Escucha Activa en React

Description

"Proyecto para creacion de test de autoevaluacion"

Estructura del proyecto

app dist node_modules public src/ |-- assets |-- components |-- ui |-- button |-- card |-- dropdown-menu |-- progress |-- scroll-area |-- tabs |-- DetailedResults.jsx |-- EscuchaActivaQuiz.jsx |-- QuestionPreview.jsx |-- Recommendations.jsx |-- theme-provider.jsx |-- ThemeToggle.jsx |-- data |-- lib |-- App.css |-- App.jsx |-- index.css |-- main.jsx gitignore components.json eslint.config.js index.html package-lock.json package.json postcss.config.js postcss.config.js tailwind.config.js tsconfig.json vite.config.js

Componentes principales

DetailedResults.jsx

'import React from 'react'; import { Card, CardContent } from "./ui/card"; import { Tabs, TabsContent, TabsList, TabsTrigger } from "./ui/tabs"; import { Progress } from "./ui/progress"; import { BarChart, Bar, XAxis, YAxis, Tooltip, ResponsiveContainer } from 'recharts'; import { CheckCircle2, AlertCircle, XCircle, BookOpen, Target, Brain, HeartHandshake, Users } from 'lucide-react';

// Mapeo de nombres cortos a nombres completos de categorías const categoryNames = { "Atención y Enfoque": "Capacidad de Atención y Enfoque", "Comprensión y Procesamiento": "Comprensión y Procesamiento de Información", "Retroalimentación": "Capacidad de Retroalimentación", "Preparación y Planificación": "Preparación y Planificación de la Comunicación", "Comportamiento en la Comunicación": "Comportamiento durante la Comunicación" };

// Iconos por categoría const categoryIcons = { "Atención y Enfoque": Brain, "Comprensión y Procesamiento": BookOpen, "Retroalimentación": HeartHandshake, "Preparación y Planificación": Target, "Comportamiento en la Comunicación": Users };

// Interpretaciones generales basadas en el puntaje total const getGeneralInterpretation = (totalScore) => { if (totalScore >= 105) { return { level: "Óptimo", description: "Has demostrado excelentes habilidades de escucha activa. Tu capacidad para atender, comprender y responder efectivamente te convierte en un comunicador destacado.", recommendation: "Continúa practicando y compartiendo tus habilidades con otros. Podrías considerar mentorear a otros en el desarrollo de estas capacidades.", color: "text-green-600", bgColor: "bg-green-50" }; } if (totalScore >= 89) { return { level: "Bueno", description: "Tienes una sólida base en escucha activa. Demuestras buenas habilidades en la mayoría de las áreas de comunicación.", recommendation: "Enfócate en perfeccionar las áreas específicas donde aún hay espacio para mejora para alcanzar un nivel óptimo.", color: "text-blue-600", bgColor: "bg-blue-50" }; } if (totalScore >= 73) { return { level: "Regular", description: "Muestras un nivel básico de escucha activa. Hay áreas específicas que requieren atención y desarrollo.", recommendation: "Practica conscientemente las habilidades de escucha en tus interacciones diarias, prestando especial atención a las áreas que necesitan mejora.", color: "text-yellow-600", bgColor: "bg-yellow-50" }; } if (totalScore >= 57) { return { level: "Básico", description: "Tu escucha activa necesita desarrollo. Hay oportunidades significativas para mejorar

en varias áreas.", recommendation: "Considera tomar un curso o taller específico sobre escucha activa y comienza a practicar las habilidades básicas de manera consciente.", color: "text-orange-600", bgColor: "bg-orange-50" }; } return { level: "Necesita Mejorar", description: "Tus habilidades de escucha activa requieren atención inmediata y desarrollo sistemático.", recommendation: "Comienza por trabajar en las habilidades fundamentales de escucha y busca orientación profesional si es posible.", color: "text-red-600", bgColor: "bg-red-50" }; };

// Componente para el resumen general const GeneralOverview = ({ totalScore, categoryScores }) => { const interpretation = getGeneralInterpretation(totalScore); const maxPossibleScore = 120; const percentage = Math.round((totalScore / maxPossibleScore) * 100);

return (

<Card className={${interpretation.bgColor} border-l-4 border-l-${interpretation.color.replace('text-', '')}}> <h3 className={text-2xl font-bold mb-2 ${interpretation.color}}> Nivel: {interpretation.level}

{interpretation.description}

Puntaje Total <span className={font-bold ${interpretation.color}}> {totalScore} / {maxPossibleScore} ({percentage}%)

Recomendación General:

{interpretation.recommendation}

```
  <Card>
    <CardContent className="pt-6">
      <h3 className="text-lg font-semibold mb-4">Resultados por
Categoría</h3>
      <div className="h-80">
        <ResponsiveContainer width="100%" height="100%">
          <BarChart data={categoryScores} margin={{ top: 20, right: 30,
left: 20, bottom: 80 }}>
            <XAxis
              dataKey="name"
              interval={0}
              angle={-45}
              textAnchor="end"
              height={100}
              tick={{ fontSize: 12 }}
            />
            <YAxis />
            <Tooltip
              content={({ active, payload }) => {
                if (active && payload && payload.length) {
                  const data = payload[0].payload;
                  return (
                    <div className="bg-white p-3 shadow-md rounded
```

```
border">
                      <p className="font-medium">
{categoryNames[data.name]}</p>
                      <p className="text-sm text-gray-600">
                        Puntaje: {data.score} / {data.maxScore}
                      </p>
                      <p className="text-sm text-gray-600">
                        ({Math.round((data.score / data.maxScore) *
100)}%)
                      </p>
                    </div>
                  );
                }
                return null;
              }}
            />
            <Bar dataKey="score" fill="#3B82F6" />
          </BarChart>
        </ResponsiveContainer>
      </div>
    </CardContent>
  </Card>
</div>
```

); };

// Componente para el detalle por categoría const CategoryDetail = ({ category, score, maxScore }) => { const Icon = categoryIcons[category] || Brain; const percentage = Math.round((score / maxScore) * 100); let status, statusColor, statusIcon;

if (percentage >= 75) { status = "Dominado"; statusColor = "text-green-600"; statusIcon = ; } else if (percentage >= 50) { status = "En Desarrollo"; statusColor = "text-yellow-600"; statusIcon = ; } else { status = "Necesita Atención"; statusColor = "text-red-600"; statusIcon = ; }

return (

{categoryNames[category]}

Esta categoría evalúa tu capacidad de {category.toLowerCase()}

<span className={font-medium ${statusColor}}> {score} / {maxScore}

```
        <div className="space-y-2 mb-4">
          <div className="flex justify-between text-sm">
            <span>Progreso</span>
            <span>{percentage}%</span>
          </div>
          <Progress value={percentage} className="h-2" />
        </div>
```

```
          <div className="flex items-center gap-2 mt-4">
            {statusIcon}
            <span className={`text-sm ${statusColor}`}>{status}</span>
          </div>
        </div>
      </div>
    </CardContent>
  </Card>
```

); };

// Componente principal export const DetailedResults = ({ categoryScores, totalScore }) => { return (Resumen Detalles Recomendaciones

```
    <TabsContent value="overview" className="mt-6">
      <GeneralOverview
        totalScore={totalScore}
        categoryScores={categoryScores}
      />
    </TabsContent>

    <TabsContent value="details" className="mt-6">
      <div className="space-y-6">
        <h2 className="text-xl font-semibold mb-4">Análisis Detallado por
Categoría</h2>
        {categoryScores.map(({ name, score, maxScore }) => (
          <CategoryDetail
            key={name}
            category={name}
            score={score}
            maxScore={maxScore}
          />
        ))}
      </div>
    </TabsContent>

    <TabsContent value="recommendations" className="mt-6">
      <div className="space-y-6">
        <Card className="mb-4">
          <CardContent className="pt-6">
            <h3 className="text-lg font-semibold mb-4">Plan de Mejora
Personalizado</h3>
            <div className="space-y-4">
              {categoryScores
                .filter(({ score, maxScore }) => (score / maxScore) * 100 <
75)
                .map(({ name, score, maxScore }) => {
                  const percentage = Math.round((score / maxScore) * 100);
                  return (
```

```
                    <div key={name} className="p-4 bg-gray-50 rounded-lg">
                      <h4 className="font-medium flex items-center gap-2">
                        {categoryIcons[name] &&
  React.createElement(categoryIcons[name], { className: "h-4 w-4" })}
                        {categoryNames[name]}
                        <span className="text-sm text-gray-500">
  ({percentage}%)</span>
                      </h4>
                      <ul className="mt-2 space-y-2">
                        {/* Aquí puedes agregar recomendaciones específicas
  */}
                        <li className="text-sm text-gray-600">• Practica
  ejercicios específicos de {name.toLowerCase()}</li>
                        <li className="text-sm text-gray-600">• Establece
  metas semanales para mejorar en esta área</li>
                        <li className="text-sm text-gray-600">• Busca
  oportunidades para aplicar estas habilidades en tu día a día</li>
                      </ul>
                    </div>
                  );
                })}
            </div>
          </CardContent>
        </Card>
      </div>
    </TabsContent>
  </Tabs>
```

); };'

EscuchaActivaQuiz.jsx

'import React, { useState } from 'react'; import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card'; import { Button } from '@/components/ui/button'; import { Progress } from '@/components/ui/progress'; import { ScrollArea } from '@/components/ui/scroll-area'; import { AlertCircle, Eye, EyeOff, ArrowLeft, ArrowRight } from 'lucide-react'; import { motion, AnimatePresence } from 'framer-motion'; import { DetailedResults } from './DetailedResults';

import { questions } from '../data/questions'; import { categories, getCategoryLevel } from '../data/categories'; import { recommendations } from '../data/recommendations';

// QuestionView Component (mantener el mismo código) const QuestionView = ({ question, onAnswer, currentAnswer }) => { const options = ['La mayoría de las veces', 'Frecuentemente', 'Ocasionalmente', 'Casi nunca'];

return (

{question.text}

{question.category}

```
  <div className="grid grid-cols-1 md:grid-cols-2 gap-4">
    {options.map((option, index) => (
      <Button
        key={option}
        onClick={() => onAnswer(4 - index)}
        variant={currentAnswer === (4 - index) ? "default" : "outline"}
        className="w-full h-auto py-4 px-6 text-sm"
      >
        {option}
      </Button>
    ))}
  </div>
</div>
```

); };

// QuestionPreview Component (mantener el mismo código) const QuestionPreview = ({ questions, currentQuestion, onQuestionClick, answers }) => { const getAnswerText = (value) => { const options = { 4: 'La mayoría de las veces', 3: 'Frecuentemente', 2: 'Ocasionalmente', 1: 'Casi nunca' }; return options[value]; };

return (

{questions.map((question, index) => (<Card key={question.id} className={ `cursor-pointer transition-all hover:shadow-md ${currentQuestion === index ? 'border-2 border-primary' : ''}` } onClick={() => onQuestionClick(index)} >
Pregunta {index + 1} {question.isReverse && (Inversa)}

{question.text}

{question.category}
{answers[question.id] && ({getAnswerText(answers[question.id])})}
))}
); };

// Componente principal actualizado export const EscuchaActivaQuiz = () => { const [currentQuestion, setCurrentQuestion] = useState(0); const [answers, setAnswers] = useState({}); const [showResults, setShowResults] = useState(false); const [showPreview, setShowPreview] = useState(false);

const handleAnswer = (value) => { const newAnswers = { ...answers, [currentQuestion + 1]: value }; setAnswers(newAnswers);

```
if (currentQuestion < questions.length - 1) {
  setCurrentQuestion(currentQuestion + 1);
} else {
  setShowResults(true);
}
```

};

const calculateCategoryScore = (category) => { let total = 0;
categories[category].questionIds.forEach(questionId => { const answer = answers[questionId]; if (answer) {
const question = questions.find(q => q.id === questionId); if (question) { total += question.isReverse ? (5 -
answer) : answer; } } }); return total; };

const resetQuiz = () => { setCurrentQuestion(0); setAnswers({}); setShowResults(false);
setShowPreview(false); };

if (showResults) { const categoryScores = Object.keys(categories).map(categoryName => { const
maxScore = categories[categoryName].maxScore; const score = calculateCategoryScore(categoryName);
return { name: categoryName, score, maxScore, recommendations: recommendations[categoryName] || []
}; });

```
const totalScore = categoryScores.reduce((acc, curr) => acc + curr.score,
0);

return (
  <div className="max-w-4xl mx-auto space-y-6">
    <Card>
      <CardHeader>
        <CardTitle>Resultados de tu Evaluación</CardTitle>
      </CardHeader>
      <CardContent>
        <DetailedResults
          categoryScores={categoryScores}
          totalScore={totalScore}
        />
        <Button
          onClick={resetQuiz}
          className="w-full mt-6"
        >
          Realizar el test nuevamente
        </Button>
      </CardContent>
    </Card>
  </div>
);
```

}

return (

Pregunta {currentQuestion + 1} de {questions.length} <Button variant="ghost" size="icon" onClick={() =>
setShowPreview(!showPreview)} > {showPreview ? : } <Progress value={(currentQuestion /
questions.length) * 100} className="mb-6" /> {showPreview ? (<motion.div initial={{ opacity: 0, y: 20 }}
animate={{ opacity: 1, y: 0 }} exit={{ opacity: 0, y: -20 }} transition={{ duration: 0.2 }} > </motion.div>) : (
<motion.div initial={{ opacity: 0, y: 20 }} animate={{ opacity: 1, y: 0 }} exit={{ opacity: 0, y: -20 }} transition={{

duration: 0.2 }} > <QuestionView question={questions[currentQuestion]} onAnswer={handleAnswer} currentAnswer={answers[currentQuestion + 1]} /> </motion.div>)}

```
    <div className="flex justify-between">
      <Button
        variant="outline"
        disabled={currentQuestion === 0}
        onClick={() => setCurrentQuestion(curr => curr - 1)}
      >
        <ArrowLeft className="h-4 w-4 mr-2" />
        Anterior
      </Button>
      <Button
        variant="outline"
        disabled={currentQuestion === questions.length - 1}
        onClick={() => setCurrentQuestion(curr => curr + 1)}
      >
        Siguiente
        <ArrowRight className="h-4 w-4 ml-2" />
      </Button>
    </div>
  </div>
```

); };'

QuestionPreview.jsx

'import React from 'react'; import { Card, CardContent } from '@/components/ui/card'; import { ScrollArea } from '@/components/ui/scroll-area'; import { cn } from '@/lib/utils';

export const QuestionPreview = ({ questions, currentQuestion, onQuestionClick, answers }) => { const getAnswerText = (value) => { const options = { 4: 'La mayoría de las veces', 3: 'Frecuentemente', 2: 'Ocasionalmente', 1: 'Casi nunca' }; return options[value] || ''; };

return (

{questions.map((question, index) => (<Card key={question.id} className={cn("p-4 cursor-pointer hover:bg-accent/50 transition-colors", currentQuestion === index && "border-2 border-primary")} onClick={() => onQuestionClick(index)} >
Pregunta {index + 1} {answers[question.id] && ({getAnswerText(answers[question.id])})}

{question.text}

{question.category} {question.isReverse && (Inversa)}
))}
); };'

Recommendations.jsx

'import React from 'react'; import { Card, CardContent } from '@/components/ui/card'; import { Button } from '@/components/ui/button'; import { BookOpen, Brain, CheckCircle2, Clock, HeartHandshake, Lightbulb, Target, Workflow } from 'lucide-react';

const RecommendationCard = ({ title, description, steps, icon: Icon }) => (

{title}

{description}

{steps.map((step, index) => (
{index + 1}

{step}

))}
);

const categoryIcons = { "Atención y Enfoque": Lightbulb, "Comprensión y Procesamiento": BookOpen, "Retroalimentación": HeartHandshake, "Preparación y Planificación": Target, "Comportamiento en la Comunicación": Workflow };

export const DetailedRecommendations = ({ categoryScores, onPracticeStart }) => { const needsImprovement = categoryScores.filter(cat => (cat.score / cat.maxScore) * 100 < 75);

if (needsImprovement.length === 0) { return (

¡Excelente! Has demostrado un buen dominio en todas las áreas.

Continúa practicando para mantener tus habilidades.

); }

return (

{needsImprovement.map(({ name, score, maxScore }) => { const percentage = Math.round((score / maxScore) * 100); const Icon = categoryIcons[name] || Lightbulb;

```
      const recommendations = {
        "Atención y Enfoque": {
          title: "Mejora tu Concentración",
          description: "Ejercicios prácticos para desarrollar tu
atención",
          steps: [
            "Practica meditación mindfulness 10 minutos diarios",
            "Establece un espacio libre de distracciones",
            "Usa técnicas de respiración consciente",
            "Implementa la regla de los 5 segundos"
          ]
        },
        "Comprensión y Procesamiento": {
```

```
          title: "Mejora tu Comprensión",
          description: "Estrategias para mejorar tu procesamiento de
información",
          steps: [
            "Lee resúmenes de libros",
            "Practica ejercicios de memoria",
            "Usa mapas mentales para organizar ideas",
            "Realiza ejercicios de lectura rápida"
          ]
        },
        "Retroalimentación": {
          title: "Mejora tu Feedback",
          description: "Estrategias para dar y recibir retroalimentación
efectiva",
          steps: [
            "Aplica la técnica del sándwich",
            "Usa frases en primera persona",
            "Practica la escucha activa",
            "Valida emociones antes de responder"
          ]
        },
        "Preparación y Planificación": {
          title: "Optimiza tu Preparación",
          description: "Métodos para planificar mejor tus comunicaciones",
          steps: [
            "Crea checklist pre-conversación",
            "Establece objetivos SMART",
            "Prepara preguntas clave",
            "Anticipa posibles escenarios"
          ]
        },
        "Comportamiento en la Comunicación": {
          title: "Mejora tu Comportamiento",
          description: "Pautas para una comunicación más efectiva",
          steps: [
            "Practica el control de impulsos",
            "Desarrolla consciencia corporal",
            "Implementa pausas estratégicas",
            "Cultiva la empatía activa"
          ]
        }
      };

      return (
        <RecommendationCard
          key={name}
          icon={Icon}
          title={recommendations[name].title}
          description={`${name}: ${percentage}% de dominio`}
          steps={recommendations[name].steps}
        />
```

```
        );
      })}
    </div>

    <Card>
      <CardContent className="pt-6">
        <h3 className="font-medium text-base mb-3 flex items-center gap-2">
          <Clock className="h-4 w-4" />
          Plan de Práctica Sugerido
        </h3>
        <div className="space-y-2 mb-4">
          <p className="text-sm text-muted-foreground">
            Dedica 15-20 minutos diarios a practicar estas habilidades.
            Comienza con el área de menor puntaje y progresa gradualmente.
          </p>
        </div>
        <Button
          className="w-full"
          onClick={onPracticeStart}
        >
          Comenzar Plan de Práctica
        </Button>
      </CardContent>
    </Card>
  </div>
```

); };'

theme-provider.jsx

'"use client";

import { createContext, useContext, useEffect, useState } from "react";

const ThemeProviderContext = createContext({});

export function ThemeProvider({ children, defaultTheme = "system", storageKey = "vite-ui-theme", ...props }) { const [theme, setTheme] = useState(() => localStorage.getItem(storageKey) || defaultTheme);

useEffect(() => { const root = window.document.documentElement;

```
  root.classList.remove("light", "dark");

  if (theme === "system") {
    const systemTheme = window.matchMedia("(prefers-color-scheme: dark)")
      .matches
      ? "dark"
      : "light";

    root.classList.add(systemTheme);
```

```
    return;
  }

  root.classList.add(theme);
```

}, [theme]);

const value = { theme, setTheme: (theme) => { localStorage.setItem(storageKey, theme); setTheme(theme); }, };

return (<ThemeProviderContext.Provider {...props} value={value}> {children} </ThemeProviderContext.Provider>); }

export const useTheme = () => { const context = useContext(ThemeProviderContext);

if (context === undefined) throw new Error("useTheme must be used within a ThemeProvider");

return context; };'

ThemeToggle.jsx

'import { Moon, Sun } from "lucide-react"; import { useTheme } from "./theme-provider"; import { Button } from "./ui/button"; import { DropdownMenu, DropdownMenuContent, DropdownMenuItem, DropdownMenuTrigger, } from "./ui/dropdown-menu";

export function ThemeToggle() { const { setTheme } = useTheme();

return (Toggle theme <DropdownMenuItem onClick={() => setTheme("light")}> Claro <DropdownMenuItem onClick={() => setTheme("dark")}> Oscuro <DropdownMenuItem onClick={() => setTheme("system")}> Sistema); }'

user interface (ui)

button.tsx

'import * as React from "react" import { Slot } from "@radix-ui/react-slot" import { cva, type VariantProps } from "class-variance-authority"

import { cn } from "@/lib/utils"

const buttonVariants = cva("inline-flex items-center justify-center whitespace-nowrap rounded-md text-sm font-medium ring-offset-background transition-colors focus-visible:outline-none focus-visible:ring-2 focus-visible:ring-ring focus-visible:ring-offset-2 disabled:pointer-events-none disabled:opacity-50", { variants: { variant: { default: "bg-primary text-primary-foreground hover:bg-primary/90", destructive: "bg-destructive text-destructive-foreground hover:bg-destructive/90", outline: "border border-input bg-background hover:bg-accent hover:text-accent-foreground", secondary: "bg-secondary text-secondary-foreground hover:bg-secondary/80", ghost: "hover:bg-accent hover:text-accent-foreground", link: "text-primary underline-offset-4 hover:underline", }, size: { default: "h-10 px-4 py-2", sm: "h-9 rounded-md px-3", lg: "h-11 rounded-md px-8", icon: "h-10 w-10", }, }, defaultVariants: { variant: "default", size: "default", }, })

export interface ButtonProps extends React.ButtonHTMLAttributes, VariantProps { asChild?: boolean }

const Button = React.forwardRef<HTMLButtonElement, ButtonProps>(({ className, variant, size, asChild = false, ...props }, ref) => { const Comp = asChild ? Slot : "button" return (<Comp className= {cn(buttonVariants({ variant, size, className }))} ref={ref} {...props} />) }) Button.displayName = "Button"

export { Button, buttonVariants }'

card.tsx

'import * as React from "react" import { cn } from "../../lib/utils"

const Card = React.forwardRef<HTMLDivElement, React.HTMLProps>(({ className, ...props }, ref) => (

)) Card.displayName = "Card"

const CardHeader = React.forwardRef<HTMLDivElement, React.HTMLProps>(({ className, ...props }, ref) => (

)) CardHeader.displayName = "CardHeader"

const CardTitle = React.forwardRef<HTMLHeadingElement, React.HTMLProps>(({ className, ...props }, ref) => (

)) CardTitle.displayName = "CardTitle"

const CardDescription = React.forwardRef<HTMLParagraphElement, React.HTMLProps>(({ className, ...props }, ref) => (

)) CardDescription.displayName = "CardDescription"

const CardContent = React.forwardRef<HTMLDivElement, React.HTMLProps>(({ className, ...props }, ref) => (

)) CardContent.displayName = "CardContent"

const CardFooter = React.forwardRef<HTMLDivElement, React.HTMLProps>(({ className, ...props }, ref) => (

)) CardFooter.displayName = "CardFooter"

export { Card, CardHeader, CardFooter, CardTitle, CardDescription, CardContent }'

dropdown-menu.tsx

'"use client"

import * as React from "react" import * as DropdownMenuPrimitive from "@radix-ui/react-dropdown-menu" import { Check, ChevronRight, Circle } from "lucide-react"

import { cn } from "@/lib/utils"

const DropdownMenu = DropdownMenuPrimitive.Root

const DropdownMenuTrigger = DropdownMenuPrimitive.Trigger

const DropdownMenuGroup = DropdownMenuPrimitive.Group

const DropdownMenuPortal = DropdownMenuPrimitive.Portal

const DropdownMenuSub = DropdownMenuPrimitive.Sub

const DropdownMenuRadioGroup = DropdownMenuPrimitive.RadioGroup

const DropdownMenuSubTrigger = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef & { inset?: boolean }

> (({ className, inset, children, ...props }, ref) => (<DropdownMenuPrimitive.SubTrigger ref={ref} className={cn("flex cursor-default select-none items-center rounded-sm px-2 py-1.5 text-sm outline-none focus:bg-accent data-[state=open]:bg-accent", inset && "pl-8", className)} {...props}

```
{children}
<ChevronRight className="ml-auto h-4 w-4" />
```

</DropdownMenuPrimitive.SubTrigger>)) DropdownMenuSubTrigger.displayName = DropdownMenuPrimitive.SubTrigger.displayName

const DropdownMenuSubContent = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, ...props }, ref) => (<DropdownMenuPrimitive.SubContent ref={ref} className={cn("z-50 min-w-[8rem] overflow-hidden rounded-md border bg-popover p-1 text-popover-foreground shadow-lg data-[state=open]:animate-in data-[state=closed]:animate-out data-[state=closed]:fade-out-0 data-[state=open]:fade-in-0 data-[state=closed]:zoom-out-95 data-[state=open]:zoom-in-95 data-[side=bottom]:slide-in-from-top-2 data-[side=left]:slide-in-from-right-2 data-[side=right]:slide-in-from-left-2 data-[side=top]:slide-in-from-bottom-2", className)} {...props} />)) DropdownMenuSubContent.displayName = DropdownMenuPrimitive.SubContent.displayName

const DropdownMenuContent = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, sideOffset = 4, ...props }, ref) => (<DropdownMenuPrimitive.Portal> <DropdownMenuPrimitive.Content ref={ref} sideOffset={sideOffset} className={cn("z-50 min-w-[8rem] overflow-hidden rounded-md border bg-popover p-1 text-popover-foreground shadow-md data-[state=open]:animate-in data-[state=closed]:animate-out data-[state=closed]:fade-out-0 data-[state=open]:fade-in-0 data-[state=closed]:zoom-out-95 data-[state=open]:zoom-in-95 data-[side=bottom]:slide-in-from-top-2 data-[side=left]:slide-in-from-right-2 data-[side=right]:slide-in-from-left-2 data-[side=top]:slide-in-from-bottom-2", className)} {...props} /> </DropdownMenuPrimitive.Portal>)) DropdownMenuContent.displayName = DropdownMenuPrimitive.Content.displayName

const DropdownMenuItem = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef & { inset?: boolean }

> (({ className, inset, ...props }, ref) => (<DropdownMenuPrimitive.Item ref={ref} className={cn("relative flex cursor-default select-none items-center rounded-sm px-2 py-1.5 text-sm outline-none transition-colors focus:bg-accent focus:text-accent-foreground data-[disabled]:pointer-events-none data-[disabled]:opacity-50", inset && "pl-8", className)} {...props} />)) DropdownMenuItem.displayName = DropdownMenuPrimitive.Item.displayName

const DropdownMenuCheckboxItem = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, children, checked, ...props }, ref) => (<DropdownMenuPrimitive.CheckboxItem ref={ref} className={cn("relative flex cursor-default select-none items-center rounded-sm py-1.5 pl-8 pr-2 text-sm outline-none transition-colors focus:bg-accent focus:text-accent-foreground data-[disabled]:pointer-events-none data-[disabled]:opacity-50", className)} checked={checked} {...props}

```
<span className="absolute left-2 flex h-3.5 w-3.5 items-center justify-
center">
  <DropdownMenuPrimitive.ItemIndicator>
    <Check className="h-4 w-4" />
  </DropdownMenuPrimitive.ItemIndicator>
</span>
{children}
```

</DropdownMenuPrimitive.CheckboxItem>)) DropdownMenuCheckboxItem.displayName = DropdownMenuPrimitive.CheckboxItem.displayName

const DropdownMenuRadioItem = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, children, ...props }, ref) => (<DropdownMenuPrimitive.RadioItem ref={ref} className={cn("relative flex cursor-default select-none items-center rounded-sm py-1.5 pl-8 pr-2 text-sm outline-none transition-colors focus:bg-accent focus:text-accent-foreground data-[disabled]:pointer-events-none data-[disabled]:opacity-50", className)} {...props}

```
<span className="absolute left-2 flex h-3.5 w-3.5 items-center justify-
center">
  <DropdownMenuPrimitive.ItemIndicator>
    <Circle className="h-2 w-2 fill-current" />
  </DropdownMenuPrimitive.ItemIndicator>
</span>
{children}
```

</DropdownMenuPrimitive.RadioItem>)) DropdownMenuRadioItem.displayName = DropdownMenuPrimitive.RadioItem.displayName

const DropdownMenuLabel = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef & { inset?: boolean }

> (({ className, inset, ...props }, ref) => (<DropdownMenuPrimitive.Label ref={ref} className={cn("px-2 py-1.5 text-sm font-semibold", inset && "pl-8", className)} {...props} />)) DropdownMenuLabel.displayName = DropdownMenuPrimitive.Label.displayName

const DropdownMenuSeparator = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, ...props }, ref) => (<DropdownMenuPrimitive.Separator ref={ref} className={cn("-mx-1 my-1 h-px bg-muted", className)} {...props} />)) DropdownMenuSeparator.displayName = DropdownMenuPrimitive.Separator.displayName

const DropdownMenuShortcut = ({ className, ...props }: React.HTMLAttributes) => { return (<span className={cn("ml-auto text-xs tracking-widest opacity-60", className)} {...props} />) } DropdownMenuShortcut.displayName = "DropdownMenuShortcut"

export { DropdownMenu, DropdownMenuTrigger, DropdownMenuContent, DropdownMenuItem, DropdownMenuCheckboxItem, DropdownMenuRadioItem, DropdownMenuLabel, DropdownMenuSeparator, DropdownMenuShortcut, DropdownMenuGroup, DropdownMenuPortal, DropdownMenuSub, DropdownMenuSubContent, DropdownMenuSubTrigger, DropdownMenuRadioGroup, }'

progress.jsx

'"use client"

import * as React from "react" import * as ProgressPrimitive from "@radix-ui/react-progress"

import { cn } from "../../lib/utils"

const Progress = React.forwardRef(({ className, value, ...props }, ref) => (<ProgressPrimitive.Root ref={ref} className={cn("relative h-4 w-full overflow-hidden rounded-full bg-secondary", className)} {...props}

```
<ProgressPrimitive.Indicator
  className="h-full w-full flex-1 bg-primary transition-all"
  style={{ transform: `translateX(-${100 - (value || 0)}%)` }}
/>
```

</ProgressPrimitive.Root>)) Progress.displayName = ProgressPrimitive.Root.displayName

export { Progress }'

progress tsx

'import * as React from "react" import * as ProgressPrimitive from "@radix-ui/react-progress"

import { cn } from "@/lib/utils"

const Progress = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, value, ...props }, ref) => (<ProgressPrimitive.Root ref={ref} className={cn("relative h-4 w-full overflow-hidden rounded-full bg-secondary", className)} {...props}

```
<ProgressPrimitive.Indicator
  className="h-full w-full flex-1 bg-primary transition-all"
  style={{ transform: `translateX(-${100 - (value || 0)}%)` }}
/>
```

</ProgressPrimitive.Root>)) Progress.displayName = ProgressPrimitive.Root.displayName

export { Progress }'

scroll-area.tsx

'"use client"

import * as React from "react" import * as ScrollAreaPrimitive from "@radix-ui/react-scroll-area"

import { cn } from "@/lib/utils"

const ScrollArea = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, children, ...props }, ref) => (<ScrollAreaPrimitive.Root ref={ref} className= {cn("relative overflow-hidden", className)} {...props}

```
<ScrollAreaPrimitive.Viewport className="h-full w-full rounded-[inherit]">
  {children}
</ScrollAreaPrimitive.Viewport>
<ScrollBar />
<ScrollAreaPrimitive.Corner />
```

</ScrollAreaPrimitive.Root>)) ScrollArea.displayName = ScrollAreaPrimitive.Root.displayName

const ScrollBar = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, orientation = "vertical", ...props }, ref) => (<ScrollAreaPrimitive.ScrollAreaScrollbar ref={ref} orientation={orientation} className={cn("flex touch-none select-none transition-colors", orientation === "vertical" && "h-full w-2.5 border-l border-l-transparent p-[1px]", orientation === "horizontal" && "h-2.5 flex-col border-t border-t-transparent p-[1px]", className)} {...props}

```
<ScrollAreaPrimitive.ScrollAreaThumb className="relative flex-1 rounded-
full bg-border" />
```

</ScrollAreaPrimitive.ScrollAreaScrollbar>)) ScrollBar.displayName = ScrollAreaPrimitive.ScrollAreaScrollbar.displayName

export { ScrollArea, ScrollBar }'

tabs

'"use client"

import * as React from "react" import * as TabsPrimitive from "@radix-ui/react-tabs"

import { cn } from "../../lib/utils"

const Tabs = TabsPrimitive.Root

const TabsList = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, ...props }, ref) => (<TabsPrimitive.List ref={ref} className={cn("inline-flex h-10 items-center justify-center rounded-md bg-muted p-1 text-muted-foreground", className)} {...props} />)) TabsList.displayName = TabsPrimitive.List.displayName

const TabsTrigger = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, ...props }, ref) => (<TabsPrimitive.Trigger ref={ref} className={cn("inline-flex items-center justify-center whitespace-nowrap rounded-sm px-3 py-1.5 text-sm font-medium ring-offset-background transition-all focus-visible:outline-none focus-visible:ring-2 focus-visible:ring-ring focus-visible:ring-offset-2 disabled:pointer-events-none disabled:opacity-50 data-[state=active]:bg-background data-[state=active]:text-foreground data-[state=active]:shadow-sm", className)} {...props} />)) TabsTrigger.displayName = TabsPrimitive.Trigger.displayName

const TabsContent = React.forwardRef< React.ElementRef, React.ComponentPropsWithoutRef

> (({ className, ...props }, ref) => (<TabsPrimitive.Content ref={ref} className={cn("mt-2 ring-offset-background focus-visible:outline-none focus-visible:ring-2 focus-visible:ring-ring focus-visible:ring-offset-2", className)} {...props} />)) TabsContent.displayName = TabsPrimitive.Content.displayName

export { Tabs, TabsList, TabsTrigger, TabsContent }'

data

categories.js

'export const categories = { "Atención y Enfoque": { questionIds: [1, 2, 6, 10, 14, 27], description: "Capacidad para mantener el foco y la concentración durante la comunicación", maxScore: 24, // 6 preguntas * 4 puntos máximos levels: { excellent: { range: [20, 24], description: "Excelente capacidad para mantener la atención y evitar distracciones" }, good: { range: [16, 19], description: "Buena capacidad de atención con ocasionales pérdidas de foco" }, needsWork: { range: [0, 15], description: "Área que requiere desarrollo para mejorar la concentración" } } }, "Comprensión y Procesamiento": { questionIds: [3, 7, 12, 13, 15], description: "Habilidad para entender y procesar la información recibida", maxScore: 20, // 5 preguntas * 4 puntos máximos levels: { excellent: { range: [17, 20], description: "Excelente capacidad de comprensión y análisis" }, good: { range: [14, 16], description: "Buena capacidad de procesamiento con áreas de mejora" }, needsWork: { range: [0, 13], description: "Necesita desarrollar habilidades de comprensión" } } }, "Retroalimentación": { questionIds: [4, 28, 29, 30], description: "Capacidad para proporcionar y recibir

feedback efectivo", maxScore: 16, // 4 preguntas * 4 puntos máximos levels: { excellent: { range: [14, 16], description: "Excelente manejo de la retroalimentación" }, good: { range: [11, 13], description: "Buen manejo del feedback con espacio para mejora" }, needsWork: { range: [0, 10], description: "Área que requiere desarrollo significativo" } } }, "Preparación y Planificación": { questionIds: [17, 18, 19, 20, 21], description: "Capacidad para prepararse y organizar la comunicación", maxScore: 20, // 5 preguntas * 4 puntos máximos levels: { excellent: { range: [17, 20], description: "Excelente preparación y planificación comunicativa" }, good: { range: [14, 16], description: "Buena planificación con aspectos a mejorar" }, needsWork: { range: [0, 13], description: "Necesita desarrollar habilidades de planificación" } } }, "Comportamiento en la Comunicación": { questionIds: [5, 8, 9, 22, 23, 24, 25, 26], description: "Conductas y actitudes durante el proceso comunicativo", maxScore: 32, // 8 preguntas * 4 puntos máximos levels: { excellent: { range: [27, 32], description: "Comportamiento comunicativo excelente" }, good: { range: [22, 26], description: "Buen comportamiento comunicativo con áreas de mejora" }, needsWork: { range: [0, 21], description: "Necesita mejorar patrones de comportamiento" } } } };

// Función auxiliar para obtener el nivel de una categoría export const getCategoryLevel = (categoryName, score) => { const category = categories[categoryName]; const { levels } = category;

```
  for (const [level, { range }] of Object.entries(levels)) {
    if (score >= range[0] && score <= range[1]) {
      return {
        level,
        description: levels[level].description
      };
    }
  }
```

};

// Función para calcular el porcentaje de progreso en una categoría export const getCategoryProgress = (categoryName, score) => { const category = categories[categoryName]; return (score / category.maxScore) * 100; };

// Obtener el total máximo posible export const getTotalMaxScore = () => { return Object.values(categories).reduce((total, category) => total + category.maxScore, 0); };

// Función para determinar si una categoría necesita mejora export const needsImprovement = (categoryName, score) => { const category = categories[categoryName]; const percentage = (score / category.maxScore) * 100; return percentage < 75; };'

questions.js

'export const questions = [{ id: 1, text: "Ocupo mi mente con otros pensamientos, principalmente cuando dicen algo que no quiero escuchar o con que no concuerdo.", category: "Atención y Enfoque", isReverse: true }, { id: 2, text: "Aunque el asunto no sea de mi interés, me concentro en lo que se está diciendo.", category: "Atención y Enfoque", isReverse: false }, { id: 3, text: "Normalmente, en medio de la conversación, sé lo que el otro está diciendo y paro de escuchar.", category: "Comprensión y Procesamiento", isReverse: true }, { id: 4, text: "Cuando el otro habla, suelo repetir para mí mismo las palabras que acaba de decir.", category: "Retroalimentación", isReverse: false }, { id: 5, text: "Aunque piensen diferente de mí, suelo

escuchar las opiniones de los otros.", category: "Comportamiento en la Comunicación", isReverse: false }, { id: 6, text: "Presto atención a las personas, pues creo que siempre voy a aprender algo.", category: "Atención y Enfoque", isReverse: false }, { id: 7, text: "Descubro lo que las palabras quieren decir cuando no son familiares para mí, prestando atención al contexto.", category: "Comprensión y Procesamiento", isReverse: false }, { id: 8, text: "No tengo paciencia para oír, por eso creo un rechazo en la mente mientras el otro está hablando.", category: "Comportamiento en la Comunicación", isReverse: true }, { id: 9, text: "Doy la impresión de ser buen oyente, aunque no escuche ninguna palabra del otro.", category: "Comportamiento en la Comunicación", isReverse: true }, { id: 10, text: "Mis pensamientos viajan mientras el otro está hablando.", category: "Atención y Enfoque", isReverse: true }, { id: 11, text: "Soy buen oyente, porque escucho el mensaje totalmente, tanto verbal como no verbal del otro.", category: "Comprensión y Procesamiento", isReverse: false }, { id: 12, text: "Reconozco que las palabras no siempre quieren decir exactamente las mismas cosas para personas diferentes.", category: "Comprensión y Procesamiento", isReverse: false }, { id: 13, text: "Tengo fuerte escucha selectiva, pues solo escucho lo que me interesa y borro de la mente el resto del mensaje.", category: "Comprensión y Procesamiento", isReverse: true }, { id: 14, text: "Miro directamente a la persona cuando está hablando.", category: "Atención y Enfoque", isReverse: false }, { id: 15, text: "Me concentro en 'qué quiere decir' el otro, de preferencia qué y cómo mira, cuando habla.", category: "Comprensión y Procesamiento", isReverse: false }, { id: 16, text: "Sé cuáles son las frases que me dejan emocionalmente tocado.", category: "Retroalimentación", isReverse: false }, { id: 17, text: "Antes de hablar, clarifico en mi mente el objetivo de mi mensaje.", category: "Preparación y Planificación", isReverse: false }, { id: 18, text: "Yo escojo la hora correcta para decir algo a alguien.", category: "Preparación y Planificación", isReverse: false }, { id: 19, text: "Antes de hablar, pienso sobre cómo el otro puede reaccionar a lo que voy a decir.", category: "Preparación y Planificación", isReverse: false }, { id: 20, text: "En mi comunicación, considero la mejor manera (escrita, hablada, email, teléfono, etc.) para emitir mi mensaje.", category: "Preparación y Planificación", isReverse: false }, { id: 21, text: "Antes de hablar, pienso sobre las características o sentimientos de la persona con quien voy a comunicarme.", category: "Preparación y Planificación", isReverse: false }, { id: 22, text: "Suelo interrumpir el habla del otro, cuando creo que ya entendí el mensaje.", category: "Comportamiento en la Comunicación", isReverse: true }, { id: 23, text: "Las personas son obvias. Por eso es normal para mí presumir lo que van a decir.", category: "Comportamiento en la Comunicación", isReverse: true }, { id: 24, text: "Cuando alguien habla o hace algo que no me gusta, me pongo a la defensiva.", category: "Comportamiento en la Comunicación", isReverse: true }, { id: 25, text: "Me preocupo por mi comunicación. Por eso, suelo entrenar para ser un oyente eficiente.", category: "Comportamiento en la Comunicación", isReverse: false }, { id: 26, text: "Suelo anotar lo que el otro dice para no olvidar.", category: "Comportamiento en la Comunicación", isReverse: false }, { id: 27, text: "Permito que los sonidos me distraigan.", category: "Atención y Enfoque", isReverse: true }, { id: 28, text: "Escucho al otro sin hacer juicios o criticarlo. Solo comprendo.", category: "Retroalimentación", isReverse: false }, { id: 29, text: "Reafirmo y repito el mensaje, para tener certeza que entendí correctamente.", category: "Retroalimentación", isReverse: false }, { id: 30, text: "Practico empatía, parafraseando aquello que creo que son los sentimientos del otro.", category: "Retroalimentación", isReverse: false }];'

recommendations.js

'export const recommendations = { "Atención y Enfoque": { basic: [{ title: "Practica la meditación mindfulness", description: "Dedica 10 minutos diarios a ejercicios de atención plena", steps: ["Encuentra un lugar tranquilo", "Concéntrate en tu respiración", "Observa tus pensamientos sin juzgarlos", "Regresa suavemente tu atención cuando te distraigas"] }, { title: "Elimina distracciones", description: "Crea un ambiente propicio para la escucha activa", steps: ["Silencia tu teléfono durante conversaciones

importantes", "Escoge lugares tranquilos para reuniones", "Mantén tu espacio de trabajo ordenado", "Evita multitareas durante conversaciones"] }, { title: "Mantén contacto visual consciente", description: "Desarrolla una conexión visual apropiada", steps: ["Mira a los ojos de forma natural y cómoda", "Alterna entre los ojos y puntos cercanos al rostro", "Evita miradas fijas o intimidantes", "Asiente ocasionalmente para mostrar atención"] }], advanced: [{ title: "Técnicas de anclaje atencional", description: "Desarrolla puntos de referencia para mantener la atención", practices: ["Usa recordatorios físicos (como un objeto en tu escritorio)", "Establece palabras clave para reconectar", "Practica ejercicios de respiración consciente", "Desarrolla rutinas de atención plena"] }] },

```
"Comprensión y Procesamiento": {
  basic: [
    {
      title: "Practica el parafraseo activo",
      description: "Reformula lo escuchado para mejorar comprensión",
      steps: [
        "Escucha el mensaje completo",
        "Resume los puntos principales",
        "Usa tus propias palabras para reformular",
        "Verifica tu comprensión con el interlocutor"
      ]
    },
    {
      title: "Desarrollo de vocabulario contextual",
      description: "Mejora tu comprensión del lenguaje en diferentes
contextos",
      steps: [
        "Lee materiales variados sobre comunicación",
        "Aprende terminología específica de tu campo",
        "Practica el uso de sinónimos",
        "Estudia expresiones comunes y su significado"
      ]
    },
    {
      title: "Análisis de lenguaje no verbal",
      description: "Mejora tu capacidad de interpretar señales no
verbales",
      steps: [
        "Observa expresiones faciales",
        "Presta atención a la postura corporal",
        "Nota los gestos y movimientos",
        "Identifica patrones de comportamiento"
      ]
    }
  ],
  advanced: [
    {
      title: "Técnicas de procesamiento profundo",
      description: "Desarrolla una comprensión más profunda",
      practices: [
```

```
        "Crea mapas mentales de conversaciones",
        "Identifica patrones de comunicación",
        "Analiza subtextos y contextos",
        "Practica la escucha crítica"
      ]
    }
  ]
},

"Retroalimentación": {
  basic: [
    {
      title: "Implementa la técnica del espejo",
      description: "Refleja el mensaje y las emociones del interlocutor",
      steps: [
        "Escucha atentamente el mensaje",
        "Identifica la emoción subyacente",
        "Repite el mensaje principal",
        "Confirma tu interpretación"
      ]
    },
    {
      title: "Validación emocional",
      description: "Reconoce y valida los sentimientos del otro",
      steps: [
        "Identifica la emoción expresada",
        "Reconoce la legitimidad del sentimiento",
        "Muestra empatía genuina",
        "Ofrece apoyo apropiado"
      ]
    },
    {
      title: "Feedback constructivo",
      description: "Aprende a dar y recibir retroalimentación efectiva",
      steps: [
        "Sé específico y objetivo",
        "Enfócate en comportamientos, no personas",
        "Sugiere mejoras concretas",
        "Mantén una actitud positiva"
      ]
    }
  ],
  advanced: [
    {
      title: "Sistemas de retroalimentación avanzada",
      description: "Desarrolla métodos sofisticados de feedback",
      practices: [
        "Implementa ciclos de retroalimentación",
        "Desarrolla marcos de referencia",
        "Crea planes de acción específicos",
        "Establece métricas de mejora"
```

```
        ]
      }
    ]
  },

  "Preparación y Planificación": {
    basic: [
      {
        title: "Establecimiento de objetivos comunicativos",
        description: "Define claramente tus metas de comunicación",
        steps: [
          "Identifica el propósito principal",
          "Define resultados esperados",
          "Establece puntos clave a comunicar",
          "Prepara preguntas relevantes"
        ]
      },
      {
        title: "Creación de ambiente propicio",
        description: "Prepara el entorno para una comunicación efectiva",
        steps: [
          "Elige el momento adecuado",
          "Selecciona el lugar apropiado",
          "Minimiza distracciones potenciales",
          "Asegura la privacidad necesaria"
        ]
      },
      {
        title: "Planificación estratégica",
        description: "Desarrolla un plan de comunicación efectivo",
        steps: [
          "Anticipa posibles respuestas",
          "Prepara ejemplos y analogías",
          "Estructura tu mensaje",
          "Considera el tiempo necesario"
        ]
      }
    ],
    advanced: [
      {
        title: "Técnicas avanzadas de preparación",
        description: "Mejora tu planificación comunicativa",
        practices: [
          "Desarrolla escenarios alternativos",
          "Crea scripts de contingencia",
          "Implementa sistemas de seguimiento",
          "Establece protocolos de comunicación"
        ]
      }
    ]
  },
```

```
  "Comportamiento en la Comunicación": {
    basic: [
      {
        title: "Desarrollo de paciencia activa",
        description: "Mejora tu capacidad de escucha sin interrumpir",
        steps: [
          "Practica el silencio consciente",
          "Cuenta hasta tres antes de responder",
          "Respira profundamente cuando sientas urgencia de interrumpir",
          "Toma notas en lugar de interrumpir"
        ]
      },
      {
        title: "Gestión de sesgos",
        description: "Identifica y maneja tus prejuicios comunicativos",
        steps: [
          "Reconoce tus sesgos personales",
          "Cuestiona tus suposiciones",
          "Busca perspectivas diferentes",
          "Practica la apertura mental"
        ]
      },
      {
        title: "Cultivo de empatía",
        description: "Desarrolla una comprensión más profunda de los demás",
        steps: [
          "Practica la perspectiva del otro",
          "Reconoce emociones ajenas",
          "Muestra interés genuino",
          "Valida experiencias diferentes"
        ]
      }
    ],
    advanced: [
      {
        title: "Comunicación adaptativa avanzada",
        description: "Desarrolla habilidades de comunicación flexible",
        practices: [
          "Adapta tu estilo según el contexto",
          "Desarrolla múltiples aproximaciones",
          "Implementa técnicas de calibración",
          "Practica la comunicación situacional"
        ]
      }
    ]
  }
```

};

export const getRecommendationsForLevel = (category, score) => { const categoryData = recommendations[category]; const percentageScore = score / 100;

```
  if (percentageScore < 0.5) {
    return {
      level: "basic",
      recommendations: categoryData.basic
    };
  } else {
    return {
      level: "advanced",
      recommendations: [...categoryData.basic, ...categoryData.advanced]
    };
  }
```

};'

lib

utils.js

'// src/lib/utils.js import { clsx } from "clsx" import { twMerge } from "tailwind-merge"

export function cn(...inputs) { return twMerge(clsx(inputs)) }'

utils.ts

'import { clsx } from "clsx" import { twMerge } from "tailwind-merge"

export function cn(...inputs: any[]) { return twMerge(clsx(inputs)) }'

App.jsx

'import { ThemeProvider } from "./components/theme-provider"; import { ThemeToggle } from "./components/ThemeToggle"; import { EscuchaActivaQuiz } from './components/EscuchaActivaQuiz';

function App() { return (

Test de Escucha Activa

); }

export default App;'

main.jsx

'import React from 'react' import ReactDOM from 'react-dom/client' import App from './App.jsx' import './index.css'

ReactDOM.createRoot(document.getElementById('root')).render(<React.StrictMode> </React.StrictMode>,)'

App.css

'#root { max-width: 1280px; margin: 0 auto; padding: 2rem; text-align: center; }

.logo { height: 6em; padding: 1.5em; will-change: filter; transition: filter 300ms; } .logo:hover { filter: drop-shadow(0 0 2em #646cffaa); } .logo.react:hover { filter: drop-shadow(0 0 2em #61dafbaa); }

@keyframes logo-spin { from { transform: rotate(0deg); } to { transform: rotate(360deg); } }

@media (prefers-reduced-motion: no-preference) { a:nth-of-type(2) .logo { animation: logo-spin infinite 20s linear; } }

.card { padding: 2em; }

.read-the-docs { color: #888; }'

index.css

'@tailwind base; @tailwind components; @tailwind utilities;

@layer base { :root { --background: 0 0% 100%; --foreground: 222.2 84% 4.9%; --card: 0 0% 100%; --card-foreground: 222.2 84% 4.9%; --popover: 0 0% 100%; --popover-foreground: 222.2 84% 4.9%; --primary: 221.2 83.2% 53.3%; --primary-foreground: 210 40% 98%; --secondary: 210 40% 96.1%; --secondary-foreground: 222.2 47.4% 11.2%; --muted: 210 40% 96.1%; --muted-foreground: 215.4 16.3% 46.9%; --accent: 210 40% 96.1%; --accent-foreground: 222.2 47.4% 11.2%; --destructive: 0 84.2% 60.2%; --destructive-foreground: 210 40% 98%; --border: 214.3 31.8% 91.4%; --input: 214.3 31.8% 91.4%; --ring: 221.2 83.2% 53.3%; --radius: 0.5rem; }

.dark { --background: 222.2 84% 4.9%; --foreground: 210 40% 98%; --card: 222.2 84% 4.9%; --card-foreground: 210 40% 98%; --popover: 222.2 84% 4.9%; --popover-foreground: 210 40% 98%; --primary: 217.2 91.2% 59.8%; --primary-foreground: 222.2 47.4% 11.2%; --secondary: 217.2 32.6% 17.5%; --secondary-foreground: 210 40% 98%; --muted: 217.2 32.6% 17.5%; --muted-foreground: 215 20.2% 65.1%; --accent: 217.2 32.6% 17.5%; --accent-foreground: 210 40% 98%; --destructive: 0 62.8% 30.6%; --destructive-foreground: 210 40% 98%; --border: 217.2 32.6% 17.5%; --input: 217.2 32.6% 17.5%; --ring: 224.3 76.3% 48%; } }

@layer base {

- { @apply border-border; } body { @apply bg-background text-foreground; } }'