SceneSense AI

Top 30 Hackathon Judge Q&A; (Final)

1) What problem are you solving?

Screenplay scenes are unstructured text. Film teams manually interpret emotion, mood, lighting, and camera style which is slow and inconsistent. SceneSense AI converts a scene into structured cinematic intent so teams align faster.

2) What is your solution in one line?

SceneSense AI turns screenplay scenes into cinematic intent JSON: emotion, narrative purpose, visual mood, camera style, and confidence.

3) Why is this needed now?

Content creation is increasing and teams need faster workflows. Indie creators and students also need tools that reduce planning time and help them learn visual storytelling.

4) Who are your target users?

Directors, cinematographers (DoP), storyboard/previz artists, film students, and indie creators.

5) What makes your solution unique?

We don’t just summarize the text. We extract actionable cinematic intent fields and return strict JSON so the output can be reused for shot planning and future integrations.

6) What is the key value you deliver?

Faster alignment. Instead of a long discussion to decide mood and shots, the team gets a structured baseline in seconds.

7) What exactly is the output?

A valid JSON object containing emotion, narrative purpose, visual mood, camera style, and a confidence score between 0 and 1.

8) Why did you choose JSON output?

Because JSON is structured, machine-readable, and easy to integrate with future tools like shot-list generation, storyboards, and production planning systems.

9) Why did you use an LLM for this?

Cinematic intent requires understanding context and subtext, not keywords. LLMs can infer emotion and purpose from language patterns much better than rule-based methods.

10) Why Groq specifically?

Groq provides very fast inference which is ideal for a hackathon demo and rapid iteration. It makes the experience feel instant.

11) What model are you using?

We use Llama 3.1 8B Instant via the Groq API for fast and consistent responses.

12) How does the system work end-to-end?

User pastes a scene → app validates input → prompt forces strict JSON → Groq LLM generates output → app parses JSON → UI shows results and confidence label.

13) What are the major components of your system?

Streamlit UI, Python inference + prompt builder, Groq LLM API, JSON parser, and .env for secure key management.

14) How do you ensure the output is consistent?

We enforce a strict JSON schema, keep temperature low, and validate outputs with JSON parsing and fallbacks.

15) What happens if the model returns invalid JSON?

We catch JSON parsing errors and show the raw output so the demo doesn’t break. The prompt also strongly reduces this risk.

16) What does confidence mean in your tool?

It represents how strongly the model believes its intent extraction matches the scene. We map it to High/Medium/Low for easy understanding.

17) How accurate is this?

This is a creative-assist tool, not a factual classifier. It provides a strong baseline. Accuracy is measured by consistency, relevance, and usefulness to a director’s decision-making.

18) How would you evaluate quality?

We would run a set of scenes across genres and check: JSON success rate, differentiation by genre, and feedback from film creators on usefulness.

19) Can this be used for full scripts?

Yes, in future. Current MVP supports one scene at a time. Next version can upload multiple scenes and generate a full breakdown.

20) What is your MVP scope today?

Single-scene input, example scenes, Analyze button, strict JSON output, confidence label, and basic error handling.

21) What features would you add next?

Multi-scene batch analysis, shot-list generation, export to PDF/CSV, role-based outputs, and integrations with storyboard tools.

22) How does this help real production teams?

It saves time in pre-production discussions and creates a shared baseline across director, DoP, and storyboard teams, reducing misalignment.

23) How does this help beginners/students?

Students learn how scenes map to visuals. They can compare outputs across genres and understand tone, mood, and framing patterns.

24) What are the limitations of your MVP?

It doesn’t generate full storyboards or guarantee perfect creative interpretation for every scene type. It’s a guidance system.

25) What if the input is too short or vague?

The tool warns the user and asks for a longer scene, because the model needs enough context for meaningful intent extraction.

26) Does your tool store user scenes?

No. The tool processes input for inference and displays results. There is no database storage in the hackathon MVP.

27) How do you handle security for API keys?

Keys are stored in .env locally. The .env file is ignored by Git so keys are never uploaded.

28) How will you scale this if many users use it?

We can add caching, request queues, batched inference, and later deploy a backend service with rate limiting and monitoring.

29) What is the business potential / monetization?

It can be offered as a SaaS tool for creators with premium features like script-level analysis, exports, and production templates.

30) If you had more time, what would you improve first?

Batch script upload, shot-list generation, and export features—these will directly convert intent into production-ready outputs.