



SONIX.AI

A curated audio visualizer — watch reactive 3D visuals to
your music.

Team Members



**Almira
Koshkina**

Full Stack
Developer



**Dinar
Ibragimov**

Full Stack
Developer



**Joe
Siburov**

Full Stack
Developer



**Lily
Meyer**

Full Stack
Developer



**Olesia
Mironenko**

Full Stack
Developer



**Sergey
Sherstobitov**

Mentor



**Serhii
Smyk**

Mentor

THE PROBLEM
your app
addresses:

Listening to music is a **flat, screen-off experience** – the audio you love never becomes something you can see.
And for events and performers, there is still **no simple, AI-powered way** to create live interactive visuals.

**THE
SOLUTION:**

Connect any audio source and watch reactive 3D visuals come alive.

Browse the library • Save your favorites • Generate your own with AI

**PROJECT
STATUS:**



✓ **Shipped**

Complete MVP – auth with tiered users
Live visualizer and Explore library
Favorites, seed scripts, test coverage
AI-generated visuals – our stretch goal, done!

→ **Next**

Visuals created by users with AI
Premium shaders via subscriptions
Collections shared between users



Tech Stack



React 19

UI · Vite · Tailwind



three.js

real-time 3D / WebGL



Gemini

AI-generated visuals



TypeScript

shared-types monorepo



MongoDB

Mongoose + GridFS

FRONTEND

TanStack React Query

Vitest · unit tests

Playwright · E2E tests

Google Stitch · design

Responsive · Accessible

BACKEND · SHARED

Node.js + Express 5

JWT auth · zero-trust

Pre-commit quality (Husky + lint)

Web Media API

CI / CD



Challenges • Working as a Team

New stack, whole team

Our first typed monorepo – learning TypeScript, generics, and strict compiler errors while shipping.

Staying out of each other's way

Branches drifted, conflicts piled up – two of us even built the same image-storage feature and had to reconcile designs.



The finish-line crunch

A wave of tickets near the deadline forced us to prioritize, split work clearly, and communicate constantly.



Challenges • Under the Hood

Real-time audio → shaders

Mic permissions, device switching, and keeping WebGL animation smooth.

AI that writes GLSL

Gemini output had to be cleaned and validated before it ever hit the renderer.

Rate limits & load

A global limiter blocked the whole team; targeted limits + cache-control fixed it.

Images at scale

GridFS storage with validated uploads, streaming, ownership, and caching.



Learnings

01

Types pay off

TypeScript caught bugs at compile time and made refactors safe.

02

Resync early, often

Small, frequent merges beat one big painful one.

03

Tests catch it first

Coverage caught regressions in CI instead of in the demo.

04

Make the API contract first

We kept changing endpoints mid-build, and every change meant rework.

05

Cache before you scale

Reusing already-fetched data erased most redundant backend load.



Most Enjoyed

Hard problems, solved together

The team clicked when things got difficult.

Learning from mentors

And from each other – every review taught something.

AI in a real product

A shipped feature users actually use – not a tech demo.

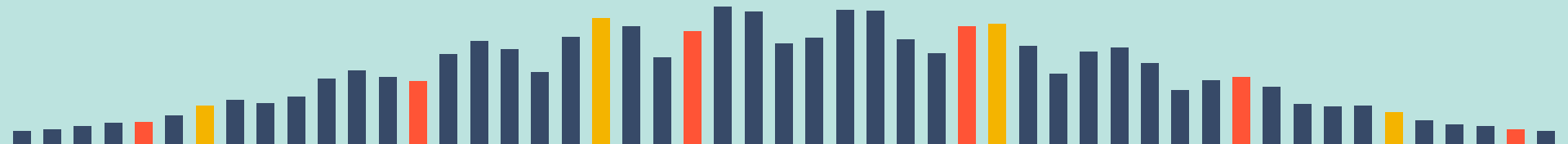
Shipping something stand-out

A project we are genuinely proud to show.



Thank You!

Shoutout to our mentors and the Code the Dream community.





Questions?

