import React, { useState } from 'react';

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

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

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

import { SubtypeType } from './QuizContainer';

interface Question {

id: string;

text: string;

options: {

text: string;

subtype: SubtypeType;

insight: string;

}[];

}

const getValidationQuestions = (subtype: SubtypeType): Question[] => {

// Alchemist subtypes validation questions

if (subtype === 'Visionary Oracle') {

return [

{

id: 'Q19',

text: "You're given a blank room and asked to design it however you want — no deadline, no instructions, just freedom. What do you naturally do first?",

options: [

{ text: "I stand in the space and feel what it needs. Then I begin placing things intuitively.", subtype: 'Energetic Empath', insight: "feels into spaces, acts through sensation" },

{ text: "I imagine the finished look instantly, then begin collecting images or pieces to match the vision.", subtype: 'Visionary Oracle', insight: "downloads the final vision and works backwards" },

{ text: "I begin sketching it out in a step-by-step plan so I don't waste energy.", subtype: 'Ultimate Alchemist', insight: "structures flow with internal precision" },

{ text: "I overthink each decision and delay starting until I'm sure it'll look right.", subtype: 'Magnetic Perfectionist', insight: "delays due to fear of imperfection" }

]

}

];

}

// Default fallback questions for all subtypes

return [

{

id: 'Q19',

text: `As a ${subtype}, how often do you find yourself in your natural flow state?`,

options: [

{ text: 'Almost always - this resonates completely', subtype: subtype, insight: 'Strong alignment' },

{ text: 'Often - this feels mostly right', subtype: subtype, insight: 'Good alignment' },

{ text: 'Sometimes - partially accurate', subtype: subtype, insight: 'Moderate alignment' },

{ text: 'Rarely - this doesn\'t feel like me', subtype: subtype, insight: 'Low alignment' }

]

},

{

id: 'Q20',

text: `When others describe your ${subtype} traits, do you recognize yourself?`,

options: [

{ text: 'Absolutely - they see the real me', subtype: subtype, insight: 'Strong recognition' },

{ text: 'Mostly - with some variations', subtype: subtype, insight: 'Good recognition' },

{ text: 'Somewhat - mixed recognition', subtype: subtype, insight: 'Moderate recognition' },

{ text: 'Not really - feels disconnected', subtype: subtype, insight: 'Low recognition' }

]

},

{

id: 'Q21',

text: 'Do you ever suppress or hide aspects of your natural entrepreneurial style?',

options: [

{ text: 'Never - I\'m authentic in all situations', subtype: subtype, insight: 'Full authenticity' },

{ text: 'Rarely - only in specific contexts', subtype: subtype, insight: 'Mostly authentic' },

{ text: 'Sometimes - when it feels necessary', subtype: subtype, insight: 'Contextual suppression' },

{ text: 'Often - I feel pressure to conform', subtype: subtype, insight: 'Regular suppression' }

]

},

{

id: 'Q22',

text: 'How aligned do you feel between your inner entrepreneur and outer actions?',

options: [

{ text: 'Completely aligned - no internal conflict', subtype: subtype, insight: 'Full alignment' },

{ text: 'Mostly aligned - minor adjustments needed', subtype: subtype, insight: 'Good alignment' },

{ text: 'Somewhat aligned - working on integration', subtype: subtype, insight: 'Moderate alignment' },

{ text: 'Misaligned - significant gaps exist', subtype: subtype, insight: 'Low alignment' }

]

}

];

};

interface Props {

subtype: SubtypeType;

onComplete: () => void;

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

}

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

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

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

const questions = getValidationQuestions(subtype);

const handleAnswer = (answer: string, selectedOption: any) => {

const questionId = questions[currentQuestion].id;

recordAnswer(questionId, answer);

if (currentQuestion < questions.length - 1) {

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

} else {

setIsAnalyzing(true);

setTimeout(() => {

onComplete();

}, 2000);

}

};

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">

Preparing your results…

</h2>

<p className="text-gray-600">Analyzing your responses</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">

Validating Your Profile

</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.text, option)}

>

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

</Button>

))}

</div>

</div>

</CardContent>

</div>

);

};

export default ValidationBlock;