/mnt/data/cleaned_project/components.json

```
{
  "$schema": "https://ui.shadcn.com/schema.json",
  "style": "new-york",
  "rsc": true,
  "tsx": true,
  "tailwind": {
    "config": "",
    "css": "app/globals.css",
    "baseColor": "neutral",
    "cssVariables": true,
    "prefix": ""
  },
  "aliases": {
    "components": "@/components",
    "utils": "@/lib/utils",
    "ui": "@/components/ui",
    "lib": "@/lib",
    "hooks": "@/hooks"
  },
  "iconLibrary": "lucide"
}
```

/mnt/data/cleaned_project/next.config.mjs

```
/** @type {import('next').NextConfig} */
const nextConfig = {
  typescript: {
    ignoreBuildErrors: true,
  },
  images: {
    unoptimized: true,
  },

}

export default nextConfig
```

/mnt/data/cleaned_project/package.json

```
{
  "name": "my-v0-project",
  "version": "0.1.0",
  "private": true,
  "scripts": {
    "build": "next build",
    "dev": "next dev",
    "lint": "eslint .",
    "start": "next start"
  },
  "dependencies": {
    "@hookform/resolvers": "^3.10.0",
    "@radix-ui/react-accordion": "1.2.2",
    "@radix-ui/react-alert-dialog": "1.1.4",
    "@radix-ui/react-aspect-ratio": "1.1.1",
    "@radix-ui/react-avatar": "1.1.2",
    "@radix-ui/react-checkbox": "1.1.3",
    "@radix-ui/react-collapsible": "1.1.2",
    "@radix-ui/react-context-menu": "2.2.4",
    "@radix-ui/react-dialog": "1.1.4",
    "@radix-ui/react-dropdown-menu": "2.1.4",
    "@radix-ui/react-hover-card": "1.1.4",
    "@radix-ui/react-label": "2.1.1",
    "@radix-ui/react-menubar": "1.1.4",
    "@radix-ui/react-navigation-menu": "1.2.3",
    "@radix-ui/react-popover": "1.1.4",
    "@radix-ui/react-progress": "1.1.1",
    "@radix-ui/react-radio-group": "1.2.2",
```

```
    "@radix-ui/react-scroll-area": "1.2.2",
    "@radix-ui/react-select": "2.1.4",
    "@radix-ui/react-separator": "1.1.1",
    "@radix-ui/react-slider": "1.2.2",
    "@radix-ui/react-slot": "1.1.1",
    "@radix-ui/react-switch": "1.1.2",
    "@radix-ui/react-tabs": "1.1.2",
    "@radix-ui/react-toast": "1.2.4",
    "@radix-ui/react-toggle": "1.1.1",
    "@radix-ui/react-toggle-group": "1.1.1",
    "@radix-ui/react-tooltip": "1.1.6",
    "@vercel/analytics": "1.3.1",
    "autoprefixer": "^10.4.20",
    "class-variance-authority": "^0.7.1",
    "clsx": "^2.1.1",
    "cmdk": "1.0.4",
    "date-fns": "4.1.0",
    "embla-carousel-react": "8.5.1",
    "input-otp": "1.4.1",
    "lucide-react": "^0.454.0",
    "next": "16.0.10",
    "next-themes": "^0.4.6",
    "react": "19.2.0",
    "react-day-picker": "9.8.0",
    "react-dom": "19.2.0",
    "react-hook-form": "^7.60.0",
    "react-resizable-panels": "^2.1.7",
    "recharts": "2.15.4",
    "sonner": "^1.7.4",
    "tailwind-merge": "^3.3.1",
    "tailwindcss-animate": "^1.0.7",
    "vaul": "^1.1.2",
    "zod": "3.25.76"
  },
  "devDependencies": {
    "@tailwindcss/postcss": "^4.1.9",
    "@types/node": "^22",
    "@types/react": "^19",
    "@types/react-dom": "^19",
    "postcss": "^8.5",
    "tailwindcss": "^4.1.9",
    "tw-animate-css": "1.3.3",
    "typescript": "^5"
  }
}
```

/mnt/data/cleaned_project/postcss.config.mjs

```
/** @type {import('postcss-load-config').Config} */
const config = {
  plugins: {
    '@tailwindcss/postcss': {},
  },
}

export default config
```

/mnt/data/cleaned_project/tsconfig.json

```
{
  "compilerOptions": {
    "lib": ["dom", "dom.iterable", "esnext"],
    "allowJs": true,
    "target": "ES6",
    "skipLibCheck": true,
    "strict": true,
    "noEmit": true,
    "esModuleInterop": true,
    "module": "esnext",
    "moduleResolution": "bundler",
```

```
    "resolveJsonModule": true,
    "isolatedModules": true,
    "jsx": "preserve",
    "incremental": true,
    "plugins": [
      {
        "name": "next"
      }
    ],
    "paths": {
      "@/*": ["./*"]
    }
  },
  "include": ["next-env.d.ts", "**/*.ts", "**/*.tsx", ".next/types/**/*.ts"],
  "exclude": ["node_modules"]
}
```

/mnt/data/cleaned_project/app/layout.tsx

```
import React from "react"
import type { Metadata } from 'next'
import { Cairo } from 'next/font/google'
import { Analytics } from '@vercel/analytics/next'
import { LanguageProvider } from '@/contexts/language-context'
import './globals.css'

const cairo = Cairo({ subsets: ["arabic", "latin"] });

export const metadata: Metadata = {
  title: 'FLEX - ■■■■■ ■■■■■■■■',
  description: '■■■■■ FLEX ■■■■■■■■ ■■■■■■■■■■■ ■■■■■■■■',
  generator: 'v0.app',
  icons: {
    icon: [
      {
        url: '/icon-light-32x32.png',
        media: '(prefers-color-scheme: light)',
      },
      {
        url: '/icon-dark-32x32.png',
        media: '(prefers-color-scheme: dark)',
      },
      {
        url: '/icon.svg',
        type: 'image/svg+xml',
      },
    ],
    apple: '/apple-icon.png',
  },
}

export default function RootLayout({
  children,
}: Readonly<{
  children: React.ReactNode
}>) {
  return (
    <html lang="ar" dir="rtl">
      <body className={`${cairo.className} antialiased`}>
        <LanguageProvider>
          {children}
        </LanguageProvider>
        <Analytics />
      </body>
    </html>
  )
}
```

/mnt/data/cleaned_project/app/page.tsx

```
"use client";

import { useState } from "react";
import { Header } from "@/components/flex-app/header";
import { AnnouncementBar } from "@/components/flex-app/announcement-bar";
import { HeroBanner } from "@/components/flex-app/hero-banner";
import { QuickActions } from "@/components/flex-app/quick-actions";
import { MovieSection } from "@/components/flex-app/movie-section";
import { BottomNavigation } from "@/components/flex-app/bottom-navigation";
import { SubscriptionsPage } from "@/components/flex-app/subscriptions-page";
import { useLanguage } from "@/contexts/language-context";

const popularMovies = [
  { id: 1, title: "Avatar", image: "/images/movie1.jpg" },
  { id: 2, title: "The Great Flood", image: "/images/movie2.jpg" },
  { id: 3, title: "Five Nights at Freddy's", image: "/images/movie3.jpg" },
  { id: 4, title: "Zootopia 2", image: "/images/movie4.jpg" },
  { id: 5, title: "Now You See Me", image: "/images/movie5.jpg" },
  { id: 6, title: "SISU", image: "/images/movie6.jpg" },
];

const newMovies = [
  { id: 7, title: "The Matrix", image: "/images/movie7.jpg" },
  { id: 8, title: "Interstellar", image: "/images/movie8.jpg" },
  { id: 9, title: "Inception", image: "/images/movie9.jpg" },
  { id: 10, title: "Dune", image: "/images/movie10.jpg" },
  { id: 11, title: "Oppenheimer", image: "/images/movie11.jpg" },
  { id: 12, title: "Barbie", image: "/images/movie12.jpg" },
];

const upcomingMovies = [
  { id: 13, title: "Avatar 3", image: "/images/movie13.jpg" },
  { id: 14, title: "Deadpool 4", image: "/images/movie14.jpg" },
  { id: 15, title: "Spider-Man 4", image: "/images/movie15.jpg" },
  { id: 16, title: "Joker 2", image: "/images/movie16.jpg" },
  { id: 17, title: "Gladiator 2", image: "/images/movie17.jpg" },
  { id: 18, title: "Mission Impossible 8", image: "/images/movie18.jpg" },
];

export default function FlexApp() {
  const { t } = useLanguage();
  const [activeTab, setActiveTab] = useState("home");
  const [userPin, setUserPin] = useState<string | null>(null);

  const handleTabChange = (tab: string) => {
    setActiveTab(tab);
  };

  const handleSetPin = (pin: string) => {
    setUserPin(pin);
  };

  if (activeTab === "subscriptions") {
    return (
      <>
        <SubscriptionsPage
          onBack={() => setActiveTab("home")}
          userPin={userPin}
          onSetPin={handleSetPin}
        />
        <BottomNavigation activeTab={activeTab} onTabChange={handleTabChange} />
      </>
    );
  }

  return (
    <main className="min-h-screen bg-gradient-to-b from-background via-background to-violet-950/10 pb
      <Header />
      <AnnouncementBar />
      <HeroBanner />
      <QuickActions />

      <MovieSection title={t("popularMovies")} movies={popularMovies} />
      <MovieSection title={t("newMovies")} movies={newMovies} />
      <MovieSection title={t("upcomingMovies")} movies={upcomingMovies} />

      <BottomNavigation activeTab={activeTab} onTabChange={handleTabChange} />
```

```
    </main>
  );
}
```

/mnt/data/cleaned_project/components/theme-provider.tsx

```
'use client'

import * as React from 'react'
import {
  ThemeProvider as NextThemesProvider,
  type ThemeProviderProps,
} from 'next-themes'

export function ThemeProvider({ children, ...props }: ThemeProviderProps) {
  return <NextThemesProvider {...props}>{children}</NextThemesProvider>
}
```

/mnt/data/cleaned_project/components/flex-app/announcement-bar.tsx

```
"use client";

import { Megaphone } from "lucide-react";
import { useLanguage } from "@/contexts/language-context";

export function AnnouncementBar() {
  const { t } = useLanguage();

  return (
    <div className="mx-4 mt-4 rounded-xl bg-gradient-to-l from-violet-950/40 via-secondary/80 to-vio
      <div className="flex items-center gap-3 py-3 px-4">
        <Megaphone className="w-5 h-5 text-violet-400 flex-shrink-0" />
        <div className="overflow-hidden flex-1">
          <p className="animate-marquee whitespace-nowrap text-sm text-white">
            {t("announcement")}
          </p>
        </div>
      </div>
    </div>
  );
}
```

/mnt/data/cleaned_project/components/flex-app/bottom-navigation.tsx

```
"use client";

import { Home, Megaphone, History, User, Crown } from "lucide-react";
import { useLanguage } from "@/contexts/language-context";

type NavItem = {
  icon: typeof Home | null;
  labelKey: string;
  isCenter?: boolean;
  id: string;
};

const navItems: NavItem[] = [
  { icon: Home, labelKey: "home", id: "home" },
  { icon: Megaphone, labelKey: "promo", id: "promo" },
  { icon: null, labelKey: "subscriptions", isCenter: true, id: "subscriptions" },
  { icon: History, labelKey: "history", id: "history" },
  { icon: User, labelKey: "account", id: "account" },
];

interface BottomNavigationProps {
  activeTab: string;
  onTabChange: (tab: string) => void;
```

```
}

export function BottomNavigation({ activeTab, onTabChange }: BottomNavigationProps) {
  const { t } = useLanguage();

  return (
    <nav className="fixed bottom-0 left-0 right-0 bg-gradient-to-t from-background via-background to
      <div className="flex items-center justify-around py-2">
        {navItems.map((item) => (
          item.isCenter ? (
            <button
              key={item.id}
              onClick={() => onTabChange(item.id)}
              className="relative -mt-6 flex flex-col items-center gap-1"
            >
              <div className={`p-5 rounded-full bg-gradient-to-br from-violet-400 via-purple-500 to-
                activeTab === item.id ? 'scale-105' : ''
              }`}>
                <Crown className="w-7 h-7 text-white" />
              </div>
              <span className="text-xs font-medium text-violet-400">{t(item.labelKey)}</span>
            </button>
          ) : (
            <button
              key={item.id}
              onClick={() => onTabChange(item.id)}
              className={`flex flex-col items-center gap-1 px-4 py-2 transition-all duration-200 ${
                activeTab === item.id
                  ? "text-violet-400 scale-105"
                  : "text-muted-foreground hover:text-violet-500/70"
              }`}
            >
              {item.icon && <item.icon className={`w-5 h-5 transition-all ${activeTab === item.id ?
              <span className="text-xs font-medium">{t(item.labelKey)}</span>
              {activeTab === item.id && (
                <div className="absolute -bottom-1 w-1 h-1 rounded-full bg-violet-400" />
              )}
            </button>
          )
        ))}
      </div>
    </nav>
  );
}
```

/mnt/data/cleaned_project/components/flex-app/header.tsx

```
"use client";

import { Globe, Headphones, Bell } from "lucide-react";
import { useState, useEffect } from "react";
import { LanguageModal } from "./language-modal";
import { NotificationsModal } from "./notifications-modal";
import { useLanguage } from "@/contexts/language-context";

export function Header() {
  const [isLanguageOpen, setIsLanguageOpen] = useState(false);
  const [isNotificationsOpen, setIsNotificationsOpen] = useState(false);
  const [scrollY, setScrollY] = useState(0);
  const { t } = useLanguage();

  useEffect(() => {
    const handleScroll = () => setScrollY(window.scrollY);
    window.addEventListener("scroll", handleScroll);
    return () => window.removeEventListener("scroll", handleScroll);
  }, []);

  const opacity = Math.min(scrollY / 100, 0.9);

  return (
    <>
      <header
        className="sticky top-0 z-40 flex items-center justify-between px-4 py-3 backdrop-blur-md bo
        style={{ backgroundColor: `rgba(15, 10, 30, ${opacity})` }}
```

```
        >
          <div className="flex items-center gap-2">
            <button
              onClick={() => setIsNotificationsOpen(true)}
              className="p-2 rounded-full hover:bg-violet-500/20 transition-colors relative"
            >
              <Bell className="w-5 h-5 text-violet-400" />
              <span className="absolute top-1 right-1 w-2 h-2 bg-red-500 rounded-full" />
            </button>
            <button
              onClick={() => setIsLanguageOpen(true)}
              className="p-2 rounded-full hover:bg-violet-500/20 transition-colors"
            >
              <Globe className="w-5 h-5 text-violet-400" />
            </button>
            <button
              className="p-2 rounded-full hover:bg-violet-500/20 transition-colors"
            >
              <Headphones className="w-5 h-5 text-violet-400" />
            </button>
          </div>
          <h1 className="text-2xl font-bold bg-gradient-to-l from-violet-300 via-purple-400 to-violet-5
            {t("flex")}
          </h1>
        </header>

        <LanguageModal isOpen={isLanguageOpen} onClose={() => setIsLanguageOpen(false)} />
        <NotificationsModal isOpen={isNotificationsOpen} onClose={() => setIsNotificationsOpen(false)}
      </>
    );
  }
```

/mnt/data/cleaned_project/components/flex-app/hero-banner.tsx

```
"use client";

import Image from "next/image";
import { useEffect, useState } from "react";
import { useLanguage } from "@/contexts/language-context";

export function HeroBanner() {
  const [currentSlide, setCurrentSlide] = useState(0);
  const { t, dir } = useLanguage();

  const bannerSlides = [
    {
      image: "/images/banner.jpg",
      title: t("bannerTitle1"),
      subtitle: t("bannerSubtitle1")
    },
    {
      image: "/images/movie4.jpg",
      title: t("bannerTitle2"),
      subtitle: t("bannerSubtitle2")
    },
    {
      image: "/images/movie6.jpg",
      title: t("bannerTitle3"),
      subtitle: t("bannerSubtitle3")
    }
  ];

  useEffect(() => {
    const interval = setInterval(() => {
      setCurrentSlide((prev) => (prev + 1) % bannerSlides.length);
    }, 4000);
    return () => clearInterval(interval);
  }, [bannerSlides.length]);

  return (
    <div className="mx-4 mt-4">
      <div className="relative rounded-2xl overflow-hidden border-2 border-violet-500/30 shadow-lg sh
        <div
          className="flex transition-transform duration-700 ease-in-out"
```

```
            style={{ transform: `translateX(${dir === "rtl" ? "" : "-"}${currentSlide * 100}%)` }}
          >
            {bannerSlides.map((slide, index) => (
              <div key={index} className="w-full flex-shrink-0 relative">
                <Image
                  src={slide.image || "/placeholder.svg"}
                  alt={slide.title}
                  width={800}
                  height={400}
                  className="w-full h-48 object-cover"
                />
                <div className="absolute inset-0 bg-gradient-to-t from-black/90 via-black/40 to-transpa
                <div className="absolute top-4 right-4 bg-gradient-to-l from-violet-400 to-purple-500 
                  <span className="text-xs font-bold text-white">FLEX</span>
                </div>
                <div className={`absolute bottom-4 right-4 left-4 ${dir === "rtl" ? "text-right" : "tex
                  <h2 className="text-lg font-bold text-white mb-1">{slide.title}</h2>
                  <p className="text-xs text-white/80">{slide.subtitle}</p>
                </div>
              </div>
            ))}
          </div>
        </div>
        <div className="flex justify-center gap-1.5 mt-3">
          {bannerSlides.map((_, index) => (
            <button
              key={index}
              onClick={() => setCurrentSlide(index)}
              className={`h-1.5 rounded-full transition-all duration-300 ${
                currentSlide === index
                  ? 'w-8 bg-gradient-to-l from-violet-400 to-purple-500'
                  : 'w-1.5 bg-muted hover:bg-violet-500/50'
              }`}
            />
          ))}
        </div>
      </div>
    );
}
```

/mnt/data/cleaned_project/components/flex-app/language-modal.tsx

```
"use client";

import { X, Check, Languages } from "lucide-react";
import { useLanguage } from "@/contexts/language-context";

interface LanguageModalProps {
  isOpen: boolean;
  onClose: () => void;
}

export function LanguageModal({ isOpen, onClose }: LanguageModalProps) {
  const { language, setLanguage, t } = useLanguage();

  if (!isOpen) return null;

  const handleLanguageSelect = (lang: "ar" | "en") => {
    setLanguage(lang);
    onClose();
  };

  return (
    <div className="fixed inset-0 z-50 flex items-center justify-center bg-black/60 backdrop-blur-sm"
      <div className="bg-card border border-violet-500/30 rounded-2xl w-[85%] max-w-sm overflow-hidde
        <div className="flex items-center justify-between p-4 border-b border-violet-500/20">
          <h2 className="text-lg font-bold text-violet-400">{t("selectLanguage")}</h2>
          <button
            onClick={onClose}
            className="p-2 rounded-full hover:bg-violet-500/20 transition-colors"
          >
            <X className="w-5 h-5 text-foreground" />
          </button>
        </div>
```

```
        <div className="p-4 flex flex-col gap-2">
          <button
            onClick={() => handleLanguageSelect("ar")}
            className={`flex items-center justify-between p-4 rounded-xl transition-all ${
              language === "ar"
                ? "bg-violet-500/20 border-2 border-violet-500"
                : "bg-secondary/50 border-2 border-transparent hover:border-violet-500/30"
            }`}
          >
            <div className="flex items-center gap-3">
              <div className="w-8 h-8 rounded-full bg-violet-500/20 flex items-center justify-center
                <Languages className="w-4 h-4 text-violet-400" />
              </div>
              <span className="font-medium text-foreground">{t("arabic")}</span>
            </div>
            {language === "ar" && <Check className="w-5 h-5 text-violet-400" />}
          </button>

          <button
            onClick={() => handleLanguageSelect("en")}
            className={`flex items-center justify-between p-4 rounded-xl transition-all ${
              language === "en"
                ? "bg-violet-500/20 border-2 border-violet-500"
                : "bg-secondary/50 border-2 border-transparent hover:border-violet-500/30"
            }`}
          >
            <div className="flex items-center gap-3">
              <div className="w-8 h-8 rounded-full bg-violet-500/20 flex items-center justify-center
                <Languages className="w-4 h-4 text-violet-400" />
              </div>
              <span className="font-medium text-foreground">{t("english")}</span>
            </div>
            {language === "en" && <Check className="w-5 h-5 text-violet-400" />}
          </button>
        </div>
      </div>
    </div>
  );
}
```

/mnt/data/cleaned_project/components/flex-app/movie-section.tsx

```
"use client";

import Image from "next/image";
import { ChevronLeft, ChevronRight, Play } from "lucide-react";
import { useLanguage } from "@/contexts/language-context";

interface Movie {
  id: number;
  title: string;
  image: string;
}

interface MovieSectionProps {
  title: string;
  movies: Movie[];
}

export function MovieSection({ title, movies }: MovieSectionProps) {
  const { t, dir } = useLanguage();
  const ChevronIcon = dir === "rtl" ? ChevronLeft : ChevronRight;

  return (
    <section className="mt-6">
      <div className="flex items-center justify-between px-4 mb-3">
        <h3 className="text-lg font-bold text-white">{title}</h3>
        <button className="flex items-center gap-1 text-sm text-white/80 hover:text-white transition
          <span>{t("viewMore")}</span>
          <ChevronIcon className="w-4 h-4" />
        </button>
      </div>
      <div className="grid grid-cols-3 gap-3 px-4">
        {movies.map((movie) => (
```

```
            <button
              key={movie.id}
              className="group cursor-pointer text-center transition-all duration-300 hover:scale-[1.0
            >
              <div className="relative rounded-xl overflow-hidden shadow-lg transition-all duration-300
                <Image
                  src={movie.image || "/placeholder.svg"}
                  alt={movie.title}
                  width={128}
                  height={180}
                  className="w-full h-40 object-cover group-hover:scale-105 transition-transform durati
                />
                <div className="absolute inset-0 bg-gradient-to-t from-black/80 via-black/20 to-transpa
                  <div className="bg-violet-500/90 rounded-full p-2">
                    <Play className="w-6 h-6 text-white fill-white" />
                  </div>
                </div>
              </div>
              <p className="mt-2 text-xs text-center text-white font-medium truncate">{movie.title}</p
            </button>
          ))}
        </div>
      </section>
    );
  }
```

/mnt/data/cleaned_project/components/flex-app/notifications-modal.tsx

```
"use client";

import { X, ArrowDownCircle, ArrowUpCircle } from "lucide-react";
import { useLanguage } from "@/contexts/language-context";

interface Notification {
  id: number;
  type: "withdrawal" | "deposit";
  amount: string;
  date: string;
  status: "pending" | "completed" | "rejected";
}

const sampleNotifications: Notification[] = [
  { id: 1, type: "deposit", amount: "$500", date: "2026-02-04", status: "completed" },
  { id: 2, type: "withdrawal", amount: "$200", date: "2026-02-03", status: "pending" },
  { id: 3, type: "deposit", amount: "$1000", date: "2026-02-02", status: "completed" },
  { id: 4, type: "withdrawal", amount: "$150", date: "2026-02-01", status: "rejected" },
];

interface NotificationsModalProps {
  isOpen: boolean;
  onClose: () => void;
}

export function NotificationsModal({ isOpen, onClose }: NotificationsModalProps) {
  const { t, dir } = useLanguage();

  if (!isOpen) return null;

  const getStatusColor = (status: string) => {
    switch (status) {
      case "completed": return "text-green-400";
      case "pending": return "text-yellow-400";
      case "rejected": return "text-red-400";
      default: return "text-foreground";
    }
  };

  const getStatusText = (status: string) => {
    switch (status) {
      case "completed": return dir === "rtl" ? "[redacted]" : "Completed";
      case "pending": return dir === "rtl" ? "[redacted]" : "Pending";
      case "rejected": return dir === "rtl" ? "[redacted]" : "Rejected";
      default: return status;
    }
```

```
  };

  return (
    <div className="fixed inset-0 z-50 flex items-center justify-center bg-black/60 backdrop-blur-sm
      <div className="bg-card border border-violet-500/30 rounded-2xl w-[90%] max-w-md max-h-[70vh] o
        <div className="flex items-center justify-between p-4 border-b border-violet-500/20">
          <h2 className="text-lg font-bold text-violet-400">{t("notifications")}</h2>
          <button
            onClick={onClose}
            className="p-2 rounded-full hover:bg-violet-500/20 transition-colors"
          >
            <X className="w-5 h-5 text-foreground" />
          </button>
        </div>

        <div className="overflow-y-auto max-h-[50vh] p-4">
          {sampleNotifications.length === 0 ? (
            <p className="text-center text-muted-foreground py-8">{t("noNotifications")}</p>
          ) : (
            <div className="flex flex-col gap-3">
              {sampleNotifications.map((notification) => (
                <div
                  key={notification.id}
                  className="flex items-center gap-3 p-3 rounded-xl bg-secondary/50 border border-vio
                >
                  <div className={`p-2 rounded-full ${notification.type === "deposit" ? "bg-green-500
                    {notification.type === "deposit" ? (
                      <ArrowDownCircle className="w-5 h-5 text-green-400" />
                    ) : (
                      <ArrowUpCircle className="w-5 h-5 text-red-400" />
                    )}
                  </div>
                  <div className="flex-1">
                    <p className="font-medium text-foreground">
                      {notification.type === "deposit" ? t("deposit") : t("withdrawal")}
                    </p>
                    <p className="text-sm text-muted-foreground">{notification.date}</p>
                  </div>
                  <div className="text-left">
                    <p className="font-bold text-foreground">{notification.amount}</p>
                    <p className={`text-xs ${getStatusColor(notification.status)}`}>
                      {getStatusText(notification.status)}
                    </p>
                  </div>
                </div>
              ))}
            </div>
          )}
        </div>
      </div>
    </div>
  );
}
```

/mnt/data/cleaned_project/components/flex-app/pin-modal.tsx

```
"use client";

import { useState, useRef, useEffect } from "react";
import { X } from "lucide-react";
import { useLanguage } from "@/contexts/language-context";

interface PinModalProps {
  isOpen: boolean;
  onClose: () => void;
  onSubmit: (pin: string) => void;
}

export function PinModal({ isOpen, onClose, onSubmit }: PinModalProps) {
  const { t } = useLanguage();
  const [pin, setPin] = useState<string[]>(["", "", "", "", "", ""]);
  const inputRefs = useRef<(HTMLInputElement | null)[]>([]);

  useEffect(() => {
```

```
  if (isOpen && inputRefs.current[0]) {
    inputRefs.current[0].focus();
  }
}, [isOpen]);

useEffect(() => {
  if (!isOpen) {
    setPin(["", "", "", "", "", ""]);
  }
}, [isOpen]);

if (!isOpen) return null;

const handleChange = (index: number, value: string) => {
  if (!/^\d*$/.test(value)) return;

  const newPin = [...pin];
  newPin[index] = value.slice(-1);
  setPin(newPin);

  if (value && index < 5) {
    inputRefs.current[index + 1]?.focus();
  }
};

const handleKeyDown = (index: number, e: React.KeyboardEvent) => {
  if (e.key === "Backspace" && !pin[index] && index > 0) {
    inputRefs.current[index - 1]?.focus();
  }
};

const handleSubmit = () => {
  const fullPin = pin.join("");
  if (fullPin.length === 6) {
    onSubmit(fullPin);
  }
};

const isComplete = pin.every((digit) => digit !== "");

return (
  <div className="fixed inset-0 z-50 flex items-center justify-center bg-black/70 backdrop-blur-sm
    <div className="bg-card border border-violet-500/30 rounded-2xl w-[90%] max-w-sm overflow-hidde
      <div className="flex items-center justify-between p-4 border-b border-violet-500/20">
        <h2 className="text-lg font-bold text-violet-400">{t("enterSecretCode")}</h2>
        <button
          onClick={onClose}
          className="p-2 rounded-full hover:bg-violet-500/20 transition-colors"
        >
          <X className="w-5 h-5 text-foreground" />
        </button>
      </div>

      <div className="p-6">
        <p className="text-center text-muted-foreground mb-6">{t("secretCodeDescription")}</p>

        <div className="flex justify-center gap-2 mb-6">
          {pin.map((digit, index) => (
            <input
              key={index}
              ref={(el) => { inputRefs.current[index] = el; }}
              type="text"
              inputMode="numeric"
              maxLength={1}
              value={digit}
              onChange={(e) => handleChange(index, e.target.value)}
              onKeyDown={(e) => handleKeyDown(index, e)}
              className="w-12 h-14 text-center text-2xl font-bold bg-secondary/50 border-2 border-v
            />
          ))}
        </div>

        <div className="flex gap-3">
          <button
            onClick={onClose}
            className="flex-1 py-3 rounded-full border-2 border-violet-500/30 text-foreground font
          >
```

```
              {t("cancel")}
            </button>
            <button
              onClick={handleSubmit}
              disabled={!isComplete}
              className={`flex-1 py-3 rounded-full font-medium transition-all ${
                isComplete
                  ? "bg-gradient-to-r from-violet-500 to-purple-600 text-white hover:from-violet-600
                  : "bg-gray-700/50 text-gray-400 cursor-not-allowed"
              }`}
            >
              {t("confirm")}
            </button>
          </div>
        </div>
      </div>
    </div>
  );
}
```

/mnt/data/cleaned_project/components/flex-app/quick-actions.tsx

```
"use client";

import { UserPlus, Crown, BookOpen, Users } from "lucide-react";
import { useState } from "react";
import { useLanguage } from "@/contexts/language-context";

const actions = [
  { icon: Users, labelKey: "partner", id: "partner" },
  { icon: BookOpen, labelKey: "intro", id: "intro" },
  { icon: Crown, labelKey: "membership", id: "membership" },
  { icon: UserPlus, labelKey: "invite", id: "invite" },
];

export function QuickActions() {
  const [activeAction, setActiveAction] = useState<string | null>(null);
  const { t } = useLanguage();

  const handleActionClick = (id: string) => {
    setActiveAction(id);
    setTimeout(() => setActiveAction(null), 400);
  };

  return (
    <div className="flex justify-around px-4 mt-6">
      {actions.map((action) => (
        <button
          key={action.id}
          onClick={() => handleActionClick(action.id)}
          className="flex flex-col items-center gap-2 group"
        >
          <div className={`p-4 rounded-full bg-gradient-to-br from-violet-400 via-purple-500 to-viol
            activeAction === action.id
              ? 'scale-90 shadow-violet-500/60'
              : 'group-hover:scale-110 group-hover:shadow-violet-500/50'
          }`}>
            <action.icon className="w-6 h-6 text-white" />
          </div>
          <span className="text-xs text-white font-medium">{t(action.labelKey)}</span>
        </button>
      ))}
    </div>
  );
}
```

/mnt/data/cleaned_project/components/flex-app/subscriptions-page.tsx

```
"use client";
import { ArrowLeft, Gift, ChevronLeft } from "lucide-react";
```

```tsx
import { useLanguage } from "@/contexts/language-context";
import { useState } from "react";
import { PinModal } from "./pin-modal";
import { TasksPage } from "./tasks-page";

interface SubscriptionsPageProps {
  onBack: () => void;
  userPin: string | null;
  onSetPin: (pin: string) => void;
}

const vipTiers = [
  { level: 1, income: 0.5, days: 75, times: 4, price: 50, available: true },
  { level: 2, income: 0.8, days: 75, times: 10, price: 200, available: true },
  { level: 3, income: 1.0, days: 75, times: 20, price: 0, available: false },
  { level: 4, income: 1.5, days: 75, times: 30, price: 0, available: false },
  { level: 5, income: 2.0, days: 75, times: 40, price: 0, available: false },
  { level: 6, income: 2.5, days: 75, times: 50, price: 0, available: false },
  { level: 7, income: 3.0, days: 75, times: 60, price: 0, available: false },
  { level: 8, income: 3.5, days: 75, times: 70, price: 0, available: false },
];

export function SubscriptionsPage({ onBack, userPin, onSetPin }: SubscriptionsPageProps) {
  const { t } = useLanguage();
  const [currentLevel, setCurrentLevel] = useState<number | null>(null);
  const [showPinModal, setShowPinModal] = useState(false);
  const [pendingActivation, setPendingActivation] = useState<number | null>(null);
  const [showTasksPage, setShowTasksPage] = useState(false);
  const [activeVipLevel, setActiveVipLevel] = useState<number | null>(null);

  const handleActivate = (level: number) => {
    setPendingActivation(level);
    setShowPinModal(true);
  };

  const handlePinSubmit = (pin: string) => {
    if (!userPin) {
      // First time - set the PIN
      onSetPin(pin);
    }

    // Activate the VIP level
    if (pendingActivation) {
      setCurrentLevel(pendingActivation);
      setActiveVipLevel(pendingActivation);
      setShowPinModal(false);
      // Navigate to tasks page
      setShowTasksPage(true);
    }
  };

  const handleTasksBack = () => {
    setShowTasksPage(false);
  };

  if (showTasksPage && activeVipLevel) {
    const tier = vipTiers.find(t => t.level === activeVipLevel);
    return (
      <TasksPage
        onBack={handleTasksBack}
        vipLevel={activeVipLevel}
        dailyTasks={tier?.times || 4}
      />
    );
  }

  return (
    <div className="min-h-screen bg-background pb-24">
      {/* Header */}
      <header className="sticky top-0 z-40 flex items-center justify-between px-4 py-4 bg-background/
        <button
          onClick={onBack}
          className="p-2 rounded-full hover:bg-violet-500/20 transition-colors"
        >
          <ArrowLeft className="w-5 h-5 text-foreground" />
        </button>
        <h1 className="text-lg font-bold text-foreground">{t("membersCenter")}</h1>
```

```
          <div className="w-9" />
        </header>

        <div className="px-4 py-4">
          {/* Current Level */}
          <div className="flex items-center gap-2 mb-4">
            <span className="text-muted-foreground">{t("currentLevel")}:</span>
            <div className="flex items-center gap-1">
              <Gift className="w-5 h-5 text-violet-400" />
              <span className="text-violet-400 font-bold">
                {currentLevel ? `VIP${currentLevel}` : t("noLevel")}
              </span>
            </div>
          </div>

          {/* Stats Cards */}
          <div className="grid grid-cols-2 gap-4 mb-6">
            <div className="bg-card border border-violet-500/20 rounded-xl p-4 text-center">
              <p className="text-2xl font-bold text-foreground">0.00</p>
              <p className="text-xs text-muted-foreground mt-1">(USDT){t("todayProfits")}</p>
            </div>
            <div className="bg-card border border-violet-500/20 rounded-xl p-4 text-center">
              <p className="text-2xl font-bold text-foreground">0.00</p>
              <p className="text-xs text-muted-foreground mt-1">(USDT){t("cumulativeIncome")}</p>
            </div>
          </div>

          {/* Special Gift Package Section */}
          <h2 className="text-lg font-bold text-foreground mb-4">{t("specialGiftPackage")}</h2>

          {/* VIP Tiers */}
          <div className="flex flex-col gap-4">
            {vipTiers.map((tier) => (
              <div
                key={tier.level}
                className={`bg-card border border-violet-500/20 rounded-xl p-4 ${!tier.available ? 'opa
              >
                {/* VIP Level Header */}
                <div className="flex items-center justify-end gap-2 mb-4">
                  <span className="font-bold text-foreground">VIP{tier.level}</span>
                  <div className="w-2 h-2 rounded-full bg-violet-500" />
                </div>

                {/* Stats Row */}
                <div className="grid grid-cols-3 gap-2 mb-4 text-center">
                  <div>
                    <p className="text-lg font-bold text-sky-400">{tier.income} <span className="text-x
```

/mnt/data/cleaned_project/components/flex-app/tasks-page.tsx

```
"use client";

import { useState, useRef } from "react";
import { ArrowLeft, Heart, ChevronLeft, Volume2, VolumeX, Play, Pause } from "lucide-react";
import { useLanguage } from "@/contexts/language-context";

interface TasksPageProps {
  onBack: () => void;
  vipLevel: number;
  dailyTasks: number;
}

const videos = [
  {
    id: 1,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/BigBuckBunny.mp4",
    title: "Adventure Awaits",
    username: "@explorer_mike",
  },
  {
    id: 2,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/ElephantsDream.mp4",
    title: "Nature's Beauty",
    username: "@nature_lover",
```

```
  },
  {
    id: 3,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/ForBiggerBlazes.mp4",
    title: "City Vibes",
    username: "@urban_explorer",
  },
  {
    id: 4,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/ForBiggerEscapes.mp4",
    title: "Amazing Journey",
    username: "@traveler_jane",
  },
  {
    id: 5,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/ForBiggerFun.mp4",
    title: "Creative Moments",
    username: "@creative_soul",
  },
  {
    id: 6,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/ForBiggerJoyrides.mp4",
    title: "Epic Story",
    username: "@story_teller",
  },
  {
    id: 7,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/ForBiggerMeltdowns.mp4",
    title: "Cinematic View",
    username: "@cinema_pro",
  },
  {
    id: 8,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/Sintel.mp4",
    title: "Action Packed",
    username: "@action_fan",
  },
  {
    id: 9,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/SubaruOutbackOnStreetAnd
    title: "Road Trip",
    username: "@road_warrior",
  },
  {
    id: 10,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/TearsOfSteel.mp4",
    title: "Sci-Fi Dreams",
    username: "@scifi_lover",
  },
  {
    id: 11,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/VolkswagenGTIReview.mp4",
    title: "Speed Review",
    username: "@car_reviews",
  },
  {
    id: 12,
    url: "https://commondatastorage.googleapis.com/gtv-videos-bucket/sample/WeAreGoingOnBullrun.mp4",
    title: "Racing Time",
    username: "@race_master",
  },
];

export function TasksPage({ onBack, vipLevel, dailyTasks }: TasksPageProps) {
  const { t } = useLanguage();
  const [currentVideoIndex, setCurrentVideoIndex] = useState(0);
  const [remainingTasks, setRemainingTasks] = useState(dailyTasks);
  const [likedVideos, setLikedVideos] = useState<Set<number>>(new Set());
  const [isMuted, setIsMuted] = useState(false);
  const [isPlaying, setIsPlaying] = useState(true);
  const [showTaskComplete, setShowTaskComplete] = useState(false);
  const videoRef = useRef<HTMLVideoElement>(null);

  const currentVideo = videos[currentVideoIndex % videos.length];

  const handleLike = () => {
    if (likedVideos.has(currentVideo.id) || remainingTasks <= 0) return;
```

```
    setLikedVideos(new Set([...likedVideos, currentVideo.id]));
    setRemainingTasks((prev) => prev - 1);
    setShowTaskComplete(true);

    setTimeout(() => {
      setShowTaskComplete(false);
      if (currentVideoIndex < videos.length - 1 && remainingTasks > 1) {
        setCurrentVideoIndex((prev) => prev + 1);
      }
    }, 1500);
  };

  const handleSwipe = (direction: "up" | "down") => {
    if (direction === "up" && currentVideoIndex < videos.length - 1) {
      setCurrentVideoIndex((prev) => prev + 1);
    } else if (direction === "down" && currentVideoIndex > 0) {
      setCurrentVideoIndex((prev) => prev - 1);
    }
  };

  const toggleMute = () => {
    setIsMuted(!isMuted);
    if (videoRef.current) {
      videoRef.current.muted = !isMuted;
    }
  };

  const togglePlay = () => {
    if (videoRef.current) {
      if (isPlaying) {
        videoRef.current.pause();
      } else {
        videoRef.current.play();
      }
      setIsPlaying(!isPlaying);
    }
  };

  const allTasksCompleted = remainingTasks <= 0;

  return (
    <div className="fixed inset-0 bg-black z-50 flex flex-col">
      {/* Header */}
      <header className="flex items-center justify-between px-4 py-4 bg-gradient-to-b from-black/80 t
        <button
          onClick={onBack}
          className="p-2 rounded-full bg-black/30 hover:bg-black/50 transition-colors"
        >
          <ArrowLeft className="w-5 h-5 text-white" />
        </button>
        <div className="flex items-center gap-2 bg-black/30 px-4 py-2 rounded-full">
          <span className="text-white font-bold">VIP{vipLevel}</span>
          <span className="text-violet-400">|</span>
          <span className="text-white">{t("tasksRemaining")}: <span className="text-violet-400 font-b
        </div>
```

/mnt/data/cleaned_project/components/ui/accordion.tsx

```
'use client'

import * as React from 'react'
import * as AccordionPrimitive from '@radix-ui/react-accordion'
import { ChevronDownIcon } from 'lucide-react'

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

function Accordion({
  ...props
}: React.ComponentProps<typeof AccordionPrimitive.Root>) {
  return <AccordionPrimitive.Root data-slot="accordion" {...props} />
}

function AccordionItem({
  className,
```

```
  ...props
}: React.ComponentProps<typeof AccordionPrimitive.Item>) {
  return (
    <AccordionPrimitive.Item
      data-slot="accordion-item"
      className={cn('border-b last:border-b-0', className)}
      {...props}
    />
  )
}

function AccordionTrigger({
  className,
  children,
  ...props
}: React.ComponentProps<typeof AccordionPrimitive.Trigger>) {
  return (
    <AccordionPrimitive.Header className="flex">
      <AccordionPrimitive.Trigger
        data-slot="accordion-trigger"
        className={cn(
          'focus-visible:border-ring focus-visible:ring-ring/50 flex flex-1 items-start justify-betwe
          className,
        )}
        {...props}
      >
        {children}
        <ChevronDownIcon className="text-muted-foreground pointer-events-none size-4 shrink-0 transla
      </AccordionPrimitive.Trigger>
    </AccordionPrimitive.Header>
  )
}

function AccordionContent({
  className,
  children,
  ...props
}: React.ComponentProps<typeof AccordionPrimitive.Content>) {
  return (
    <AccordionPrimitive.Content
      data-slot="accordion-content"
      className="data-[state=closed]:animate-accordion-up data-[state=open]:animate-accordion-down ov
      {...props}
    >
      <div className={cn('pt-0 pb-4', className)}>{children}</div>
    </AccordionPrimitive.Content>
  )
}

export { Accordion, AccordionItem, AccordionTrigger, AccordionContent }
```

/mnt/data/cleaned_project/components/ui/alert-dialog.tsx

```
'use client'

import * as React from 'react'
import * as AlertDialogPrimitive from '@radix-ui/react-alert-dialog'

import { cn } from '@/lib/utils'
import { buttonVariants } from '@/components/ui/button'

function AlertDialog({
  ...props
}: React.ComponentProps<typeof AlertDialogPrimitive.Root>) {
  return <AlertDialogPrimitive.Root data-slot="alert-dialog" {...props} />
}

function AlertDialogTrigger({
  ...props
}: React.ComponentProps<typeof AlertDialogPrimitive.Trigger>) {
  return (
    <AlertDialogPrimitive.Trigger data-slot="alert-dialog-trigger" {...props} />
  )
}
```

```
function AlertDialogPortal({
  ...props
}: React.ComponentProps<typeof AlertDialogPrimitive.Portal>) {
  return (
    <AlertDialogPrimitive.Portal data-slot="alert-dialog-portal" {...props} />
  )
}

function AlertDialogOverlay({
  className,
  ...props
}: React.ComponentProps<typeof AlertDialogPrimitive.Overlay>) {
  return (
    <AlertDialogPrimitive.Overlay
      data-slot="alert-dialog-overlay"
      className={cn(
        'data-[state=open]:animate-in data-[state=closed]:animate-out data-[state=closed]:fade-out-0
        className,
      )}
      {...props}
    />
  )
}

function AlertDialogContent({
  className,
  ...props
}: React.ComponentProps<typeof AlertDialogPrimitive.Content>) {
  return (
    <AlertDialogPortal>
      <AlertDialogOverlay />
      <AlertDialogPrimitive.Content
        data-slot="alert-dialog-content"
        className={cn(
          'bg-background data-[state=open]:animate-in data-[state=closed]:animate-out data-[state=cl
          className,
        )}
        {...props}
      />
    </AlertDialogPortal>
  )
}

function AlertDialogHeader({
  className,
  ...props
}: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="alert-dialog-header"
      className={cn('flex flex-col gap-2 text-center sm:text-left', className)}
      {...props}
    />
  )
}

function AlertDialogFooter({
  className,
  ...props
}: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="alert-dialog-footer"
      className={cn(
        'flex flex-col-reverse gap-2 sm:flex-row sm:justify-end',
        className,
      )}
      {...props}
    />
  )
}

function AlertDialogTitle({
  className,
  ...props
}: React.ComponentProps<typeof AlertDialogPrimitive.Title>) {
  return (
```

```
    <AlertDialogPrimitive.Title
      data-slot="alert-dialog-title"
      className={cn('text-lg font-semibold', className)}
      {...props}
    />
  )
}

function AlertDialogDescription({
  className,
  ...props
}: React.ComponentProps<typeof AlertDialogPrimitive.Description>) {
  return (
    <AlertDialogPrimitive.Description
      data-slot="alert-dialog-description"
      className={cn('text-muted-foreground text-sm', className)}
      {...props}
    />
  )
}

function AlertDialogAction({
  className,
  ...props
}: React.ComponentProps<typeof AlertDialogPrimitive.Action>) {
  return (
    <AlertDialogPrimitive.Action
      className={cn(buttonVariants(), className)}
      {...props}
    />
  )
}

function AlertDialogCancel({
  className,
  ...props
}: React.ComponentProps<typeof AlertDialogPrimitive.Cancel>) {
  return (
    <AlertDialogPrimitive.Cancel
      className={cn(buttonVariants({ variant: 'outline' }), className)}
      {...props}
    />
  )
}

export {
  AlertDialog,
  AlertDialogPortal,
  AlertDialogOverlay,
  AlertDialogTrigger,
  AlertDialogContent,
  AlertDialogHeader,
  AlertDialogFooter,
  AlertDialogTitle,
  AlertDialogDescription,
  AlertDialogAction,
  AlertDialogCancel,
}
```

/mnt/data/cleaned_project/components/ui/alert.tsx

```
import * as React from 'react'
import { cva, type VariantProps } from 'class-variance-authority'

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

const alertVariants = cva(
  'relative w-full rounded-lg border px-4 py-3 text-sm grid has-[>svg]:grid-cols-[calc(var(--spacing
  {
    variants: {
      variant: {
        default: 'bg-card text-card-foreground',
        destructive:
          'text-destructive bg-card [&>svg]:text-current *:data-[slot=alert-description]:text-destruc
```

```
      },
    },
    defaultVariants: {
      variant: 'default',
    },
  },
)

function Alert({
  className,
  variant,
  ...props
}: React.ComponentProps<'div'> & VariantProps<typeof alertVariants>) {
  return (
    <div
      data-slot="alert"
      role="alert"
      className={cn(alertVariants({ variant }), className)}
      {...props}
    />
  )
}

function AlertTitle({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="alert-title"
      className={cn(
        'col-start-2 line-clamp-1 min-h-4 font-medium tracking-tight',
        className,
      )}
      {...props}
    />
  )
}

function AlertDescription({
  className,
  ...props
}: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="alert-description"
      className={cn(
        'text-muted-foreground col-start-2 grid justify-items-start gap-1 text-sm [&_p]:leading-rela
        className,
      )}
      {...props}
    />
  )
}

export { Alert, AlertTitle, AlertDescription }
```

/mnt/data/cleaned_project/components/ui/aspect-ratio.tsx

```
'use client'

import * as AspectRatioPrimitive from '@radix-ui/react-aspect-ratio'

function AspectRatio({
  ...props
}: React.ComponentProps<typeof AspectRatioPrimitive.Root>) {
  return <AspectRatioPrimitive.Root data-slot="aspect-ratio" {...props} />
}

export { AspectRatio }
```

/mnt/data/cleaned_project/components/ui/avatar.tsx

```
'use client'

import * as React from 'react'
import * as AvatarPrimitive from '@radix-ui/react-avatar'

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

function Avatar({
  className,
  ...props
}: React.ComponentProps<typeof AvatarPrimitive.Root>) {
  return (
    <AvatarPrimitive.Root
      data-slot="avatar"
      className={cn(
        'relative flex size-8 shrink-0 overflow-hidden rounded-full',
        className,
      )}
      {...props}
    />
  )
}

function AvatarImage({
  className,
  ...props
}: React.ComponentProps<typeof AvatarPrimitive.Image>) {
  return (
    <AvatarPrimitive.Image
      data-slot="avatar-image"
      className={cn('aspect-square size-full', className)}
      {...props}
    />
  )
}

function AvatarFallback({
  className,
  ...props
}: React.ComponentProps<typeof AvatarPrimitive.Fallback>) {
  return (
    <AvatarPrimitive.Fallback
      data-slot="avatar-fallback"
      className={cn(
        'bg-muted flex size-full items-center justify-center rounded-full',
        className,
      )}
      {...props}
    />
  )
}

export { Avatar, AvatarImage, AvatarFallback }
```

/mnt/data/cleaned_project/components/ui/badge.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 badgeVariants = cva(
  'inline-flex items-center justify-center rounded-md border px-2 py-0.5 text-xs font-medium w-fit wh
  {
    variants: {
      variant: {
        default:
          'border-transparent bg-primary text-primary-foreground [a&]:hover:bg-primary/90',
        secondary:
          'border-transparent bg-secondary text-secondary-foreground [a&]:hover:bg-secondary/90',
        destructive:
          'border-transparent bg-destructive text-white [a&]:hover:bg-destructive/90 focus-visible:ri
        outline:
```

```
          'text-foreground [a&]:hover:bg-accent [a&]:hover:text-accent-foreground',
      },
    },
    defaultVariants: {
      variant: 'default',
    },
  },
)

function Badge({
  className,
  variant,
  asChild = false,
  ...props
}: React.ComponentProps<'span'> &
  VariantProps<typeof badgeVariants> & { asChild?: boolean }) {
  const Comp = asChild ? Slot : 'span'

  return (
    <Comp
      data-slot="badge"
      className={cn(badgeVariants({ variant }), className)}
      {...props}
    />
  )
}

export { Badge, badgeVariants }
```

/mnt/data/cleaned_project/components/ui/breadcrumb.tsx

```
import * as React from 'react'
import { Slot } from '@radix-ui/react-slot'
import { ChevronRight, MoreHorizontal } from 'lucide-react'

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

function Breadcrumb({ ...props }: React.ComponentProps<'nav'>) {
  return <nav aria-label="breadcrumb" data-slot="breadcrumb" {...props} />
}

function BreadcrumbList({ className, ...props }: React.ComponentProps<'ol'>) {
  return (
    <ol
      data-slot="breadcrumb-list"
      className={cn(
        'text-muted-foreground flex flex-wrap items-center gap-1.5 text-sm break-words sm:gap-2.5',
        className,
      )}
      {...props}
    />
  )
}

function BreadcrumbItem({ className, ...props }: React.ComponentProps<'li'>) {
  return (
    <li
      data-slot="breadcrumb-item"
      className={cn('inline-flex items-center gap-1.5', className)}
      {...props}
    />
  )
}

function BreadcrumbLink({
  asChild,
  className,
  ...props
}: React.ComponentProps<'a'> & {
  asChild?: boolean
}) {
  const Comp = asChild ? Slot : 'a'

  return (
```

```
    <Comp
      data-slot="breadcrumb-link"
      className={cn('hover:text-foreground transition-colors', className)}
      {...props}
    />
  )
}

function BreadcrumbPage({ className, ...props }: React.ComponentProps<'span'>) {
  return (
    <span
      data-slot="breadcrumb-page"
      role="link"
      aria-disabled="true"
      aria-current="page"
      className={cn('text-foreground font-normal', className)}
      {...props}
    />
  )
}

function BreadcrumbSeparator({
  children,
  className,
  ...props
}: React.ComponentProps<'li'>) {
  return (
    <li
      data-slot="breadcrumb-separator"
      role="presentation"
      aria-hidden="true"
      className={cn('[&>svg]:size-3.5', className)}
      {...props}
    >
      {children ?? <ChevronRight />}
    </li>
  )
}

function BreadcrumbEllipsis({
  className,
  ...props
}: React.ComponentProps<'span'>) {
  return (
    <span
      data-slot="breadcrumb-ellipsis"
      role="presentation"
      aria-hidden="true"
      className={cn('flex size-9 items-center justify-center', className)}
      {...props}
    >
      <MoreHorizontal className="size-4" />
      <span className="sr-only">More</span>
    </span>
  )
}

export {
  Breadcrumb,
  BreadcrumbList,
  BreadcrumbItem,
  BreadcrumbLink,
  BreadcrumbPage,
  BreadcrumbSeparator,
  BreadcrumbEllipsis,
}
```

/mnt/data/cleaned_project/components/ui/button-group.tsx

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

import { cn } from '@/lib/utils'
import { Separator } from '@/components/ui/separator'
```

```
const buttonGroupVariants = cva(
  "flex w-fit items-stretch [&>*]:focus-visible:z-10 [&>*]:focus-visible:relative [&>[data-slot=selec
  {
    variants: {
      orientation: {
        horizontal:
          '[&>*:not(:first-child)]:rounded-l-none [&>*:not(:first-child)]:border-l-0 [&>*:not(:last-c
        vertical:
          'flex-col [&>*:not(:first-child)]:rounded-t-none [&>*:not(:first-child)]:border-t-0 [&>*:no
      },
    },
    defaultVariants: {
      orientation: 'horizontal',
    },
  },
)

function ButtonGroup({
  className,
  orientation,
  ...props
}: React.ComponentProps<'div'> & VariantProps<typeof buttonGroupVariants>) {
  return (
    <div
      role="group"
      data-slot="button-group"
      data-orientation={orientation}
      className={cn(buttonGroupVariants({ orientation }), className)}
      {...props}
    />
  )
}

function ButtonGroupText({
  className,
  asChild = false,
  ...props
}: React.ComponentProps<'div'> & {
  asChild?: boolean
}) {
  const Comp = asChild ? Slot : 'div'

  return (
    <Comp
      className={cn(
        "bg-muted flex items-center gap-2 rounded-md border px-4 text-sm font-medium shadow-xs [&_svg
        className,
      )}
      {...props}
    />
  )
}

function ButtonGroupSeparator({
  className,
  orientation = 'vertical',
  ...props
}: React.ComponentProps<typeof Separator>) {
  return (
    <Separator
      data-slot="button-group-separator"
      orientation={orientation}
      className={cn(
        'bg-input relative !m-0 self-stretch data-[orientation=vertical]:h-auto',
        className,
      )}
      {...props}
    />
  )
}

export {
  ButtonGroup,
  ButtonGroupSeparator,
  ButtonGroupText,
  buttonGroupVariants,
}
```

/mnt/data/cleaned_project/components/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 gap-2 whitespace-nowrap rounded-md text-sm font-medium tra
  {
    variants: {
      variant: {
        default: 'bg-primary text-primary-foreground hover:bg-primary/90',
        destructive:
          'bg-destructive text-white hover:bg-destructive/90 focus-visible:ring-destructive/20 dark:f
        outline:
          'border bg-background shadow-xs hover:bg-accent hover:text-accent-foreground dark:bg-input/
        secondary:
          'bg-secondary text-secondary-foreground hover:bg-secondary/80',
        ghost:
          'hover:bg-accent hover:text-accent-foreground dark:hover:bg-accent/50',
        link: 'text-primary underline-offset-4 hover:underline',
      },
      size: {
        default: 'h-9 px-4 py-2 has-[>svg]:px-3',
        sm: 'h-8 rounded-md gap-1.5 px-3 has-[>svg]:px-2.5',
        lg: 'h-10 rounded-md px-6 has-[>svg]:px-4',
        icon: 'size-9',
        'icon-sm': 'size-8',
        'icon-lg': 'size-10',
      },
    },
    defaultVariants: {
      variant: 'default',
      size: 'default',
    },
  },
)

function Button({
  className,
  variant,
  size,
  asChild = false,
  ...props
}: React.ComponentProps<'button'> &
  VariantProps<typeof buttonVariants> & {
    asChild?: boolean
  }) {
  const Comp = asChild ? Slot : 'button'

  return (
    <Comp
      data-slot="button"
      className={cn(buttonVariants({ variant, size, className }))}
      {...props}
    />
  )
}

export { Button, buttonVariants }
```

/mnt/data/cleaned_project/components/ui/calendar.tsx

```
'use client'

import * as React from 'react'
import {
```

```
  ChevronDownIcon,
  ChevronLeftIcon,
  ChevronRightIcon,
} from 'lucide-react'
import { DayButton, DayPicker, getDefaultClassNames } from 'react-day-picker'

import { cn } from '@/lib/utils'
import { Button, buttonVariants } from '@/components/ui/button'

function Calendar({
  className,
  classNames,
  showOutsideDays = true,
  captionLayout = 'label',
  buttonVariant = 'ghost',
  formatters,
  components,
  ...props
}: React.ComponentProps<typeof DayPicker> & {
  buttonVariant?: React.ComponentProps<typeof Button>['variant']
}) {
  const defaultClassNames = getDefaultClassNames()

  return (
    <DayPicker
      showOutsideDays={showOutsideDays}
      className={cn(
        'bg-background group/calendar p-3 [--cell-size:--spacing(8)] [[data-slot=card-content]_&]:bg
        String.raw`rtl:**:[.rdp-button\_next>svg]:rotate-180`,
        String.raw`rtl:**:[.rdp-button\_previous>svg]:rotate-180`,
        className,
      )}
      captionLayout={captionLayout}
      formatters={{
        formatMonthDropdown: (date) =>
          date.toLocaleString('default', { month: 'short' }),
        ...formatters,
      }}
      classNames={{
        root: cn('w-fit', defaultClassNames.root),
        months: cn(
          'flex gap-4 flex-col md:flex-row relative',
          defaultClassNames.months,
        ),
        month: cn('flex flex-col w-full gap-4', defaultClassNames.month),
        nav: cn(
          'flex items-center gap-1 w-full absolute top-0 inset-x-0 justify-between',
          defaultClassNames.nav,
        ),
        button_previous: cn(
          buttonVariants({ variant: buttonVariant }),
          'size-(--cell-size) aria-disabled:opacity-50 p-0 select-none',
          defaultClassNames.button_previous,
        ),
        button_next: cn(
          buttonVariants({ variant: buttonVariant }),
          'size-(--cell-size) aria-disabled:opacity-50 p-0 select-none',
          defaultClassNames.button_next,
        ),
        month_caption: cn(
          'flex items-center justify-center h-(--cell-size) w-full px-(--cell-size)',
          defaultClassNames.month_caption,
        ),
        dropdowns: cn(
          'w-full flex items-center text-sm font-medium justify-center h-(--cell-size) gap-1.5',
          defaultClassNames.dropdowns,
        ),
        dropdown_root: cn(
          'relative has-focus:border-ring border border-input shadow-xs has-focus:ring-ring/50 has-f
          defaultClassNames.dropdown_root,
        ),
        dropdown: cn(
          'absolute bg-popover inset-0 opacity-0',
          defaultClassNames.dropdown,
        ),
        caption_label: cn(
          'select-none font-medium',
```

```
            captionLayout === 'label'
              ? 'text-sm'
              : 'rounded-md pl-2 pr-1 flex items-center gap-1 text-sm h-8 [&>svg]:text-muted-foreground
            defaultClassNames.caption_label,
          ),
          table: 'w-full border-collapse',
          weekdays: cn('flex', defaultClassNames.weekdays),
          weekday: cn(
            'text-muted-foreground rounded-md flex-1 font-normal text-[0.8rem] select-none',
            defaultClassNames.weekday,
          ),
          week: cn('flex w-full mt-2', defaultClassNames.week),
          week_number_header: cn(
            'select-none w-(--cell-size)',
            defaultClassNames.week_number_header,
          ),
          week_number: cn(
            'text-[0.8rem] select-none text-muted-foreground',
            defaultClassNames.week_number,
          ),
          day: cn(
            'relative w-full h-full p-0 text-center [&:first-child[data-selected=true]_button]:rounded
            defaultClassNames.day,
          ),
          range_start: cn(
            'rounded-l-md bg-accent',
            defaultClassNames.range_start,
          ),
          range_middle: cn('rounded-none', defaultClassNames.range_middle),
          range_end: cn('rounded-r-md bg-accent', defaultClassNames.range_end),
          today: cn(
            'bg-accent text-accent-foreground rounded-md data-[selected=true]:rounded-none',
            defaultClassNames.today,
          ),
          outside: cn(
            'text-muted-foreground aria-selected:text-muted-foreground',
            defaultClassNames.outside,
          ),
          disabled: cn(
            'text-muted-foreground opacity-50',
            defaultClassNames.disabled,
          ),
          hidden: cn('invisible', defaultClassNames.hidden),
          ...classNames,
        }}
        components={{
          Root: ({ className, rootRef, ...props }) => {
            return (
              <div
                data-slot="calendar"
                ref={rootRef}
                className={cn(className)}
                {...props}
              />
            )
          },
          Chevron: ({ className, orientation, ...props }) => {
```

/mnt/data/cleaned_project/components/ui/card.tsx

```
import * as React from 'react'

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

function Card({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="card"
      className={cn(
        'bg-card text-card-foreground flex flex-col gap-6 rounded-xl border py-6 shadow-sm',
        className,
      )}
      {...props}
    />
```

```
  )
}

function CardHeader({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="card-header"
      className={cn(
        '@container/card-header grid auto-rows-min grid-rows-[auto_auto] items-start gap-2 px-6 has-
        className,
      )}
      {...props}
    />
  )
}

function CardTitle({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="card-title"
      className={cn('leading-none font-semibold', className)}
      {...props}
    />
  )
}

function CardDescription({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="card-description"
      className={cn('text-muted-foreground text-sm', className)}
      {...props}
    />
  )
}

function CardAction({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="card-action"
      className={cn(
        'col-start-2 row-span-2 row-start-1 self-start justify-self-end',
        className,
      )}
      {...props}
    />
  )
}

function CardContent({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="card-content"
      className={cn('px-6', className)}
      {...props}
    />
  )
}

function CardFooter({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="card-footer"
      className={cn('flex items-center px-6 [.border-t]:pt-6', className)}
      {...props}
    />
  )
}

export {
  Card,
  CardHeader,
  CardFooter,
  CardTitle,
  CardAction,
  CardDescription,
  CardContent,
```

```
}
```

/mnt/data/cleaned_project/components/ui/carousel.tsx

```
'use client'

import * as React from 'react'
import useEmblaCarousel, {
  type UseEmblaCarouselType,
} from 'embla-carousel-react'
import { ArrowLeft, ArrowRight } from 'lucide-react'

import { cn } from '@/lib/utils'
import { Button } from '@/components/ui/button'

type CarouselApi = UseEmblaCarouselType[1]
type UseCarouselParameters = Parameters<typeof useEmblaCarousel>
type CarouselOptions = UseCarouselParameters[0]
type CarouselPlugin = UseCarouselParameters[1]

type CarouselProps = {
  opts?: CarouselOptions
  plugins?: CarouselPlugin
  orientation?: 'horizontal' | 'vertical'
  setApi?: (api: CarouselApi) => void
}

type CarouselContextProps = {
  carouselRef: ReturnType<typeof useEmblaCarousel>[0]
  api: ReturnType<typeof useEmblaCarousel>[1]
  scrollPrev: () => void
  scrollNext: () => void
  canScrollPrev: boolean
  canScrollNext: boolean
} & CarouselProps

const CarouselContext = React.createContext<CarouselContextProps | null>(null)

function useCarousel() {
  const context = React.useContext(CarouselContext)

  if (!context) {
    throw new Error('useCarousel must be used within a <Carousel />')
  }

  return context
}

function Carousel({
  orientation = 'horizontal',
  opts,
  setApi,
  plugins,
  className,
  children,
  ...props
}: React.ComponentProps<'div'> & CarouselProps) {
  const [carouselRef, api] = useEmblaCarousel(
    {
      ...opts,
      axis: orientation === 'horizontal' ? 'x' : 'y',
    },
    plugins,
  )
  const [canScrollPrev, setCanScrollPrev] = React.useState(false)
  const [canScrollNext, setCanScrollNext] = React.useState(false)

  const onSelect = React.useCallback((api: CarouselApi) => {
    if (!api) return
    setCanScrollPrev(api.canScrollPrev())
    setCanScrollNext(api.canScrollNext())
  }, [])

  const scrollPrev = React.useCallback(() => {
```

```
    api?.scrollPrev()
  }, [api])

  const scrollNext = React.useCallback(() => {
    api?.scrollNext()
  }, [api])

  const handleKeyDown = React.useCallback(
    (event: React.KeyboardEvent<HTMLDivElement>) => {
      if (event.key === 'ArrowLeft') {
        event.preventDefault()
        scrollPrev()
      } else if (event.key === 'ArrowRight') {
        event.preventDefault()
        scrollNext()
      }
    },
    [scrollPrev, scrollNext],
  )

  React.useEffect(() => {
    if (!api || !setApi) return
    setApi(api)
  }, [api, setApi])

  React.useEffect(() => {
    if (!api) return
    onSelect(api)
    api.on('reInit', onSelect)
    api.on('select', onSelect)

    return () => {
      api?.off('select', onSelect)
    }
  }, [api, onSelect])

  return (
    <CarouselContext.Provider
      value={{
        carouselRef,
        api: api,
        opts,
        orientation:
          orientation || (opts?.axis === 'y' ? 'vertical' : 'horizontal'),
        scrollPrev,
        scrollNext,
        canScrollPrev,
        canScrollNext,
      }}
    >
      <div
        onKeyDownCapture={handleKeyDown}
        className={cn('relative', className)}
        role="region"
        aria-roledescription="carousel"
        data-slot="carousel"
        {...props}
      >
        {children}
      </div>
    </CarouselContext.Provider>
  )
}

function CarouselContent({ className, ...props }: React.ComponentProps<'div'>) {
  const { carouselRef, orientation } = useCarousel()

  return (
    <div
      ref={carouselRef}
      className="overflow-hidden"
      data-slot="carousel-content"
    >
      <div
        className={cn(
          'flex',
          orientation === 'horizontal' ? '-ml-4' : '-mt-4 flex-col',
```

```
          className,
        )}
        {...props}
      />
    </div>
  )
}

function CarouselItem({ className, ...props }: React.ComponentProps<'div'>) {
  const { orientation } = useCarousel()

  return (
    <div
      role="group"
      aria-roledescription="slide"
      data-slot="carousel-item"
      className={cn(
        'min-w-0 shrink-0 grow-0 basis-full',
        orientation === 'horizontal' ? 'pl-4' : 'pt-4',
        className,
      )}
      {...props}
    />
  )
}

function CarouselPrevious({
  className,
  variant = 'outline',
  size = 'icon',
  ...props
}: React.ComponentProps<typeof Button>) {
  const { orientation, scrollPrev, canScrollPrev } = useCarousel()

  return (
    <Button
      data-slot="carousel-previous"
      variant={variant}
      size={size}
      className={cn(
        'absolute size-8 rounded-full',
        orientation === 'horizontal'
          ? 'top-1/2 -left-12 -translate-y-1/2'
          : '-top-12 left-1/2 -translate-x-1/2 rotate-90',
        className,
      )}
      disabled={!canScrollPrev}
      onClick={scrollPrev}
      {...props}
    >
      <ArrowLeft />
      <span className="sr-only">Previous slide</span>
    </Button>
  )
}

function CarouselNext({
  className,
  variant = 'outline',
  size = 'icon',
  ...props
}: React.ComponentProps<typeof Button>) {
  const { orientation, scrollNext, canScrollNext } = useCarousel()

  return (
    <Button
      data-slot="carousel-next"
```

/mnt/data/cleaned_project/components/ui/chart.tsx

```
'use client'

import * as React from 'react'
import * as RechartsPrimitive from 'recharts'
```

```tsx
import { cn } from '@/lib/utils'

// Format: { THEME_NAME: CSS_SELECTOR }
const THEMES = { light: '', dark: '.dark' } as const

export type ChartConfig = {
  [k in string]: {
    label?: React.ReactNode
    icon?: React.ComponentType
  } & (
    | { color?: string; theme?: never }
    | { color?: never; theme: Record<keyof typeof THEMES, string> }
  )
}

type ChartContextProps = {
  config: ChartConfig
}

const ChartContext = React.createContext<ChartContextProps | null>(null)

function useChart() {
  const context = React.useContext(ChartContext)

  if (!context) {
    throw new Error('useChart must be used within a <ChartContainer />')
  }

  return context
}

function ChartContainer({
  id,
  className,
  children,
  config,
  ...props
}: React.ComponentProps<'div'> & {
  config: ChartConfig
  children: React.ComponentProps<
    typeof RechartsPrimitive.ResponsiveContainer
  >['children']
}) {
  const uniqueId = React.useId()
  const chartId = `chart-${id || uniqueId.replace(/:/g, '')}`

  return (
    <ChartContext.Provider value={{ config }}>
      <div
        data-slot="chart"
        data-chart={chartId}
        className={cn(
          "[&_.recharts-cartesian-axis-tick_text]:fill-muted-foreground [&_.recharts-cartesian-grid_
          className,
        )}
        {...props}
      >
        <ChartStyle id={chartId} config={config} />
        <RechartsPrimitive.ResponsiveContainer>
          {children}
        </RechartsPrimitive.ResponsiveContainer>
      </div>
    </ChartContext.Provider>
  )
}

const ChartStyle = ({ id, config }: { id: string; config: ChartConfig }) => {
  const colorConfig = Object.entries(config).filter(
    ([, config]) => config.theme || config.color,
  )

  if (!colorConfig.length) {
    return null
  }

  return (
    <style
```

```
      dangerouslySetInnerHTML={{
        __html: Object.entries(THEMES)
          .map(
            ([theme, prefix]) => `
${prefix} [data-chart=${id}] {
${colorConfig
  .map(([key, itemConfig]) => {
    const color =
      itemConfig.theme?.[theme as keyof typeof itemConfig.theme] ||
      itemConfig.color
    return color ? `  --color-${key}: ${color};` : null
  })
  .join('\n')}
}
`,
          )
          .join('\n'),
      }}
    />
  )
}

const ChartTooltip = RechartsPrimitive.Tooltip

function ChartTooltipContent({
  active,
  payload,
  className,
  indicator = 'dot',
  hideLabel = false,
  hideIndicator = false,
  label,
  labelFormatter,
  labelClassName,
  formatter,
  color,
  nameKey,
  labelKey,
}: React.ComponentProps<typeof RechartsPrimitive.Tooltip> &
  React.ComponentProps<'div'> & {
    hideLabel?: boolean
    hideIndicator?: boolean
    indicator?: 'line' | 'dot' | 'dashed'
    nameKey?: string
    labelKey?: string
  }) {
  const { config } = useChart()

  const tooltipLabel = React.useMemo(() => {
    if (hideLabel || !payload?.length) {
      return null
    }

    const [item] = payload
    const key = `${labelKey || item?.dataKey || item?.name || 'value'}`
    const itemConfig = getPayloadConfigFromPayload(config, item, key)
    const value =
      !labelKey && typeof label === 'string'
        ? config[label as keyof typeof config]?.label || label
        : itemConfig?.label

    if (labelFormatter) {
      return (
        <div className={cn('font-medium', labelClassName)}>
          {labelFormatter(value, payload)}
        </div>
      )
    }

    if (!value) {
      return null
    }

    return <div className={cn('font-medium', labelClassName)}>{value}</div>
  }, [
    label,
    labelFormatter,
```

```
    payload,
    hideLabel,
    labelClassName,
    config,
    labelKey,
  ])

  if (!active || !payload?.length) {
    return null
  }

  const nestLabel = payload.length === 1 && indicator !== 'dot'

  return (
    <div
      className={cn(
        'border-border/50 bg-background grid min-w-[8rem] items-start gap-1.5 rounded-lg border px-2
        className,
      )}
    >
      {!nestLabel ? tooltipLabel : null}
      <div className="grid gap-1.5">
        {payload.map((item, index) => {
          const key = `${nameKey || item.name || item.dataKey || 'value'}`
          const itemConfig = getPayloadConfigFromPayload(config, item, key)
          const indicatorColor = color || item.payload.fill || item.color

          return (
```

/mnt/data/cleaned_project/components/ui/checkbox.tsx

```
'use client'

import * as React from 'react'
import * as CheckboxPrimitive from '@radix-ui/react-checkbox'
import { CheckIcon } from 'lucide-react'

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

function Checkbox({
  className,
  ...props
}: React.ComponentProps<typeof CheckboxPrimitive.Root>) {
  return (
    <CheckboxPrimitive.Root
      data-slot="checkbox"
      className={cn(
        'peer border-input dark:bg-input/30 data-[state=checked]:bg-primary data-[state=checked]:tex
        className,
      )}
      {...props}
    >
      <CheckboxPrimitive.Indicator
        data-slot="checkbox-indicator"
        className="flex items-center justify-center text-current transition-none"
      >
        <CheckIcon className="size-3.5" />
      </CheckboxPrimitive.Indicator>
    </CheckboxPrimitive.Root>
  )
}

export { Checkbox }
```

/mnt/data/cleaned_project/components/ui/collapsible.tsx

```
'use client'

import * as CollapsiblePrimitive from '@radix-ui/react-collapsible'
function Collapsible({
```

```
  ...props
}: React.ComponentProps<typeof CollapsiblePrimitive.Root>) {
  return <CollapsiblePrimitive.Root data-slot="collapsible" {...props} />
}

function CollapsibleTrigger({
  ...props
}: React.ComponentProps<typeof CollapsiblePrimitive.CollapsibleTrigger>) {
  return (
    <CollapsiblePrimitive.CollapsibleTrigger
      data-slot="collapsible-trigger"
      {...props}
    />
  )
}

function CollapsibleContent({
  ...props
}: React.ComponentProps<typeof CollapsiblePrimitive.CollapsibleContent>) {
  return (
    <CollapsiblePrimitive.CollapsibleContent
      data-slot="collapsible-content"
      {...props}
    />
  )
}

export { Collapsible, CollapsibleTrigger, CollapsibleContent }
```

/mnt/data/cleaned_project/components/ui/command.tsx

```
'use client'

import * as React from 'react'
import { Command as CommandPrimitive } from 'cmdk'
import { SearchIcon } from 'lucide-react'

import { cn } from '@/lib/utils'
import {
  Dialog,
  DialogContent,
  DialogDescription,
  DialogHeader,
  DialogTitle,
} from '@/components/ui/dialog'

function Command({
  className,
  ...props
}: React.ComponentProps<typeof CommandPrimitive>) {
  return (
    <CommandPrimitive
      data-slot="command"
      className={cn(
        'bg-popover text-popover-foreground flex h-full w-full flex-col overflow-hidden rounded-md',
        className,
      )}
      {...props}
    />
  )
}

function CommandDialog({
  title = 'Command Palette',
  description = 'Search for a command to run...',
  children,
  className,
  showCloseButton = true,
  ...props
}: React.ComponentProps<typeof Dialog> & {
  title?: string
  description?: string
  className?: string
  showCloseButton?: boolean
```

```
}) {
  return (
    <Dialog {...props}>
      <DialogHeader className="sr-only">
        <DialogTitle>{title}</DialogTitle>
        <DialogDescription>{description}</DialogDescription>
      </DialogHeader>
      <DialogContent
        className={cn('overflow-hidden p-0', className)}
        showCloseButton={showCloseButton}
      >
        <Command className="[&_[cmdk-group-heading]]:text-muted-foreground **:data-[slot=command-inpu
          {children}
        </Command>
      </DialogContent>
    </Dialog>
  )
}

function CommandInput({
  className,
  ...props
}: React.ComponentProps<typeof CommandPrimitive.Input>) {
  return (
    <div
      data-slot="command-input-wrapper"
      className="flex h-9 items-center gap-2 border-b px-3"
    >
      <SearchIcon className="size-4 shrink-0 opacity-50" />
      <CommandPrimitive.Input
        data-slot="command-input"
        className={cn(
          'placeholder:text-muted-foreground flex h-10 w-full rounded-md bg-transparent py-3 text-sm
          className,
        )}
        {...props}
      />
    </div>
  )
}

function CommandList({
  className,
  ...props
}: React.ComponentProps<typeof CommandPrimitive.List>) {
  return (
    <CommandPrimitive.List
      data-slot="command-list"
      className={cn(
        'max-h-[300px] scroll-py-1 overflow-x-hidden overflow-y-auto',
        className,
      )}
      {...props}
    />
  )
}

function CommandEmpty({
  ...props
}: React.ComponentProps<typeof CommandPrimitive.Empty>) {
  return (
    <CommandPrimitive.Empty
      data-slot="command-empty"
      className="py-6 text-center text-sm"
      {...props}
    />
  )
}

function CommandGroup({
  className,
  ...props
}: React.ComponentProps<typeof CommandPrimitive.Group>) {
  return (
    <CommandPrimitive.Group
      data-slot="command-group"
      className={cn(
```

```
        'text-foreground [&_[cmdk-group-heading]]:text-muted-foreground overflow-hidden p-1 [&_[cmdk
        className,
      )}
      {...props}
    />
  )
}

function CommandSeparator({
  className,
  ...props
}: React.ComponentProps<typeof CommandPrimitive.Separator>) {
  return (
    <CommandPrimitive.Separator
      data-slot="command-separator"
      className={cn('bg-border -mx-1 h-px', className)}
      {...props}
    />
  )
}

function CommandItem({
  className,
  ...props
}: React.ComponentProps<typeof CommandPrimitive.Item>) {
  return (
    <CommandPrimitive.Item
      data-slot="command-item"
      className={cn(
        "data-[selected=true]:bg-accent data-[selected=true]:text-accent-foreground [&_svg:not([class
        className,
      )}
      {...props}
    />
  )
}

function CommandShortcut({
  className,
  ...props
}: React.ComponentProps<'span'>) {
  return (
    <span
      data-slot="command-shortcut"
      className={cn(
        'text-muted-foreground ml-auto text-xs tracking-widest',
        className,
      )}
      {...props}
    />
  )
}

export {
  Command,
  CommandDialog,
  CommandInput,
  CommandList,
  CommandEmpty,
  CommandGroup,
  CommandItem,
  CommandShortcut,
  CommandSeparator,
}
```

/mnt/data/cleaned_project/components/ui/context-menu.tsx

```
'use client'

import * as React from 'react'
import * as ContextMenuPrimitive from '@radix-ui/react-context-menu'
import { CheckIcon, ChevronRightIcon, CircleIcon } from 'lucide-react'

import { cn } from '@/lib/utils'
```

```
function ContextMenu({
  ...props
}: React.ComponentProps<typeof ContextMenuPrimitive.Root>) {
  return <ContextMenuPrimitive.Root data-slot="context-menu" {...props} />
}

function ContextMenuTrigger({
  ...props
}: React.ComponentProps<typeof ContextMenuPrimitive.Trigger>) {
  return (
    <ContextMenuPrimitive.Trigger data-slot="context-menu-trigger" {...props} />
  )
}

function ContextMenuGroup({
  ...props
}: React.ComponentProps<typeof ContextMenuPrimitive.Group>) {
  return (
    <ContextMenuPrimitive.Group data-slot="context-menu-group" {...props} />
  )
}

function ContextMenuPortal({
  ...props
}: React.ComponentProps<typeof ContextMenuPrimitive.Portal>) {
  return (
    <ContextMenuPrimitive.Portal data-slot="context-menu-portal" {...props} />
  )
}

function ContextMenuSub({
  ...props
}: React.ComponentProps<typeof ContextMenuPrimitive.Sub>) {
  return <ContextMenuPrimitive.Sub data-slot="context-menu-sub" {...props} />
}

function ContextMenuRadioGroup({
  ...props
}: React.ComponentProps<typeof ContextMenuPrimitive.RadioGroup>) {
  return (
    <ContextMenuPrimitive.RadioGroup
      data-slot="context-menu-radio-group"
      {...props}
    />
  )
}

function ContextMenuSubTrigger({
  className,
  inset,
  children,
  ...props
}: React.ComponentProps<typeof ContextMenuPrimitive.SubTrigger> & {
  inset?: boolean
}) {
  return (
    <ContextMenuPrimitive.SubTrigger
      data-slot="context-menu-sub-trigger"
      data-inset={inset}
      className={cn(
        "focus:bg-accent focus:text-accent-foreground data-[state=open]:bg-accent data-[state=open]:
        className,
      )}
      {...props}
    >
      {children}
      <ChevronRightIcon className="ml-auto" />
    </ContextMenuPrimitive.SubTrigger>
  )
}

function ContextMenuSubContent({
  className,
  ...props
}: React.ComponentProps<typeof ContextMenuPrimitive.SubContent>) {
  return (
    <ContextMenuPrimitive.SubContent
```

```
      data-slot="context-menu-sub-content"
      className={cn(
        'bg-popover text-popover-foreground data-[state=open]:animate-in data-[state=closed]:animate
        className,
      )}
      {...props}
    />
  )
}

function ContextMenuContent({
  className,
  ...props
}: React.ComponentProps<typeof ContextMenuPrimitive.Content>) {
  return (
    <ContextMenuPrimitive.Portal>
      <ContextMenuPrimitive.Content
        data-slot="context-menu-content"
        className={cn(
          'bg-popover text-popover-foreground data-[state=open]:animate-in data-[state=closed]:anima
          className,
        )}
        {...props}
      />
    </ContextMenuPrimitive.Portal>
  )
}

function ContextMenuItem({
  className,
  inset,
  variant = 'default',
  ...props
}: React.ComponentProps<typeof ContextMenuPrimitive.Item> & {
  inset?: boolean
  variant?: 'default' | 'destructive'
}) {
  return (
    <ContextMenuPrimitive.Item
      data-slot="context-menu-item"
      data-inset={inset}
      data-variant={variant}
      className={cn(
        "focus:bg-accent focus:text-accent-foreground data-[variant=destructive]:text-destructive da
        className,
      )}
```

/mnt/data/cleaned_project/components/ui/dialog.tsx

```
'use client'

import * as React from 'react'
import * as DialogPrimitive from '@radix-ui/react-dialog'
import { XIcon } from 'lucide-react'

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

function Dialog({
  ...props
}: React.ComponentProps<typeof DialogPrimitive.Root>) {
  return <DialogPrimitive.Root data-slot="dialog" {...props} />
}

function DialogTrigger({
  ...props
}: React.ComponentProps<typeof DialogPrimitive.Trigger>) {
  return <DialogPrimitive.Trigger data-slot="dialog-trigger" {...props} />
}

function DialogPortal({
  ...props
}: React.ComponentProps<typeof DialogPrimitive.Portal>) {
  return <DialogPrimitive.Portal data-slot="dialog-portal" {...props} />
}
```

```
function DialogClose({
  ...props
}: React.ComponentProps<typeof DialogPrimitive.Close>) {
  return <DialogPrimitive.Close data-slot="dialog-close" {...props} />
}

function DialogOverlay({
  className,
  ...props
}: React.ComponentProps<typeof DialogPrimitive.Overlay>) {
  return (
    <DialogPrimitive.Overlay
      data-slot="dialog-overlay"
      className={cn(
        'data-[state=open]:animate-in data-[state=closed]:animate-out data-[state=closed]:fade-out-0
        className,
      )}
      {...props}
    />
  )
}

function DialogContent({
  className,
  children,
  showCloseButton = true,
  ...props
}: React.ComponentProps<typeof DialogPrimitive.Content> & {
  showCloseButton?: boolean
}) {
  return (
    <DialogPortal data-slot="dialog-portal">
      <DialogOverlay />
      <DialogPrimitive.Content
        data-slot="dialog-content"
        className={cn(
          'bg-background data-[state=open]:animate-in data-[state=closed]:animate-out data-[state=cl
          className,
        )}
        {...props}
      >
        {children}
        {showCloseButton && (
          <DialogPrimitive.Close
            data-slot="dialog-close"
            className="ring-offset-background focus:ring-ring data-[state=open]:bg-accent data-[stat
          >
            <XIcon />
            <span className="sr-only">Close</span>
          </DialogPrimitive.Close>
        )}
      </DialogPrimitive.Content>
    </DialogPortal>
  )
}

function DialogHeader({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="dialog-header"
      className={cn('flex flex-col gap-2 text-center sm:text-left', className)}
      {...props}
    />
  )
}

function DialogFooter({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="dialog-footer"
      className={cn(
        'flex flex-col-reverse gap-2 sm:flex-row sm:justify-end',
        className,
      )}
      {...props}
    />
  )
```

```
}

function DialogTitle({
  className,
  ...props
}: React.ComponentProps<typeof DialogPrimitive.Title>) {
  return (
    <DialogPrimitive.Title
      data-slot="dialog-title"
      className={cn('text-lg leading-none font-semibold', className)}
      {...props}
    />
  )
}

function DialogDescription({
  className,
  ...props
}: React.ComponentProps<typeof DialogPrimitive.Description>) {
  return (
    <DialogPrimitive.Description
      data-slot="dialog-description"
      className={cn('text-muted-foreground text-sm', className)}
      {...props}
    />
  )
}

export {
  Dialog,
  DialogClose,
  DialogContent,
  DialogDescription,
  DialogFooter,
  DialogHeader,
  DialogOverlay,
  DialogPortal,
  DialogTitle,
  DialogTrigger,
}
```

/mnt/data/cleaned_project/components/ui/drawer.tsx

```
'use client'

import * as React from 'react'
import { Drawer as DrawerPrimitive } from 'vaul'

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

function Drawer({
  ...props
}: React.ComponentProps<typeof DrawerPrimitive.Root>) {
  return <DrawerPrimitive.Root data-slot="drawer" {...props} />
}

function DrawerTrigger({
  ...props
}: React.ComponentProps<typeof DrawerPrimitive.Trigger>) {
  return <DrawerPrimitive.Trigger data-slot="drawer-trigger" {...props} />
}

function DrawerPortal({
  ...props
}: React.ComponentProps<typeof DrawerPrimitive.Portal>) {
  return <DrawerPrimitive.Portal data-slot="drawer-portal" {...props} />
}

function DrawerClose({
  ...props
}: React.ComponentProps<typeof DrawerPrimitive.Close>) {
  return <DrawerPrimitive.Close data-slot="drawer-close" {...props} />
}
function DrawerOverlay({
```

```
  className,
  ...props
}: React.ComponentProps<typeof DrawerPrimitive.Overlay>) {
  return (
    <DrawerPrimitive.Overlay
      data-slot="drawer-overlay"
      className={cn(
        'data-[state=open]:animate-in data-[state=closed]:animate-out data-[state=closed]:fade-out-0
        className,
      )}
      {...props}
    />
  )
}

function DrawerContent({
  className,
  children,
  ...props
}: React.ComponentProps<typeof DrawerPrimitive.Content>) {
  return (
    <DrawerPortal data-slot="drawer-portal">
      <DrawerOverlay />
      <DrawerPrimitive.Content
        data-slot="drawer-content"
        className={cn(
          'group/drawer-content bg-background fixed z-50 flex h-auto flex-col',
          'data-[vaul-drawer-direction=top]:inset-x-0 data-[vaul-drawer-direction=top]:top-0 data-[va
          'data-[vaul-drawer-direction=bottom]:inset-x-0 data-[vaul-drawer-direction=bottom]:bottom-0
          'data-[vaul-drawer-direction=right]:inset-y-0 data-[vaul-drawer-direction=right]:right-0 da
          'data-[vaul-drawer-direction=left]:inset-y-0 data-[vaul-drawer-direction=left]:left-0 data-
          className,
        )}
        {...props}
      >
        <div className="bg-muted mx-auto mt-4 hidden h-2 w-[100px] shrink-0 rounded-full group-data-
        {children}
      </DrawerPrimitive.Content>
    </DrawerPortal>
  )
}

function DrawerHeader({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="drawer-header"
      className={cn(
        'flex flex-col gap-0.5 p-4 group-data-[vaul-drawer-direction=bottom]/drawer-content:text-cen
        className,
      )}
      {...props}
    />
  )
}

function DrawerFooter({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="drawer-footer"
      className={cn('mt-auto flex flex-col gap-2 p-4', className)}
      {...props}
    />
  )
}

function DrawerTitle({
  className,
  ...props
}: React.ComponentProps<typeof DrawerPrimitive.Title>) {
  return (
    <DrawerPrimitive.Title
      data-slot="drawer-title"
      className={cn('text-foreground font-semibold', className)}
      {...props}
    />
  )
}
```

```
function DrawerDescription({
  className,
  ...props
}: React.ComponentProps<typeof DrawerPrimitive.Description>) {
  return (
    <DrawerPrimitive.Description
      data-slot="drawer-description"
      className={cn('text-muted-foreground text-sm', className)}
      {...props}
    />
  )
}

export {
  Drawer,
  DrawerPortal,
  DrawerOverlay,
  DrawerTrigger,
  DrawerClose,
  DrawerContent,
  DrawerHeader,
  DrawerFooter,
  DrawerTitle,
  DrawerDescription,
}
```

/mnt/data/cleaned_project/components/ui/dropdown-menu.tsx

```
'use client'

import * as React from 'react'
import * as DropdownMenuPrimitive from '@radix-ui/react-dropdown-menu'
import { CheckIcon, ChevronRightIcon, CircleIcon } from 'lucide-react'

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

function DropdownMenu({
  ...props
}: React.ComponentProps<typeof DropdownMenuPrimitive.Root>) {
  return <DropdownMenuPrimitive.Root data-slot="dropdown-menu" {...props} />
}

function DropdownMenuPortal({
  ...props
}: React.ComponentProps<typeof DropdownMenuPrimitive.Portal>) {
  return (
    <DropdownMenuPrimitive.Portal data-slot="dropdown-menu-portal" {...props} />
  )
}

function DropdownMenuTrigger({
  ...props
}: React.ComponentProps<typeof DropdownMenuPrimitive.Trigger>) {
  return (
    <DropdownMenuPrimitive.Trigger
      data-slot="dropdown-menu-trigger"
      {...props}
    />
  )
}

function DropdownMenuContent({
  className,
  sideOffset = 4,
  ...props
}: React.ComponentProps<typeof DropdownMenuPrimitive.Content>) {
  return (
    <DropdownMenuPrimitive.Portal>
      <DropdownMenuPrimitive.Content
        data-slot="dropdown-menu-content"
        sideOffset={sideOffset}
        className={cn(
          'bg-popover text-popover-foreground data-[state=open]:animate-in data-[state=closed]:anima
          className,
```

```
        )}
        {...props}
      />
    </DropdownMenuPrimitive.Portal>
  )
}

function DropdownMenuGroup({
  ...props
}: React.ComponentProps<typeof DropdownMenuPrimitive.Group>) {
  return (
    <DropdownMenuPrimitive.Group data-slot="dropdown-menu-group" {...props} />
  )
}

function DropdownMenuItem({
  className,
  inset,
  variant = 'default',
  ...props
}: React.ComponentProps<typeof DropdownMenuPrimitive.Item> & {
  inset?: boolean
  variant?: 'default' | 'destructive'
}) {
  return (
    <DropdownMenuPrimitive.Item
      data-slot="dropdown-menu-item"
      data-inset={inset}
      data-variant={variant}
      className={cn(
        "focus:bg-accent focus:text-accent-foreground data-[variant=destructive]:text-destructive da
        className,
      )}
      {...props}
    />
  )
}

function DropdownMenuCheckboxItem({
  className,
  children,
  checked,
  ...props
}: React.ComponentProps<typeof DropdownMenuPrimitive.CheckboxItem>) {
  return (
    <DropdownMenuPrimitive.CheckboxItem
      data-slot="dropdown-menu-checkbox-item"
      className={cn(
        "focus:bg-accent focus:text-accent-foreground relative flex cursor-default items-center gap-
        className,
      )}
      checked={checked}
      {...props}
    >
      <span className="pointer-events-none absolute left-2 flex size-3.5 items-center justify-center
        <DropdownMenuPrimitive.ItemIndicator>
          <CheckIcon className="size-4" />
        </DropdownMenuPrimitive.ItemIndicator>
      </span>
      {children}
    </DropdownMenuPrimitive.CheckboxItem>
  )
}

function DropdownMenuRadioGroup({
  ...props
}: React.ComponentProps<typeof DropdownMenuPrimitive.RadioGroup>) {
  return (
    <DropdownMenuPrimitive.RadioGroup
      data-slot="dropdown-menu-radio-group"
      {...props}
    />
  )
}

function DropdownMenuRadioItem({
  className,
```

```
  children,
  ...props
}: React.ComponentProps<typeof DropdownMenuPrimitive.RadioItem>) {
  return (
    <DropdownMenuPrimitive.RadioItem
      data-slot="dropdown-menu-radio-item"
      className={cn(
        "focus:bg-accent focus:text-accent-foreground relative flex cursor-default items-center gap-
        className,
      )}
      {...props}
    >
      <span className="pointer-events-none absolute left-2 flex size-3.5 items-center justify-
```

/mnt/data/cleaned_project/components/ui/empty.tsx

```
import { cva, type VariantProps } from 'class-variance-authority'

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

function Empty({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="empty"
      className={cn(
        'flex min-w-0 flex-1 flex-col items-center justify-center gap-6 rounded-lg border-dashed p-6
        className,
      )}
      {...props}
    />
  )
}

function EmptyHeader({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="empty-header"
      className={cn(
        'flex max-w-sm flex-col items-center gap-2 text-center',
        className,
      )}
      {...props}
    />
  )
}

const emptyMediaVariants = cva(
  'flex shrink-0 items-center justify-center mb-2 [&_svg]:pointer-events-none [&_svg]:shrink-0',
  {
    variants: {
      variant: {
        default: 'bg-transparent',
        icon: "bg-muted text-foreground flex size-10 shrink-0 items-center justify-center rounded-lg
      },
    },
    defaultVariants: {
      variant: 'default',
    },
  },
)

function EmptyMedia({
  className,
  variant = 'default',
  ...props
}: React.ComponentProps<'div'> & VariantProps<typeof emptyMediaVariants>) {
  return (
    <div
      data-slot="empty-icon"
      data-variant={variant}
      className={cn(emptyMediaVariants({ variant, className }))}
      {...props}
    />
  )
```

```
}

function EmptyTitle({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="empty-title"
      className={cn('text-lg font-medium tracking-tight', className)}
      {...props}
    />
  )
}

function EmptyDescription({ className, ...props }: React.ComponentProps<'p'>) {
  return (
    <div
      data-slot="empty-description"
      className={cn(
        'text-muted-foreground [&>a:hover]:text-primary text-sm/relaxed [&>a]:underline [&>a]:underli
        className,
      )}
      {...props}
    />
  )
}

function EmptyContent({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="empty-content"
      className={cn(
        'flex w-full max-w-sm min-w-0 flex-col items-center gap-4 text-sm text-balance',
        className,
      )}
      {...props}
    />
  )
}

export {
  Empty,
  EmptyHeader,
  EmptyTitle,
  EmptyDescription,
  EmptyContent,
  EmptyMedia,
}
```

/mnt/data/cleaned_project/components/ui/field.tsx

```
'use client'

import { useMemo } from 'react'
import { cva, type VariantProps } from 'class-variance-authority'

import { cn } from '@/lib/utils'
import { Label } from '@/components/ui/label'
import { Separator } from '@/components/ui/separator'

function FieldSet({ className, ...props }: React.ComponentProps<'fieldset'>) {
  return (
    <fieldset
      data-slot="field-set"
      className={cn(
        'flex flex-col gap-6',
        'has-[>[data-slot=checkbox-group]]:gap-3 has-[>[data-slot=radio-group]]:gap-3',
        className,
      )}
      {...props}
    />
  )
}

function FieldLegend({
  className,
```

```
  variant = 'legend',
  ...props
}: React.ComponentProps<'legend'> & { variant?: 'legend' | 'label' }) {
  return (
    <legend
      data-slot="field-legend"
      data-variant={variant}
      className={cn(
        'mb-3 font-medium',
        'data-[variant=legend]:text-base',
        'data-[variant=label]:text-sm',
        className,
      )}
      {...props}
    />
  )
}

function FieldGroup({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="field-group"
      className={cn(
        'group/field-group @container/field-group flex w-full flex-col gap-7 data-[slot=checkbox-grou
        className,
      )}
      {...props}
    />
  )
}

const fieldVariants = cva(
  'group/field flex w-full gap-3 data-[invalid=true]:text-destructive',
  {
    variants: {
      orientation: {
        vertical: ['flex-col [&>*]:w-full [&>.sr-only]:w-auto'],
        horizontal: [
          'flex-row items-center',
          '[&>[data-slot=field-label]]:flex-auto',
          'has-[>[data-slot=field-content]]:items-start has-[>[data-slot=field-content]]:[&>[role=che
        ],
        responsive: [
          'flex-col [&>*]:w-full [&>.sr-only]:w-auto @md/field-group:flex-row @md/field-group:items-c
          '@md/field-group:[&>[data-slot=field-label]]:flex-auto',
          '@md/field-group:has-[>[data-slot=field-content]]:items-start @md/field-group:has-[>[data-s
        ],
      },
    },
    defaultVariants: {
      orientation: 'vertical',
    },
  },
)

function Field({
  className,
  orientation = 'vertical',
  ...props
}: React.ComponentProps<'div'> & VariantProps<typeof fieldVariants>) {
  return (
    <div
      role="group"
      data-slot="field"
      data-orientation={orientation}
      className={cn(fieldVariants({ orientation }), className)}
      {...props}
    />
  )
}

function FieldContent({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="field-content"
      className={cn(
        'group/field-content flex flex-1 flex-col gap-1.5 leading-snug',
```

```
        className,
      )}
      {...props}
    />
  )
}

function FieldLabel({
  className,
  ...props
}: React.ComponentProps<typeof Label>) {
  return (
    <Label
      data-slot="field-label"
      className={cn(
        'group/field-label peer/field-label flex w-fit gap-2 leading-snug group-data-[disabled=true]/
        'has-[>[data-slot=field]]:w-full has-[>[data-slot=field]]:flex-col has-[>[data-slot=field]]:
        'has-data-[state=checked]:bg-primary/5 has-data-[state=checked]:border-primary dark:has-data
        className,
      )}
      {...props}
    />
  )
}

function FieldTitle({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="field-label"
      className={cn(
        'flex w-fit items-center gap-2 text-sm leading-snug font-medium group-data-[disabled=true]/fi
        className,
      )}
      {...props}
    />
  )
}

function FieldDescription({ className, ...props }: React.ComponentProps<'p'>) {
  return (
    <p
      data-slot="field-description"
      className={cn(
        'text-muted-foreground text-sm leading-normal font-normal group-has-[[data-orientation=horiz
        'last:mt-0 nth-last-2:-mt-1 [[data-variant=legend]+&]:-mt-1.5',
        '[&>a:hover]:text-primary [&>a]:underline [&>a]:underline-offset-4',
        className,
      )}
      {...props}
    />
  )
}

function FieldSeparator({
  children,
  className,
  ...props
}: React.ComponentProps<'div'> & {
  children?: React.ReactNode
}) {
  return (
    <div
      data-slot="field-separator"
      data-content={!!children}
      className={cn(
        'relative -my-2 h-5 text-sm group-data-[variant=outline]/field-group:-mb-2',
        className,
      )}
      {...props}
    >
      <Separator className="absolute inset-0 top-1/2" />
      {children && (
        <span
          className="bg-background text-muted-foreground relative mx-auto block w-fit px-2"
          data-slot="field-separator-content"
        >
          {children}
```

/mnt/data/cleaned_project/components/ui/form.tsx

```
'use client'

import * as React from 'react'
import * as LabelPrimitive from '@radix-ui/react-label'
import { Slot } from '@radix-ui/react-slot'
import {
  Controller,
  FormProvider,
  useFormContext,
  useFormState,
  type ControllerProps,
  type FieldPath,
  type FieldValues,
} from 'react-hook-form'

import { cn } from '@/lib/utils'
import { Label } from '@/components/ui/label'

const Form = FormProvider

type FormFieldContextValue<
  TFieldValues extends FieldValues = FieldValues,
  TName extends FieldPath<TFieldValues> = FieldPath<TFieldValues>,
> = {
  name: TName
}

const FormFieldContext = React.createContext<FormFieldContextValue>(
  {} as FormFieldContextValue,
)

const FormField = <
  TFieldValues extends FieldValues = FieldValues,
  TName extends FieldPath<TFieldValues> = FieldPath<TFieldValues>,
>({
  ...props
}: ControllerProps<TFieldValues, TName>) => {
  return (
    <FormFieldContext.Provider value={{ name: props.name }}>
      <Controller {...props} />
    </FormFieldContext.Provider>
  )
}

const useFormField = () => {
  const fieldContext = React.useContext(FormFieldContext)
  const itemContext = React.useContext(FormItemContext)
  const { getFieldState } = useFormContext()
  const formState = useFormState({ name: fieldContext.name })
  const fieldState = getFieldState(fieldContext.name, formState)

  if (!fieldContext) {
    throw new Error('useFormField should be used within <FormField>')
  }

  const { id } = itemContext

  return {
    id,
    name: fieldContext.name,
    formItemId: `${id}-form-item`,
    formDescriptionId: `${id}-form-item-description`,
    formMessageId: `${id}-form-item-message`,
    ...fieldState,
  }
}

type FormItemContextValue = {
  id: string
}
const FormItemContext = React.createContext<FormItemContextValue>(
```

```
  {} as FormItemContextValue,
)

function FormItem({ className, ...props }: React.ComponentProps<'div'>) {
  const id = React.useId()

  return (
    <FormItemContext.Provider value={{ id }}>
      <div
        data-slot="form-item"
        className={cn('grid gap-2', className)}
        {...props}
      />
    </FormItemContext.Provider>
  )
}

function FormLabel({
  className,
  ...props
}: React.ComponentProps<typeof LabelPrimitive.Root>) {
  const { error, formItemId } = useFormField()

  return (
    <Label
      data-slot="form-label"
      data-error={!!error}
      className={cn('data-[error=true]:text-destructive', className)}
      htmlFor={formItemId}
      {...props}
    />
  )
}

function FormControl({ ...props }: React.ComponentProps<typeof Slot>) {
  const { error, formItemId, formDescriptionId, formMessageId } = useFormField()

  return (
    <Slot
      data-slot="form-control"
      id={formItemId}
      aria-describedby={
        !error
          ? `${formDescriptionId}`
          : `${formDescriptionId} ${formMessageId}`
      }
      aria-invalid={!!error}
      {...props}
    />
  )
}

function FormDescription({ className, ...props }: React.ComponentProps<'p'>) {
  const { formDescriptionId } = useFormField()

  return (
    <p
      data-slot="form-description"
      id={formDescriptionId}
      className={cn('text-muted-foreground text-sm', className)}
      {...props}
    />
  )
}

function FormMessage({ className, ...props }: React.ComponentProps<'p'>) {
  const { error, formMessageId } = useFormField()
  const body = error ? String(error?.message ?? '') : props.children

  if (!body) {
    return null
  }

  return (
    <p
      data-slot="form-message"
      id={formMessageId}
```

```
      className={cn('text-destructive text-sm', className)}
      {...props}
    >
      {body}
    </p>
  )
}

export {
  useFormField,
  Form,
  FormItem,
  FormLabel,
  FormControl,
  FormDescription,
  FormMessage,
  FormField,
}
```

/mnt/data/cleaned_project/components/ui/hover-card.tsx

```
'use client'

import * as React from 'react'
import * as HoverCardPrimitive from '@radix-ui/react-hover-card'

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

function HoverCard({
  ...props
}: React.ComponentProps<typeof HoverCardPrimitive.Root>) {
  return <HoverCardPrimitive.Root data-slot="hover-card" {...props} />
}

function HoverCardTrigger({
  ...props
}: React.ComponentProps<typeof HoverCardPrimitive.Trigger>) {
  return (
    <HoverCardPrimitive.Trigger data-slot="hover-card-trigger" {...props} />
  )
}

function HoverCardContent({
  className,
  align = 'center',
  sideOffset = 4,
  ...props
}: React.ComponentProps<typeof HoverCardPrimitive.Content>) {
  return (
    <HoverCardPrimitive.Portal data-slot="hover-card-portal">
      <HoverCardPrimitive.Content
        data-slot="hover-card-content"
        align={align}
        sideOffset={sideOffset}
        className={cn(
          'bg-popover text-popover-foreground data-[state=open]:animate-in data-[state=closed]:anima
          className,
        )}
        {...props}
      />
    </HoverCardPrimitive.Portal>
  )
}

export { HoverCard, HoverCardTrigger, HoverCardContent }
```

/mnt/data/cleaned_project/components/ui/input-group.tsx

```
'use client'
import { cva, type VariantProps } from 'class-variance-authority'
```

```tsx
import { cn } from '@/lib/utils'
import { Button } from '@/components/ui/button'
import { Input } from '@/components/ui/input'
import { Textarea } from '@/components/ui/textarea'

function InputGroup({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="input-group"
      role="group"
      className={cn(
        'group/input-group border-input dark:bg-input/30 relative flex w-full items-center rounded-m
        'h-9 has-[>textarea]:h-auto',

        // Variants based on alignment.
        'has-[>[data-align=inline-start]]:[&>input]:pl-2',
        'has-[>[data-align=inline-end]]:[&>input]:pr-2',
        'has-[>[data-align=block-start]]:h-auto has-[>[data-align=block-start]]:flex-col has-[>[data
        'has-[>[data-align=block-end]]:h-auto has-[>[data-align=block-end]]:flex-col has-[>[data-alig

        // Focus state.
        'has-[[data-slot=input-group-control]:focus-visible]:border-ring has-[[data-slot=input-group

        // Error state.
        'has-[[data-slot][aria-invalid=true]]:ring-destructive/20 has-[[data-slot][aria-invalid=true]

        className,
      )}
      {...props}
    />
  )
}

const inputGroupAddonVariants = cva(
  "text-muted-foreground flex h-auto cursor-text items-center justify-center gap-2 py-1.5 text-sm fo
  {
    variants: {
      align: {
        'inline-start':
          'order-first pl-3 has-[>button]:ml-[-0.45rem] has-[>kbd]:ml-[-0.35rem]',
        'inline-end':
          'order-last pr-3 has-[>button]:mr-[-0.4rem] has-[>kbd]:mr-[-0.35rem]',
        'block-start':
          'order-first w-full justify-start px-3 pt-3 [.border-b]:pb-3 group-has-[>input]/input-grou
        'block-end':
          'order-last w-full justify-start px-3 pb-3 [.border-t]:pt-3 group-has-[>input]/input-group
      },
    },
    defaultVariants: {
      align: 'inline-start',
    },
  },
)

function InputGroupAddon({
  className,
  align = 'inline-start',
  ...props
}: React.ComponentProps<'div'> & VariantProps<typeof inputGroupAddonVariants>) {
  return (
    <div
      role="group"
      data-slot="input-group-addon"
      data-align={align}
      className={cn(inputGroupAddonVariants({ align }), className)}
      onClick={(e) => {
        if ((e.target as HTMLElement).closest('button')) {
          return
        }
        e.currentTarget.parentElement?.querySelector('input')?.focus()
      }}
      {...props}
    />
  )
}

const inputGroupButtonVariants = cva(
```

```
  'text-sm shadow-none flex gap-2 items-center',
  {
    variants: {
      size: {
        xs: "h-6 gap-1 px-2 rounded-[calc(var(--radius)-5px)] [&>svg:not([class*='size-'])]:size-3.5
        sm: 'h-8 px-2.5 gap-1.5 rounded-md has-[>svg]:px-2.5',
        'icon-xs':
          'size-6 rounded-[calc(var(--radius)-5px)] p-0 has-[>svg]:p-0',
        'icon-sm': 'size-8 p-0 has-[>svg]:p-0',
      },
    },
    defaultVariants: {
      size: 'xs',
    },
  },
)

function InputGroupButton({
  className,
  type = 'button',
  variant = 'ghost',
  size = 'xs',
  ...props
}: Omit<React.ComponentProps<typeof Button>, 'size'> &
  VariantProps<typeof inputGroupButtonVariants>) {
  return (
    <Button
      type={type}
      data-size={size}
      variant={variant}
      className={cn(inputGroupButtonVariants({ size }), className)}
      {...props}
    />
  )
}

function InputGroupText({ className, ...props }: React.ComponentProps<'span'>) {
  return (
    <span
      className={cn(
        "text-muted-foreground flex items-center gap-2 text-sm [&_svg]:pointer-events-none [&_svg:no
        className,
      )}
      {...props}
    />
  )
}

function InputGroupInput({
  className,
  ...props
}: React.ComponentProps<'input'>) {
  return (
    <Input
      data-slot="input-group-control"
      className={cn(
        'flex-1 rounded-none border-0 bg-transparent shadow-none focus-visible:ring-0 dark:bg-transpa
        className,
      )}
      {...props}
    />
  )
}

function InputGroupTextarea({
  className,
  ...props
}: React.ComponentProps<'textarea'>) {
  return (
    <Textarea
      data-slot="input-group-control"
      className={cn(
        'flex-1 resize-none rounded-none border-0 bg-transparent py-3 shadow-none focus-visible:ring
        className,
      )}
      {...props}
    />
```

```
  )
}

export {
  InputGroup,
  InputGroupAddon,
  InputGroupButton,
  InputGroupText,
  InputGroup
```

/mnt/data/cleaned_project/components/ui/input-otp.tsx

```
'use client'

import * as React from 'react'
import { OTPInput, OTPInputContext } from 'input-otp'
import { MinusIcon } from 'lucide-react'

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

function InputOTP({
  className,
  containerClassName,
  ...props
}: React.ComponentProps<typeof OTPInput> & {
  containerClassName?: string
}) {
  return (
    <OTPInput
      data-slot="input-otp"
      containerClassName={cn(
        'flex items-center gap-2 has-disabled:opacity-50',
        containerClassName,
      )}
      className={cn('disabled:cursor-not-allowed', className)}
      {...props}
    />
  )
}

function InputOTPGroup({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="input-otp-group"
      className={cn('flex items-center', className)}
      {...props}
    />
  )
}

function InputOTPSlot({
  index,
  className,
  ...props
}: React.ComponentProps<'div'> & {
  index: number
}) {
  const inputOTPContext = React.useContext(OTPInputContext)
  const { char, hasFakeCaret, isActive } = inputOTPContext?.slots[index] ?? {}

  return (
    <div
      data-slot="input-otp-slot"
      data-active={isActive}
      className={cn(
        'data-[active=true]:border-ring data-[active=true]:ring-ring/50 data-[active=true]:aria-inva
        className,
      )}
      {...props}
    >
      {char}
      {hasFakeCaret && (
        <div className="pointer-events-none absolute inset-0 flex items-center justify-center">
          <div className="animate-caret-blink bg-foreground h-4 w-px duration-1000" />
```

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

function InputOTPSeparator({ ...props }: React.ComponentProps<'div'>) {
  return (
    <div data-slot="input-otp-separator" role="separator" {...props}>
      <MinusIcon />
    </div>
  )
}

export { InputOTP, InputOTPGroup, InputOTPSlot, InputOTPSeparator }
```

/mnt/data/cleaned_project/components/ui/input.tsx

```
import * as React from 'react'

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

function Input({ className, type, ...props }: React.ComponentProps<'input'>) {
  return (
    <input
      type={type}
      data-slot="input"
      className={cn(
        'file:text-foreground placeholder:text-muted-foreground selection:bg-primary selection:text-p
        'focus-visible:border-ring focus-visible:ring-ring/50 focus-visible:ring-[3px]',
        'aria-invalid:ring-destructive/20 dark:aria-invalid:ring-destructive/40 aria-invalid:border-d
        className,
      )}
      {...props}
    />
  )
}

export { Input }
```

/mnt/data/cleaned_project/components/ui/item.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'
import { Separator } from '@/components/ui/separator'

function ItemGroup({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      role="list"
      data-slot="item-group"
      className={cn('group/item-group flex flex-col', className)}
      {...props}
    />
  )
}

function ItemSeparator({
  className,
  ...props
}: React.ComponentProps<typeof Separator>) {
  return (
    <Separator
      data-slot="item-separator"
      orientation="horizontal"
      className={cn('my-0', className)}
      {...props}
```

```
    />
  )
}

const itemVariants = cva(
  'group/item flex items-center border border-transparent text-sm rounded-md transition-colors [a&]:h
  {
    variants: {
      variant: {
        default: 'bg-transparent',
        outline: 'border-border',
        muted: 'bg-muted/50',
      },
      size: {
        default: 'p-4 gap-4 ',
        sm: 'py-3 px-4 gap-2.5',
      },
    },
    defaultVariants: {
      variant: 'default',
      size: 'default',
    },
  },
)

function Item({
  className,
  variant = 'default',
  size = 'default',
  asChild = false,
  ...props
}: React.ComponentProps<'div'> &
  VariantProps<typeof itemVariants> & { asChild?: boolean }) {
  const Comp = asChild ? Slot : 'div'
  return (
    <Comp
      data-slot="item"
      data-variant={variant}
      data-size={size}
      className={cn(itemVariants({ variant, size, className }))}
      {...props}
    />
  )
}

const itemMediaVariants = cva(
  'flex shrink-0 items-center justify-center gap-2 group-has-[[data-slot=item-description]]/item:self
  {
    variants: {
      variant: {
        default: 'bg-transparent',
        icon: "size-8 border rounded-sm bg-muted [&_svg:not([class*='size-'])]:size-4",
        image:
          'size-10 rounded-sm overflow-hidden [&_img]:size-full [&_img]:object-cover',
      },
    },
    defaultVariants: {
      variant: 'default',
    },
  },
)

function ItemMedia({
  className,
  variant = 'default',
  ...props
}: React.ComponentProps<'div'> & VariantProps<typeof itemMediaVariants>) {
  return (
    <div
      data-slot="item-media"
      data-variant={variant}
      className={cn(itemMediaVariants({ variant, className }))}
      {...props}
    />
  )
}
function ItemContent({ className, ...props }: React.ComponentProps<'div'>) {
```

```
  return (
    <div
      data-slot="item-content"
      className={cn(
        'flex flex-1 flex-col gap-1 [&+[data-slot=item-content]]:flex-none',
        className,
      )}
      {...props}
    />
  )
}

function ItemTitle({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="item-title"
      className={cn(
        'flex w-fit items-center gap-2 text-sm leading-snug font-medium',
        className,
      )}
      {...props}
    />
  )
}

function ItemDescription({ className, ...props }: React.ComponentProps<'p'>) {
  return (
    <p
      data-slot="item-description"
      className={cn(
        'text-muted-foreground line-clamp-2 text-sm leading-normal font-normal text-balance',
        '[&>a:hover]:text-primary [&>a]:underline [&>a]:underline-offset-4',
        className,
      )}
      {...props}
    />
  )
}

function ItemActions({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="item-actions"
      className={cn('flex items-center gap-2', className)}
      {...props}
    />
  )
}

function ItemHeader({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="item-header"
      className={cn(
        'flex basis-full items-center justify-between gap-2',
        className,
      )}
      {...props}
    />
  )
}

function ItemFooter({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="item-footer"
      className={cn(
        'flex basis-full items-center justify-between gap-2',
        className,
      )}
      {...props}
    />
  )
}

export {
  Item,
```

```
  ItemMedia,
  ItemContent,
  ItemActions,
  ItemGroup,
  ItemSeparator,
  ItemTitle,
  ItemDescription,
  ItemHeader,
  ItemFooter,
}
```

/mnt/data/cleaned_project/components/ui/kbd.tsx

```
import { cn } from '@/lib/utils'

function Kbd({ className, ...props }: React.ComponentProps<'kbd'>) {
  return (
    <kbd
      data-slot="kbd"
      className={cn(
        'bg-muted w-fit text-muted-foreground pointer-events-none inline-flex h-5 min-w-5 items-cente
        "[&_svg:not([class*='size-'])]:size-3",
        '[[data-slot=tooltip-content]_&]:bg-background/20 [[data-slot=tooltip-content]_&]:text-backgr
        className,
      )}
      {...props}
    />
  )
}

function KbdGroup({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <kbd
      data-slot="kbd-group"
      className={cn('inline-flex items-center gap-1', className)}
      {...props}
    />
  )
}

export { Kbd, KbdGroup }
```

/mnt/data/cleaned_project/components/ui/label.tsx

```
'use client'

import * as React from 'react'
import * as LabelPrimitive from '@radix-ui/react-label'

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

function Label({
  className,
  ...props
}: React.ComponentProps<typeof LabelPrimitive.Root>) {
  return (
    <LabelPrimitive.Root
      data-slot="label"
      className={cn(
        'flex items-center gap-2 text-sm leading-none font-medium select-none group-data-[disabled=tr
        className,
      )}
      {...props}
    />
  )
}

export { Label }
```

/mnt/data/cleaned_project/components/ui/menubar.tsx

```
'use client'

import * as React from 'react'
import * as MenubarPrimitive from '@radix-ui/react-menubar'
import { CheckIcon, ChevronRightIcon, CircleIcon } from 'lucide-react'

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

function Menubar({
  className,
  ...props
}: React.ComponentProps<typeof MenubarPrimitive.Root>) {
  return (
    <MenubarPrimitive.Root
      data-slot="menubar"
      className={cn(
        'bg-background flex h-9 items-center gap-1 rounded-md border p-1 shadow-xs',
        className,
      )}
      {...props}
    />
  )
}

function MenubarMenu({
  ...props
}: React.ComponentProps<typeof MenubarPrimitive.Menu>) {
  return <MenubarPrimitive.Menu data-slot="menubar-menu" {...props} />
}

function MenubarGroup({
  ...props
}: React.ComponentProps<typeof MenubarPrimitive.Group>) {
  return <MenubarPrimitive.Group data-slot="menubar-group" {...props} />
}

function MenubarPortal({
  ...props
}: React.ComponentProps<typeof MenubarPrimitive.Portal>) {
  return <MenubarPrimitive.Portal data-slot="menubar-portal" {...props} />
}

function MenubarRadioGroup({
  ...props
}: React.ComponentProps<typeof MenubarPrimitive.RadioGroup>) {
  return (
    <MenubarPrimitive.RadioGroup data-slot="menubar-radio-group" {...props} />
  )
}

function MenubarTrigger({
  className,
  ...props
}: React.ComponentProps<typeof MenubarPrimitive.Trigger>) {
  return (
    <MenubarPrimitive.Trigger
      data-slot="menubar-trigger"
      className={cn(
        'focus:bg-accent focus:text-accent-foreground data-[state=open]:bg-accent data-[state=open]:
        className,
      )}
      {...props}
    />
  )
}

function MenubarContent({
  className,
  align = 'start',
  alignOffset = -4,
  sideOffset = 8,
  ...props
}: React.ComponentProps<typeof MenubarPrimitive.Content>) {
  return (
```

```
    <MenubarPortal>
      <MenubarPrimitive.Content
        data-slot="menubar-content"
        align={align}
        alignOffset={alignOffset}
        sideOffset={sideOffset}
        className={cn(
          'bg-popover text-popover-foreground data-[state=open]:animate-in data-[state=closed]:fade-
          className,
        )}
        {...props}
      />
    </MenubarPortal>
  )
}

function MenubarItem({
  className,
  inset,
  variant = 'default',
  ...props
}: React.ComponentProps<typeof MenubarPrimitive.Item> & {
  inset?: boolean
  variant?: 'default' | 'destructive'
}) {
  return (
    <MenubarPrimitive.Item
      data-slot="menubar-item"
      data-inset={inset}
      data-variant={variant}
      className={cn(
        "focus:bg-accent focus:text-accent-foreground data-[variant=destructive]:text-destructive da
        className,
      )}
      {...props}
    />
  )
}

function MenubarCheckboxItem({
  className,
  children,
  checked,
  ...props
}: React.ComponentProps<typeof MenubarPrimitive.CheckboxItem>) {
  return (
    <MenubarPrimitive.CheckboxItem
      data-slot="menubar-checkbox-item"
      className={cn(
        "focus:bg-accent focus:text-accent-foreground relative flex cursor-default items-center gap-
        className,
      )}
      checked={checked}
      {...props}
    >
      <span className="pointer-events-none absolute left-2 flex size-3.5 items-center justify-center
        <MenubarPrimitive.ItemIndicator>
          <CheckIcon className="size-4" />
        </MenubarPrimitive.ItemIndicator>
      </span>
      {children}
    </MenubarPrimitive.CheckboxItem>
  )
}

function MenubarRadioItem({
  className,
  children,
  ...props
}: React.ComponentProps<typeof MenubarPrimitive.RadioItem>) {
  return (
    <MenubarPrimitive.RadioItem
      data-slot="menubar-radio-item"
      className={cn(
        "focus:bg-accent focus:text-accent-foreground relative flex cursor-default items-center gap-
```

/mnt/data/cleaned_project/components/ui/navigation-menu.tsx

```
import * as React from 'react'
import * as NavigationMenuPrimitive from '@radix-ui/react-navigation-menu'
import { cva } from 'class-variance-authority'
import { ChevronDownIcon } from 'lucide-react'

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

function NavigationMenu({
  className,
  children,
  viewport = true,
  ...props
}: React.ComponentProps<typeof NavigationMenuPrimitive.Root> & {
  viewport?: boolean
}) {
  return (
    <NavigationMenuPrimitive.Root
      data-slot="navigation-menu"
      data-viewport={viewport}
      className={cn(
        'group/navigation-menu relative flex max-w-max flex-1 items-center justify-center',
        className,
      )}
      {...props}
    >
      {children}
      {viewport && <NavigationMenuViewport />}
    </NavigationMenuPrimitive.Root>
  )
}

function NavigationMenuList({
  className,
  ...props
}: React.ComponentProps<typeof NavigationMenuPrimitive.List>) {
  return (
    <NavigationMenuPrimitive.List
      data-slot="navigation-menu-list"
      className={cn(
        'group flex flex-1 list-none items-center justify-center gap-1',
        className,
      )}
      {...props}
    />
  )
}

function NavigationMenuItem({
  className,
  ...props
}: React.ComponentProps<typeof NavigationMenuPrimitive.Item>) {
  return (
    <NavigationMenuPrimitive.Item
      data-slot="navigation-menu-item"
      className={cn('relative', className)}
      {...props}
    />
  )
}

const navigationMenuTriggerStyle = cva(
  'group inline-flex h-9 w-max items-center justify-center rounded-md bg-background px-4 py-2 text-sm
)

function NavigationMenuTrigger({
  className,
  children,
  ...props
}: React.ComponentProps<typeof NavigationMenuPrimitive.Trigger>) {
  return (
    <NavigationMenuPrimitive.Trigger
      data-slot="navigation-menu-trigger"
      className={cn(navigationMenuTriggerStyle(), 'group', className)}
      {...props}
```

```
    >
      {children}{' '}
      <ChevronDownIcon
        className="relative top-[1px] ml-1 size-3 transition duration-300 group-data-[state=open]:ro
        aria-hidden="true"
      />
    </NavigationMenuPrimitive.Trigger>
  )
}

function NavigationMenuContent({
  className,
  ...props
}: React.ComponentProps<typeof NavigationMenuPrimitive.Content>) {
  return (
    <NavigationMenuPrimitive.Content
      data-slot="navigation-menu-content"
      className={cn(
        'data-[motion^=from-]:animate-in data-[motion^=to-]:animate-out data-[motion^=from-]:fade-in
        'group-data-[viewport=false]/navigation-menu:bg-popover group-data-[viewport=false]/navigatio
        className,
      )}
      {...props}
    />
  )
}

function NavigationMenuViewport({
  className,
  ...props
}: React.ComponentProps<typeof NavigationMenuPrimitive.Viewport>) {
  return (
    <div
      className={'absolute top-full left-0 isolate z-50 flex justify-center'}
    >
      <NavigationMenuPrimitive.Viewport
        data-slot="navigation-menu-viewport"
        className={cn(
          'origin-top-center bg-popover text-popover-foreground data-[state=open]:animate-in data-[st
          className,
        )}
        {...props}
```

/mnt/data/cleaned_project/components/ui/pagination.tsx

```
import * as React from 'react'
import {
  ChevronLeftIcon,
  ChevronRightIcon,
  MoreHorizontalIcon,
} from 'lucide-react'

import { cn } from '@/lib/utils'
import { Button, buttonVariants } from '@/components/ui/button'

function Pagination({ className, ...props }: React.ComponentProps<'nav'>) {
  return (
    <nav
      role="navigation"
      aria-label="pagination"
      data-slot="pagination"
      className={cn('mx-auto flex w-full justify-center', className)}
      {...props}
    />
  )
}

function PaginationContent({
  className,
  ...props
}: React.ComponentProps<'ul'>) {
  return (
    <ul
      data-slot="pagination-content"
```

```
      className={cn('flex flex-row items-center gap-1', className)}
      {...props}
    />
  )
}

function PaginationItem({ ...props }: React.ComponentProps<'li'>) {
  return <li data-slot="pagination-item" {...props} />
}

type PaginationLinkProps = {
  isActive?: boolean
} & Pick<React.ComponentProps<typeof Button>, 'size'> &
  React.ComponentProps<'a'>

function PaginationLink({
  className,
  isActive,
  size = 'icon',
  ...props
}: PaginationLinkProps) {
  return (
    <a
      aria-current={isActive ? 'page' : undefined}
      data-slot="pagination-link"
      data-active={isActive}
      className={cn(
        buttonVariants({
          variant: isActive ? 'outline' : 'ghost',
          size,
        }),
        className,
      )}
      {...props}
    />
  )
}

function PaginationPrevious({
  className,
  ...props
}: React.ComponentProps<typeof PaginationLink>) {
  return (
    <PaginationLink
      aria-label="Go to previous page"
      size="default"
      className={cn('gap-1 px-2.5 sm:pl-2.5', className)}
      {...props}
    >
      <ChevronLeftIcon />
      <span className="hidden sm:block">Previous</span>
    </PaginationLink>
  )
}

function PaginationNext({
  className,
  ...props
}: React.ComponentProps<typeof PaginationLink>) {
  return (
    <PaginationLink
      aria-label="Go to next page"
      size="default"
      className={cn('gap-1 px-2.5 sm:pr-2.5', className)}
      {...props}
    >
      <span className="hidden sm:block">Next</span>
      <ChevronRightIcon />
    </PaginationLink>
  )
}

function PaginationEllipsis({
  className,
  ...props
}: React.ComponentProps<'span'>) {
  return (
```

```
    <span
      aria-hidden
      data-slot="pagination-ellipsis"
      className={cn('flex size-9 items-center justify-center', className)}
      {...props}
    >
      <MoreHorizontalIcon className="size-4" />
      <span className="sr-only">More pages</span>
    </span>
  )
}

export {
  Pagination,
  PaginationContent,
  PaginationLink,
  PaginationItem,
  PaginationPrevious,
  PaginationNext,
  PaginationEllipsis,
}
```

/mnt/data/cleaned_project/components/ui/popover.tsx

```
'use client'

import * as React from 'react'
import * as PopoverPrimitive from '@radix-ui/react-popover'

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

function Popover({
  ...props
}: React.ComponentProps<typeof PopoverPrimitive.Root>) {
  return <PopoverPrimitive.Root data-slot="popover" {...props} />
}

function PopoverTrigger({
  ...props
}: React.ComponentProps<typeof PopoverPrimitive.Trigger>) {
  return <PopoverPrimitive.Trigger data-slot="popover-trigger" {...props} />
}

function PopoverContent({
  className,
  align = 'center',
  sideOffset = 4,
  ...props
}: React.ComponentProps<typeof PopoverPrimitive.Content>) {
  return (
    <PopoverPrimitive.Portal>
      <PopoverPrimitive.Content
        data-slot="popover-content"
        align={align}
        sideOffset={sideOffset}
        className={cn(
          'bg-popover text-popover-foreground data-[state=open]:animate-in data-[state=closed]:anima
          className,
        )}
        {...props}
      />
    </PopoverPrimitive.Portal>
  )
}

function PopoverAnchor({
  ...props
}: React.ComponentProps<typeof PopoverPrimitive.Anchor>) {
  return <PopoverPrimitive.Anchor data-slot="popover-anchor" {...props} />
}

export { Popover, PopoverTrigger, PopoverContent, PopoverAnchor }
```

/mnt/data/cleaned_project/components/ui/progress.tsx

```
'use client'

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

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

function Progress({
  className,
  value,
  ...props
}: React.ComponentProps<typeof ProgressPrimitive.Root>) {
  return (
    <ProgressPrimitive.Root
      data-slot="progress"
      className={cn(
        'bg-primary/20 relative h-2 w-full overflow-hidden rounded-full',
        className,
      )}
      {...props}
    >
      <ProgressPrimitive.Indicator
        data-slot="progress-indicator"
        className="bg-primary h-full w-full flex-1 transition-all"
        style={{ transform: `translateX(-${100 - (value || 0)}%)` }}
      />
    </ProgressPrimitive.Root>
  )
}

export { Progress }
```

/mnt/data/cleaned_project/components/ui/radio-group.tsx

```
'use client'

import * as React from 'react'
import * as RadioGroupPrimitive from '@radix-ui/react-radio-group'
import { CircleIcon } from 'lucide-react'

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

function RadioGroup({
  className,
  ...props
}: React.ComponentProps<typeof RadioGroupPrimitive.Root>) {
  return (
    <RadioGroupPrimitive.Root
      data-slot="radio-group"
      className={cn('grid gap-3', className)}
      {...props}
    />
  )
}

function RadioGroupItem({
  className,
  ...props
}: React.ComponentProps<typeof RadioGroupPrimitive.Item>) {
  return (
    <RadioGroupPrimitive.Item
      data-slot="radio-group-item"
      className={cn(
        'border-input text-primary focus-visible:border-ring focus-visible:ring-ring/50 aria-invalid
        className,
      )}
      {...props}
    >
      <RadioGroupPrimitive.Indicator
        data-slot="radio-group-indicator"
        className="relative flex items-center justify-center"
```

```
      >
        <CircleIcon className="fill-primary absolute top-1/2 left-1/2 size-2 -translate-x-1/2 -trans
      </RadioGroupPrimitive.Indicator>
    </RadioGroupPrimitive.Item>
  )
}

export { RadioGroup, RadioGroupItem }
```

/mnt/data/cleaned_project/components/ui/resizable.tsx

```
'use client'

import * as React from 'react'
import { GripVerticalIcon } from 'lucide-react'
import * as ResizablePrimitive from 'react-resizable-panels'

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

function ResizablePanelGroup({
  className,
  ...props
}: React.ComponentProps<typeof ResizablePrimitive.PanelGroup>) {
  return (
    <ResizablePrimitive.PanelGroup
      data-slot="resizable-panel-group"
      className={cn(
        'flex h-full w-full data-[panel-group-direction=vertical]:flex-col',
        className,
      )}
      {...props}
    />
  )
}

function ResizablePanel({
  ...props
}: React.ComponentProps<typeof ResizablePrimitive.Panel>) {
  return <ResizablePrimitive.Panel data-slot="resizable-panel" {...props} />
}

function ResizableHandle({
  withHandle,
  className,
  ...props
}: React.ComponentProps<typeof ResizablePrimitive.PanelResizeHandle> & {
  withHandle?: boolean
}) {
  return (
    <ResizablePrimitive.PanelResizeHandle
      data-slot="resizable-handle"
      className={cn(
        'bg-border focus-visible:ring-ring relative flex w-px items-center justify-center after:abso
        className,
      )}
      {...props}
    >
      {withHandle && (
        <div className="bg-border z-10 flex h-4 w-3 items-center justify-center rounded-xs border">
          <GripVerticalIcon className="size-2.5" />
        </div>
      )}
    </ResizablePrimitive.PanelResizeHandle>
  )
}

export { ResizablePanelGroup, ResizablePanel, ResizableHandle }
```

/mnt/data/cleaned_project/components/ui/scroll-area.tsx

```
'use client'

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

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

function ScrollArea({
  className,
  children,
  ...props
}: React.ComponentProps<typeof ScrollAreaPrimitive.Root>) {
  return (
    <ScrollAreaPrimitive.Root
      data-slot="scroll-area"
      className={cn('relative', className)}
      {...props}
    >
      <ScrollAreaPrimitive.Viewport
        data-slot="scroll-area-viewport"
        className="focus-visible:ring-ring/50 size-full rounded-[inherit] transition-[color,box-shado
      >
        {children}
      </ScrollAreaPrimitive.Viewport>
      <ScrollBar />
      <ScrollAreaPrimitive.Corner />
    </ScrollAreaPrimitive.Root>
  )
}

function ScrollBar({
  className,
  orientation = 'vertical',
  ...props
}: React.ComponentProps<typeof ScrollAreaPrimitive.ScrollAreaScrollbar>) {
  return (
    <ScrollAreaPrimitive.ScrollAreaScrollbar
      data-slot="scroll-area-scrollbar"
      orientation={orientation}
      className={cn(
        'flex touch-none p-px transition-colors select-none',
        orientation === 'vertical' &&
          'h-full w-2.5 border-l border-l-transparent',
        orientation === 'horizontal' &&
          'h-2.5 flex-col border-t border-t-transparent',
        className,
      )}
      {...props}
    >
      <ScrollAreaPrimitive.ScrollAreaThumb
        data-slot="scroll-area-thumb"
        className="bg-border relative flex-1 rounded-full"
      />
    </ScrollAreaPrimitive.ScrollAreaScrollbar>
  )
}

export { ScrollArea, ScrollBar }
```

/mnt/data/cleaned_project/components/ui/select.tsx

```
'use client'

import * as React from 'react'
import * as SelectPrimitive from '@radix-ui/react-select'
import { CheckIcon, ChevronDownIcon, ChevronUpIcon } from 'lucide-react'

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

function Select({
  ...props
}: React.ComponentProps<typeof SelectPrimitive.Root>) {
  return <SelectPrimitive.Root data-slot="select" {...props} />
}
```

```
function SelectGroup({
  ...props
}: React.ComponentProps<typeof SelectPrimitive.Group>) {
  return <SelectPrimitive.Group data-slot="select-group" {...props} />
}

function SelectValue({
  ...props
}: React.ComponentProps<typeof SelectPrimitive.Value>) {
  return <SelectPrimitive.Value data-slot="select-value" {...props} />
}

function SelectTrigger({
  className,
  size = 'default',
  children,
  ...props
}: React.ComponentProps<typeof SelectPrimitive.Trigger> & {
  size?: 'sm' | 'default'
}) {
  return (
    <SelectPrimitive.Trigger
      data-slot="select-trigger"
      data-size={size}
      className={cn(
        "border-input data-[placeholder]:text-muted-foreground [&_svg:not([class*='text-'])]:text-mu
        className,
      )}
      {...props}
    >
      {children}
      <SelectPrimitive.Icon asChild>
        <ChevronDownIcon className="size-4 opacity-50" />
      </SelectPrimitive.Icon>
    </SelectPrimitive.Trigger>
  )
}

function SelectContent({
  className,
  children,
  position = 'popper',
  ...props
}: React.ComponentProps<typeof SelectPrimitive.Content>) {
  return (
    <SelectPrimitive.Portal>
      <SelectPrimitive.Content
        data-slot="select-content"
        className={cn(
          'bg-popover text-popover-foreground data-[state=open]:animate-in data-[state=closed]:anima
          position === 'popper' &&
            'data-[side=bottom]:translate-y-1 data-[side=left]:-translate-x-1 data-[side=right]:trans
          className,
        )}
        position={position}
        {...props}
      >
        <SelectScrollUpButton />
        <SelectPrimitive.Viewport
          className={cn(
            'p-1',
            position === 'popper' &&
              'h-[var(--radix-select-trigger-height)] w-full min-w-[var(--radix-select-trigger-width
          )}
        >
          {children}
        </SelectPrimitive.Viewport>
        <SelectScrollDownButton />
      </SelectPrimitive.Content>
    </SelectPrimitive.Portal>
  )
}

function SelectLabel({
  className,
  ...props
}: React.ComponentProps<typeof SelectPrimitive.Label>) {
```

```
  return (
    <SelectPrimitive.Label
      data-slot="select-label"
      className={cn('text-muted-foreground px-2 py-1.5 text-xs', className)}
      {...props}
    />
  )
}

function SelectItem({
  className,
  children,
  ...props
}: React.ComponentProps<typeof SelectPrimitive.Item>) {
  return (
    <SelectPrimitive.Item
      data-slot="select-item"
      className={cn(
        "focus:bg-accent focus:text-accent-foreground [&_svg:not([class*='text-'])]:text-muted-foregr
        className,
      )}
      {...props}
    >
      <span className="absolute right-2 flex size-3.5 items-center justify-center">
        <SelectPrimitive.ItemIndicator>
          <CheckIcon className="size-4" />
        </SelectPrimitive.ItemIndicator>
      </span>
      <SelectPrimitive.ItemText>{children}</SelectPrimitive.ItemText>
    </SelectPrimitive.Item>
  )
}

function SelectSeparator({
  className,
  ...props
}: React.ComponentProps<typeof SelectPrimitive.Separator
```

/mnt/data/cleaned_project/components/ui/separator.tsx

```
'use client'

import * as React from 'react'
import * as SeparatorPrimitive from '@radix-ui/react-separator'

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

function Separator({
  className,
  orientation = 'horizontal',
  decorative = true,
  ...props
}: React.ComponentProps<typeof SeparatorPrimitive.Root>) {
  return (
    <SeparatorPrimitive.Root
      data-slot="separator"
      decorative={decorative}
      orientation={orientation}
      className={cn(
        'bg-border shrink-0 data-[orientation=horizontal]:h-px data-[orientation=horizontal]:w-full
        className,
      )}
      {...props}
    />
  )
}

export { Separator }
```

/mnt/data/cleaned_project/components/ui/sheet.tsx

```
'use client'

import * as React from 'react'
import * as SheetPrimitive from '@radix-ui/react-dialog'
import { XIcon } from 'lucide-react'

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

function Sheet({ ...props }: React.ComponentProps<typeof SheetPrimitive.Root>) {
  return <SheetPrimitive.Root data-slot="sheet" {...props} />
}

function SheetTrigger({
  ...props
}: React.ComponentProps<typeof SheetPrimitive.Trigger>) {
  return <SheetPrimitive.Trigger data-slot="sheet-trigger" {...props} />
}

function SheetClose({
  ...props
}: React.ComponentProps<typeof SheetPrimitive.Close>) {
  return <SheetPrimitive.Close data-slot="sheet-close" {...props} />
}

function SheetPortal({
  ...props
}: React.ComponentProps<typeof SheetPrimitive.Portal>) {
  return <SheetPrimitive.Portal data-slot="sheet-portal" {...props} />
}

function SheetOverlay({
  className,
  ...props
}: React.ComponentProps<typeof SheetPrimitive.Overlay>) {
  return (
    <SheetPrimitive.Overlay
      data-slot="sheet-overlay"
      className={cn(
        'data-[state=open]:animate-in data-[state=closed]:animate-out data-[state=closed]:fade-out-0
        className,
      )}
      {...props}
    />
  )
}

function SheetContent({
  className,
  children,
  side = 'right',
  ...props
}: React.ComponentProps<typeof SheetPrimitive.Content> & {
  side?: 'top' | 'right' | 'bottom' | 'left'
}) {
  return (
    <SheetPortal>
      <SheetOverlay />
      <SheetPrimitive.Content
        data-slot="sheet-content"
        className={cn(
          'bg-background data-[state=open]:animate-in data-[state=closed]:animate-out fixed z-50 flex
          side === 'right' &&
            'data-[state=closed]:slide-out-to-right data-[state=open]:slide-in-from-right inset-y-0
          side === 'left' &&
            'data-[state=closed]:slide-out-to-left data-[state=open]:slide-in-from-left inset-y-0 le
          side === 'top' &&
            'data-[state=closed]:slide-out-to-top data-[state=open]:slide-in-from-top inset-x-0 top-0
          side === 'bottom' &&
            'data-[state=closed]:slide-out-to-bottom data-[state=open]:slide-in-from-bottom inset-x-0
          className,
        )}
        {...props}
      >
        {children}
        <SheetPrimitive.Close className="ring-offset-background focus:ring-ring data-[state=open]:bg
          <XIcon className="size-4" />
          <span className="sr-only">Close</span>
```

```
        </SheetPrimitive.Close>
      </SheetPrimitive.Content>
    </SheetPortal>
  )
}

function SheetHeader({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="sheet-header"
      className={cn('flex flex-col gap-1.5 p-4', className)}
      {...props}
    />
  )
}

function SheetFooter({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="sheet-footer"
      className={cn('mt-auto flex flex-col gap-2 p-4', className)}
      {...props}
    />
  )
}

function SheetTitle({
  className,
  ...props
}: React.ComponentProps<typeof SheetPrimitive.Title>) {
  return (
    <SheetPrimitive.Title
      data-slot="sheet-title"
      className={cn('text-foreground font-semibold', className)}
      {...props}
    />
  )
}

function SheetDescription({
  className,
  ...props
}: React.ComponentProps<typeof SheetPrimitive.Description>) {
  return (
    <SheetPrimitive.Description
      data-slot="sheet-description"
      className={cn('text-muted-foreground text-sm', className)}
      {...props}
    />
  )
}

export {
  Sheet,
  SheetTrigger,
  SheetClose,
  SheetContent,
  SheetHeader,
  SheetFooter,
  SheetTitle,
  SheetDescription,
}
```

/mnt/data/cleaned_project/components/ui/sidebar.tsx

```
'use client'

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

import { useIsMobile } from '@/hooks/use-mobile'
import { cn } from '@/lib/utils'
```

```
import { Button } from '@/components/ui/button'
import { Input } from '@/components/ui/input'
import { Separator } from '@/components/ui/separator'
import {
  Sheet,
  SheetContent,
  SheetDescription,
  SheetHeader,
  SheetTitle,
} from '@/components/ui/sheet'
import { Skeleton } from '@/components/ui/skeleton'
import {
  Tooltip,
  TooltipContent,
  TooltipProvider,
  TooltipTrigger,
} from '@/components/ui/tooltip'

const SIDEBAR_COOKIE_NAME = 'sidebar_state'
const SIDEBAR_COOKIE_MAX_AGE = 60 * 60 * 24 * 7
const SIDEBAR_WIDTH = '16rem'
const SIDEBAR_WIDTH_MOBILE = '18rem'
const SIDEBAR_WIDTH_ICON = '3rem'
const SIDEBAR_KEYBOARD_SHORTCUT = 'b'

type SidebarContextProps = {
  state: 'expanded' | 'collapsed'
  open: boolean
  setOpen: (open: boolean) => void
  openMobile: boolean
  setOpenMobile: (open: boolean) => void
  isMobile: boolean
  toggleSidebar: () => void
}

const SidebarContext = React.createContext<SidebarContextProps | null>(null)

function useSidebar() {
  const context = React.useContext(SidebarContext)
  if (!context) {
    throw new Error('useSidebar must be used within a SidebarProvider.')
  }

  return context
}

function SidebarProvider({
  defaultOpen = true,
  open: openProp,
  onOpenChange: setOpenProp,
  className,
  style,
  children,
  ...props
}: React.ComponentProps<'div'> & {
  defaultOpen?: boolean
  open?: boolean
  onOpenChange?: (open: boolean) => void
}) {
  const isMobile = useIsMobile()
  const [openMobile, setOpenMobile] = React.useState(false)

  // This is the internal state of the sidebar.
  // We use openProp and setOpenProp for control from outside the component.
  const [_open, _setOpen] = React.useState(defaultOpen)
  const open = openProp ?? _open
  const setOpen = React.useCallback(
    (value: boolean | ((value: boolean) => boolean)) => {
      const openState = typeof value === 'function' ? value(open) : value
      if (setOpenProp) {
        setOpenProp(openState)
      } else {
        _setOpen(openState)
      }

      // This sets the cookie to keep the sidebar state.
      document.cookie = `${SIDEBAR_COOKIE_NAME}=${openState}; path=/; max-age=${SIDEBAR_COOKIE_MAX_A
```

```
    },
    [setOpenProp, open],
  )

  // Helper to toggle the sidebar.
  const toggleSidebar = React.useCallback(() => {
    return isMobile ? setOpenMobile((open) => !open) : setOpen((open) => !open)
  }, [isMobile, setOpen, setOpenMobile])

  // Adds a keyboard shortcut to toggle the sidebar.
  React.useEffect(() => {
    const handleKeyDown = (event: KeyboardEvent) => {
      if (
        event.key === SIDEBAR_KEYBOARD_SHORTCUT &&
        (event.metaKey || event.ctrlKey)
      ) {
        event.preventDefault()
        toggleSidebar()
      }
    }

    window.addEventListener('keydown', handleKeyDown)
    return () => window.removeEventListener('keydown', handleKeyDown)
  }, [toggleSidebar])

  // We add a state so that we can do data-state="expanded" or "collapsed".
  // This makes it easier to style the sidebar with Tailwind classes.
  const state = open ? 'expanded' : 'collapsed'

  const contextValue = React.useMemo<SidebarContextProps>(
    () => ({
      state,
      open,
      setOpen,
      isMobile,
      openMobile,
      setOpenMobile,
      toggleSidebar,
    }),
    [state, open, setOpen, isMobile, openMobile, setOpenMobile, toggleSidebar],
  )

  return (
    <SidebarContext.Provider value={contextValue}>
      <TooltipProvider delayDuration={0}>
        <div
          data-slot="sidebar-wrapper"
          style={
            {
              '--sidebar-width': SIDEBAR_WIDTH,
              '--sidebar-width-icon': SIDEBAR_WIDTH_ICON,
              ...style,
            } as React.CSSProperties
          }
          className={cn(
            'group/sidebar-wrapper has-data-[variant=inset]:bg-sidebar flex min-h-svh w-full',
            className,
          )}
          {...props}
        >
          {children}
        </div>
      </TooltipProvider>
    </SidebarContext.Provider>
  )
}

function Sidebar({
  side = 'left',
  variant = 'sidebar',
  collapsible = 'offcanvas',
  className,
  children,
  ...props
}: React.ComponentProps<'div'> & {
  side?: 'left' | 'right'
  variant?: 'sidebar' | 'floating' | 'inset'
```

```
  collapsible?: 'offcanvas' | 'icon' | 'none'
}) {
  const { isMobile, state, openMobile, setOpenMobile } = useSidebar()

  if (collapsible === 'none') {
    return (
      <div
        data-slot="sidebar"
        className={cn(
          'bg-sidebar text-sidebar-foreground flex h-full w-(--sidebar-width) flex-col',
          className,
        )}
        {...props}
      >
        {children}
      </div>
    )
  }

  if (isMobile) {
    return (
      <Sheet open={openMobile} onOpenChange=
```

/mnt/data/cleaned_project/components/ui/skeleton.tsx

```
import { cn } from '@/lib/utils'

function Skeleton({ className, ...props }: React.ComponentProps<'div'>) {
  return (
    <div
      data-slot="skeleton"
      className={cn('bg-accent animate-pulse rounded-md', className)}
      {...props}
    />
  )
}

export { Skeleton }
```

/mnt/data/cleaned_project/components/ui/slider.tsx

```
'use client'

import * as React from 'react'
import * as SliderPrimitive from '@radix-ui/react-slider'

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

function Slider({
  className,
  defaultValue,
  value,
  min = 0,
  max = 100,
  ...props
}: React.ComponentProps<typeof SliderPrimitive.Root>) {
  const _values = React.useMemo(
    () =>
      Array.isArray(value)
        ? value
        : Array.isArray(defaultValue)
          ? defaultValue
          : [min, max],
    [value, defaultValue, min, max],
  )

  return (
    <SliderPrimitive.Root
      data-slot="slider"
      defaultValue={defaultValue}
```

```
      value={value}
      min={min}
      max={max}
      className={cn(
        'relative flex w-full touch-none items-center select-none data-[disabled]:opacity-50 data-[or
        className,
      )}
      {...props}
    >
      <SliderPrimitive.Track
        data-slot="slider-track"
        className={
          'bg-muted relative grow overflow-hidden rounded-full data-[orientation=horizontal]:h-1.5 da
        }
      >
        <SliderPrimitive.Range
          data-slot="slider-range"
          className={
            'bg-primary absolute data-[orientation=horizontal]:h-full data-[orientation=vertical]:w-f
          }
        />
      </SliderPrimitive.Track>
      {Array.from({ length: _values.length }, (_, index) => (
        <SliderPrimitive.Thumb
          data-slot="slider-thumb"
          key={index}
          className="border-primary ring-ring/50 block size-4 shrink-0 rounded-full border bg-white s
        />
      ))}
    </SliderPrimitive.Root>
  )
}

export { Slider }
```

/mnt/data/cleaned_project/components/ui/sonner.tsx

```
'use client'

import { useTheme } from 'next-themes'
import { Toaster as Sonner, ToasterProps } from 'sonner'

const Toaster = ({ ...props }: ToasterProps) => {
  const { theme = 'system' } = useTheme()

  return (
    <Sonner
      theme={theme as ToasterProps['theme']}
      className="toaster group"
      style={
        {
          '--normal-bg': 'var(--popover)',
          '--normal-text': 'var(--popover-foreground)',
          '--normal-border': 'var(--border)',
        } as React.CSSProperties
      }
      {...props}
    />
  )
}

export { Toaster }
```

/mnt/data/cleaned_project/components/ui/spinner.tsx

```
import { Loader2Icon } from 'lucide-react'

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

function Spinner({ className, ...props }: React.ComponentProps<'svg'>) {
```

```
  return (
    <Loader2Icon
      role="status"
      aria-label="Loading"
      className={cn('size-4 animate-spin', className)}
      {...props}
    />
  )
}

export { Spinner }
```

/mnt/data/cleaned_project/components/ui/switch.tsx

```
'use client'

import * as React from 'react'
import * as SwitchPrimitive from '@radix-ui/react-switch'

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

function Switch({
  className,
  ...props
}: React.ComponentProps<typeof SwitchPrimitive.Root>) {
  return (
    <SwitchPrimitive.Root
      data-slot="switch"
      className={cn(
        'peer data-[state=checked]:bg-primary data-[state=unchecked]:bg-input focus-visible:border-ri
        className,
      )}
      {...props}
    >
      <SwitchPrimitive.Thumb
        data-slot="switch-thumb"
        className={
          'bg-background dark:data-[state=unchecked]:bg-foreground dark:data-[state=checked]:bg-prima
        }
      />
    </SwitchPrimitive.Root>
  )
}

export { Switch }
```

/mnt/data/cleaned_project/components/ui/table.tsx

```
'use client'

import * as React from 'react'

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

function Table({ className, ...props }: React.ComponentProps<'table'>) {
  return (
    <div
      data-slot="table-container"
      className="relative w-full overflow-x-auto"
    >
      <table
        data-slot="table"
        className={cn('w-full caption-bottom text-sm', className)}
        {...props}
      />
    </div>
  )
}

function TableHeader({ className, ...props }: React.ComponentProps<'thead'>) {
```

```
  return (
    <thead
      data-slot="table-header"
      className={cn('[&_tr]:border-b', className)}
      {...props}
    />
  )
}

function TableBody({ className, ...props }: React.ComponentProps<'tbody'>) {
  return (
    <tbody
      data-slot="table-body"
      className={cn('[&_tr:last-child]:border-0', className)}
      {...props}
    />
  )
}

function TableFooter({ className, ...props }: React.ComponentProps<'tfoot'>) {
  return (
    <tfoot
      data-slot="table-footer"
      className={cn(
        'bg-muted/50 border-t font-medium [&>tr]:last:border-b-0',
        className,
      )}
      {...props}
    />
  )
}

function TableRow({ className, ...props }: React.ComponentProps<'tr'>) {
  return (
    <tr
      data-slot="table-row"
      className={cn(
        'hover:bg-muted/50 data-[state=selected]:bg-muted border-b transition-colors',
        className,
      )}
      {...props}
    />
  )
}

function TableHead({ className, ...props }: React.ComponentProps<'th'>) {
  return (
    <th
      data-slot="table-head"
      className={cn(
        'text-foreground h-10 px-2 text-left align-middle font-medium whitespace-nowrap [&:has([role=
        className,
      )}
      {...props}
    />
  )
}

function TableCell({ className, ...props }: React.ComponentProps<'td'>) {
  return (
    <td
      data-slot="table-cell"
      className={cn(
        'p-2 align-middle whitespace-nowrap [&:has([role=checkbox])]:pr-0 [&>[role=checkbox]]:transla
        className,
      )}
      {...props}
    />
  )
}

function TableCaption({
  className,
  ...props
}: React.ComponentProps<'caption'>) {
  return (
    <caption
```

```
      data-slot="table-caption"
      className={cn('text-muted-foreground mt-4 text-sm', className)}
      {...props}
    />
  )
}

export {
  Table,
  TableHeader,
  TableBody,
  TableFooter,
  TableHead,
  TableRow,
  TableCell,
  TableCaption,
}
```

/mnt/data/cleaned_project/components/ui/tabs.tsx

```
'use client'

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

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

function Tabs({
  className,
  ...props
}: React.ComponentProps<typeof TabsPrimitive.Root>) {
  return (
    <TabsPrimitive.Root
      data-slot="tabs"
      className={cn('flex flex-col gap-2', className)}
      {...props}
    />
  )
}

function TabsList({
  className,
  ...props
}: React.ComponentProps<typeof TabsPrimitive.List>) {
  return (
    <TabsPrimitive.List
      data-slot="tabs-list"
      className={cn(
        'bg-muted text-muted-foreground inline-flex h-9 w-fit items-center justify-center rounded-lg
        className,
      )}
      {...props}
    />
  )
}

function TabsTrigger({
  className,
  ...props
}: React.ComponentProps<typeof TabsPrimitive.Trigger>) {
  return (
    <TabsPrimitive.Trigger
      data-slot="tabs-trigger"
      className={cn(
        "data-[state=active]:bg-background dark:data-[state=active]:text-foreground focus-visible:bor
        className,
      )}
      {...props}
    />
  )
}

function TabsContent({
  className,
```

```
  ...props
}: React.ComponentProps<typeof TabsPrimitive.Content>) {
  return (
    <TabsPrimitive.Content
      data-slot="tabs-content"
      className={cn('flex-1 outline-none', className)}
      {...props}
    />
  )
}

export { Tabs, TabsList, TabsTrigger, TabsContent }
```

/mnt/data/cleaned_project/components/ui/textarea.tsx

```
import * as React from 'react'

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

function Textarea({ className, ...props }: React.ComponentProps<'textarea'>) {
  return (
    <textarea
      data-slot="textarea"
      className={cn(
        'border-input placeholder:text-muted-foreground focus-visible:border-ring focus-visible:ring
        className,
      )}
      {...props}
    />
  )
}

export { Textarea }
```

/mnt/data/cleaned_project/components/ui/toast.tsx

```
'use client'

import * as React from 'react'
import * as ToastPrimitives from '@radix-ui/react-toast'
import { cva, type VariantProps } from 'class-variance-authority'
import { X } from 'lucide-react'

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

const ToastProvider = ToastPrimitives.Provider

const ToastViewport = React.forwardRef<
  React.ElementRef<typeof ToastPrimitives.Viewport>,
  React.ComponentPropsWithoutRef<typeof ToastPrimitives.Viewport>
>(({ className, ...props }, ref) => (
  <ToastPrimitives.Viewport
    ref={ref}
    className={cn(
      'fixed top-0 z-[100] flex max-h-screen w-full flex-col-reverse p-4 sm:bottom-0 sm:right-0 sm:to
      className,
    )}
    {...props}
  />
))
ToastViewport.displayName = ToastPrimitives.Viewport.displayName

const toastVariants = cva(
  'group pointer-events-auto relative flex w-full items-center justify-between space-x-4 overflow-hi
  {
    variants: {
      variant: {
        default: 'border bg-background text-foreground',
        destructive:
          'destructive group border-destructive bg-destructive text-destructive-foreground',
```

```
      },
    },
    defaultVariants: {
      variant: 'default',
    },
  },
)

const Toast = React.forwardRef<
  React.ElementRef<typeof ToastPrimitives.Root>,
  React.ComponentPropsWithoutRef<typeof ToastPrimitives.Root> &
    VariantProps<typeof toastVariants>
>(({ className, variant, ...props }, ref) => {
  return (
    <ToastPrimitives.Root
      ref={ref}
      className={cn(toastVariants({ variant }), className)}
      {...props}
    />
  )
})
Toast.displayName = ToastPrimitives.Root.displayName

const ToastAction = React.forwardRef<
  React.ElementRef<typeof ToastPrimitives.Action>,
  React.ComponentPropsWithoutRef<typeof ToastPrimitives.Action>
>(({ className, ...props }, ref) => (
  <ToastPrimitives.Action
    ref={ref}
    className={cn(
      'inline-flex h-8 shrink-0 items-center justify-center rounded-md border bg-transparent px-3 te
      className,
    )}
    {...props}
  />
))
ToastAction.displayName = ToastPrimitives.Action.displayName

const ToastClose = React.forwardRef<
  React.ElementRef<typeof ToastPrimitives.Close>,
  React.ComponentPropsWithoutRef<typeof ToastPrimitives.Close>
>(({ className, ...props }, ref) => (
  <ToastPrimitives.Close
    ref={ref}
    className={cn(
      'absolute right-2 top-2 rounded-md p-1 text-foreground/50 opacity-0 transition-opacity hover:te
      className,
    )}
    toast-close=""
    {...props}
  >
    <X className="h-4 w-4" />
  </ToastPrimitives.Close>
))
ToastClose.displayName = ToastPrimitives.Close.displayName

const ToastTitle = React.forwardRef<
  React.ElementRef<typeof ToastPrimitives.Title>,
  React.ComponentPropsWithoutRef<typeof ToastPrimitives.Title>
>(({ className, ...props }, ref) => (
  <ToastPrimitives.Title
    ref={ref}
    className={cn('text-sm font-semibold', className)}
    {...props}
  />
))
ToastTitle.displayName = ToastPrimitives.Title.displayName

const ToastDescription = React.forwardRef<
  React.ElementRef<typeof ToastPrimitives.Description>,
  React.ComponentPropsWithoutRef<typeof ToastPrimitives.Description>
>(({ className, ...props }, ref) => (
  <ToastPrimitives.Description
    ref={ref}
    className={cn('text-sm opacity-90', className)}
    {...props}
  />
```

```
))
ToastDescription.displayName = ToastPrimitives.Description.displayName

type ToastProps = React.ComponentPropsWithoutRef<typeof Toast>

type ToastActionElement = React.ReactElement<typeof ToastAction>

export {
  type ToastProps,
  type ToastActionElement,
  ToastProvider,
  ToastViewport,
  Toast,
  ToastTitle,
  ToastDescription,
  ToastClose,
  ToastAction,
}
```

/mnt/data/cleaned_project/components/ui/toaster.tsx

```
'use client'

import { useToast } from '@/hooks/use-toast'
import {
  Toast,
  ToastClose,
  ToastDescription,
  ToastProvider,
  ToastTitle,
  ToastViewport,
} from '@/components/ui/toast'

export function Toaster() {
  const { toasts } = useToast()

  return (
    <ToastProvider>
      {toasts.map(function ({ id, title, description, action, ...props }) {
        return (
          <Toast key={id} {...props}>
            <div className="grid gap-1">
              {title && <ToastTitle>{title}</ToastTitle>}
              {description && (
                <ToastDescription>{description}</ToastDescription>
              )}
            </div>
            {action}
            <ToastClose />
          </Toast>
        )
      })}
      <ToastViewport />
    </ToastProvider>
  )
}
```

/mnt/data/cleaned_project/components/ui/toggle-group.tsx

```
'use client'

import * as React from 'react'
import * as ToggleGroupPrimitive from '@radix-ui/react-toggle-group'
import { type VariantProps } from 'class-variance-authority'

import { cn } from '@/lib/utils'
import { toggleVariants } from '@/components/ui/toggle'

const ToggleGroupContext = React.createContext<
  VariantProps<typeof toggleVariants>
```

```
>({
  size: 'default',
  variant: 'default',
})

function ToggleGroup({
  className,
  variant,
  size,
  children,
  ...props
}: React.ComponentProps<typeof ToggleGroupPrimitive.Root> &
  VariantProps<typeof toggleVariants>) {
  return (
    <ToggleGroupPrimitive.Root
      data-slot="toggle-group"
      data-variant={variant}
      data-size={size}
      className={cn(
        'group/toggle-group flex w-fit items-center rounded-md data-[variant=outline]:shadow-xs',
        className,
      )}
      {...props}
    >
      <ToggleGroupContext.Provider value={{ variant, size }}>
        {children}
      </ToggleGroupContext.Provider>
    </ToggleGroupPrimitive.Root>
  )
}

function ToggleGroupItem({
  className,
  children,
  variant,
  size,
  ...props
}: React.ComponentProps<typeof ToggleGroupPrimitive.Item> &
  VariantProps<typeof toggleVariants>) {
  const context = React.useContext(ToggleGroupContext)

  return (
    <ToggleGroupPrimitive.Item
      data-slot="toggle-group-item"
      data-variant={context.variant || variant}
      data-size={context.size || size}
      className={cn(
        toggleVariants({
          variant: context.variant || variant,
          size: context.size || size,
        }),
        'min-w-0 flex-1 shrink-0 rounded-none shadow-none first:rounded-l-md last:rounded-r-md focus
        className,
      )}
      {...props}
    >
      {children}
    </ToggleGroupPrimitive.Item>
  )
}

export { ToggleGroup, ToggleGroupItem }
```

/mnt/data/cleaned_project/components/ui/toggle.tsx

```
'use client'

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

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

const toggleVariants = cva(
```

```
  "inline-flex items-center justify-center gap-2 rounded-md text-sm font-medium hover:bg-muted hover
  {
    variants: {
      variant: {
        default: 'bg-transparent',
        outline:
          'border border-input bg-transparent shadow-xs hover:bg-accent hover:text-accent-foreground
      },
      size: {
        default: 'h-9 px-2 min-w-9',
        sm: 'h-8 px-1.5 min-w-8',
        lg: 'h-10 px-2.5 min-w-10',
      },
    },
    defaultVariants: {
      variant: 'default',
      size: 'default',
    },
  },
)

function Toggle({
  className,
  variant,
  size,
  ...props
}: React.ComponentProps<typeof TogglePrimitive.Root> &
  VariantProps<typeof toggleVariants>) {
  return (
    <TogglePrimitive.Root
      data-slot="toggle"
      className={cn(toggleVariants({ variant, size, className }))}
      {...props}
    />
  )
}

export { Toggle, toggleVariants }
```

/mnt/data/cleaned_project/components/ui/tooltip.tsx

```
'use client'

import * as React from 'react'
import * as TooltipPrimitive from '@radix-ui/react-tooltip'

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

function TooltipProvider({
  delayDuration = 0,
  ...props
}: React.ComponentProps<typeof TooltipPrimitive.Provider>) {
  return (
    <TooltipPrimitive.Provider
      data-slot="tooltip-provider"
      delayDuration={delayDuration}
      {...props}
    />
  )
}

function Tooltip({
  ...props
}: React.ComponentProps<typeof TooltipPrimitive.Root>) {
  return (
    <TooltipProvider>
      <TooltipPrimitive.Root data-slot="tooltip" {...props} />
    </TooltipProvider>
  )
}

function TooltipTrigger({
  ...props
}: React.ComponentProps<typeof TooltipPrimitive.Trigger>) {
```

```
  return <TooltipPrimitive.Trigger data-slot="tooltip-trigger" {...props} />
}

function TooltipContent({
  className,
  sideOffset = 0,
  children,
  ...props
}: React.ComponentProps<typeof TooltipPrimitive.Content>) {
  return (
    <TooltipPrimitive.Portal>
      <TooltipPrimitive.Content
        data-slot="tooltip-content"
        sideOffset={sideOffset}
        className={cn(
          'bg-foreground text-background animate-in fade-in-0 zoom-in-95 data-[state=closed]:animate
          className,
        )}
        {...props}
      >
        {children}
        <TooltipPrimitive.Arrow className="bg-foreground fill-foreground z-50 size-2.5 translate-y-[
      </TooltipPrimitive.Content>
    </TooltipPrimitive.Portal>
  )
}

export { Tooltip, TooltipTrigger, TooltipContent, TooltipProvider }
```

/mnt/data/cleaned_project/components/ui/use-mobile.tsx

```
import * as React from 'react'

const MOBILE_BREAKPOINT = 768

export function useIsMobile() {
  const [isMobile, setIsMobile] = React.useState<boolean | undefined>(undefined)

  React.useEffect(() => {
    const mql = window.matchMedia(`(max-width: ${MOBILE_BREAKPOINT - 1}px)`)
    const onChange = () => {
      setIsMobile(window.innerWidth < MOBILE_BREAKPOINT)
    }
    mql.addEventListener('change', onChange)
    setIsMobile(window.innerWidth < MOBILE_BREAKPOINT)
    return () => mql.removeEventListener('change', onChange)
  }, [])

  return !!isMobile
}
```

/mnt/data/cleaned_project/components/ui/use-toast.ts

```
'use client'

// Inspired by react-hot-toast library
import * as React from 'react'

import type { ToastActionElement, ToastProps } from '@/components/ui/toast'

const TOAST_LIMIT = 1
const TOAST_REMOVE_DELAY = 1000000

type ToasterToast = ToastProps & {
  id: string
  title?: React.ReactNode
  description?: React.ReactNode
  action?: ToastActionElement
}
const actionTypes = {
```

```
  ADD_TOAST: 'ADD_TOAST',
  UPDATE_TOAST: 'UPDATE_TOAST',
  DISMISS_TOAST: 'DISMISS_TOAST',
  REMOVE_TOAST: 'REMOVE_TOAST',
} as const

let count = 0

function genId() {
  count = (count + 1) % Number.MAX_SAFE_INTEGER
  return count.toString()
}

type ActionType = typeof actionTypes

type Action =
  | {
      type: ActionType['ADD_TOAST']
      toast: ToasterToast
    }
  | {
      type: ActionType['UPDATE_TOAST']
      toast: Partial<ToasterToast>
    }
  | {
      type: ActionType['DISMISS_TOAST']
      toastId?: ToasterToast['id']
    }
  | {
      type: ActionType['REMOVE_TOAST']
      toastId?: ToasterToast['id']
    }

interface State {
  toasts: ToasterToast[]
}

const toastTimeouts = new Map<string, ReturnType<typeof setTimeout>>()

const addToRemoveQueue = (toastId: string) => {
  if (toastTimeouts.has(toastId)) {
    return
  }

  const timeout = setTimeout(() => {
    toastTimeouts.delete(toastId)
    dispatch({
      type: 'REMOVE_TOAST',
      toastId: toastId,
    })
  }, TOAST_REMOVE_DELAY)

  toastTimeouts.set(toastId, timeout)
}

export const reducer = (state: State, action: Action): State => {
  switch (action.type) {
    case 'ADD_TOAST':
      return {
        ...state,
        toasts: [action.toast, ...state.toasts].slice(0, TOAST_LIMIT),
      }

    case 'UPDATE_TOAST':
      return {
        ...state,
        toasts: state.toasts.map((t) =>
          t.id === action.toast.id ? { ...t, ...action.toast } : t,
        ),
      }

    case 'DISMISS_TOAST': {
      const { toastId } = action

      // ! Side effects ! - This could be extracted into a dismissToast() action,
      // but I'll keep it here for simplicity
      if (toastId) {
```

```
        addToRemoveQueue(toastId)
      } else {
        state.toasts.forEach((toast) => {
          addToRemoveQueue(toast.id)
        })
      }

      return {
        ...state,
        toasts: state.toasts.map((t) =>
          t.id === toastId || toastId === undefined
            ? {
                ...t,
                open: false,
              }
            : t,
        ),
      }
    }
    case 'REMOVE_TOAST':
      if (action.toastId === undefined) {
        return {
          ...state,
          toasts: [],
        }
      }
      return {
        ...state,
        toasts: state.toasts.filter((t) => t.id !== action.toastId),
      }
  }
}

const listeners: Array<(state: State) => void> = []

let memoryState: State = { toasts: [] }

function dispatch(action: Action) {
  memoryState = reducer(memoryState, action)
  listeners.forEach((listener) => {
    listener(memoryState)
  })
}

type Toast = Omit<ToasterToast, 'id'>

function toast({ ...props }: Toast) {
  const id = genId()

  const update = (props: ToasterToast) =>
    dispatch({
      type: 'UPDATE_TOAST',
      toast: { ...props, id },
    })
  const dismiss = () => dispatch({ type: 'DISMISS_TOAST', toastId: id })

  dispatch({
    type: 'ADD_TOAST',
    toast: {
      ...props,
      id,
      open: true,
      onOpenChange: (open) => {
        if (!open) dismiss()
      },
    },
  })

  return {
    id: id,
    dismiss,
    update,
  }
}

function useToast() {
  const [state, setState] = React.useState<State>(memoryState)
```

```
  React.useEffect(() => {
    listeners.push(setState)
    return () => {
      const index = listeners.indexOf(setState)
      if (index > -1) {
        listeners.splice(index, 1)
      }
    }
  }, [state])

  return {
    ...state,
    toast,
    dismiss: (toastId?: string) => dispatch({ type: 'DISMISS_TOAST', toastId }),
  }
}

export { useToast, toast }
```

/mnt/data/cleaned_project/contexts/language-context.tsx

```
"use client";

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

type Language = "ar" | "en";

interface Translations {
  [key: string]: {
    ar: string;
    en: string;
  };
}

export const translations: Translations = {
  flex: { ar: "FLEX", en: "FLEX" },
  home: { ar: "[illegible]", en: "Home" },
  promo: { ar: "[illegible]", en: "Promo" },
  history: { ar: "[illegible]", en: "History" },
  account: { ar: "[illegible]", en: "Account" },
  popularMovies: { ar: "[illegible] [illegible]", en: "Popular Movies" },
  newMovies: { ar: "[illegible] [illegible]", en: "New Movies" },
  upcomingMovies: { ar: "[illegible] / [illegible] [illegible]", en: "Coming Soon" },
  viewMore: { ar: "[illegible] [illegible]", en: "View More" },
  partner: { ar: "[illegible] [illegible]", en: "City Partner" },
  intro: { ar: "[illegible] [illegible]", en: "System Intro" },
  membership: { ar: "[illegible]", en: "Membership" },
  invite: { ar: "[illegible] [illegible]", en: "Invite Friend" },
  announcement: { ar: "[illegible] VIP [illegible] [illegible]! [illegible] [illegible] [illegible] [illegible] - [illegible] [illegible] [illegible]
  selectLanguage: { ar: "[illegible] [illegible]", en: "Select Language" },
  arabic: { ar: "[illegible]", en: "Arabic" },
  english: { ar: "[illegible]", en: "English" },
  notifications: { ar: "[illegible]", en: "Notifications" },
  withdrawal: { ar: "[illegible]", en: "Withdrawal" },
  deposit: { ar: "[illegible]", en: "Deposit" },
  noNotifications: { ar: "[illegible] [illegible] [illegible]", en: "No notifications" },
  close: { ar: "[illegible]", en: "Close" },
  bannerTitle1: { ar: "[illegible] [illegible] [illegible] FLEX", en: "Join FLEX City Partners" },
  bannerSubtitle1: { ar: "[illegible] [illegible] [illegible] [illegible] [illegible] [illegible]", en: "Entrepreneurial oppo
  bannerTitle2: { ar: "[illegible] [illegible] [illegible]", en: "New Exclusive Movies" },
  bannerSubtitle2: { ar: "[illegible] [illegible] [illegible] [illegible] [illegible] [illegible]", en: "Watch the latest glo
  bannerTitle3: { ar: "[illegible] [illegible] [illegible]", en: "Diverse Premium Content" },
  bannerSubtitle3: { ar: "[illegible] [illegible] [illegible] [illegible] [illegible] [illegible]", en: "Thousands of movies an
  subscriptions: { ar: "[illegible]", en: "Subscriptions" },
  membersCenter: { ar: "[illegible] [illegible]", en: "Members Center" },
  currentLevel: { ar: "[illegible] [illegible]", en: "Current Level" },
  todayProfits: { ar: "[illegible] [illegible]", en: "Today's Profits" },
  cumulativeIncome: { ar: "[illegible] [illegible]", en: "Cumulative Income" },
  specialGiftPackage: { ar: "[illegible] [illegible] [illegible]", en: "Special Gift Package" },
  movieViewingIncome: { ar: "[illegible] [illegible] [illegible]", en: "Movie Viewing Income" },
  expirationTime: { ar: "[illegible] [illegible] [illegible]", en: "Expiration Time" },
  dailyRevenueCount: { ar: "[illegible] [illegible] [illegible]", en: "Daily Revenue Count" },
  earnProfits: { ar: "[illegible] [illegible]", en: "Earn Profits" },
  back: { ar: "[illegible]", en: "Back" },
```

```
  day: { ar: "[redacted]", en: "Day" },
  times: { ar: "[redacted]", en: "Times" },
  openNow: { ar: "[redacted]", en: "Open Now" },
  comingSoon: { ar: "Coming Soon", en: "Coming Soon" },
  noLevel: { ar: "-", en: "-" },
  enterSecretCode: { ar: "[redacted]", en: "Enter Secret Code" },
  secretCodeDescription: { ar: "[redacted]", en: "Enter a 6-digit code" },
  confirm: { ar: "[redacted]", en: "Confirm" },
  cancel: { ar: "[redacted]", en: "Cancel" },
  dailyTasks: { ar: "[redacted]", en: "Daily Tasks" },
  tasksRemaining: { ar: "[redacted]", en: "Tasks Remaining" },
  likeToComplete: { ar: "[redacted]", en: "Tap the heart to complete task" },
  taskCompleted: { ar: "[redacted]", en: "Task Completed!" },
  allTasksCompleted: { ar: "[redacted]", en: "All tasks completed today!" },
  team: { ar: "[redacted]", en: "Team" },
  totalUserIncome: { ar: "[redacted]", en: "Total User Income" },
  todayIncome: { ar: "[redacted]", en: "Today's Income" },
  benefitsAnalysis: { ar: "[redacted]", en: "Benefits Analysis" },
  totalReturn: { ar: "[redacted]", en: "Total Return" },
  ratingIncome: { ar: "[redacted]", en: "Rating Income" },
  teamIncome: { ar: "[redacted]", en: "Team Income" },
  investmentIncome: { ar: "[redacted]", en: "Investment Income" },
  teamMembersList: { ar: "[redacted]", en: "Team Members List" },
  totalTeamMembers: { ar: "[redacted]", en: "Total Team Members" },
  addedToday: { ar: "[redacted]", en: "Added Today" },
  addedLastWeek: { ar: "[redacted]", en: "Added Last Week" },
  addedThisWeek: { ar: "[redacted]", en: "Added This Week" },
  generation1: { ar: "[redacted]", en: "Generation 1" },
  generation2: { ar: "[redacted]", en: "Generation 2" },
  generation3: { ar: "[redacted]", en: "Generation 3" },
  subscribe: { ar: "[redacted]", en: "Subscribe" },
  like: { ar: "[redacted]", en: "Like" },
  dislike: { ar: "[redacted]", en: "Dislike" },
  comments: { ar: "[redacted]", en: "Comments" },
  share: { ar: "[redacted]", en: "Share" },
  remix: { ar: "[redacted]", en: "Remix" },
};

interface LanguageContext
```

/mnt/data/cleaned_project/hooks/use-mobile.ts

```
import * as React from 'react'

const MOBILE_BREAKPOINT = 768

export function useIsMobile() {
  const [isMobile, setIsMobile] = React.useState<boolean | undefined>(undefined)

  React.useEffect(() => {
    const mql = window.matchMedia(`(max-width: ${MOBILE_BREAKPOINT - 1}px)`)
    const onChange = () => {
      setIsMobile(window.innerWidth < MOBILE_BREAKPOINT)
    }
    mql.addEventListener('change', onChange)
    setIsMobile(window.innerWidth < MOBILE_BREAKPOINT)
    return () => mql.removeEventListener('change', onChange)
  }, [])

  return !!isMobile
}
```

/mnt/data/cleaned_project/hooks/use-toast.ts

```
'use client'

// Inspired by react-hot-toast library
import * as React from 'react'

import type { ToastActionElement, ToastProps } from '@/components/ui/toast'
```

```
const TOAST_LIMIT = 1
const TOAST_REMOVE_DELAY = 1000000

type ToasterToast = ToastProps & {
  id: string
  title?: React.ReactNode
  description?: React.ReactNode
  action?: ToastActionElement
}

const actionTypes = {
  ADD_TOAST: 'ADD_TOAST',
  UPDATE_TOAST: 'UPDATE_TOAST',
  DISMISS_TOAST: 'DISMISS_TOAST',
  REMOVE_TOAST: 'REMOVE_TOAST',
} as const

let count = 0

function genId() {
  count = (count + 1) % Number.MAX_SAFE_INTEGER
  return count.toString()
}

type ActionType = typeof actionTypes

type Action =
  | {
      type: ActionType['ADD_TOAST']
      toast: ToasterToast
    }
  | {
      type: ActionType['UPDATE_TOAST']
      toast: Partial<ToasterToast>
    }
  | {
      type: ActionType['DISMISS_TOAST']
      toastId?: ToasterToast['id']
    }
  | {
      type: ActionType['REMOVE_TOAST']
      toastId?: ToasterToast['id']
    }

interface State {
  toasts: ToasterToast[]
}

const toastTimeouts = new Map<string, ReturnType<typeof setTimeout>>()

const addToRemoveQueue = (toastId: string) => {
  if (toastTimeouts.has(toastId)) {
    return
  }

  const timeout = setTimeout(() => {
    toastTimeouts.delete(toastId)
    dispatch({
      type: 'REMOVE_TOAST',
      toastId: toastId,
    })
  }, TOAST_REMOVE_DELAY)

  toastTimeouts.set(toastId, timeout)
}

export const reducer = (state: State, action: Action): State => {
  switch (action.type) {
    case 'ADD_TOAST':
      return {
        ...state,
        toasts: [action.toast, ...state.toasts].slice(0, TOAST_LIMIT),
      }

    case 'UPDATE_TOAST':
      return {
        ...state,
```

```
        toasts: state.toasts.map((t) =>
          t.id === action.toast.id ? { ...t, ...action.toast } : t,
        ),
      }

    case 'DISMISS_TOAST': {
      const { toastId } = action

      // ! Side effects ! - This could be extracted into a dismissToast() action,
      // but I'll keep it here for simplicity
      if (toastId) {
        addToRemoveQueue(toastId)
      } else {
        state.toasts.forEach((toast) => {
          addToRemoveQueue(toast.id)
        })
      }

      return {
        ...state,
        toasts: state.toasts.map((t) =>
          t.id === toastId || toastId === undefined
            ? {
                ...t,
                open: false,
              }
            : t,
        ),
      }
    }
    case 'REMOVE_TOAST':
      if (action.toastId === undefined) {
        return {
          ...state,
          toasts: [],
        }
      }
      return {
        ...state,
        toasts: state.toasts.filter((t) => t.id !== action.toastId),
      }
  }
}

const listeners: Array<(state: State) => void> = []

let memoryState: State = { toasts: [] }

function dispatch(action: Action) {
  memoryState = reducer(memoryState, action)
  listeners.forEach((listener) => {
    listener(memoryState)
  })
}

type Toast = Omit<ToasterToast, 'id'>

function toast({ ...props }: Toast) {
  const id = genId()

  const update = (props: ToasterToast) =>
    dispatch({
      type: 'UPDATE_TOAST',
      toast: { ...props, id },
    })
  const dismiss = () => dispatch({ type: 'DISMISS_TOAST', toastId: id })

  dispatch({
    type: 'ADD_TOAST',
    toast: {
      ...props,
      id,
      open: true,
      onOpenChange: (open) => {
        if (!open) dismiss()
      },
    },
```

```
  })

  return {
    id: id,
    dismiss,
    update,
  }
}

function useToast() {
  const [state, setState] = React.useState<State>(memoryState)

  React.useEffect(() => {
    listeners.push(setState)
    return () => {
      const index = listeners.indexOf(setState)
      if (index > -1) {
        listeners.splice(index, 1)
      }
    }
  }, [state])

  return {
    ...state,
    toast,
    dismiss: (toastId?: string) => dispatch({ type: 'DISMISS_TOAST', toastId }),
  }
}

export { useToast, toast }
```

/mnt/data/cleaned_project/lib/utils.ts

```
import { clsx, type ClassValue } from 'clsx'
import { twMerge } from 'tailwind-merge'

export function cn(...inputs: ClassValue[]) {
  return twMerge(clsx(inputs))
}
```