import React, { useState } from 'react';

import { Button } from '@/components/ui/button';

import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card';

import { Progress } from '@/components/ui/progress';

import { DNAType } from './QuizContainer';

interface Question {

id: string;

text: string;

options: {

text: string;

type: 'Architect' | 'Alchemist' | 'Blurred' | 'Neutral';

}[];

}

const questions: Question[] = [

{

id: 'Q1',

text: "You're going away for the weekend. How do you prepare the night before?",

options: [

{ text: 'I mentally run through what I need and pack once — essentials are covered.', type: 'Architect' },

{ text: 'I write a full list, check everything off, repack a few times, still feel uneasy.', type: 'Blurred' },

{ text: "I throw things in last minute and trust it'll be fine.", type: 'Alchemist' },

{ text: 'I pack, unpack, and get overwhelmed deciding what I even need.', type: 'Blurred' }

]

},

{

id: 'Q2',

text: 'A close friend unintentionally hurts your feelings. How do you respond?',

options: [

{ text: "I won't say anything — they'll figure it out or I'll quietly move on.", type: 'Architect' },

{ text: "I'll express it — maybe now, maybe later — but it will come out.", type: 'Alchemist' },

{ text: 'I react suddenly, then second-guess if I was overdramatic.', type: 'Blurred' },

{ text: 'I feel stuck about whether I should say something or not.', type: 'Blurred' }

]

},

{

id: 'Q3',

text: 'You walk into a room full of strangers. What do you do?',

options: [

{ text: 'I observe quietly, scan the room, and engage when it makes strategic sense.', type: 'Architect' },

{ text: 'I tune into the energy — I might light up the room or stay quiet, depending how I feel.', type: 'Alchemist' },

{ text: "I pause and wait for someone to approach — I'm not sure how to show up.", type: 'Neutral' },

{ text: "I keep switching between acting confident and feeling unsure — I want to be seen but don't know how.", type: 'Blurred' }

]

},

{

id: 'Q4',

text: "You've committed to waking up at 6am for a week. Day 3, you're exhausted. What happens?",

options: [

{ text: "I stick to it. Fatigue doesn't override commitment unless it's serious.", type: 'Architect' },

{ text: 'I ask myself if the reason still matters — if not, I adjust without guilt.', type: 'Alchemist' },

{ text: 'I sleep in, feel bad, and try again tomorrow.', type: 'Neutral' },

{ text: "I feel torn — I want to keep going but can't force myself either.", type: 'Blurred' }

]

},

{

id: 'Q5',

text: "You've completed a project and it performs well. How do you feel about it?",

options: [

{ text: "If the result is strong, I'm satisfied — no need to change anything.", type: 'Architect' },

{ text: 'I immediately wonder how it could have been even better.', type: 'Alchemist' },

{ text: 'I feel good but uneasy — maybe I missed something important.', type: 'Blurred' },

{ text: "I can't tell if I'm happy or not — depends what others say.", type: 'Blurred' }

]

},

{

id: 'Q6',

text: "You're pursuing a goal no one else has achieved. How do you think about it?",

options: [

{ text: "I need to see a path or example — otherwise I'm not sure it's achievable.", type: 'Architect' },

{ text: "Even if no one's done it, I know it's possible — I just need the steps.", type: 'Alchemist' },

{ text: 'I doubt myself, but I still try in case it works out.', type: 'Blurred' },

{ text: 'I switch between confidence and confusion depending on the day.', type: 'Blurred' }

]

}

];

interface Props {

onComplete: (dna: DNAType) => void;

recordAnswer: (questionId: string, answer: string) => void;

}

const DefaultDNABlock: React.FC<Props> = ({ onComplete, recordAnswer }) => {

const [currentQuestion, setCurrentQuestion] = useState(0);

const [answers, setAnswers] = useState<Record<string, string>>({});

const [isAnalyzing, setIsAnalyzing] = useState(false);

const handleAnswer = (option: { text: string; type: string }) => {

const questionId = questions[currentQuestion].id;

const newAnswers = { ...answers, [questionId]: option.type };

setAnswers(newAnswers);

recordAnswer(questionId, option.type);

if (currentQuestion < questions.length - 1) {

setTimeout(() => setCurrentQuestion(currentQuestion + 1), 300);

} else {

setIsAnalyzing(true);

setTimeout(() => {

const dnaResult = calculateDNA(newAnswers);

onComplete(dnaResult);

}, 2000);

}

};

const calculateDNA = (answers: Record<string, string>): DNAType => {

const counts = { Architect: 0, Alchemist: 0 };

Object.values(answers).forEach(answer => {

if (answer === 'Architect') counts.Architect++;

if (answer === 'Alchemist') counts.Alchemist++;

});

// 5-6 Architect answers → Default DNA = Architect

if (counts.Architect >= 5) return 'Architect';

// 5-6 Alchemist answers → Default DNA = Alchemist

if (counts.Alchemist >= 5) return 'Alchemist';

// 1-4 mixed → Default DNA = Blurred Identity

return 'Blurred';

};

const progress = ((currentQuestion + 1) / questions.length) \* 100;

if (isAnalyzing) {

return (

<CardContent className="p-12 text-center">

<div className="mb-8">

<div className="animate-spin rounded-full h-16 w-16 border-b-2 border-purple-600 mx-auto mb-4"></div>

<h2 className="text-2xl font-bold text-gray-800 mb-2">

Analysing your Entrepreneurial DNA…

</h2>

<p className="text-gray-600">Processing your responses to determine your core DNA type</p>

</div>

</CardContent>

);

}

return (

<div className="p-8">

<CardHeader className="text-center pb-6">

<CardTitle className="text-2xl font-bold text-gray-800 mb-2">

Discovering Your Entrepreneurial DNA…

</CardTitle>

<div className="mb-4">

<Progress value={progress} className="w-full h-2" />

<p className="text-sm text-gray-600 mt-2">

Question {currentQuestion + 1} of {questions.length}

</p>

</div>

</CardHeader>

<CardContent>

<div className="mb-8">

<h3 className="text-xl font-semibold text-gray-800 mb-6 leading-relaxed">

🧭 {questions[currentQuestion].text}

</h3>

<div className="space-y-3">

{questions[currentQuestion].options.map((option, index) => (

<Button

key={index}

variant="outline"

className="w-full p-6 text-left justify-start hover:scale-[1.02] transition-all duration-200 hover:shadow-md"

onClick={() => handleAnswer(option)}

>

<div className="flex items-center">

<div className={`w-3 h-3 rounded-full mr-4 ${

option.type === 'Architect' ? 'bg-purple-500' :

option.type === 'Alchemist' ? 'bg-orange-500' :

option.type === 'Blurred' ? 'bg-red-500' : 'bg-gray-400'

}`}></div>

<span className="text-base">{option.text}</span>

</div>

</Button>

))}

</div>

</div>

</CardContent>

</div>

);

};

export default DefaultDNABlock;