Build this as my initial prototype

Copy-paste this component to /components/ui folder:

```tsx
globe.tsx
"use client"

import createGlobe, { COBEOptions } from "cobe"
import { useCallback, useEffect, useRef, useState } from "react"

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

const GLOBE_CONFIG: COBEOptions = {
  width: 800,
  height: 800,
  onRender: () => {},
  devicePixelRatio: 2,
  phi: 0,
  theta: 0.3,
  dark: 0,
  diffuse: 0.4,
  mapSamples: 16000,
  mapBrightness: 1.2,
  baseColor: [1, 1, 1],
  markerColor: [251 / 255, 100 / 255, 21 / 255],
  glowColor: [1, 1, 1],
  markers: [
    { location: [14.5995, 120.9842], size: 0.03 },
    { location: [19.076, 72.8777], size: 0.1 },
    { location: [23.8103, 90.4125], size: 0.05 },
    { location: [30.0444, 31.2357], size: 0.07 },
    { location: [39.9042, 116.4074], size: 0.08 },
    { location: [-23.5505, -46.6333], size: 0.1 },
    { location: [19.4326, -99.1332], size: 0.1 },
    { location: [40.7128, -74.006], size: 0.1 },
    { location: [34.6937, 135.5022], size: 0.05 },
    { location: [41.0082, 28.9784], size: 0.06 },
  ],
}

export function Globe({
  className,
  config = GLOBE_CONFIG,
}: {
  className?: string
  config?: COBEOptions
}) {
  let phi = 0
  let width = 0
  const canvasRef = useRef<HTMLCanvasElement>(null)
  const pointerInteracting = useRef(null)
  const pointerInteractionMovement = useRef(0)
  const [r, setR] = useState(0)

  const updatePointerInteraction = (value: any) => {
    pointerInteracting.current = value
    if (canvasRef.current) {
      canvasRef.current.style.cursor = value ? "grabbing" : "grab"
    }
  }

  const updateMovement = (clientX: any) => {
    if (pointerInteracting.current !== null) {
      const delta = clientX - pointerInteracting.current
      pointerInteractionMovement.current = delta
      setR(delta / 200)
    }
  }

  const onRender = useCallback(
    (state: Record<string, any>) => {
      if (!pointerInteracting.current) phi += 0.005
      state.phi = phi + r
      state.width = width * 2
      state.height = width * 2
    },
    [r],
  )

  const onResize = () => {
    if (canvasRef.current) {
      width = canvasRef.current.offsetWidth
    }
  }

  useEffect(() => {
    window.addEventListener("resize", onResize)
    onResize()
    const globe = createGlobe(canvasRef.current!, {
      ...config,
      width: width * 2,
      height: width * 2,
      onRender,
    })

    setTimeout(() => (canvasRef.current!.style.opacity = "1"))
    return () => globe.destroy()
  }, [])

  return (
    <div
      className={cn(
        "absolute inset-0 mx-auto aspect-[1/1] w-full max-w-[600px]",
        className,
      )}
    >
      <canvas
        className={cn(
          "size-full opacity-0 transition-opacity duration-500 [contain:layout_paint_size]",
        )}
        ref={canvasRef}
        onPointerDown={(e) =>
          updatePointerInteraction(
            e.clientX - pointerInteractionMovement.current,
          )
        }
        onPointerUp={() => updatePointerInteraction(null)}
        onPointerOut={() => updatePointerInteraction(null)}
        onMouseMove={(e) => updateMovement(e.clientX)}
        onTouchMove={(e) =>
          e.touches[0] && updateMovement(e.touches[0].clientX)
        }
      />
    </div>
  )
}
```

demo.tsx

```
import { Globe } from "@/components/ui/globe"
```

```
export function GlobeDemo() {
  return (
    <div className="relative flex size-full max-w-lg items-center justify-center overflow-hidden rounded-lg border bg-background px-40 pb-40 pt-8 md:pb-60 md:shadow-xl">
      <span className="pointer-events-none whitespace-pre-wrap bg-gradient-to-b from-black to-gray-300/80 bg-clip-text text-center text-8xl font-semibold leading-none text-transparent dark:from-white dark:to-slate-900/10">
        Globe
      </span>
      <Globe className="top-28" />
      <div className="pointer-events-none absolute inset-0 h-full bg-[radial-gradient(circle_at_50%_200%,rgba(0,0,0,0.2),rgba(255,255,255,0))]" />
    </div>
  )
}

```

Install these NPM dependencies:

```bash
cobe
```