
 border-radius: 10px;  
 box-shadow: 0px 4px 15px rgba(0,0,0,0.5);  
}  
  
.setting-group {  
 display: flex;  
 flex-direction: column;  
 width: 80%;  
 padding: 10px;  
 border-radius: 8px;  
 background: rgba(50, 50, 50, 0.8);  
}  
  
.setting-group h3 {  
 color: #ffcc00;  
 font-size: 20px;  
 margin-bottom: 10px;  
 text-align: center;  
}  
  
.setting-group label {  
 display: flex;  
 align-items: center;  
 justify-content: space-between;  
}  
  
  
\* {  
 margin: 0;  
 padding: 0;  
 box-sizing: border-box;  
 font-family: 'Poppins', sans-serif;  
}  
  
body {  
 display: flex;  
 flex-direction: column;  
 align-items: center;  
 justify-content: flex-start;  
 background-color: #121212;  
 color: #ffffff;  
 min-height: 100vh;  
 padding-top: 40px;  
 text-align: center;  
}  
  
h1 {  
 margin-bottom: 20px;  
 font-size: 36px;  
 font-weight: 700;  
 color: #ffcc00;  
 text-shadow: 2px 2px #000;  
}  
  
canvas {  
 border: 3px solid #ffffff;  
 width: 500px;  
 height: 500px;  
 border-radius: 15px;  
 box-shadow: 0px 0px 20px rgba(255, 255, 255, 0.3);  
}  
  
#mainMenu, #settings, #gameScreen {  
 padding: 20px;  
 background-color: rgba(30, 30, 30, 0.9);  
 border-radius: 10px;  
 box-shadow: 0px 4px 15px rgba(0,0,0,0.5);  
 margin-top: 20px;  
 display: none;  
}  
  
button {  
 width: 220px;  
 padding: 14px;  
 font-size: 18px;  
 font-weight: bold;  
 border: none;  
 border-radius: 30px;  
 cursor: pointer;  
 background: linear-gradient(135deg, #3498db, #2980b9);  
 color: white;  
 transition: all 0.3s ease;  
 margin: 10px;  
 box-shadow: 0px 4px 10px rgba(0,0,0,0.3);  
}  
  
button:hover {  
 background: linear-gradient(135deg, #5dade2, #3498db);  
 transform: scale(1.05);  
}  
select {  
 width: 220px;  
 padding: 12px;  
 font-size: 18px;  
 font-weight: bold;  
 border: 2px solid #3498db;  
 border-radius: 10px;  
 cursor: pointer;  
 background: linear-gradient(135deg, #2c3e50, #3498db);  
 color: white;  
 transition: all 0.3s ease;  
 appearance: none; /\* Ẩn kiểu mặc định của trình duyệt \*/  
 text-align: center;  
}  
  
select:hover {  
 background: linear-gradient(135deg, #5dade2, #3498db);  
 transform: scale(1.05);  
}  
  
select:focus {  
 outline: none;  
 box-shadow: 0px 0px 10px rgba(52, 152, 219, 0.8);  
}  
  
option {  
 background-color: #2c3e50;  
 color: white;  
 padding: 10px;  
}