

```

import { GameContainer } from "@components/game/GameContainer";

function App() {
  return (
    <div className="min-h-screen w-full bg-background overflow-hidden">
      <GameContainer />
    </div>
  );
}

export default App;
import { createRoot } from "react-dom/client";
import App from "./App";
import "./index.css";

createRoot(document.getElementById("root")!).render(<App />);

@import "tailwindcss";
@import "tw-animate-css";

@custom-variant dark (&:is(.dark *));

@theme inline {
  --font-cairo: 'Cairo', sans-serif;
  --font-changa: 'Changa', sans-serif;
  --font-pixel: 'Press Start 2P', cursive;

  --color-background: hsl(35 30% 90%);
  --color-foreground: hsl(20 10% 15%);

  --color-card: hsl(35 30% 95%);
  --color-card-foreground: hsl(20 10% 15%);

  --color-primary: hsl(25 70% 40%);
  --color-primary-foreground: hsl(35 30% 95%);

  --color-secondary: hsl(180 40% 30%);
  --color-secondary-foreground: hsl(35 30% 95%);

  --color-accent: hsl(45 80% 60%);
  --color-accent-foreground: hsl(20 10% 15%);

  --color-muted: hsl(30 20% 80%);
  --color-muted-foreground: hsl(20 10% 40%);

  --color-border: hsl(30 20% 70%);
  --color-input: hsl(30 20% 85%);

  --radius-sm: 0px;
  --radius-md: 4px;
  --radius-lg: 8px;
}

@layer base {
  * {
    @apply border-border;
  }
  body {
    @apply font-cairo antialiased bg-background text-foreground;
    background-color: #f5f0e6;
    image-rendering: pixelated; /* Crucial for pixel art feel */
  }
}

/* Custom Scrollbar for Webkit */
::-webkit-scrollbar {
  width: 10px;
}
::-webkit-scrollbar-track {
  background: #d6cfc2;
}
::-webkit-scrollbar-thumb {
  background: #8c7b65;
  border: 2px solid #d6cfc2;
}

```

```

::-webkit-scrollbar-thumb:hover {
  background: #6b5d4b;
}

/* Pixel Art UI Utilities */
.pixel-border {
  box-shadow:
    -4px 0 0 0 #000,
    4px 0 0 0 #000,
    0 -4px 0 0 #000,
    0 4px 0 0 #000;
  margin: 4px;
}

.pixel-corners {
  clip-path: polygon(
    0px 4px, 4px 4px, 4px 0px,
    calc(100% - 4px) 0px, calc(100% - 4px) 4px, 100% 4px,
    100% calc(100% - 4px), calc(100% - 4px) calc(100% - 4px), calc(100% - 4px) 100%,
    4px 100%, 4px calc(100% - 4px), 0px calc(100% - 4px)
  );
}

.pixel-text {
  font-family: 'Changa', sans-serif;
  font-weight: 700;
}

/* For character sprites to blend */
.multiply-blend {
  mix-blend-mode: multiply;
}

export type CharacterType = 'saladin' | 'richard' | 'omar' | 'sophronius' | 'soldier' | 'civilian'
| 'narrator' | 'doctor' | 'messenger';

export interface DialogueLine {
  speaker: CharacterType;
  text: string;
  isNarrator?: boolean;
  audio?: {
    file?: string;
    start: number;
    end: number;
  };
};

export interface MiniGame {
  type: 'text_input' | 'rewrite' | 'compare' | 'order' | 'reflection' | 'drag_match' | 'true_false'
| 'select_char';
  title: string;
  description: string;
  items?: string[]; // For ordering or choices
  correctAnswer?: string | string[]; // For validation
  options?: { label: string; value: string }[];
  targets?: { id: string; image: string; label: string }[]; // For drag match
}

export interface SceneData {
  id: number;
  title: string;
  background: string;
  characters: {
    type: CharacterType;
    position: 'left' | 'center' | 'right' | 'mid-left' | 'mid-right';
    animation?: 'idle' | 'gesture' | 'fear' | 'writing' | 'listening';
  }[];
  dialogue: DialogueLine[];
  miniGame: MiniGame;
}

export interface BalloonQuestion {
  id: number;
  question: string;
  options: string[];
}

```

```

correct: string;
}

export const balloonQuestions: BalloonQuestion[] = [
  { id: 1, question: "ما هو أساس التعامل مع الأسرى؟", options: ["التجاهل", "الرحمة والكرامة", "القسوة"], correct: "الرحمة والكرامة" },
  { id: 2, question: "كيف رد صلاح الدين على مرض ريتشارد؟", options: ["أرسل له طبيباً", "انتظر موته", "هاجمه"], correct: "أرسل له طبيباً" },
  { id: 3, question: "ماذا ضمنت العهدة العمرية؟", options: ["الخروج من المدينة", "دفع المال فقط", "حرية العبادة"], correct: "حرية العبادة" },
  { id: 4, question: "...التسامح هو علامة على", options: ["الضعف", "الشجاعة والقوة", "الاستسلام"], correct: "الشجاعة والقوة" },
  { id: 5, question: "كيف دخل عمر بن الخطاب القدس؟", options: ["بجيش جرار", "بتواضع وتسامح", "بالقوة"], correct: "بتواضع وتسامح" },
];

export const introQuestions: MiniGame[] = [
  {
    type: 'select_char',
    title: "من هو القائد؟",
    description: "اختر الشخصية التي ترمز للتسامح في قصتنا (اضغط على الصورة)",
    targets: [
      { id: 'saladin', image: '/assets/characters/saladin.png', label: 'صلاح الدين' },
      { id: 'soldier', image: '/assets/characters/soldier.png', label: 'جندي عادي' }
    ],
    correctAnswer: 'saladin'
  }
];

export const scenes: SceneData[] = [
  // --- PART 1: Saladin & Richard ---
  {
    id: 1,
    title: "حماية المدنيين",
    background: "/assets/backgrounds/jerusalem-walls.png",
    characters: [
      { type: 'saladin', position: 'right', animation: 'gesture' },
      { type: 'soldier', position: 'left', animation: 'idle' }
    ],
    dialogue: [
      { speaker: 'narrator', text: "اقتربت الحرب من القدس، لكن قرارًا واحدًا غير مسار الأحداث.", isNarrator: true,
        audio: { file: "/assets/audio/narrator.mp3", start: 0, end: 5 } },
      {
        speaker: 'saladin',
        text: "قبل أن تُشهر السيوف، تذكرُوا: نحن نقاتل من يحمل السلاح، لا من يحمل الخوف",
        audio: { file: "/assets/audio/salahuddin.mp3", start: 0, end: 7 }
      },
      {
        speaker: 'soldier',
        text: "ولكنهم من مدينة العدو يا مولاي",
        audio: { file: "/assets/audio/soldier_messenger.mp3", start: 0, end: 3 }
      },
      {
        speaker: 'saladin',
        text: "العدو هو الظلم، لا الإنسان. من آذى بريئاً، فقد خسر شرفه",
        audio: { file: "/assets/audio/salahuddin.mp3", start: 8, end: 14 }
      },
      { speaker: 'narrator', text: "هكذا ولد التسامح في قلب الحرب", isNarrator: true, audio: { file: "/assets/audio/narrator.mp3", start: 6, end: 8 } }
    ],
    miniGame: {
      type: 'true_false',
      title: "حكم أخلاقي",
      description: "هل يجوز إيذاء المدنيين إذا كانوا من مدينة العدو؟",

```

```

    options: [
      { label: "نعم، هم أعداء", value: "false" },
      { label: "لا، العداء لا يبرر الظلم", value: "true" }
    ],
    correctAnswer: "true"
  }
},
{
  id: 2,
  title: "مرض ريتشارد الأول",
  background: "/assets/backgrounds/camp-night.png",
  characters: [
    { type: 'richard', position: 'left', animation: 'idle' },
    { type: 'doctor', position: 'right', animation: 'idle' },
    { type: 'messenger', position: 'mid-right', animation: 'gesture' }
  ],
  dialogue: [
    { speaker: 'narrator', text: "اشتد المرض على ريتشارد الأول حتى خاف جنوده عليه", isNarrator: true, audio: {
file: "/assets/audio/narrator.mp3", start: 9, end: 13 } },
    { speaker: 'doctor', text: "لا نملك دواءً كافياً", audio: { file: "/assets/audio/doctor.mp3", start:
0, end: 2 } } },
    {
      speaker: 'messenger',
      text: "صلاح الدين يرسل طبيبه الخاص، ومعه فاكهة وتلج",
      audio: { file: "/assets/audio/soldier_messenger.mp3", start: 3, end: 8 }
    },
    {
      speaker: 'richard',
      text: "عدوي... يختار إنقاذي؟ هذا فعل إنسان قبل أن يكون قائدًا",
      audio: { file: "/assets/audio/richard.mp3", start: 0, end: 6 }
    },
    { speaker: 'narrator', text: "حين مرض الجسد شفيت القلوب", isNarrator: true, audio: { file:
"/assets/audio/narrator.mp3", start: 13, end: 16 } }
  ],
  miniGame: {
    type: 'rewrite',
    title: "أعد كتابة الشعور",
    description: "أعد صياغة عبارة ريتشارد الأخيرة بأسلوبك، معبّرًا عن الاحترام والامتنان"
  }
},
{
  id: 3,
  title: "الاحترام بين الأعداء",
  background: "/assets/backgrounds/open-land.png",
  characters: [
    { type: 'saladin', position: 'right', animation: 'idle' },
    { type: 'messenger', position: 'left', animation: 'gesture' }
  ],
  dialogue: [
    {
      speaker: 'messenger',
      text: "الملك ريتشارد يقول: لو لم نكن أعداء، لكننا إخوة في الشرف",
      audio: { file: "/assets/audio/soldier_messenger.mp3", start: 8, end: 14 }
    },
    {
      speaker: 'saladin',
      text: "الحرب تنتهي، أما الكرامة فتبقى",
      audio: { file: "/assets/audio/salahuddin.mp3", start: 15, end: 19 }
    },
    { speaker: 'narrator', text: "حتى الخصومة يمكن أن تُدار بإنسانية", isNarrator: true, audio: { file:
"/assets/audio/narrator.mp3", start: 16, end: 19 } }
  ],
  miniGame: {
    type: 'drag_match',
    title: "لمن هذا التصرف؟",
    description: "اسحب 'احترام العدو' إلى الشخصية المناسبة",
    targets: [

```

```

    { id: 'saladin', image: '/assets/characters/saladin.png', label: 'صلاح الدين'},
    { id: 'soldier', image: '/assets/characters/soldier.png', label: 'جندي متعصب' }
  ],
  items: ['احترام العدو'],
  correctAnswer: 'saladin'
}
},

// --- PART 2: Omar Bin Al-Khattab ---
{
  id: 4,
  title: "دخول عمر بن الخطاب القدس",
  background: "/assets/backgrounds/jerusalem-gates.png",
  characters: [
    { type: 'omar', position: 'right', animation: 'gesture' },
    { type: 'sophronius', position: 'left', animation: 'idle' }
  ],
  dialogue: [
    { speaker: 'narrator', text: "سلمت مفاتيح القدس بلا قتال", isNarrator: true, audio: { file:
"/assets/audio/narrator.mp3", start: 20, end: 22 } },
    {
      speaker: 'sophronius',
      text: "أهذا هو الحاكم؟ بلا تاج ولا جيش؟",
      audio: { file: "/assets/audio/zephronius.mp3", start: 0, end: 5 }
    },
    {
      speaker: 'omar',
      text: "نحن قوم نقيم العدل، لا المظاهر",
      audio: { file: "/assets/audio/omar.mp3", start: 0, end: 4 }
    },
    { speaker: 'narrator', text: "العدل فتح ما لم تفتحهِ السيوف", isNarrator: true, audio: { file:
"/assets/audio/narrator.mp3", start: 23, end: 25 } }
  ],
  miniGame: {
    type: 'order',
    title: "رتب القيم",
    description: "رتب الكلمات التالية حسب الأهمية في شخصية عمر",
    items: ["العدل", "التواضع", "المظاهر", "القوة"]
  }
},
{
  id: 5,
  title: "الصلاة والكنيسة",
  background: "/assets/backgrounds/church-interior.png",
  characters: [
    { type: 'omar', position: 'right', animation: 'idle' },
    { type: 'sophronius', position: 'left', animation: 'gesture' }
  ],
  dialogue: [
    {
      speaker: 'sophronius',
      text: "صَلِّ هنا، المكان آمن",
      audio: { file: "/assets/audio/zephronius.mp3", start: 8, end: 11 }
    },
    {
      speaker: 'omar',
      text: "لا. أخاف أن يأتي من بعدي من لا يفهم، فيظلم هذا المكان",
      audio: { file: "/assets/audio/omar.mp3", start: 4, end: 10 }
    },
    { speaker: 'narrator', text: "التسامح يفكر في المستقبل قبل الحاضر", isNarrator: true, audio: { file:
"/assets/audio/narrator.mp3", start: 25, end: 29 } }
  ],
  miniGame: {
    type: 'reflection',
    title: "ماذا لو؟",
    description: "لماذا رفض عمر الصلاة في الكنيسة؟ (اكتب إجابتك)"
  }
}

```

```

    },
    {
      id: 6,
      title: "العهد العمرية",
      background: "/assets/backgrounds/quiet-room.png",
      characters: [
        { type: 'omar', position: 'right', animation: 'writing' },
        { type: 'sophronius', position: 'left', animation: 'listening' }
      ],
      dialogue: [
        {
          speaker: 'omar',
          text: "هذا ما أعطى عبد الله عمر أمير المؤمنين أهل إيلياء من الأمان: لأنفسهم، وأموالهم، وكنائسهم... لا يُكْرَهُونَ على دينهم",
          audio: { file: "/assets/audio/omar.mp3", start: 10, end: 23 }
        },
        {
          speaker: 'sophronius',
          text: "هذا عهد سيبقى شاهداً على العدل",
          audio: { file: "/assets/audio/zephronius.mp3", start: 14, end: 21 }
        },
        { speaker: 'narrator', text: "هنا أصبح التسامح قانوناً", isNarrator: true, audio: { file:
"/assets/audio/narrator.mp3", start: 29, end: 32 } }
      ],
      miniGame: {
        type: 'text_input',
        title: "عهدك الشخصي",
        description: "اكتب عهداً صغيراً تقطعه على نفسك لتكون متسامحاً مع زملائك"
      }
    }
  ];

```

```

export const getCharacterName = (type: CharacterType) => {
  const names: Record<CharacterType, string> = {
    saladin: "صلاح الدين",
    richard: "ريتشارد الأول",
    omar: "عمر بن الخطاب",
    sophronius: "صفرونيوس",
    soldier: "جندي",
    civilian: "مدني",
    narrator: "الراوي",
    doctor: "الطبيب",
    messenger: "المرسال"
  };
  return names[type];
};

```

```

export const getCharacterImage = (type: CharacterType) => {
  const map: Record<CharacterType, string> = {
    saladin: "/assets/characters/saladin.png",
    richard: "/assets/characters/richard.png",
    omar: "/assets/characters/omar.png",
    sophronius: "/assets/characters/sophronius.png",
    soldier: "/assets/characters/soldier.png",
    civilian: "/assets/characters/civilian.png",
    narrator: "",
    doctor: "/assets/characters/civilian.png",
    messenger: "/assets/characters/soldier.png"
  };
  return map[type];
};

```

```

import { useState } from "react";
import { balloonQuestions } from "@data/scenes";
import { Button } from "@components/ui/button";
import { motion, AnimatePresence } from "framer-motion";
import { Student } from "@lib/storage";
import { saveProgress } from "@lib/storage";

```

```

interface BalloonGameProps {
  student: Student;
  onComplete: () => void;
}

export function BalloonGame({ student, onComplete }: BalloonGameProps) {
  const [activeQuestion, setActiveQuestion] = useState<number | null>(null);
  const [poppedBalloons, setPoppedBalloons] = useState<number[]>([]);
  const [score, setScore] = useState(0);

  const audioCtx = new (window.AudioContext || (window as any).webkitAudioContext)();

  const playPop = () => {
    const osc = audioCtx.createOscillator();
    const gain = audioCtx.createGain();

    osc.type = "triangle";
    osc.frequency.setValueAtTime(1000, audioCtx.currentTime);
    osc.frequency.exponentialRampToValueAtTime(200, audioCtx.currentTime + 0.15);
    gain.gain.setValueAtTime(0.2, audioCtx.currentTime);

    osc.connect(gain);
    gain.connect(audioCtx.destination);

    osc.start();
    osc.stop(audioCtx.currentTime + 0.15);
  };

  const playClick = () => {
    const osc = audioCtx.createOscillator();
    const gain = audioCtx.createGain();
    osc.type = "sine";
    osc.frequency.setValueAtTime(400, audioCtx.currentTime);
    gain.gain.setValueAtTime(0.1, audioCtx.currentTime);
    osc.connect(gain);
    gain.connect(audioCtx.destination);
    osc.start();
    osc.stop(audioCtx.currentTime + 0.1);
  };

  const handleBalloonClick = (id: number) => {
    playPop();
    setActiveQuestion(id);
  };

  const playCorrect = () => {
    const osc = audioCtx.createOscillator();
    const gain = audioCtx.createGain();

    osc.type = "sine";
    osc.frequency.setValueAtTime(600, audioCtx.currentTime);
    gain.gain.setValueAtTime(0.1, audioCtx.currentTime);

    osc.connect(gain);
    gain.connect(audioCtx.destination);

    osc.start();
    osc.stop(audioCtx.currentTime + 0.2);
  };

  const playWrong = () => {
    const osc = audioCtx.createOscillator();
    const gain = audioCtx.createGain();

    osc.type = "sawtooth";
    osc.frequency.setValueAtTime(200, audioCtx.currentTime);
    gain.gain.setValueAtTime(0.1, audioCtx.currentTime);

    osc.connect(gain);
    gain.connect(audioCtx.destination);

    osc.start();
    osc.stop(audioCtx.currentTime + 0.3);
  };
}

```

```

};

const handleAnswer = (answer: string) => {
  if (activeQuestion === null) return;

  const question = balloonQuestions.find(q => q.id === activeQuestion);
  if (!question) return;

  if (answer === question.correct) {
    setScore(prev => prev + 10);
    playCorrect();
  } else {
    playWrong();
  }

  saveProgress(student, `balloon_${activeQuestion}`, answer, answer === question.correct ? 10 :
0);
  setPoppedBalloons(prev => [...prev, activeQuestion]);
  setActiveQuestion(null);

  if (poppedBalloons.length + 1 === balloonQuestions.length) {
    setTimeout(onComplete, 2000);
  }
};

return (
  <div className="relative w-full h-full bg-sky-200 overflow-hidden font-cairo">
    {/* Background Clouds */}
    <div className="absolute inset-0 opacity-50">
      <div className="absolute top-10 left-10 w-32 h-16 bg-white rounded-full blur-xl" />
      <div className="absolute top-40 right-20 w-48 h-24 bg-white rounded-full blur-xl" />
    </div>

    <div className="absolute top-4 left-4 z-20 bg-white/80 p-2 rounded pixel-border">
      النقاط: {score}
    </div>

    <h2 className="absolute top-10 w-full text-center text-3xl font-changa text-primary z-10
drop-shadow-md">
      اجمالي بالونات التسامح
    </h2>

    {/* Balloons */}
    {balloonQuestions.map((q, idx) => {
      if (poppedBalloons.includes(q.id)) return null;

      // Random positions and delays for floating effect
      // generate random position between 10% and 90%
      // We use a deterministic way based on index to keep it stable during renders but "random"
looking
      const randomLeft = [15, 85, 30, 70, 50][idx % 5];
      const leftPos = `${randomLeft}%`;
      const delay = idx * 1.5; // More spacing between balloons
      const duration = 10 + (idx % 3) * 2; // Varying speeds
      const color = ['bg-red-400', 'bg-blue-400', 'bg-green-400', 'bg-yellow-400', 'bg-purple-
400'][idx % 5];

      return (
        <motion.div
          key={q.id}
          initial={{ y: "110vh", x: 0 }}
          animate={{
            y: "-20vh",
            x: [0, 20, -20, 0] // Swaying motion
          }}
          transition={{
            y: {
              duration: duration,
              delay: delay,
              repeat: Infinity,
              ease: "linear"
            },
            x: {

```



```

        duration: 4,
        repeat: Infinity,
        ease: "easeInOut"
      }
    }}
    onClick={() => handleBalloonClick(q.id)}
    className={`absolute ${leftPos} cursor-pointer group`}
  >
    <div className={`w-24 h-32 ${color} rounded-[50%] opacity-90 shadow-inner relative flex
items-center justify-center border-2 border-black/10 transform group-hover:scale-110 transition-
transform`}>
      <span className="text-4xl">🎈</span>
      { /* String */ }
      <div className="absolute bottom-[-40px] left-1/2 w-0.5 h-10 bg-gray-600" />
    </div>
  </motion.div>
);
}}

{ /* Question Modal */ }
<AnimatePresence>
  {activeQuestion !== null && (
    <div className="absolute inset-0 z-50 flex items-center justify-center bg-black/60
backdrop-blur-sm p-4">
      <motion.div
        initial={{ scale: 0.8, opacity: 0 }}
        animate={{ scale: 1, opacity: 1 }}
        exit={{ scale: 0.8, opacity: 0 }}
        className="bg-white p-8 rounded-lg max-w-lg w-full pixel-border"
      >
        <h3 className="text-xl font-bold mb-6 text-center">
          {balloonQuestions.find(q => q.id === activeQuestion)?.question}
        </h3>
        <div className="grid gap-4">
          {balloonQuestions.find(q => q.id === activeQuestion)?.options.map(opt => (
            <Button
              key={opt}
              onClick={() => handleAnswer(opt)}
              className="text-lg py-6 bg-slate-100 hover:bg-primary hover:text-white
text-gray-800 transition-colors"
            >
              {opt}
            </Button>
          ))}
        </div>
      </motion.div>
    </div>
  )}
</AnimatePresence>
</div>
);
}
import { motion } from "framer-motion";
import { useEffect, useState, useRef } from "react";
import { DialogueLine, getCharacterName } from "@/data/scenes";

interface DialogueBoxProps {
  line: DialogueLine;
  onComplete: () => void;
  isTyping: boolean;
  setIsTyping: (typing: boolean) => void;
}

export function DialogueBox({ line, onComplete, isTyping, setIsTyping }: DialogueBoxProps) {
  const [displayedText, setDisplayedText] = useState("");
  const fullText = line.text;
  const audioRef = useRef<HTMLAudioElement | null>(null);

  useEffect(() => {
    setDisplayedText("");
    setIsTyping(true);
    let index = 0;

    // Audio Handling

```

```

    if (line.audio) {
      const audioFile = line.audio.file || "/assets/audio/salahuddin.mp3";
      if (!audioRef.current || audioRef.current.src !== window.location.origin + audioFile) {
        audioRef.current = new Audio(audioFile);
      }
      const audio = audioRef.current;
      audio.currentTime = line.audio.start;
      audio.play().catch(e => console.log("Audio play blocked", e));

      const checkTime = () => {
        if (audio.currentTime >= (line.audio?.end || 0)) {
          audio.pause();
          audio.removeEventListener('timeupdate', checkTime);
        }
      };
      audio.addEventListener('timeupdate', checkTime);
    }

    const interval = setInterval(() => {
      index++;
      setDisplayedText(fullText.substring(0, index));
      if (index >= fullText.length) {
        clearInterval(interval);
        setIsTyping(false);
      }
    }, 30);

    return () => {
      clearInterval(interval);
      if (audioRef.current) {
        audioRef.current.pause();
      }
    };
  }, [line, fullText, setIsTyping]);

  const playClick = () => {
    const audio = new Audio("https://assets.mixkit.co/active_storage/sfx/2568/2568-preview.mp3");
    audio.volume = 0.2;
    audio.play().catch(e => console.log("Audio play blocked", e));
  };

  const handleSkip = () => {
    playClick();
    if (isTyping) {
      setDisplayedText(fullText);
      setIsTyping(false);
      if (audioRef.current) {
        audioRef.current.pause();
      }
    } else {
      onComplete();
    }
  };

  return (
    // Fixed at bottom, semi-transparent, clear text
    <div
      className="absolute bottom-4 left-4 right-4 md:left-[15%] md:right-[15%] bg-black/85 text-white p-6 rounded-lg border-2 border-white/20 cursor-pointer select-none z-50 min-h-[140px] flex flex-col justify-start items-start shadow-2xl backdrop-blur-md"
      onClick={handleSkip}
    >
      {!line.isNarrator && (
        <div className="text-accent text-xl mb-2 font-bold font-changa bg-black/50 px-3 py-1 rounded inline-block border border-accent/30">
          {getCharacterName(line.speaker)}
        </div>
      )}
      <div className={`text-xl md:text-2xl leading-relaxed font-cairo w-full ${line.isNarrator ? 'italic text-yellow-100/90 text-center mt-2' : ''}`>
        {displayedText}
        {!isTyping && <span className="animate-pulse inline-block mr-2 text-accent">▼</span>}
      </div>
    </div>
  );

```

```

    {!isTyping && (
      <div className="absolute bottom-2 left-4 text-xs text-gray-400 font-sans opacity-70">
        اضغط للمتابعة
      </div>
    )}
  </div>
);
}
import { useState } from "react";
import { scenes } from "@data/scenes";
import { Scene } from "../Scene";
import { DialogueBox } from "../DialogueBox";
import { MiniGame } from "../MiniGame";
import { Login } from "../Login";
import { TeacherDashboard } from "../TeacherDashboard";
import { IntroSection } from "../IntroSection";
import { BalloonGame } from "../BalloonGame";
import { Button } from "@components/ui/button";
import { motion, AnimatePresence } from "framer-motion";
import { Student } from "@lib/storage";
import { Route, Switch } from "wouter";

export function GameContainer() {
  const [student, setStudent] = useState<Student | null>(null);
  const [gameState, setGameState] = useState<'intro' | 'scenes' | 'balloon' | 'end'>('intro');

  const [currentSceneIndex, setCurrentSceneIndex] = useState(0);
  const [dialogueIndex, setDialogueIndex] = useState(0);
  const [showMiniGame, setShowMiniGame] = useState(false);
  const [isTyping, setIsTyping] = useState(false);

  const currentScene = scenes[currentSceneIndex];
  const currentDialogue = currentScene?.dialogue[dialogueIndex];

  // Handler for Login
  const handleLogin = (studentData: Student) => {
    setStudent(studentData);
    setGameState('intro');
  };

  // Handler for Intro Completion
  const handleIntroComplete = () => {
    setGameState('scenes');
  };

  // Handler for Dialogue Progression
  const handleNextDialogue = () => {
    if (isTyping) return;

    if (dialogueIndex < currentScene.dialogue.length - 1) {
      setDialogueIndex(prev => prev + 1);
    } else {
      setShowMiniGame(true);
    }
  };

  // Handler for MiniGame Completion
  const handleMiniGameComplete = () => {
    setShowMiniGame(false);
    if (currentSceneIndex < scenes.length - 1) {
      setCurrentSceneIndex(prev => prev + 1);
      setDialogueIndex(0);
    } else {
      // All scenes done -> Go to Balloon Game
      setGameState('balloon');
    }
  };

  const handleBalloonComplete = () => {
    setGameState('end');
  };

  const resetGame = () => {
    setCurrentSceneIndex(0);
  };

```

```

setDialogueIndex(0);
setShowMiniGame(false);
setGameState('intro');
// Keep student logged in or logout? "Refresh page" usually logs out, keeping logged in for
replay is nicer.
};

// 1. ROUTING: Dashboard vs Game
return (
  <Switch>
    <Route path="/dashboard" component={TeacherDashboard} />
    <Route path="/">
      <div className="relative w-full h-screen overflow-hidden bg-gray-900 select-none font-
cairo" dir="rtl">

        {/* Header always visible except login */}
        {student && (
          <div className="absolute top-0 left-0 right-0 z-30 p-4 flex justify-between items-
start pointer-events-none">
            <div className="text-white/90 text-sm md:text-base font-bold drop-shadow-md bg-
black/40 px-3 py-1 rounded backdrop-blur-sm border border-white/10">
              مدرسة بنات حجة الثانوية
            </div>
            <div className="text-center">
              <div className="text-accent text-2xl font-changa drop-shadow-lg text-
shadow-lg">عبر التاريخ</div>
              <div className="text-xs text-white/70 mt-1">أهلاً بك: {student.name}</div>
            </div>
          </div>
        )}

        {/* GAME FLOW */}
        {!student ? (
          <Login onLogin={handleLogin} />
        ) : (
          <>
            {/* INTRO */}
            {gameState === 'intro' && (
              <IntroSection onComplete={handleIntroComplete} student={student} />
            )}

            {/* SCENES */}
            {gameState === 'scenes' && (
              <>
                <AnimatePresence mode="wait">
                  <motion.div
                    key={currentSceneIndex}
                    initial={{ opacity: 0 }}
                    animate={{ opacity: 1 }}
                    exit={{ opacity: 0 }}
                    transition={{ duration: 0.8 }}
                    className="absolute inset-0 w-full h-full"
                  >
                    <Scene
                      scene={currentScene}
                      activeSpeaker={currentDialogue?.speaker}
                    />
                  </motion.div>
                </AnimatePresence>

                <AnimatePresence>
                  {showMiniGame ? (
                    <MiniGame
                      key="minigame"
                      gameData={currentScene.miniGame}
                      onComplete={handleMiniGameComplete}
                      student={student}
                    />
                  ) : (
                    <DialogueBox
                      key={`_${currentSceneIndex}-${dialogueIndex}`}
                      line={currentDialogue}
                      onComplete={handleNextDialogue}

```

```

        isTyping={isTyping}
        setIsTyping={setIsTyping}
      />
    )}
  </AnimatePresence>
</>
)}

{/* BALLOON GAME */}
{gameState === 'balloon' && (
  <BalloonGame student={student} onComplete={handleBalloonComplete} />
)}

{/* END SCREEN */}
{gameState === 'end' && (
  <div className="min-h-screen bg-black flex flex-col items-center justify-
center p-4 text-white text-center">
    <motion.div
      initial={{ opacity: 0 }}
      animate={{ opacity: 1 }}
      transition={{ duration: 2 }}
      className="max-w-3xl"
    >
      <h2 className="text-4xl md:text-6xl font-changa mb-8 text-accent
leading-normal">
        "التسامح ليس ضعفًا، بل شجاعة أخلاقية تصنع السلام عبر التاريخ."
      </h2>
      <p className="text-gray-400 mb-8">يا {student.name}! لقد أتممتِ يا
الرحلة بنجاح
      </p>
      <Button
        onClick={resetGame}
        variant="outline"
        className="mt-8 text-xl px-8 py-4 bg-transparent border-2
border-white text-white hover:bg-white hover:text-black font-cairo"
      >
        إعادة الرحلة
      </Button>
    </motion.div>
  </div>
)}
</>
)}
</div>
</Route>
</Switch>
);
}
import { introQuestions } from "@/data/scenes";
import { MiniGame } from "@/components/game/MiniGame";
import { motion } from "framer-motion";
import { useState } from "react";
import { Button } from "@/components/ui/button";
import { Student } from "@/lib/storage";

interface IntroSectionProps {
  onComplete: () => void;
  student: Student;
}

export function IntroSection({ onComplete, student }: IntroSectionProps) {
  const [step, setStep] = useState<'text' | 'game'>('text');

  const handleGameComplete = () => {
    onComplete();
  };

  if (step === 'text') {
    return (
      <div className="h-full flex flex-col items-center justify-center p-8 text-center bg-[#fdf6e3]
relative overflow-hidden">
        {/* Background pattern */}
        <div className="absolute inset-0 opacity-10 bg-[url('/assets/backgrounds/jerusalem-
walls.png')] bg-cover" />

```

```

<motion.div
  initial={{ opacity: 0, y: 20 }}
  animate={{ opacity: 1, y: 0 }}
  className="z-10 max-w-3xl bg-white/90 p-8 rounded-lg pixel-border shadow-xl"
>
  <h2 className="text-4xl font-changa text-primary mb-6">ما هو التسامح؟</h2>
  <p className="text-xl font-cairo leading-loose text-gray-800 mb-8">
    التسامح ليس مجرد كلمة، بل هو فعل الأقوياء. هو أن تغفو وأنت قادر على الانتقام، وأن تحترم من يختلف معك.
    عبر التاريخ، سطر قادتنا أعظم الأمثلة في الإنسانية والرحمة، حتى في أشد لحظات الحرب.
  </p>
  <Button
    onClick={() => setStep('game')}
    className="text-xl px-8 py-6 font-changa bg-primary hover:bg-primary/90 text-white
pixel-corners"
  >
    اختبر معلوماتك لنبدأ
  </Button>
</motion.div>
</div>
);
}

return (
  <div className="h-full relative bg-gray-100">
    <MiniGame
      gameData={introQuestions[0]}
      onComplete={handleGameComplete}
      student={student}
    />
    {/* Force MiniGame to look like part of the page instead of modal for intro if desired, but
reusing component is efficient */}
  </div>
);
}

import { useState } from "react";
import { Button } from "@components/ui/button";
import { Input } from "@components/ui/input";
import { Student } from "@lib/storage";
import { motion } from "framer-motion";
import { useLocation } from "wouter";

interface LoginProps {
  onLogin: (student: Student) => void;
}

const TEACHER_PASS = "2042010YABB";

export function Login({ onLogin }: LoginProps) {
  const [name, setName] = useState("");
  const [className, setClassName] = useState("");
  const [, setLocation] = useLocation();

  const handleSubmit = (e: React.FormEvent) => {
    e.preventDefault();
    if (name && className) {
      localStorage.setItem("currentStudent", name);
      // Generate a random ID since we removed the input
      const generatedId = Math.random().toString(36).substr(2, 9);
      onLogin({ name, className, id: generatedId });
    }
  };

  const handleTeacherLogin = () => {
    const pass = prompt("كلمة سر المعلم:");
    if (pass === TEACHER_PASS) {
      setLocation("/dashboard");
    } else if (pass !== null) {
      alert("كلمة السر خاطئة");
    }
  };
};

```

```

return (
  <div className="min-h-screen flex items-center justify-center bg-gray-900 p-4 relative
overflow-hidden font-cairo" dir="rtl">
    {/* Background with blur */}
    <div className="absolute inset-0 bg-[url('/assets/backgrounds/jerusalem-walls.png')] bg-cover
opacity-30 blur-sm pointer-events-none" />

    <motion.div
      initial={{ scale: 0.9, opacity: 0 }}
      animate={{ scale: 1, opacity: 1 }}
      className="bg-white/95 p-8 rounded-lg shadow-2xl max-w-md w-full pixel-border relative z-
10"
    >
      <div className="text-center mb-8">
        <h2 className="text-sm font-bold text-gray-500 mb-2">مدرسة بنات حجة الثانوية</h2>
        <h1 className="text-3xl font-changa text-primary">تسجيل الدخول</h1>
      </div>

      <form onSubmit={handleSubmit} className="space-y-6">
        <div className="space-y-2">
          <label className="text-lg font-bold text-gray-700 block">الاسم الرباعي</label>
          <Input
            value={name}
            onChange={(e) => setName(e.target.value)}
            placeholder="اكتب اسمك هنا..."
            className="font-cairo text-lg bg-white"
            required
            data-testid="input-name"
          />
        </div>

        <div className="space-y-2">
          <label className="text-lg font-bold text-gray-700 block">الصف</label>
          <Input
            value={className}
            onChange={(e) => setClassName(e.target.value)}
            placeholder="مثال: العاشر أ"
            className="font-cairo text-lg bg-white"
            required
            data-testid="input-class"
          />
        </div>

        <Button
          type="submit"
          className="w-full text-xl py-6 bg-primary hover:bg-primary/90 text-white font-changa
shadow-lg hover:shadow-xl transition-all pixel-corners mt-4"
          data-testid="button-start"
        >
          بدء الرحلة
        </Button>
      </form>
    </motion.div>

    {/* Teacher Mode Button */}
    <button
      onClick={handleTeacherLogin}
      className="absolute bottom-2 left-2 text-[10px] bg-gray-800/50 hover:bg-gray-800 text-white
px-2 py-1 rounded transition-colors z-50 font-cairo"
      data-testid="button-teacher-mode"
    >
      وضع المعلم
    </button>
  </div>
);
}
import { useState } from "react";
import { MiniGame as MiniGameType } from "@data/scenes";
import { Button } from "@components/ui/button";
import { Textarea } from "@components/ui/textarea";
import { motion } from "framer-motion";

```

```

import { Student, saveProgress } from "@lib/storage";

interface MiniGameProps {
  gameData: MiniGameType;
  onComplete: () => void;
  student: Student | null; // Pass student for saving
}

export function MiniGame({ gameData, onComplete, student }: MiniGameProps) {
  const [input, setInput] = useState("");
  const [order, setOrder] = useState<string[]>(gameData.items || []);
  const [completed, setCompleted] = useState(false);
  const [feedback, setFeedback] = useState<string | null>(null);

  const audioCtx = new (window.AudioContext || (window as any).webkitAudioContext)();

  const playCorrect = () => {
    const osc = audioCtx.createOscillator();
    const gain = audioCtx.createGain();
    osc.type = "sine";
    osc.frequency.setValueAtTime(600, audioCtx.currentTime);
    gain.gain.setValueAtTime(0.1, audioCtx.currentTime);
    osc.connect(gain);
    gain.connect(audioCtx.destination);
    osc.start();
    osc.stop(audioCtx.currentTime + 0.2);
  };

  const playWrong = () => {
    const osc = audioCtx.createOscillator();
    const gain = audioCtx.createGain();
    osc.type = "sawtooth";
    osc.frequency.setValueAtTime(200, audioCtx.currentTime);
    gain.gain.setValueAtTime(0.1, audioCtx.currentTime);
    osc.connect(gain);
    gain.connect(audioCtx.destination);
    osc.start();
    osc.stop(audioCtx.currentTime + 0.3);
  };

  const saveAndComplete = (answer: string, score: number) => {
    if (student) {
      saveProgress(student, gameData.title, answer, score);
    }
    setCompleted(true);
    setTimeout(onComplete, 2000);
  };

  const handleSubmit = () => {
    // Text Input / Rewrite / Reflection / Compare
    if (['text_input', 'rewrite', 'compare', 'reflection'].includes(gameData.type)) {
      if (input.trim().length > 0) {
        saveAndComplete(input, 10);
      }
    }
    // Order Game
    else if (gameData.type === 'order') {
      saveAndComplete(order.join(" -> "), 10);
    }
  };

  const handleTrueFalse = (value: string) => {
    const isCorrect = value === gameData.correctAnswer;
    if (isCorrect) {
      playCorrect();
      setFeedback("إجابة صحيحة! التسامح هو أساس القوة.");
      saveAndComplete(value, 10);
    } else {
      playWrong();
      setFeedback("حاول مرة أخرى. فكر في الإنسانية أولاً.");
    }
  };
};

```



```

const handleSelectChar = (id: string) => {
  const isCorrect = id === gameData.correctAnswer;
  if (isCorrect) {
    playCorrect();
    saveAndComplete(id, 10);
  } else {
    playWrong();
    setFeedback("هذا ليس الاختيار الأمثل. حاول مرة أخرى.");
  }
};

const handleReorder = (item: string) => {
  setOrder(prev => [item, ...prev.filter(i => i !== item)]);
};

if (completed) {
  return (
    <motion.div
      initial={{ scale: 0.8, opacity: 0 }}
      animate={{ scale: 1, opacity: 1 }}
      className="flex items-center justify-center h-full w-full bg-black/80 backdrop-blur-md
absolute inset-0 z-50 font-cairo"
    >
      <div className="text-center p-8 border-4 border-accent bg-[#fdf6e3] rounded-lg shadow-2xl
transform rotate-1 max-w-lg">
        <h2 className="text-4xl md:text-5xl font-changa text-primary mb-4">ممتاز!</h2>
        <p className="text-xl text-gray-700">تم حفظ إجابتك بنجاح</p>
      </div>
    </motion.div>
  );
}

return (
  <div className="absolute inset-0 z-40 flex items-center justify-center bg-black/70 backdrop-
blur-sm p-4 font-cairo">
    <motion.div
      initial={{ y: 50, opacity: 0 }}
      animate={{ y: 0, opacity: 1 }}
      className="w-full max-w-2xl bg-paper-pattern bg-[#fdf6e3] p-8 rounded-lg shadow-2xl pixel-
border relative overflow-hidden"
    >
      <div className="absolute top-0 left-0 w-full h-2 bg-accent/50" />

      <h2 className="text-3xl font-changa text-primary mb-2 text-center border-b-2 border-
primary/20 pb-4">
        {gameData.title}
      </h2>

      <p className="text-xl text-gray-800 mb-8 text-center leading-relaxed font-bold">
        {gameData.description}
      </p>

      {/* FEEDBACK MSG */}
      {feedback && (
        <motion.div
          initial={{ opacity: 0 }} animate={{ opacity: 1 }}
          className="bg-orange-100 text-orange-800 p-3 rounded text-center mb-4 border
border-orange-200"
        >
          {feedback}
        </motion.div>
      )}

      {/* GAME TYPES */}

      {/* 1. ORDER GAME */}
      {gameData.type === 'order' && gameData.items && (
        <div className="flex flex-wrap gap-4 justify-center mb-8">
          {order.map((item, idx) => (
            <motion.button
              key={item}
              layout
              onClick={() => handleReorder(item)}
            >

```

```

        className="px-6 py-3 bg-white border-2 border-secondary text-secondary font-bold
text-lg rounded shadow-md hover:bg-secondary hover:text-white transition-colors"
    >
        {idx + 1}. {item}
    </motion.button>
    )))
    <p className="w-full text-center text-sm text-gray-500 mt-2">(اضغط على الكلمة لنقلها إلى البداية)
</p>

    <div className="w-full text-center mt-6">
        <Button onClick={handleSubmit} className="text-xl px-8 bg-primary text-white font-
changa">تأكيد الترتيب</Button>
    </div>
</div>
))}

{/* 2. TRUE / FALSE */}
{gameData.type === 'true_false' && gameData.options && (
    <div className="grid grid-cols-1 gap-4 mb-4">
        {gameData.options.map((opt) => (
            <Button
                key={opt.value}
                onClick={() => handleTrueFalse(opt.value)}
                className="text-lg py-6 bg-white border-2 border-primary/20 hover:border-
primary hover:bg-primary/5 text-gray-800 justify-start px-6"
            >
                {opt.label}
            </Button>
        ))}
    </div>
)}

{/* 3. SELECT CHARACTER / DRAG MATCH SIMULATION */}
{(gameData.type === 'select_char' || gameData.type === 'drag_match') && gameData.targets &&
(
    <div className="flex justify-center gap-8 mb-6">
        {gameData.targets.map(target => (
            <div
                key={target.id}
                onClick={() => handleSelectChar(target.id)}
                className="cursor-pointer group flex flex-col items-center"
            >
                <div className="w-32 h-32 bg-white/50 rounded-full border-4 border-
transparent group-hover:border-accent transition-all overflow-hidden relative shadow-inner">
                    <img src={target.image} alt={target.label} className="w-full h-full
object-contain transform group-hover:scale-110 transition-transform" />
                </div>
                <span className="mt-2 font-bold text-gray-700">{target.label}</span>
            </div>
        ))}
    </div>
)}

{/* 4. TEXT INPUT TYPES */}
{['text_input', 'rewrite', 'compare', 'reflection'].includes(gameData.type) && (
    <>
        <Textarea
            value={input}
            onChange={(e) => setInput(e.target.value)}
            placeholder="اكتب إجابتك هنا..."
            className="w-full min-h-[150px] text-lg font-cairo p-4 bg-white/50 border-2
border-primary/30 focus:border-primary rounded-md mb-6 resize-none shadow-inner"
        />
        <div className="flex justify-center">
            <Button
                onClick={handleSubmit}
                className="bg-primary hover:bg-primary/90 text-white text-xl px-12 py-6 h-
auto font-changa pixel-corners shadow-lg hover:shadow-xl transform hover:-translate-y-1 transition-
all"
                disabled={!input.trim()}
            >
                تأكيد الإجابة
            </Button>
        </div>
    </>
)
```

```

    </>
  })
  </motion.div>
</div>
);
}
import { SceneData, getCharacterImage } from "@data/scenes";
import { motion, AnimatePresence } from "framer-motion";

interface SceneProps {
  scene: SceneData;
  activeSpeaker: string; // Add this prop to know who is talking
}

export function Scene({ scene, activeSpeaker }: SceneProps) {
  return (
    <div className="relative w-full h-full overflow-hidden bg-black font-cairo">
      {/* Background */}
      <motion.div
        key={scene.background}
        initial={{ opacity: 0 }}
        animate={{ opacity: 1 }}
        exit={{ opacity: 0 }}
        transition={{ duration: 1 }}
        className="absolute inset-0 w-full h-full"
      >
        <img
          src={scene.background}
          alt="background"
          className="w-full h-full object-cover pixelated"
        />
        <div className="absolute inset-0 bg-gradient-to-t from-black/40 via-transparent to-black/10" />
      </motion.div>

      {/* Characters Container Area - Fixed bottom area ensuring no overlap with text */}
      <div className="absolute bottom-[25%] left-0 right-0 h-[60%] flex items-end justify-center
px-8 z-10 pointer-events-none">

        <AnimatePresence mode="popLayout">
          {scene.characters.map((char, index) => {
            const isLeft = char.position.includes('left');
            const isActive = char.type === activeSpeaker;

            // Explicit horizontal spacing using flex logic or absolute percentages
            // Using absolute percentages for precise control as requested ("horizontal
distribution with clear spaces")
            let leftPos = "50%";
            if (char.position === 'left') leftPos = "15%";
            if (char.position === 'mid-left') leftPos = "30%";
            if (char.position === 'center') leftPos = "50%";
            if (char.position === 'mid-right') leftPos = "70%";
            if (char.position === 'right') leftPos = "85%";

            return (
              <motion.div
                key={`-${scene.id}-${char.type}-${index}`}
                initial={{ opacity: 0, x: isLeft ? -50 : 50 }}
                animate={{ opacity: 1, x: "-50%" }} // Centered on the position point
                exit={{ opacity: 0 }}
                transition={{ delay: 0.2 + index * 0.1, duration: 0.5 }}
                className="absolute bottom-0 w-48 md:w-64 lg:w-72 h-auto flex flex-col items-
center"

                style={{ left: leftPos }}
              >
                {/* Character Container to ensure no overlap */}
                <div className="relative w-full h-full">
                  <motion.img
                    src={getCharacterImage(char.type)}
                    alt={char.type}
                    className={`
w-full h-auto object-contain
${isLeft ? 'scale-x-[-1]' : ''}

```

```

        pixelated
        drop-shadow-2xl
      `}
      // Animation Logic
      animate={
        isActive
        ? {
            y: [0, -5, 0], // Bobbing when talking
            scale: [1, 1.02, 1], // Slight pulse
          }
        : {
            y: [0, -2, 0], // Subtle breathing when idle
          }
      }
    }
    transition={{
      repeat: Infinity,
      duration: isActive ? 0.5 : 2.5, // Faster when talking
      ease: "easeInOut"
    }}
  />
</div>
</motion.div>
);
}}}
</AnimatePresence>

</div>
</div>
);
}
import { getAllProgress, clearAllData } from "@lib/storage";
import { Button } from "@components/ui/button";
import { motion } from "framer-motion";
import { useState } from "react";
import { Link } from "wouter";

export function TeacherDashboard() {
  const [data, setData] = useState(getAllProgress());

  const handleClear = () => {
    if (confirm("هل أنت متأكد من حذف جميع بيانات الطالب؟")) {
      clearAllData();
      localStorage.removeItem('project_answers');
      setData({});
    }
  };

  const students = Object.values(data);
  const rawAnswers = JSON.parse(localStorage.getItem('project_answers') || '[]');

  return (
    <div className="min-h-screen bg-[#fdf6e3] p-4 md:p-8 font-cairo" dir="rtl">
      <div className="max-w-6xl mx-auto">
        <div className="flex justify-between items-center mb-8 bg-white/80 backdrop-blur-sm p-6 rounded-xl shadow-sm border border-primary/10">
          <div>
            <h1 className="text-3xl font-changa text-primary">سجل إجابات الطالب</h1>
            <p className="text-gray-500 mt-1">عرض ومتابعة المشاركات النصية</p>
          </div>
          <div className="flex gap-4">
            <Link href="/">
              <Button variant="outline" className="border-primary/20 hover:bg-primary/5">العودة</Button>
            </Link>
            <Button variant="destructive" onClick={handleClear} className="bg-red-500 hover:bg-red-600">مسح السجل</Button>
          </div>
        </div>
        <div className="bg-white/90 backdrop-blur-sm rounded-2xl shadow-xl border border-primary/5 overflow-hidden">
          <div className="overflow-x-auto max-h-[70vh] custom-scrollbar">

```

```

<table className="w-full text-right border-collapse">
  <thead className="sticky top-0 z-10">
    <tr className="bg-primary/5 border-b border-primary/10 backdrop-blur-md">
      <th className="p-5 font-changa text-primary text-lg w-16">#</th>
      <th className="p-5 font-changa text-primary text-lg">اسم الطالبة</th>
      <th className="p-5 font-changa text-primary text-lg">السؤال</th>
      <th className="p-5 font-changa text-primary text-lg">الإجابة النصية</th>
      <th className="p-5 font-changa text-primary text-lg w-48">الوقت</th>
    </tr>
  </thead>
  <tbody className="divide-y divide-primary/5">
    {rawAnswers.length === 0 ? (
      <tr>
        <td colspan={5} className="p-20 text-center text-gray-400 text-xl font-cairo">
          <div className="flex flex-col items-center gap-4">
            <span className="text-5xl opacity-20">📝</span>
            لا توجد إجابات مسجلة حتى الآن
          </div>
        </td>
      </tr>
    ) : (
      rawAnswers.map((item: any, idx: number) => (
        <tr key={idx} className="hover:bg-primary/[0.02] transition-colors group">
          <td className="p-5 text-gray-400 font-mono text-sm">{idx + 1}</td>
          <td className="p-5">
            <span className="font-bold text-gray-800 block">{item.student}</span>
          </td>
          <td className="p-5">
            <span className="bg-primary/5 text-primary px-3 py-1 rounded-lg text-sm font-bold border border-primary/10">
              {item.question}
            </span>
          </td>
          <td className="p-5">
            <div className="text-gray-700 leading-relaxed bg-white/50 p-3 rounded-lg border border-gray-100 group-hover:border-primary/20 transition-colors whitespace-pre-wrap">
              {item.answer}
            </div>
          </td>
          <td className="p-5">
            <div className="text-xs text-gray-400 flex flex-col items-start gap-1 font-mono">
              <span className="bg-gray-100 px-2 py-0.5 rounded">⌚ 17</span>
              {item.time?.split(',')[0] || item.time}</span>
              <span className="bg-gray-100 px-2 py-0.5 rounded">🕒</span>
              {item.time?.split(',')[1] || ''}</span>
            </div>
          </td>
        </tr>
      ))
    )
  </tbody>
</table>
</div>
</div>

<div className="mt-6 text-center text-gray-400 text-sm">
  مشاركة : {rawAnswers.length} عدد المشاركات الإجمالي
</div>

{/* Keeping summary for progress records if needed */}
{students.length > 0 && (
  <div className="mt-12">
    <h2 className="text-2xl font-changa mb-4 text-gray-800 border-r-4 border-primary pr-4">ملخص الطالبات</h2>
    <div className="grid gap-6">
      {students.map((record, idx) => (
        <motion.div
          key={idx}
          initial={{ opacity: 0, y: 20 }}
          animate={{ opacity: 1, y: 0 }}
        </motion.div>
      ))}
    </div>
  </div>
)}

```

```

        transition={{ delay: idx * 0.1 }}
        className="bg-white rounded-lg shadow overflow-hidden border border-gray-200"
      >
        <div className="bg-gray-50 p-4 border-b flex justify-between items-center">
          <div>
            <span className="font-bold text-lg text-primary ml-2">{record.student.name}</span>

            <span className="text-gray-600">({record.student.className})</span>
          </div>
          <span className="text-sm text-gray-400">{new
Date(record.timestamp).toLocaleDateString('ar-EG')}</span>
        </div>

        <div className="p-4 grid gap-4 md:grid-cols-2">
          <div className="space-y-2">
            <h3 className="font-bold text-gray-700 border-b pb-1 mb-2">الإجابات</h3>
            {Object.entries(record.answers).map(([qid, ans]) => (
              <div key={qid} className="bg-amber-50 p-3 rounded text-sm">
                <span className="font-bold text-amber-800 block mb-1">سؤال: {qid}</span>
                <p>{ans}</p>
              </div>
            ))}
          </div>

          <div className="space-y-2">
            <h3 className="font-bold text-gray-700 border-b pb-1 mb-2">النتائج الرقمية</h3>
            {Object.entries(record.scores).map(([qid, score]) => (
              <div key={qid} className="flex justify-between bg-green-50 p-3 rounded
text-sm">
                <span className="font-bold text-green-800">{qid}</span>
                <span className="font-bold bg-white px-2 rounded border">{score}</span>
              </div>
            ))}
          </div>
        </div>
      </motion.div>
    </div>
  </div>
);
}

```