'use client';

import React, { useEffect, useRef } from 'react';

import gsap from 'gsap';

const TileEffectBg = ({ imageUrl }) => {

const TILE\_SIZE = 60; // ✅ Place this inside component

const containerRef = useRef(null);

useEffect(() => {

const tiles = gsap.utils.toArray('.tile');

gsap.set(tiles, { opacity: 1 });

const tl = gsap.timeline({ repeat: -1, repeatDelay: 1 });

tl.to(tiles, {

duration: 1,

x: () => gsap.utils.random(-150, 150),

y: () => gsap.utils.random(-150, 150),

rotation: () => gsap.utils.random(-180, 180),

opacity: 0,

ease: "power2.inOut",

stagger: {

each: 0.008,

from: "center",

},

});

tl.to(tiles, {

duration: 1.2,

x: 0,

y: 0,

rotation: 0,

opacity: 1,

ease: "power3.out",

stagger: {

each: 0.008,

from: "center",

},

});

}, []);

const containerWidth = 1920;

const containerHeight = 1080;

const rows = Math.ceil(containerHeight / TILE\_SIZE);

const cols = Math.ceil(containerWidth / TILE\_SIZE);

return (

<div

className="absolute top-0 left-0 z-0 w-full h-full opacity-[0.1] pointer-events-none overflow-hidden"

ref={containerRef}

>

{[...Array(rows \* cols)].map((\_, i) => {

const row = Math.floor(i / cols);

const col = i % cols;

return (

<div

key={i}

className="tile absolute"

style={{

width: TILE\_SIZE,

height: TILE\_SIZE,

top: row \* TILE\_SIZE,

left: col \* TILE\_SIZE,

backgroundImage: `url(${imageUrl})`,

backgroundSize: `${cols \* TILE\_SIZE}px ${rows \* TILE\_SIZE}px`,

backgroundPosition: `-${col \* TILE\_SIZE}px -${row \* TILE\_SIZE}px`,

}}

/>

);

})}

</div>

);

};

export default TileEffectBg;